Medical Ultrasound Imaging and Interventional Component (MUSiiC) Framework for Advanced Ultrasound Image-guided Therapy

by

Hyun Jae Kang

A dissertation submitted to Johns Hopkins University in conformity with the requirements for the degree of Doctor of Philosophy

Baltimore, Maryland

July, 2015

Abstract

Medical ultrasound (US) imaging is a popular and convenient medical imaging modality thanks to its mobility, non-ionizing radiation, ease-of-use, and real-time data acquisition. Conventional US brightness mode (B-Mode) is one type of diagnostic medical imaging modality that represents tissue morphology by collecting and displaying the intensity information of a reflected acoustic wave. Moreover, US B-Mode imaging is frequently integrated with tracking systems and robotic systems in image-guided therapy (IGT) systems. Recently, these systems have also begun to incorporate advanced US imaging such as US elasticity imaging, photoacoustic imaging, and thermal imaging. Several software frameworks and toolkits have been developed for US imaging research and the integration of US data acquisition, processing and display with existing IGT systems. However, there is no software framework or toolkit that supports advanced US imaging research and advanced US IGT systems by providing low-level US data (channel data or radio-frequency (RF) data) essential for advanced US imaging.

In this dissertation, we propose a new medical US imaging and interventional component framework for advanced US image-guided therapy based on network-distributed modularity, real-time computation and communication, and open-interface design specifications. Consequently, the framework can provide a modular research environment by supporting communication interfaces between heterogeneous systems to allow for flexible interventional US imaging research, and easy reconfiguration of an entire interventional US imaging system by adding or removing devices or equipment specific to each therapy. In addition, our proposed framework offers real-time synchronization between data from multiple data acquisition devices for advanced interventional US imaging research and integration of the US imaging system with other IGT systems. Moreover, we can easily implement and test new advanced ultrasound imaging techniques inside the proposed framework in real-time because our software framework is designed and optimized for advanced ultrasound research. The system's flexibility, real-time performance, and open-interface are demonstrated and evaluated through performing experimental tests for several applications.

Advisor: **Emad M. Boctor, Ph. D.**
Assistant Professor, Radiology and Radiological Science, Department of Computer Science/Electrical and Computer Engineering and
Director, Medical Ultrasound Imaging and Intervention Collaboration (MUSiiC) Research Laboratory
The Johns Hopkins University

Readers: **Russell H. Taylor, Ph. D.**
JOHN C. MALONE Professor, Department of Computer Science
Director, Engineering Research Center for Computer-Integrated Surgical System and Technology (CISST-ERC)
The Johns Hopkins University

Peter Kazanzides, Ph. D.
Research Professor, Department of Computer Science
Chief Systems and Robotic Engineer, Engineering Research Center for Computer-Integrated Surgical System and Technology (CISST-ERC)
The Johns Hopkins University

Acknowledgements

I am grateful to my advisor, Dr. Emad M. Boctor for providing me with the opportunity to work in his lab and it has been an honor to have started my professional career as a computer scientist with him. He has been a great advisor and mentor and I have learned so much under his guidance. Without his guidance and persistent help, this dissertation would not have been possible. I am sincerely appreciative for all he has done.

I would also like to thank Dr. Russell H. Taylor and Dr. Peter Kazanzides for their support and for serving on my thesis committee. Their persistence and dedication for graduate education inspired me to learn about the knowledge of computer science. I never would have accomplished this goal and pursued a Ph. D., if it was not for the mentorship and guidance of Dr. Russell H. Taylor and Dr. Peter Kazanzides.

I want to extend a special thanks to all current and former MUSiiC lab members. Particular thanks to Dr. Philipp J. Stolka, MD. Daniel A. Carnegie, Dr. Hassan Rivaz, Dr. Pezhman Foroughi, Dr. Ioana Fleming, Dr. Nathanael Kuo, Dr. Xiaoyu Guo, Dr. Muyinatu A. Lediju Bell, Dr. Behnoosh Tavakoli and Dr. Lei Chen who played fundamental roles in developing my science and engineering knowledge. I would like to specifically thank to Nishikant Deshmukh, Seth Billings, Alexis Cheng, Haichong Zhang, Fereshteh Aalamifar and Younsu Kim for being my friend/mentor/great lab-mate. Alexis Cheng looked closely at the final version of the thesis for English style and grammar, correcting both and offering suggestions for improvement, Thanks!

Special thanks to all of my best friends, Dr. Jin Seob Kim, Nathan Cho, Dr. Min Yang Jung, Dr. Sungmin Kim, Byunggu Ahn, Dr. Jong Hyun Lim, Dr. JeongGil Ko, Dr. Chunwoo Kim, Jaepyeong Cha, Gyeongwoo Cheon, Doyoung Chang, Travis Jin Choi, Jung Yoon, Dr. Ben Park, Dr. Ga Young Park and Dr. Moon Chul Jung. I could not have finished my graduate studies without your friendship and support.

Finally, I would like to specifically thank Dr. Sunghee Kim, who is my friend/mentor/family for support, encouragement throughout this process. I would not have accomplished this goal without her selfless support. I am very grateful and thankful for all of the support and patience my parents and brother have provided me throughout my education.

July, 2015

Table of Contents

Page
Number

Chapter 1: Introduction

Chapter 2: Background and Significance

Chapter 3: Real-time medical ultrasound imaging and interventional component framework

Chapter 4: Use Cases of Interventional Ultrasound Elastography

Chapter 6: Applications of Interventional Photoacoustic Imaging

Chapter 7: Spatial angular compounding of photoacoustic image

Chapter 8: Conclusions and Future work

Appendix: *OpenIGTLinkMUSiiC* tutorial

List of Figures, Tables and Snippets

Page
Number

Chapter 1: Introduction

Chapter 2: Background and Significance

Chapter 3: Real-time medical ultrasound imaging and interventional component framework

Chapter 4: Use Cases of Interventional Ultrasound Elastography

Chapter 5: Real-time photoacoustic imaging and interventional component framework

Chapter 6: Applications of Interventional Photoacoustic Imaging

Chapter 7: Spatial angular compounding of photoacoustic image

Chapter 8: Conclusions and Future work

Appendix: *OpenIGTLinkMUSiiC* tutorial

Chapter 1:

Introduction

1.1. Motivation

Image guided therapy (IGT) is one of the promising minimally invasive or non-invasive approaches to treat cancer or remove malignant tissue based on visual information from pre- or intra-operative medical imaging modalities such as medical ultrasound (US) imaging, computed tomography (CT), X-ray, and magnetic resonance imaging (MRI) [1]. In general, IGT procedures typically require many sources of synchronized information such as a surgical instrument's tracking information, a surgical robot's end-effector pose information, a region's temperature value from a thermocouple, and multiple medical imaging modalities to improve the localization and targeting of diseased tissue or surgical instruments and to monitor or control treatments. In addition, IGT systems generally include medical image processing functionalities such as image reconstruction, image analysis, segmentation, registration, and visualization [1].

The first clinical application of IGT was in a surgery to remove a sewing needle from a woman's hand with X-ray imaging, which was performed by a surgeon in Birmingham, England [2]. A month later, John Cox, a professor of physics at McGill University in Montreal, reported a success in removing a bullet from a victim's leg with radiography of the victim's limbs [3]. In 1908, Horsley and Clarke introduced a device for modern image-guided interventions, called the stereotactic frame, which allowed minimally invasive surgery by providing a 3D position of a small target inside the body [4]. After the introduction of CT scanning in the early 1970s and the personal computer

(PC) in 1981, a PC-based neurosurgery system for planning stereotactic procedures with CT, MRI, or digital subtraction angiography was reported by Peters et al. [5,6]. In 1986, David Roberts's lab at Dartmouth investigated a frameless stereotactic approach for neurosurgery with an operating microscope and a single dynamically updated tomographic image [7,8]. Galloway and his colleagues developed an articulated arm for neurosurgery with the capability of displaying a surgical position on 3D CT planes [9]. In a similar fashion, a number of IGT systems or applications based on CT, MRI, or US images were introduced and investigated in orthopedics [10-12]. Moreover, US image guided therapies were reported in various clinical applications such as injections, biopsies, ablations, cannulations and other surgical procedures [13-17]. As computational power and display performance of the PC improved, computer integrated surgery (CIS) systems and technologies were introduced to assist surgical interventions and to improve the surgical environment by combining engineering, robotics, and computer technologies [18].

Among the medical imaging modalities used with IGT systems, US imaging has attracted much recent attention due to its mobility, non-ionizing radiation, ease-of-use and real-time data acquisition. US imaging is a popular and convenient medical imaging modality, employed for diagnostic imaging or interventional guidance, and as either a standalone or complementary modality. In addition, conventional US imaging is frequently integrated with other medical imaging modalities such as pre-operative models (CT, MRI), tracking systems and robotic systems in an IGT system [19-21]. Furthermore, many research directions have sprung up around new medical ultrasound imaging modalities in the past few years including US elasticity imaging [22-24], US photoacoustic (PA) imaging [25-28], and US thermal imaging [29]. These new advanced ultrasound imaging modalities are also integrated with IGT systems [24, 30-33].

US imaging systems provide real-time 2D and 3D US Brightness Mode (B-mode) images which represent tissue-morphology by collecting and displaying the intensity information of reflected ultrasound acoustic waves leading to their main use in diagnosis and monitoring. In a B-mode image, the location of a medical instrument or implant can be tracked in real-time by using various ultrasound image segmentation algorithms. On the other hand, US elasticity images provide information about tissue stiffness and its Young's module [22, 23]. Since the abnormal tissue associated with tumors has a different stiffness compared to normal surrounding tissue, the area and boundary of the tumor can be recognized by using US elastography. In addition, PA imaging has been studied to identify tumor tissue and guide a surgical needle or implants [25, 28, 34-38]. In addition to the above advantages, US imaging systems represent a low cost, non-invasive technology to acquire deep penetration scanning inside the body. They can also provide good spatial and temporal localization and have shown compatibility with existing surgical instruments. These characteristics present US imaging as a system with quite a bit of potential in the clinical environment.

Several software frameworks and toolkits have been developed to integrate US data acquisition, processing and display with existing IGT systems [39-43]. Boctor et al. developed the Computer-integrated Surgery Guided by US imaging (CISUS) [39] toolkit to integrate US B-mode images with Slicer 2 [44] with 3D US reconstruction and US calibration functionalities. SynchroGrab [20] provides modular interfaces for the US machine and tracking devices to generate 3D US volume data. The public software library for ultrasound (PLUS) [42] is an open source toolkit for US guided intervention systems, which supports US calibration, data processing, visualization, and data transfer with 3D Slicer using OpenIGTLink [45] messages. Recently, the NifTK software platform was introduced [43], which is comprised of a small independent module for data collection (NiftyGuide), a main workstation program for IGT applications (NiftyIGI), and a messaging library based on OpenIGTLink [45] and the Qt software development toolkit (SDK) [46] to support communication between NiftyGuide and NiftyIGI. However, there is no open-source software framework or toolkit that supports advanced US imaging research and advanced US-guided therapy systems by providing low-level US data (channel data or radio-frequency (RF) data) essential for advanced US imaging.

In this research, we propose a new medical US imaging and interventional component framework for advanced US image guided therapy. **Figure 1.1** represents the concept of the proposed framework. As shown in the figure, all of the independent systems related to US image guided therapy are connected and can share their data or commands in our framework. In addition, this framework can provide a modular research environment by supporting communication interfaces between heterogeneous systems to allow for flexible ultrasound image guided therapy. Thus, we can easily reconfigure an entire US IGT system by adding or removing devices or equipment specific to each therapy. Moreover, we can easily implement and test new advanced ultrasound imaging techniques inside our framework in real-time because our software framework is designed and optimized for advanced ultrasound research.

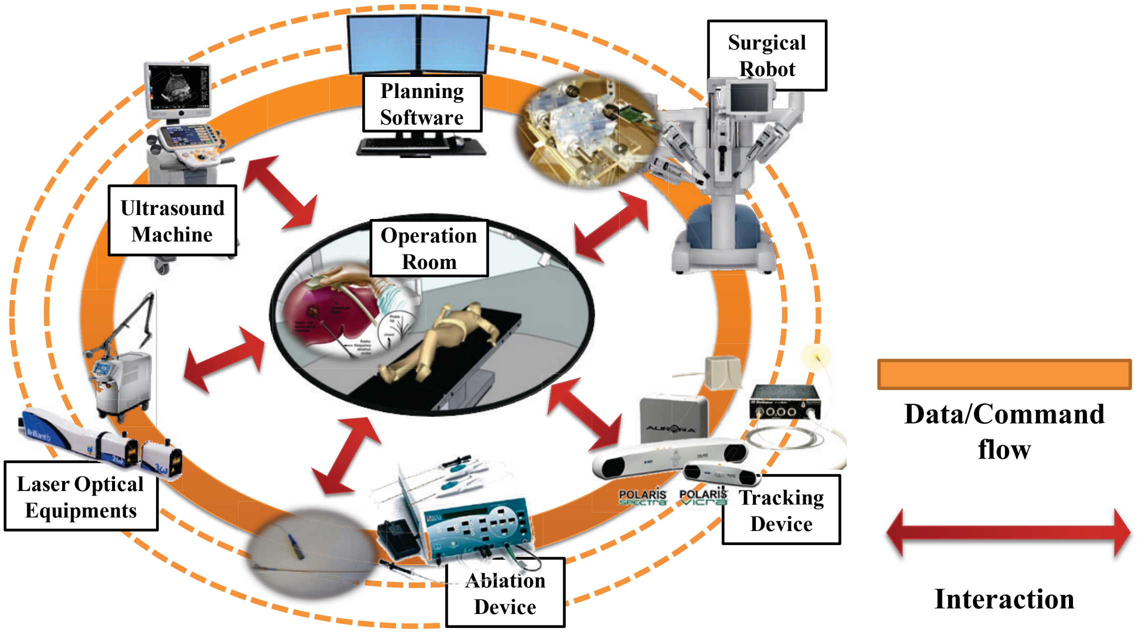


Figure 1.1. Concept of medical ultrasound imaging and interventional component framework for advanced ultrasound image-guided therapy.

1.2. Thesis statement

A software component framework of advanced ultrasound image-guided therapy with the properties of network-distributed modularity, real-time computation, and well-defined open-interfaces enables vendor independent advanced ultrasound imaging research, novel ultrasound interventional imaging research, and easy integration with existing image-guided therapy systems or components.

1.3. Contributions

This dissertation presents contributions in the area of software component frameworks for advanced medical ultrasound imaging research, advanced ultrasound image guided therapy applications, and a reconstruction algorithm for enhanced US imaging.

- Designing and building the *OpenIGTLinkMUSiiC* [47] library to provide the properties of network distributed modular architecture, real-time computation, and open-interface in our software framework. The design specification and software architecture are described in Chapter 3. In specific, main features of *OpenIGTLinkMUSiiC* include the following:
    - New *OpenIGTLink* compatible messages (*USMessage*, *EIMessage*, *GenMessage*, *ArgMessage*, and *FileMessage*) for interventional ultrasound imaging research providing the feature of open-interface to the software framework.
    - New concurrent data interface (*MUSiiCVector*) and a management class (*MUSiiCVectorSet*) of the concurrent data interface to support real-time thread-safe communication of other software multi-threaded tasks.
    - A customized callback function interface (*MUSiiCCallbackInterface*) with the compatibility of *OpenIGTLink*'s message and its management class (*MUSiiCCallbackInterfaceControl*) to support asynchronous communication inside an executable software module.
    - Customized task classes (*MUSiiCTaskObject*, *MUSiiCTaskAbstract*, and *MUSiiCTaskInterfaceAbstract*) with the compatibility of *OpenIGTLink*'s message and multi-threading techniques to support various patterns of real-time computation or multi-threaded task-network efficiently.

  - Two network classes (*MUSiiCTCPServer* and *MUSiiCTCPClient*) to establish a TCP/IP communication between software modules and broadcast data to multiple connected clients efficiently.

- Designing and developing *MUSiiC* modules based on the *OpenIGTLinkMUSiiC* library, accomplished by implementing the signal processing for US image reconstruction. *MUSiiC* modules include the following software modules: *MUSiiC-RFServer, MUSiiC-TrackerServer, MUSiiC-BMode, MUSiiC-Elasticity, MUSiiC-ImageViewer, MUSiiC-StreamWriter, and MUSiiC-FileLoader*. The detailed information of each software module is presented in Chapter 3.3. Connecting several MUSiiC modules can compose a US data processing pipeline. The US data processing pipeline based on the *MUSiiC* modules enables the following:
  - Vendor independent and customized US imaging researching by providing low-level and intermediate US data.
  - Easy reconfiguration of US imaging system for several specific US imaging and interventional US imaging research applications.
  - Easy integration with existing OpenIGTLink compatible IGT system.

- Designing and implementing *MUSiiC-Sync* module to receive multiple messages of OpenIGTLink and *OpenIGTLinkMUSiiC* from different data sources and to synchronize the received messages with their respective time-stamp. This software module enables us to combine multiple data, such as US image, tracking information or other sensor data in real-time for interventional US imaging research. The detail explanation of the synchronization mechanism and implementation are discussed in Chapter 3.2.4 and 3.3.7.

- Integrating a real-time GPU based US elastography system with an interventional US thermal ablation system (Acoustic MedSystems TeraVision^{TM}), studying real-time thermal monitoring for acoustic ablation therapy, and validating the performance through a series of *in vivo* experiments. The motivation and specific description of the system is presented in Chapter 4.1.

- Developing a spatially tracked US data acquisition system with *MUSiiC-RFServer*, *MUSiiC-TrackerServer* and *MUSiiC-Sync* modules. This system improves the quality of US elastograpy by implementing the method of tracked US elastography. The system setup and the results of a phantom test are described in Chapter 4.3.

- Developing real-time spatially tracked PA image acquisition system for an interventional PA surgical system with a real-time US channel data acquisition module (*MUSiiC-DAQ*), a spatial tracking software module (*MUSiiC-TrackerServer*) with electromagnetic tracking system, and *MUSiiC-Sync* software module to synchronize the acquired 2D PA image stream and the tracking data stream in real-time. This PA image collecting system enables 3D PA imaging and interventional PA imaging research. Detailed information regarding software modules and the system are presented in Chapter 5.

- Studying the localization of prostate brachytherapy seeds with a clinical ultrasound system and the PA effect, and demonstrating the feasibility of target localization with PA images. The system setup and results of a phantom test are discussed in Chapter 6.1.

- Proposing and inventing a novel needle detection method with a single PA image, a needle including optical fiber, and customized needle detection algorithm. The basic idea of the proposed needle detection approach and the results of a phantom test are presented in Chapter 6.2.

- Inventing and investigating three freehand spatial-angular compounding algorithms (averaging, weighted-averaging, and selective-averaging) utilizing a spatial registration technique and the contents information of the PA image, and a frame selection method of PA images to enhance certain image quality metrics: contrast, signal-to-noise ratio (SNR), contrast-to-noise ratio (CNR), and axial/lateral resolutions. The detailed description of each compounding method and the comparison of the compounded image's image quality metrics are discussed in Chapter 7.

1.4. Outline

In the next chapter, we discuss the background of US-guided therapy. First, US imaging modalities and their reconstruction methods are briefly described. Next, various software toolkits or frameworks for IGT systems are introduced. Finally, the significance of developing a real-time software component framework for advanced US research and advanced US-guided therapies is presented.

Chapter 3 introduces our proposed software component framework. This section can be divided into two parts: description of *OpenIGTLinkMUSiiC* [47] and *MUSiiC Notes/ MUSiiC* modules following the structure of this software framework. *OpenIGTLinkMUSiiC* is an extended version of OpenIGTLink [45], providing additional communication interfaces for network distributed modules, new message types for advanced US research, and abstract classes for real-time computation to *MUSiiC Notes/MUSiiC* modules. *MUSiiC Notes* is the collection of software components related to an US IGT system. Each sub-chapter includes a more detailed explanation of the corresponding software design and functionalities.

IGT applications using interventional US elastography based on our software framework are discussed in Chapter 4. First, real-time thermal monitoring of acoustic ablation therapy [41] is introduced. In this research, US thermal imaging based on the change in speed of sound according to a medium's temperature is implemented in our real-time thermal monitoring system [29, 48]. Our system is synchronized with an acoustic ablation system by a transistor-transistor logic (TTL) signal to collect pre-/intra-ablation US RF data efficiently [41]. Second, a robotic system for real-time laparoscopic US elastography [30] is presented. Our real-time US stain imaging system is integrated with a daVinci surgical robotic system to generate enhanced US elastrography images consistently and provide elasticity information of the target material to the surgical robotic system in real-time. Finally, a real-time tracked US elastography system is proposed to consistently generate high quality US elastograms. The system utilizes a spatially tracked US RF data stream to find proper pairs of RF frames, and a graphics processing unit (GPU) architecture to improve computational performance.

In Chapter 5, the extended software framework with additional *MUSiiC* modules for PA imaging research [49, 50] and its interventional applications is introduced. *MUSiiC-DAQServer* is developed to collect US channel data including PA acoustic waves in real-time. A beamforming algorithm (Delay-and-Sum) for reconstructing PA images was implemented in a *MUSiiC-Beamformer* module. Also, the *MUSiiC-OPO* module was developed with the provided application program interface (API) from the manufacturer to control the laser beam's wavelength and save the received US channel data from *MUSiiC-DAQServer* to a local hard disk. In addition, a spatially-tracked pre-beamformed RF data acquisition system with a freehand clinical US transducer is introduced for 3D PA imaging research, spatial compounding of PA images, and interventional PA surgical systems.

Similar to Chapter 4, IGT applications of interventional PA imaging research are presented with results of their experimental tests in Chapter 6. First, a real-time US pre-beamformed RF data acquisition system for brachytherapy seeds visualization [50] is discussed. Generally, localization of clinical metal implants such as brachytherapy seeds in a clinical US B-mode image is difficult because of US artifacts caused by mismatch of acoustic impedance of the material of the implants with surrounding background material [25, 26]. Therefore, we proposed a real-time PA imaging system and demonstrated PA imaging of brachytherapy seeds with the proposed system. Second, a new technique for needle visualization using the PA effect is introduced. A clinical biopsy needle similar to a brachytherapy seed generates noise artifacts on a US B-mode image [51].

Consequently, real-time tracking of a needle is not easy in an operation room. To address this limitation, we proposed a new method based on a single PA image and an image-processing pipeline in this research. In the end, an image-guided assistance system for PA imaging is suggested. The system consists of spatial-tracking and an US imaging system, and a PA navigation software module. The software module provides visual guidance to align the laser source, the US probe, and the target object to collect a proper PA signal. In this study, we investigated the accuracy of the system with a phantom experiment.

Chapter 7 presents a study on spatial-angular compounding for PA imaging to improve its image quality. A frame selection algorithm based on pose information of each PA image to find proper compounding PA images and three spatial-angular compounding algorithms are presented. The averaging compounding method is very similar to conventional compounding algorithms for US B-mode and is only based on the spatial transformation between multiple images. The new proposed compounding algorithms (weighted- and selective-compounding) rely on both the spatial transformation and the PA image content to enhance the image quality and reduce the spatial tracking error. Finally, a comparison and a discussion about the relative image qualities of the compounded PA images from these three methods are presented.

The last chapter concludes this thesis, providing future directions of the proposed software framework and US IGT applications. There are numerous necessary functions that our software framework must support as more advanced US modalities become available and various IGT systems for other applications are developed.

Chapter 2:

Background and Significance

2.1. Medical ultrasound image

2.1.1. B-mode ultrasound

Ultrasound (US) brightness mode (B-mode) images are one type of diagnostic medical image provided by a medical ultrasound system. They represent tissue morphology by collecting and displaying the intensity information of reflected ultrasound waves in real-time [52]. **Figure 2.1** illustrates the data flowchart for generating ultrasound B-mode images. The process can be divided into three main steps: radio-frequency (RF) data acquisition, envelope detection, and compression and scan conversion.

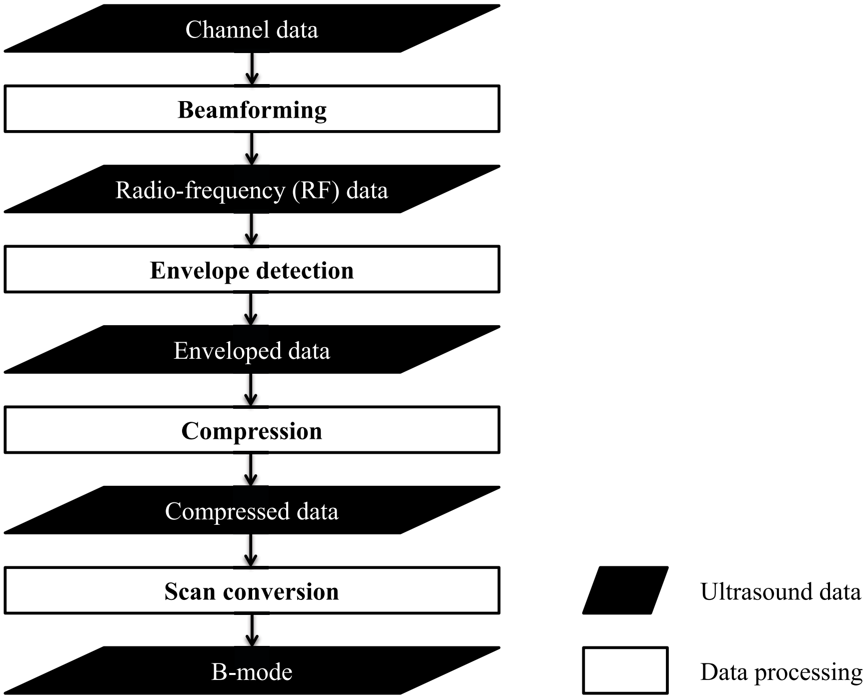
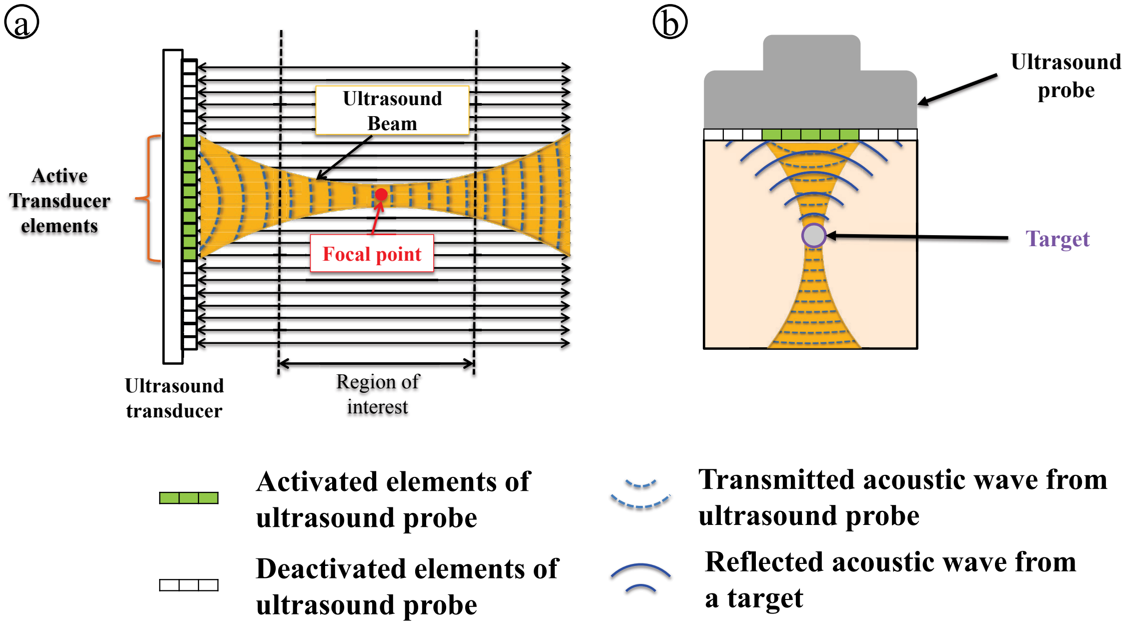


Figure 2.1. Flowchart for ultrasound B-mode image reconstruction.

2.1.1.1 RF-data acquisition

Figure 2.2 shows the basic concepts of an ultrasound system. One of the fundamental requirements of an ultrasound system is to make a narrower US beam for better US B-mode image resolution [52, 53]. In general, this is accomplished by combining multiple acoustic waves emitted from multiple neighboring US transducer elements with a set of time-delays to focus at a particular focal point (see **Figure 2.2** (**a**)). As the US waves move toward the focal point, the US waves are reflected by any objects in their path. The amplitude of the reflected acoustic waves is measured by several US transducer elements (receiving aperture) and recorded as US channel data.



a
Active Transducer elements
Ultrasound Beam
Focal point
Ultrasound transducer
Region of interest
b
Ultrasound probe
Target
Activated elements of ultrasound probe
Deactivated elements of ultrasound probe
Transmitted acoustic wave from ultrasound probe
Reflected acoustic wave from a target

Figure 2.2. Concept diagram of generating ultrasound B-mode images.

Figure 2.3 shows the block diagram of ultrasound system for getting an RF-data [52]. "Beam-former controller unit" creates a narrow US beam using a set of time-delayed acoustic waves with the US transmit parameters presented in **Table 2.1**. In the receiving mode (see Table 2.2), recorded channel data is composed with the "Beam-former controller unit", and regulated with "Digital gain" to reduce any effect of US attenuation. RF A-line data is sequentially generated by the "Accumulator" [54] process.

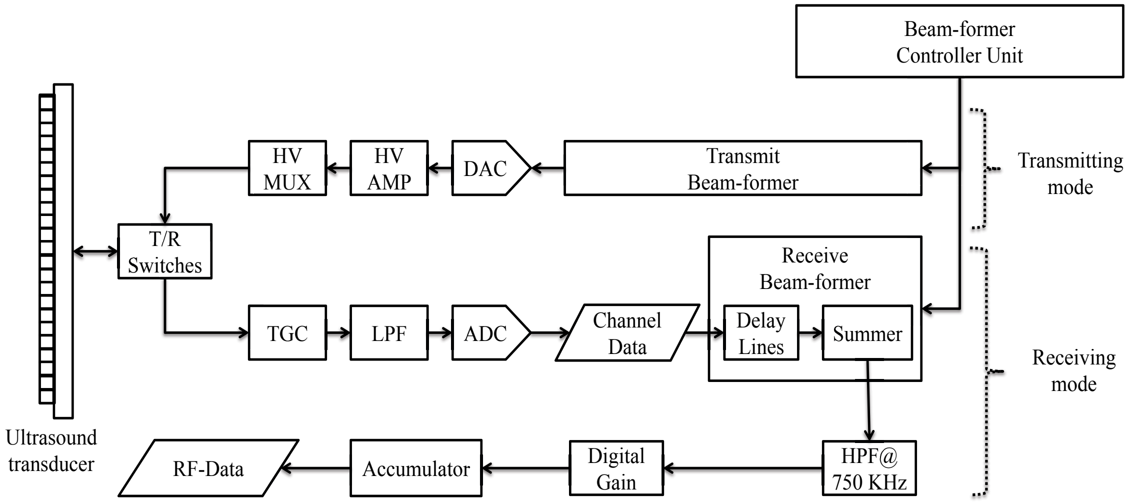


Figure 2.3. Flowchart for ultrasound B-mode image reconstruction. (Ref. **[54]**)

Table 2.1. Transmit (Tx) parameters of Ultrasound system.

Parameter	***Description***
Tx Focus Depth (mm)	The depth of the focal point.
Tx Aperture (element)	The aperture size of transmitting ultrasound pulse in ultrasound transducer elements.
Tx Style	The pulse shape that is applied for RF-data.
Tx Frequency (MHz)	The transmit frequency that is output to the ultrasound transducer for RF-data.
Tx Angle (degree)	The angle of transmitting ultrasound pulses for beam-steering RF-data.

Table 2.2. Receiving (Rx) parameters of Ultrasound system.

Parameter	***Description***
Rx Depth (mm)	The depth of RF-data in millimeter units.
Rx Aperture (element)	The aperture size of receiving reflected ultrasound waves in ultrasound transducer elements.
Rx Save Delay (mm)	The depth of delay in the acquisition RF-data with millimeter unit.
Rx Max Aperture (mm)	The maximum half aperture that the receive profile curve will fit to.
Rx Angle (degree)	The angle of receiving reflected ultrasound waves for beam-steering RF-data.
Gain (%)	The general overall gain. Maximum value is 100 %.

2.1.1.2 Envelope detection

In general, clinical US B-mode images represent tissue morphology by collecting and displaying the intensity of the reflected ultrasound wave. The intensity information is computed by the envelope detection of the received RF-signal. The envelope signal is computed as the magnitude of the analytic signal, which is a complex signal extracted from a received RF-data. There are two methods for computing an analytic signal from a reflected RF signal. One method uses the Hilbert transform [52, 55]. Although this operation is independent of the actual frequency of the echo RF signal and imaging modes of medical ultrasound imaging, its implementation is complex. Another method uses a complex rotator to demodulate the signal in its baseband. The second method is simpler than the previous method, but the center frequency of the reflected RF signal is required. In this project, an envelope signal is created by the first method, using the Hilbert transformation for independent B-Mode generation.

An analytic signal ($f_A(x)$) of real signal ($f(x)$) can be defined as shown in equation (2.1).

$$f_A(x) = f(x) - i\, f_H(x) \tag{2.1}$$

In equation (2.1), $f_H(x)$ is the Hilbert transform of the real signal and this is defined by equation (2.2):

$$f_H(x) = \frac{1}{\pi} \int_{-\infty}^{\infty} \frac{f(x')}{x' - x} dx' \tag{2.2}$$

To get the analytic signal from the real signal, we transformed equation (2.1) and (2.2) into Fourier Space using a Fourier Transform as follows.

$$F_H(\gamma) = F(\gamma) \cdot i\, \mathrm{sgn}(\gamma) \tag{2.3}$$

$$F_A(\gamma) = F(\gamma) - i\, F_H(\gamma) \tag{2.4}$$

Where, $F(\gamma)$ is the Fourier transformed of $f(x)$. From equation (2.3) and (2.4), we can get the analytic signal in Fourier Space ($F_A(\gamma)$) as shown in equation (2.5):

$$\begin{aligned} F_A(\gamma) &= F(\gamma) - i\, F_H(\gamma) \\ &= F(\gamma) - i\, F(\gamma) \cdot i\, \mathrm{sgn}(\gamma) \\ &= F(\gamma) \cdot [1 + \mathrm{sgn}(\gamma)] \end{aligned} \tag{2.5}$$

Where, $\mathrm{sgn}(\gamma)$ is the sign function that extracts the sign of a signal. To get the analytic signal, we apply an inverse Fourier Transform of equation (2.5). As a result, the local phase ($\phi(x)$) and the envelope signal (the local energy), A(x), of the real signal is expressed by the following equations.

$$\phi(x) = \tan^{-1}(f(x)/f_H(x)) \tag{2.6}$$

$$A(x) = \sqrt{(f^2(x) + f_H^2(x))} \tag{2.7}$$

In summary, an analytic signal is computed by calculating the Fourier transform of the RF-signal, multiplying the first half by two, multiplying second half by zero, and then taking the inverse Fourier transform. After this, an Envelope signal is computed by calculating the magnitude of the analytic signal (see equation (2.7)). **Figure 2.4** represents a received RF-signal and its corresponding enveloped signal by the envelope detection operator.

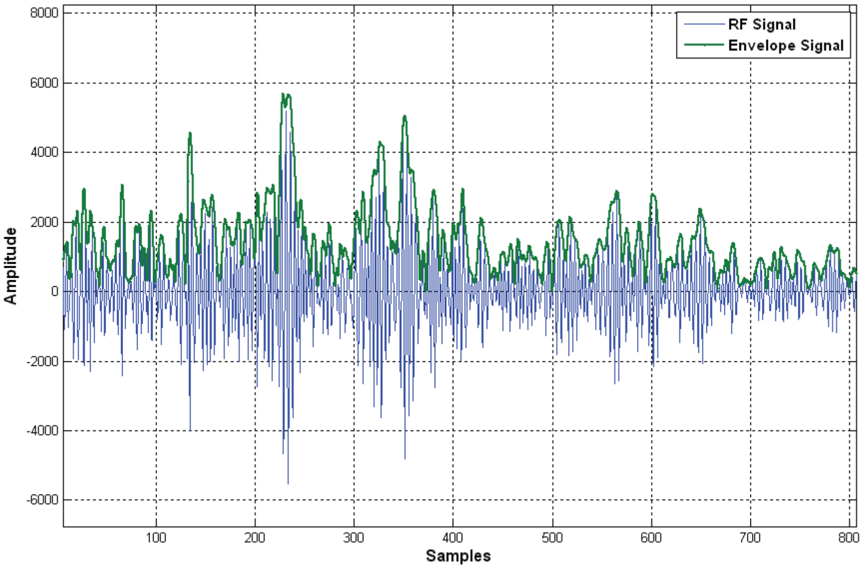


Figure 2.4. RF-signal and an envelope-detected data.

2.1.1.3 Compression and Digital scan conversion

In this section, envelope data compression and B-mode image scan conversion are presented. Since the dynamic range of an envelope-data (generally 12 bit) is still too large to display, the data should be compressed to a compatible range (8bit). Equation (2.8) represents a well-known a logarithmic-compression method for compressing envelope data [54].

$$\boldsymbol{B}[\boldsymbol{i}] = \frac{exp\ (\boldsymbol{\alpha}\times \log(\boldsymbol{Env}[\boldsymbol{i}]))}{exp\ (\boldsymbol{\alpha}\times \log(65535))/255} \tag{2.8}$$

Where ***i*** is the index of the data array, ***Env[]*** and ***B[]*** represents the data buffer for envelope-data and the B-mode image, respectively, and $\boldsymbol{\alpha}$ is a compression coefficient. When $\boldsymbol{\alpha}$ is 0.5, this compression operation becomes a square root compression.

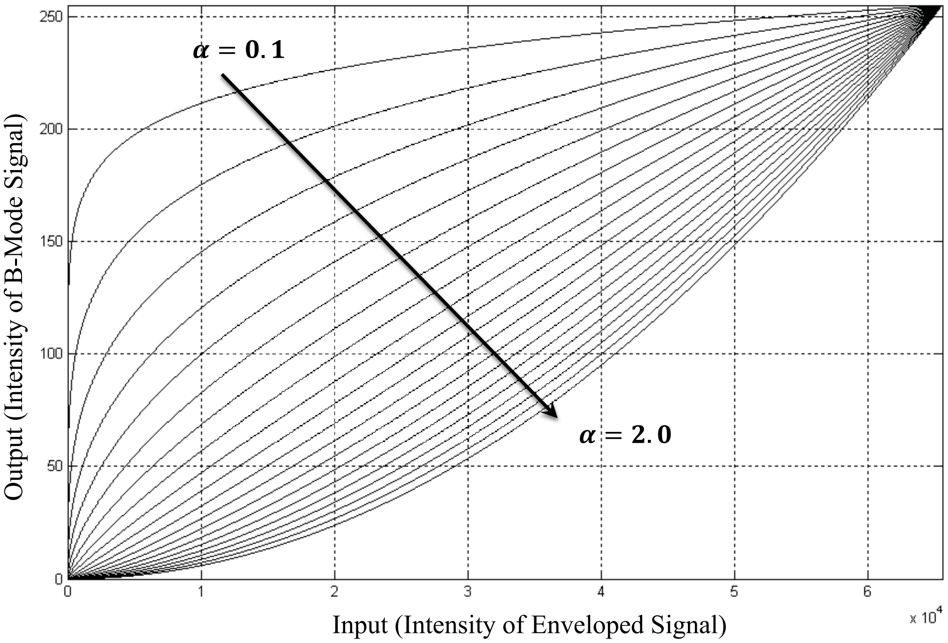
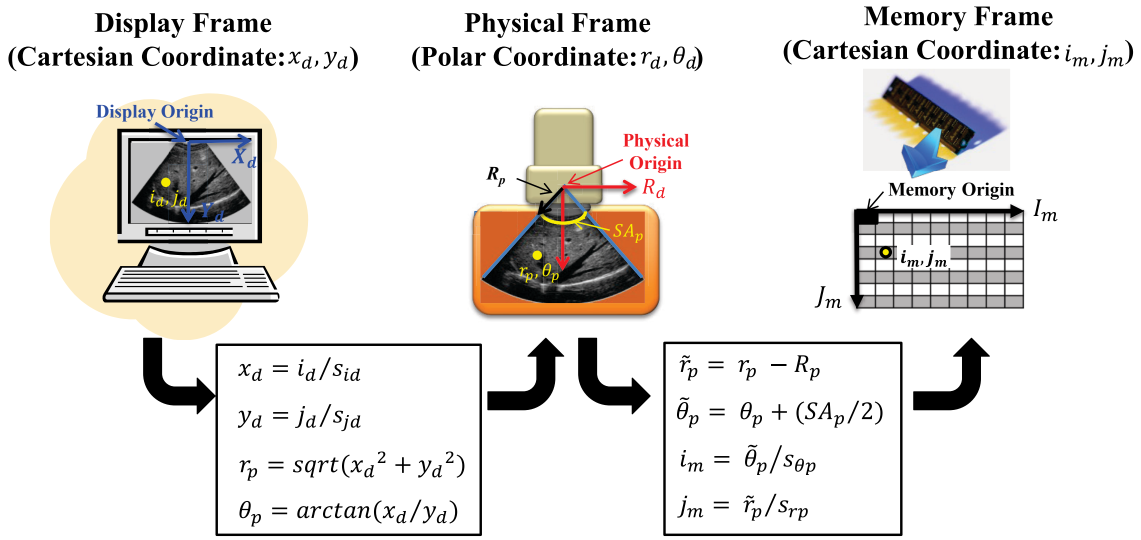


Figure 2.5. RF-signal and an envelope-detected data.

Figure 2.5 shows the results of this logarithmic-compression operation with different values of $\boldsymbol{\alpha}$. The dynamic range of a B-mode image can be controlled by the value of $\boldsymbol{\alpha}$**.**

Digital scan conversion in US B-mode imaging is the conversion from the coordinates of the stored acoustic signal in memory (memory frame) to the coordinates of an US scanner (physical frame) and then to the coordinates of display windows (display frame) to display the retrieved digital visual information (US B-mode image) on screen [56]. **Figure 2.6** represents the concept of a backward digital scan conversion algorithm for a curvilinear US probe.



$$x_d = i_d/s_{id}$$
$$y_d = j_d/s_{jd}$$
$$r_p = sqrt({x_d}^2 + {y_d}^2)$$
$$\theta_p = arctan(x_d/y_d)$$

$$\tilde{r}_p = r_p - R_p$$
$$\tilde{\theta}_p = \theta_p + (SA_p/2)$$
$$i_m = \tilde{\theta}_p/s_{\theta p}$$
$$j_m = \tilde{r}_p/s_{rp}$$

Figure 2.6. Diagram of the sequence of a backward digital scan conversion for a curved US probe.

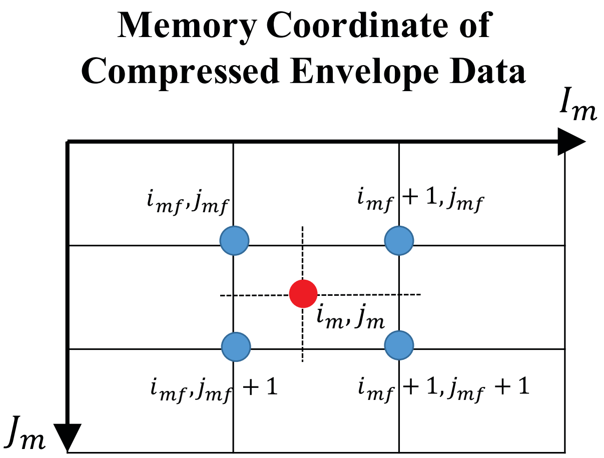
The display and memory frame can be defined in Cartesian coordinates, and the physical coordinate of the curvilinear US probe is defined in Polar coordinates. Indices i_d and j_d are the pixel locations in a US B-mode image, and x_d and y_d are the position of the point in the display frame with units of mm. r_p and θ_p represents the location of the considered point in polar coordinates (physical frame) with units of mm and degree (°), respectively. The converted r_p and θ_p are updated to $\tilde{r}_p$ and $\tilde{\theta}_p$ for computing the memory indices (i_m and j_m) that represent the position of the point on the memory frame with scale values ($s_{\theta p}$ and s_{rp}). These scale values are computed by the following equations.

$$s_{\theta p} = SA_p/Pitch_p \tag{2.8}$$

$$s_{rp} = SoS/f_{sp} \tag{2.9}$$

Where, SA_p and $Pitch_p$ represent the scan angle and the transducer element's pitch, respectively. SoS and f_{sp} indicate the background material's speed of sound and the sampling frequency of the US machine.

Although several interpolation algorithms (e.g. nearest neighbor, linear interpolation, and bilinear interpolation) are suggested for US digital scan conversion, bilinear interpolation (see **Figure 2.7**) is most often implemented in US systems after considering the computation speed and the quality of the B-mode image [57].



$$i_{mf} = floor(i_m)$$
$$j_{mf} = floor(j_m)$$
$$\Delta i_m = i_m - i_{mf}$$
$$\Delta j_m = j_m - j_{mf}$$

$$CEnv1 = CEnv[i_{mf}][j_{mf}]$$
$$CEnv2 = CEnv[i_{mf} + 1][j_{mf}]$$
$$CEnv3 = CEnv[i_{mf}][j_{mf} + 1]$$
$$CEnv4 = CEnv[i_{mf} + 1][j_{mf} + 1]$$

Figure 2.7. Bilinear interpolation for digital scan conversion.

Through the conversions between frames described earlier, there is some memory location (i_m, j_m) for every display location (i_d, j_d). i_{mf} and j_{mf} are the floor values of a desired memory location (i_m, j_m), and those differences are expressed as Δi_m and Δj_m, respectively. *CEnv* represents the memory frame of a compressed envelope-detected data, and the intensities of the four surrounding corners (see **Figure 2.7**) are expressed as *CEnv1*, *CEnv2*, *CEnv3*, and *CEnv4*, respectively. The pixel value at the position (i_d, j_d) will correspond with the intensity value at the memory location (i_m, j_m) as computed with the following equation.

$$\begin{aligned} Pixel(i_d, j_d) = CEnv(i_m, j_m) &= (1-\Delta j_m)\times\left[(1-\Delta i_m)\times CEnv1 + \Delta i_m \times CEnv2\right] \\ &+ \Delta j_m \times\left[(1-\Delta i_m)\times CEnv3 + \Delta i_m \times CEnv4\right] \end{aligned} \tag{2.10}$$

2.1.2. Ultrasound elastography

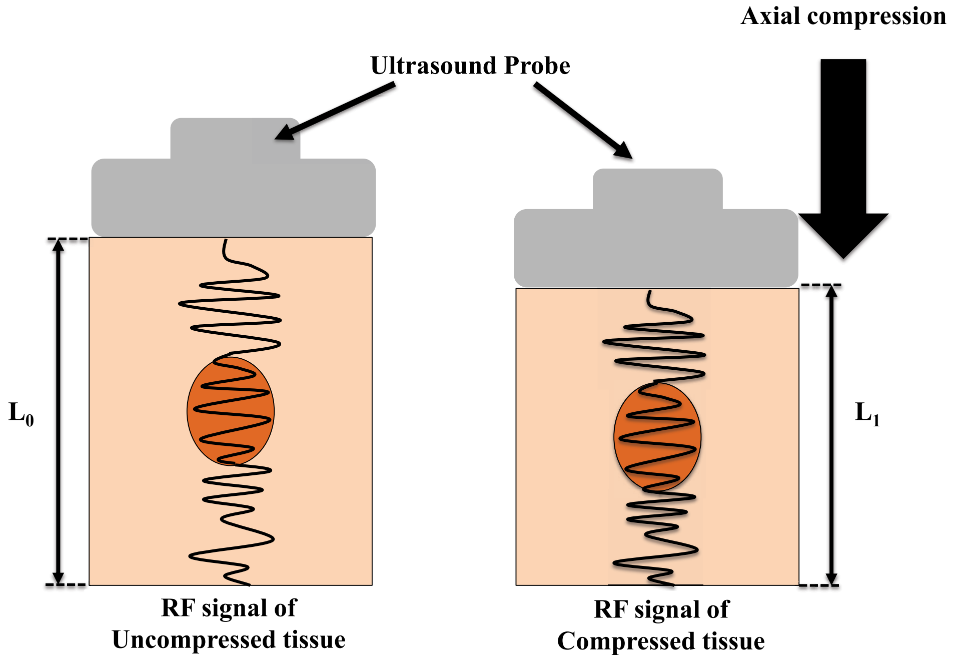


Figure 2.8. Principle of quasi-static US elastography.

The basic idea of US elastography is to estimate internal displacement of tissue caused by internal or external force. Many US elastography imaging methods have been developed since the 1990s [58-61]. Quasi-static elastography imaging is one of the first methods suggested by Ophir *et al.* [58]. In this method, US elastography is generated by the comparison between an echo RF signal with and without external force compression, as shown in **Figure 2.8**. The displacement between these two RF signals is calculated by the correlation between these two signals. The strain of the target can then be computed with the following equation under an assumption of uniform stress.

$$E = \frac{\sigma}{\varepsilon} \tag{2.11}$$

Where E is the Young's modulus, also known as elastic modulus, which represents the stiffness of a tissue, and σ and ε are tensile stress and strain, respectively. Therefore, we can estimate the following tissue properties using US elastography: displacement, strain, Young's module, and acoustic wave propagation speed [62]. **Figure 2.9** shows a flowchart for the reconstruction of an ultrasound quasi-static elastography image.

Normalized cross correlation (NCC) (see equation 2.12) is used for building a correlation map between two RF frames (with and without compression) by measuring speckle shifts [48].

$$\gamma(u,v) = \frac{\sum_{x,y}\left[f(x,y) - \bar{f}_{u,v}\right]\left[t(x-u,y-v) - \bar{t}\right]}{\left\{\sum_{x,y}\left[f(x,y) - \bar{f}_{u,v}\right]^2 \sum_{x,y}\left[t(x-u,y-v) - \bar{t}\right]^2\right\}^{0.5}} \tag{2.12}$$

Where, f is a template window in the RF frame without compression, and t is a target window in the RF frame with compression. $\bar{f}$ and $\bar{t}$ represent the mean value of f and t, respectively. x, y, v, and u represent pixel indices within the windowed region. A displacement map is generated from the correlation map by sub-sampling with cosine fit interpolation. Then, moving average and median filters are applied on the displacement map to remove outliers [48, 63]. A strain map is estimated with least squares strain estimation [48]. Finally, a quasi-static US elastography image generated by applying the compression and scan conversion operations on the strain map can be displayed on screen.

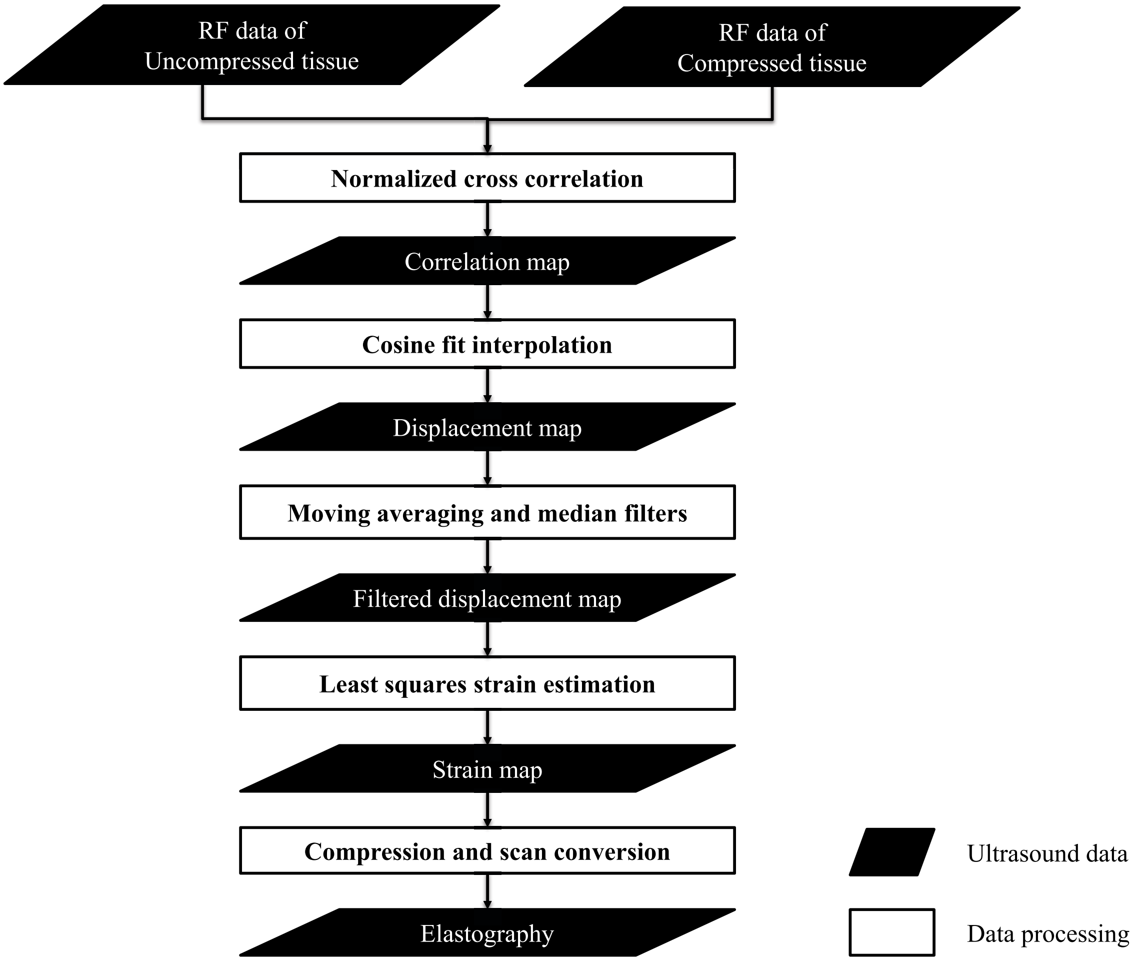

RF data of Uncompressed tissue
RF data of Compressed tissue
Normalized cross correlation
Correlation map
Cosine fit interpolation
Displacement map
Moving averaging and median filters
Filtered displacement map
Least squares strain estimation
Strain map
Compression and scan conversion
Elastography
Ultrasound data
Data processing

Figure 2.9. Flowchart for ultrasound elastography reconstruction.

2.1.3. Photoacoustic imaging

Photoacoustic imaging (PAI) is a hybrid medical imaging modality that exploits the properties of light and ultrasound by scanning the optical absorption spectrum of some material or medium. PAI is based on the photoacoustic (PA) effect, which is the conversion of light waves to acoustic waves through optical absorption, localized thermal excitation, and localized pressure transient of tissue [34, 35, 64]. In addition, different material types and different concentration ratios of a specific material have different optical absorption coefficient values dependent on the laser source wavelength as shown in **Figure 2.10**. Therefore, PAI has been researched in biomedical imaging areas for various structural and functional imaging applications [35, 64].

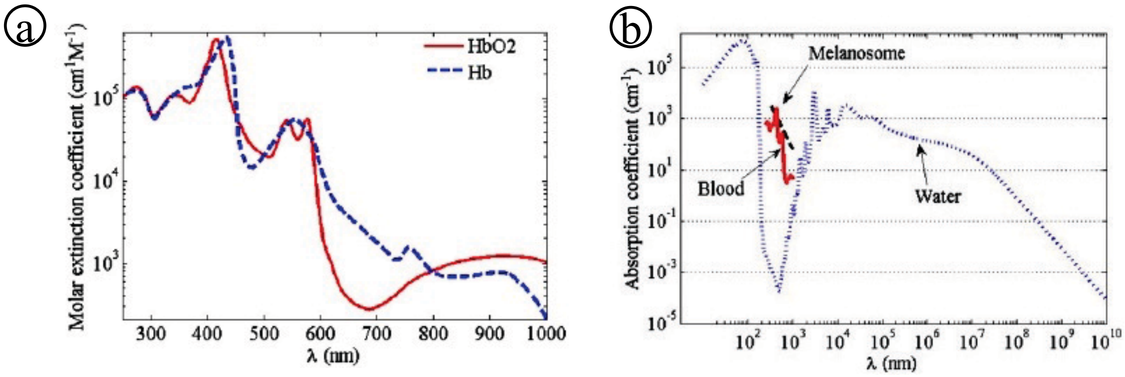


Figure 2.10. (a) Absorption spectra of oxyhemoglobin (HbO2) and deoxyhemoglobin (Hb). (b) Absorption coefficients of blood, melanosomes and water. (Ref. **[34])**

Figure 2.11 shows the difference between US B-mode imaging and PAI. Generally, US B-mode images are generated by receiving echo acoustic waves from a target with limited US transmit elements (see **Figure 2.11** (**a**)). In PAI technology, acoustic waves are generated simultaneously from the regions in the irradiated tissue volume that passes the PA generation threshold. To analyze the distribution of optical absorption in tissue using acoustic waves generated from the PA effect, the acoustic waves should be recorded on every element of an ultrasound transducer array simultaneously. US channel data (pre-beamformed RF data) acquired by all the transducer's elements is necessary to create a PA image.

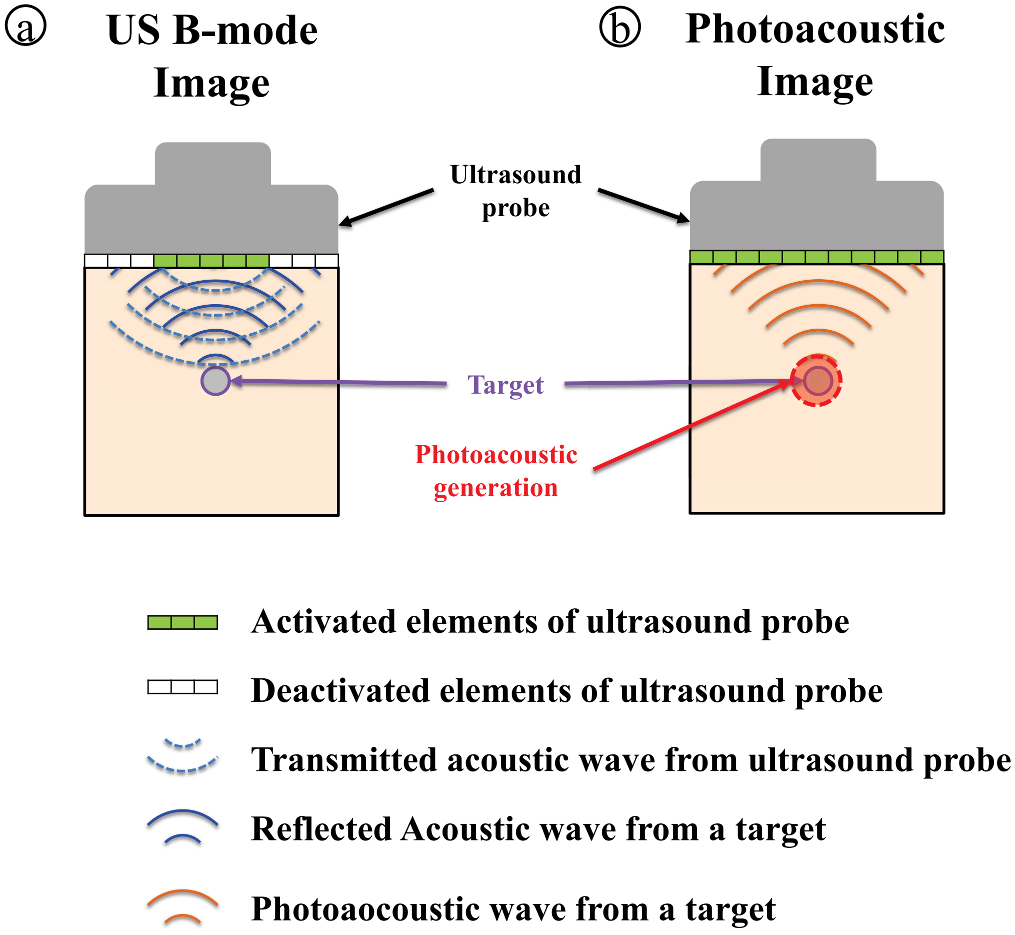

ⓐ
US B-mode Image
ⓑ
Photoacoustic Image
Ultrasound probe
Target
Photoacoustic generation
Activated elements of ultrasound probe
Deactivated elements of ultrasound probe
Transmitted acoustic wave from ultrasound probe
Reflected Acoustic wave from a target
Photoaocoustic wave from a target

Figure 2.11. Concept diagram showing the difference between US B-mode imaging and photoacoustic imaging.

The flowchart for PA image reconstruction is presented in **Figure 2.12.** The sequence is very similar to US B-mode image reconstruction. However, the time-of-flight delay in the beamformer is not the same between the US B-mode and PA images as shown in the **Figure 2.11**. Therefore, we need to apply a different beamforming algorithm to generate PA images from captured PA waves. However, the remaining processing components (envelope detection, compression, and scan conversion) are identical.

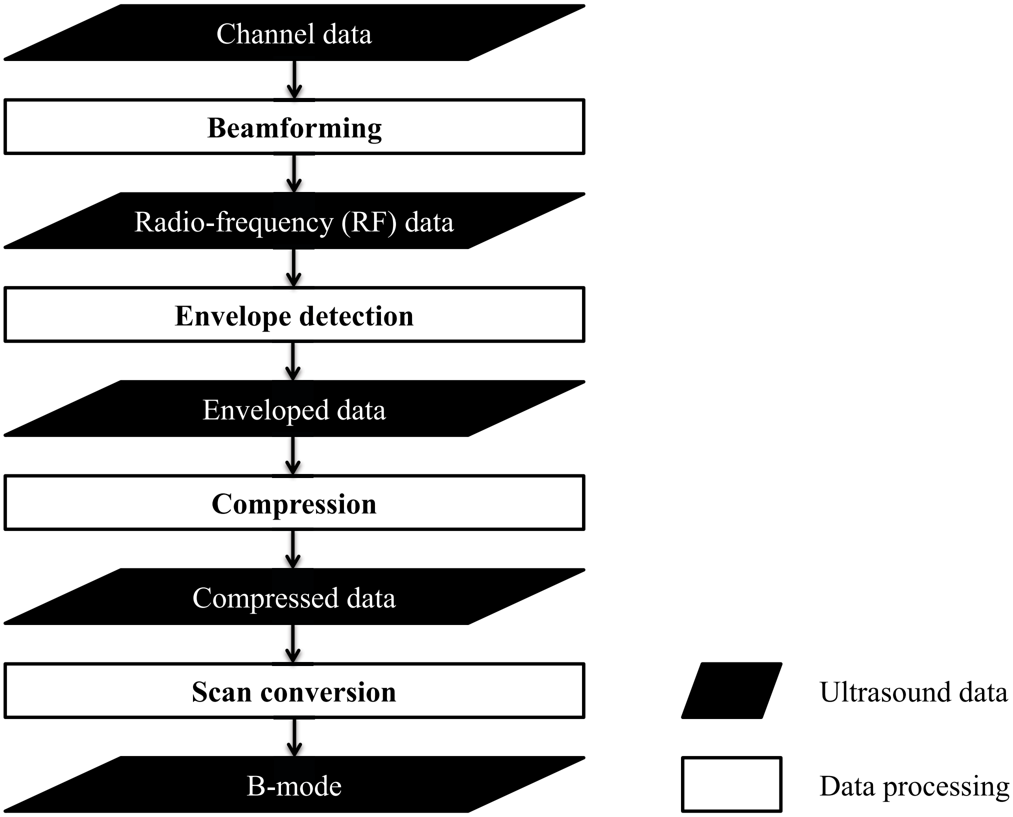


Figure 2.12. Flowchart for photoacoustic image reconstruction.

Several groups have performed research on beamformers to enhance PA image quality [34, 65-71]. In this research, we used delay-and-sum (DAS) [34] and short-lag-spatial coherence (SLSC) [67, 68] beamforming algorithms. These two methods are based on time-delayed US channel data as illustrated in **Figure 2.13**.

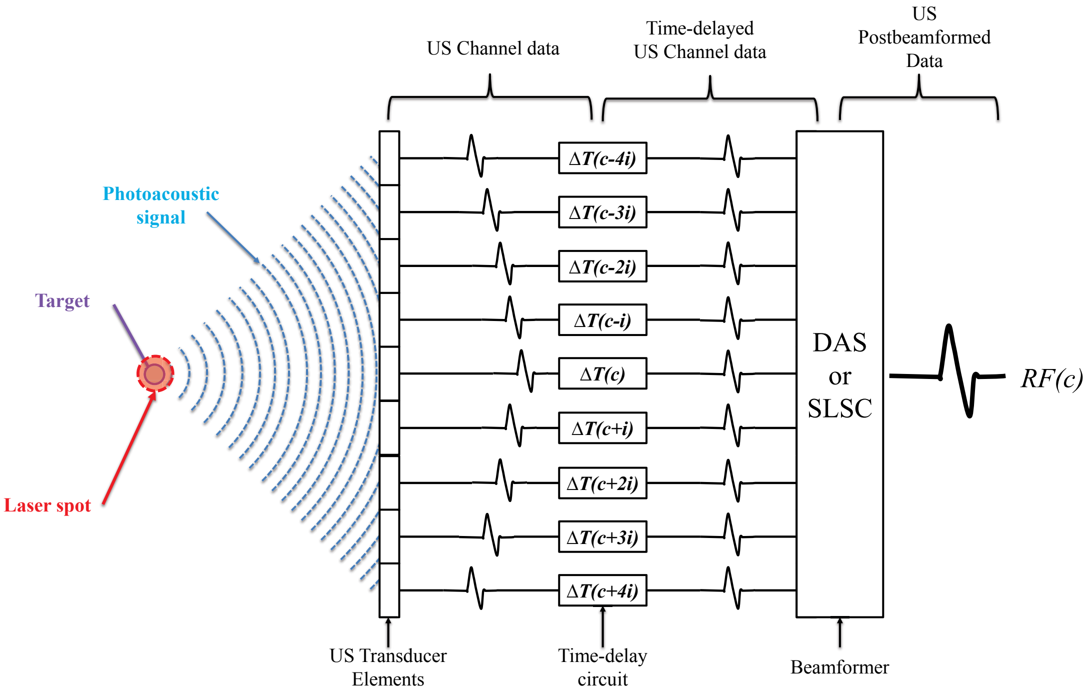


Figure 2.13. Concept diagram of time-delays for delay-and-sum (DAS) and short-lag spatial coherence (SLSC) beamforming algorithm.

DAS beamforming is the most conventional and popular beamforming method that is implemented in most US machines to generate US B-mode images. However, DAS for PA considers one-way acoustic travel time instead of two-way travel to calculate a time-delay for each channel [67, 69]. Then, DAS generates post-beamformed RF data ($S_{DAS}(c)$) at the *c-th* element that is the center of a receiving aperture by synthesizing the time-delayed US channel data (s_i) with hanning apodization weight factors (w_i) and a half size of receiving aperture size (n) as in the following equation.

$$S_{DAS}(c) = \sum_{i=c-n}^{c+n} w_i s_i \tag{2.13}$$

On the other hand, SLSC [67, 68] is based on spatial coherence in the short-lag region of an acquired US channel frame, and can be expressed by the following equation.

$$\hat{R}(m) = \frac{1}{N-m} \sum_{i=\left(c+\frac{N}{2}\right)-m}^{\left(c+\frac{N}{2}\right)+m} \frac{\sum_{d=d_1}^{d_2} s_i(d) s_{i+m}(d)}{\sqrt{\sum_{d=d_1}^{d_2} {s_i}^2(d) \sum_{d=d_1}^{d_2} {s_{i+m}}^2(d)}} \tag{2.14}$$

$$S_{slsc}\,(c) = \sum_{m=1}^{M} \hat{R}(m) \tag{2.15}$$

Where, $\hat{R}$ is the normalized spatial correlation of the time-delayed signal, m is the distance of short-lag between two elements, and N is the size of receive aperture that is defined by users. s_i is the intensity of the time-delayed and zero-mean signal that is received by the *i-th* element. d is the sample depth, and M is the number of lags included in the short-lag sum. Finally, $S_{slsc}\,(c)$ represents the short-lag spatial coherence [67, 68].

2.2. Software for image guided therapy (IGT)

A considerable amount of toolkits, frameworks, and components populate the field of image-guided surgical/therapy systems research and development. Apart from the low-level vendor application program interfaces (API) and related research toolkits, several open-source projects are reported.

- OpenIGTLink:

One open-source toolkit aimed at modular abstraction in image-guided therapy (IGT) in the widest sense deals with the problem at the networking level: OpenIGTLink has been suggested by the NAMIC group for providing a standardized TCP/IP network mechanism for IGT systems [45]. This protocol enables the communication of position, transformation, and image data, as well as commands or system status messages between multiple processes in a distributed IGT system. Due to its advantages of being open-source, simple, operating system agnostic, and ready-to-use, several research groups and commercial companies have used this toolkit in their environments [20, 41, 47, 50, 72, 73]. Recently, OpenIGTLink has received upgrades with the addition of new message types, supporting several IGT-specific data types: tracking-data, image-meta data, fiducial points, trajectory data, etc. [74]. However, there are still no message types for containing general ultrasound data or results of advanced ultrasound imaging modalities such as US elastograpy or PA imaging on even the newest version of OpenIGTLink [45, 74].

- The Image-Guided Surgery Toolkit (IGSTK):

The Image-Guided Surgery Toolkit (IGSTK) [40] recognized the overhead involved in re-inventing Image-Guided Surgery (IGS)-related functionality and provided extensive support for hardware-independent tracking, volume representations, file I/O (e.g. DICOM), networking (with OpenIGTLink [45]) registration, calibration, Graphic-User-interface (GUI) building, and many others. However, it does not support real-time US imaging.

- CISST:

One development that had a similar goal as IGSTK was the CISST libraries [75, 76]. It incorporated support for robotics, algebra, operating system abstractions, vision, and intra-operative imaging. However, these libraries did not have any specific functions related to ultrasound imaging because they have been focused on the development of computer assisted intervention systems.

- The Robot operating system (ROS):

  ROS [77, 78] is an open source infrastructure toolkit for developing robotic applications by providing hardware abstraction, low-level device control, libraries, and visualization. Moreover, ROS supports a structured communication method (a publish/subscribe communication mechanism) for distributed inter-process/inter-machine communication and configuration. Therefore, many research and commercial groups are using this framework for developing their robotic applications [78]. Although this software framework is well-known and popular to robotics developers, there are no modules which can manipulate US data such as US channel or RF data.

- Computer-integrated Surgery Guided by US Imaging (CISUS):

  'Computer-integrated Surgery Guided by US Imaging' (CISUS) developed by Boctor et al. [39] was a software module that integrated with Slicer 2 [44]. This module was designed to reconstruct 3D US volume data from US images with tracking information, and visualize and/or process the 3D volume in Slicer 2 for diagnostic visualization and surgical planning. This software could directly control a robotic system for image-guided needle placement with the functionality of its own automatic US calibration [79]. Since Slicer 2 was an open-source software package available on multiple operating systems for medical image visualization and processing, CISUS was a promising approach for US based computer-integrated surgical systems. However, CISUS is no longer available because it was dependent on the architecture of Slicer 2 and the Slicer's architecture has changed during its major revisions [44, 80].

- SynchroGrab:

SynchroGrab [20] encapsulated the low-level interfacing with US machines and tracking devices to provide 3D US functionality. The generated volume data was handed over via OpenIGTLink [45, 74] to compatible clients, e.g., Slicer [44, 80] for visualization and intervention guidance. Due to the modular architecture of this software, it could be easily extended to support other ultrasound systems or tracking devices. However, SynchroGrab did not provide any functionality for advanced ultrasound data processing such as ultrasound elastograpy even though it supported a well-designed interface for ultrasound image-guided surgical systems.

- Public software Library for Ultrasound Imaging research (PLUS):

PLUS [42] is an open source toolkit for rapid development of US-guided intervention systems. This toolkit provides access to various US imaging and tool tracking systems with the functionalities for US data processing, visualization, data transfer and US calibration. In addition, PLUS uses OpenIGTLink [45, 74] messages to communicate with 3D Slicer [44, 80]. However, this toolkit utilizes only US B-mode images for US-guided intervention applications. There are no functionalities for US elastography or PA image –guided interventional applications.

- NifTK platform

The NifTK [43] software platform was also developed for image-guided interventional applications. This platform is comprised of NiftyIGI, NiftyGuide, and

NiftyLink. NiftyGuide are small individual applications for acquiring sensor data (e.g. tracking data or US data) and sending the collected data via NiftyLink. NiftyIGI is the main workstation application for visualizing the collected data (e.g. video stream, US data, and tracking information). NiftyLink is a small messaging library based on OpenIGTLink [45, 74] and the Qt SDK [46], which provides a simple loosely coupled connection model of client and server. Therefore, the Nifty platform can communicate with the PLUS toolkit and 3D Slicer applications. However, this platform also does not have any pipeline for data processing of advanced US imaging modalities such as US elastography or PAI.

2.3. Significance

US imaging is a popular and convenient medical imaging modality. Recently, conventional US imaging has been frequently integrated with other medical imaging modalities (CT, MRI), tracking systems and robotic systems for IGT [19-21]. These systems are also exploring the use of advanced US imaging such as US elastography, PA imaging, and thermal imaging [19, 30, 81]. However, in order to integrate the results of advanced ultrasound research with existing IGT systems, three important things must be considered: the collection of low-level US data, the capability for real-time data processing, and data communication with existing IGT systems.

First of all, the collection of low-level ultrasound data, RF data or channel data (pre-beamformed RF data), must be in real time. This low-level ultrasound data is essential for computing advanced ultrasound image modalities, such as ultrasound elastography or ultrasound PAI, as shown by the data processing pipe lines for US B-mode, elastography, and PAI in **Figure 2.14**. Although some systems provide the collection and storage of RF data or US channel data through their own proprietary or open interfaces (e.g. Ultrasonix [82], Siemens [83], Hitachi [84], ZONARE [85], or Philips/Agilent [86]), these methods are relatively non-configurable, and the data is often asynchronously saved as files. Commercial US systems do not support real-time access to low-level US data.

Secondly, another consideration is the capability of real-time data processing for advanced ultrasound imaging. This requires access to ultrasound pre- / post-beamformed RF data in a high-bandwidth, low-latency system and subsequent processing using similarly high-performance algorithms. Finally, it must have data communication protocols with existing IGT systems. IGT procedures typically require many sources of information including surgical instrument tracking data and pre-/ intra-operative medical images. With continued research in integrating advanced US and IGT systems, there is a growing demand for the synchronization of data from these many sources. There is also a demand for a bidirectional communication mechanism between modules in a US-IGT system.

As shown in **Figure 2.14**, there are common parts of data-processing to reconstruct US B-mode, elastography, and PA images: e.g. beamforming, envelope detection, compression, and scan conversion. However, most existing US systems implement the computational functions on hardware such as field programmable gate arrays (FPGA). Therefore, this architecture is not conducive to applying new reconstruction algorithms for US imaging research because the implemented computational architecture is usually inflexible and does not allow any modification. To resolve this limitation, a distributed modular architecture is additionally required for improving the efficiency of research and development. We could achieve efficient resource allocation across different computers for real-time data computation and reuse of the common computational functions by breaking down the large collection of processes of existing US systems into a collection of independent and communicating modules.

Several software frameworks and toolkits have been developed to integrate US data acquisition, processing and visualization with existing IGT systems [19, 20, 39, 86]. However, there is no toolkit for advanced US research and integration with existing IGT systems. To solve these issues, we propose a real-time software component framework for advanced interventional ultrasound research.

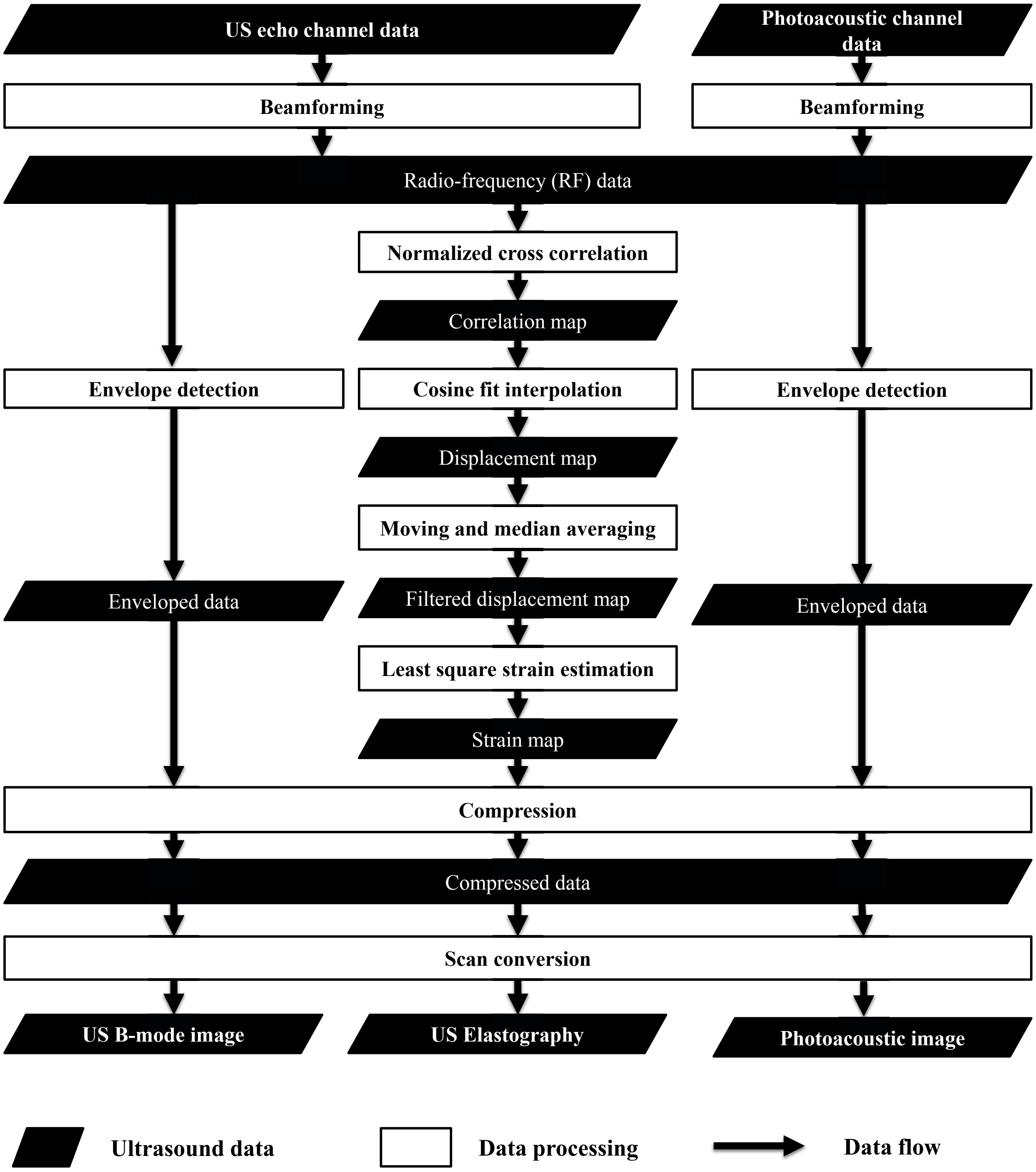


Figure 2.14. Flowcharts for US B-mode, elastography and photoacoustic imaging reconstruction.

Chapter 3:

Real-time medical ultrasound imaging and interventional component framework

3.1. Overview

Although there have been many software toolkits, frameworks and executable components developed for image-guided therapy, most of them do not provide a modular research environment for flexible ultrasound-imaging research systems. Moreover, according to our study, these software frameworks are not optimized to allow for simple integration of advanced ultrasound imaging such as ultrasound elastography imaging and photoacoustic imaging. Therefore, we introduce a new real-time software framework for advanced US research, called the *MUSiiC Toolkit*, which has the following properties:

- **Network distributed modularity:** To integrate heterogeneous IGT systems, a distributed system architecture is needed. Therefore, inter-module communication mechanisms should be implemented using network connections or inter-process communication methods [87, 88]. Moreover, each module of the toolkit follows the pattern of 'functional decomposition organized by flow of data' [89] to provide only specialized functionalities for easy configuration, high reliability, and minimal hardware footprints.

- **Open interface:** Communication protocols, data-type, and file formats saved on hard disk have to be well-defined, well-documented and backward-compatible for interoperability between existing toolkits and third-party components.
- **Real-time communication and processing (parallel computing):** For clinical applications, high-bandwidth US data has to be collected and processed with minimal latency. With multi-core CPUs and GPU, high-bandwidth applications can be executed and balanced easily when implemented with parallel algorithms and independent execution threads.
- **Portable and operating system agnostic:** For flexibility when integrating a new advanced ultrasound research module with existing IGT systems, all modules should be portable across major operating systems such as Windows, Linux, or Mac OS-X.

Figure 3.1 shows an overview of the *MUSiiC ToolKit* [72]. Our software framework can be classified into four categories: OpenIGTLink [45], *OpenIGTLinkMUSiiC* [47, 72], *MUSiiC Notes*, and *MUSiiC Modules*, a collection of executable programs. OpenIGTLink is a software library proposed by J. Tokuda [45] using a standard TCP/IP-based message protocol for the integration of medical IGT systems. In this library, multiple data types are defined as TCP/IP messages for real-time communication between subsystems of an IGT system, and there are serialization and deserialization mechanisms for each message.

OpenIGTLinkMUSiiC [47] is the extended version of OpenIGTLink by adding a special US data message and other message types (*USMessage*, *EIMessage*, *GenMessage*, *ArgMessage*, and *FileMessage*) for advanced US research. In addition, *OpenIGTLinkMUSiiC* includes active task classes that have their own independent task thread. These task threads are based on multithreaded techniques and thread-safe inter-process communication (IPC) [87, 88] for efficient real-time US data computation. We also implemented the Observer design pattern [90] in this library for bidirectional communication between applications or active task class components. Since *OpenIGTLinkMUSiiC* provides the basic functionalities, such as abstract active-task classes (*MUSiiCTaskObject*, *MUSiiCTaskAbstract*, and *MUSiiCTaskInterfaceAbstract*), thread-safe data interfaces (*MUSiiCVector* and *MUSiiCVectorSet*), and callback interfaces (*MUSiiCCallbackInterface*, *MUSiiCCallbackInterfaceControl*), it is therefore considered the fundamental library in *MUSiiC ToolKit* [72].

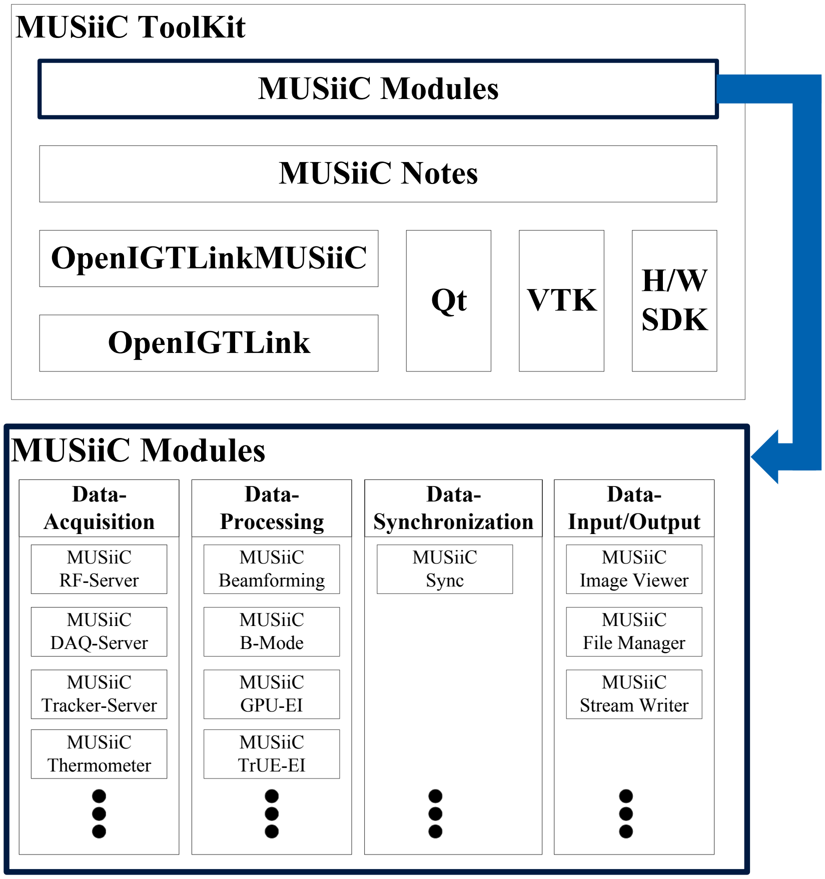


Figure 3.1. Overview of the *MUSiiC ToolKit*

MUSiiC Notes is the component library in our software framework. It provides US research-oriented task classes, such as US data acquisition, beamforming, envelope detection, scan conversion of ultrasound data, data synchronization, and Graphic User Interface (GUI) units for each task class or executable program. As shown in **Figure 3.1**, *MUSiiC Notes* depends on OpenIGTLink, *OpenIGTLinkMUSiiC*, the Qt SDK [46], the Visualization ToolKit (VTK) [91], and hardware-dependent SDKs such as the Ultrasonix SDK (Ultrasonix Co.) [54] and the 3D Guidance medSAFE / driveBay SDK (Ascension Technology Co.) [92]. *MUSiiC Modules* is a collection of executable programs related to US research. All of these programs are based on a network distributed computing system to improve the performance of US data processing and the flexibility to reconfigure the US research system. To support these features, all programs have their own network classes (*MUSiiCTCPServer* and *MUSiiCTCPClient*) and file input/output (I/O) classes (*MUSiiCFileIO)*. **Figure 3.1** also shows that there are four types of executable modules: data acquisition modules, data processing modules, data synchronization module, and data I/O modules. Different data acquisition modules support collecting US pre-beamformed and post-beamformed RF data, tracking information of a medical device, or temperature information in real-time. In the data processing modules, US B-mode or US elasticity images are computed using the data collected from the data acquisition modules. The data synchronization module can combine different data from multiple data sources using the timestamps of each data. We can also include additional information such as ultrasound calibration information to the synchronized data at this module [93]. The data I/O modules allow us to efficiently display the data in various formats on the computer monitor. It also gives us the capability to read and write the data to files on the local hard disk in real-time.

3.2. *OpenIGTLinkMUSiiC*

In 2008, a standard TCP/IP network protocol for IGT systems named OpenIGTLink was announced. This protocol enabled the communication of position, transformation, and image data, as well as commands or system status messages between two processes in distributed IGT systems [45, 73]. Thanks to its simple and extensible nature, several research groups and commercial companies have taken up OpenIGTLink in their development [20, 42, 43, 47, 50, 72]. Also, our group chose OpenIGTLink for communicating US data, the outputs of our advanced US modules, and localization data from tracker devices between modules in our *MUSiiC Toolkit* and existing research IGT systems such as 3D Slicer [44, 80].

However, the existing OpenIGTLink turned out to be insufficient for this application scenario, as it did not include specific messages that can contain low-level US data or output values of advanced US computation. For these reasons, we introduce our extended version of OpenIGTLink, *OpenIGTLinkMUSiiC*, for advanced US research such as US elastography, thermal imaging and photoacoustic imaging in real-time, and other applications [47, 72].

3.2.1. New message types in *OpenIGTLinkMUSiiC*

In *OpenIGTLinkMUSiiC*, we add five new message types. Two message types are for transmission of general ultrasound data and the results of our advanced ultrasound research modules. The remaining three messages are for communicating text-based control arguments and files.

3.2.1.1. New ultrasound messages in *OpenIGTLinkMUSiiC*

USMessage and *EIMessage* message types in *OpenIGTLinkMUSiiC* are extensions of the *ImageMessage* in OpenIGTLink, containing general ultrasound data with its additional information and elastography data with the results of its computation, respectively. **Figure 3.2** presents the class diagram and the structure for these two messages with *ImageMessage* of OpenIGTLink. As seen from **Figure 3.2**, the two message types have identical structure: each message is a subclass of *ImageMessage*, and each carries its own tag after the *ImageMessage* data block.

Table 3.1. US_Tag of USMessage

Data	**Type**	**Description**
txf	32bit int	Transmit frequency of ultrasound beam
sf	32bit int	Sampling frequency of ultrasound data
dr	32bit int	Frame rate
ls	32bit int	Line density
sa	32bit int	Steering angle
probe	32bit int	ID of ultrasound probe
ea	32bit int	Extension angle
elements	32bit int	The number of elements in the probe
pitch	32bit int	The spacing between elements
radius	32bit int	The curvature of the probe
probe_angle	32bit int	The field of view of the probe
tx_offset	32bit int	The offset in the steering image (phased array)

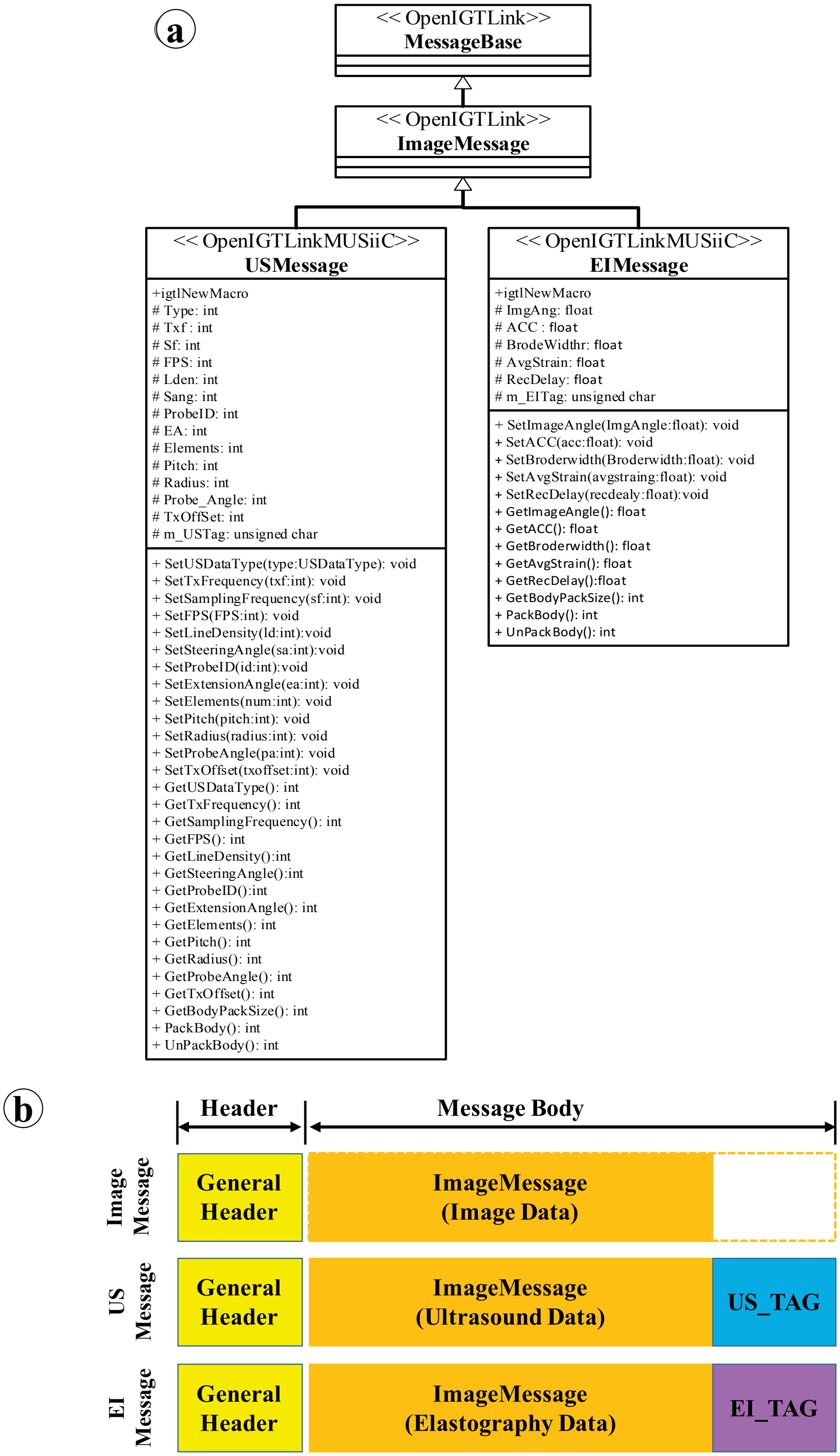


Figure 3.2. Two new message types – *USMessage* and *EIMessage*: (a) class diagram, (b) general structure

Table 3.2. EI_Tag of EIMessage

Data	**Type**	**Description**
ImagAng	32bit float	Extension angle
ACC	32bit float	Average Cross-correlation
BorderWidth	32bit float	Border-Width of Elastography
AveStrain	32bit float	Average Strain value
RecDelay	32bit float	The delay of Receiving

More specifically, general ultrasound data will be contained in the *ImageMessage* block of *USMessage* with its metric information, data matrix size, position, and orientation in the *US_Tag*. Additionally, ultrasound-specific data is stored in the US Tag (**Table 3.1**) of *USMessage*. *EIMessage* sores elastography the same way; the *ImageMessage* part of *EIMessage* contains the elastography image and the EI Tag (**Table 3.2**) has additional information of the EI data. As seen in **Figure 3.2**, each message has its own serialization (*PackBody()*) and deserialization(*UnPackBody()*) functions for its own tag. Because these two message types are subclasses of *ImageMessage*, they can contain 2D/3D ultrasound data or elastography data. In the case of 3D data transfers, sub-volume data can be partially updated using these two messages. Moreover, these messages are backwards-compatible with *ImageMessage* in deserialization procedures.

3.2.1.2. New control messages in *OpenIGTLinkMUSiiC*

From our experience, we identified the need to communicate a general user-defined message such as “any number of arguments” or “any kind of file”. Although there is a *StatusMessage* type in OpenIGTLink, the message has a predefined set of status parameters. For more universal data transfer and communication, we built three new message types in *OpenIGTLinkMUSiiC*: *GenMessage*, *ArgMessage*, and *FileMessage*.

GenMessage has the general header of OpenIGTLink for compatibility with *Pack()* and *UnPack()* functions of the existing OpenIGTLink protocol. There is no definition of any protocol inside the message body part of *GenMessage*. This means that we can input and retrieve any user-defined content. *ArgMessage* has the specific goal of communicating text-based argument messages or control parameters. *ArgMessage* is a subclass of the *GenMessage* class (see **Figure 3.3**). That is, the arguments of *ArgMessage* will be in the message-body section of *GenMessage*. *ArgMessage* has only serialization and deserialization methods for its text-based arguments or control parameters. An example code snippet using *ArgMessage* to collect US B-Mode image could look like the following:

Snippet 3.1. Packing *ArgMessage* – Add arguments using the function *AddArgument()*

```
char argument1[] =  “USBMode”       // Image mode for ultrasound data acquisition
char argument2[] =  “ImageDepth 5.0”  // Image Depth of US B-Mode
igtl::ArgMessage::Pointer argMsg = igtl::ArgMessage::New();
argMsg->AddArgument(argument1);
argMsg->AddArgument(argument2);
argMsg->AllocateArguments();
argMsg->Pack();
```

Snippet 3.2. Unpacking *ArgMessage* – Extract arguments using the function *GetArgument()*

```
argMsg->Unpack(1);
int sz = argMsg->GetNumOfArguments();
char* arg1 = argMsg->GetArgument(0);  // “USBMode”
char* arg2 = argMsg->GetArgument(1);  // “ImageDepth 5.0”
```

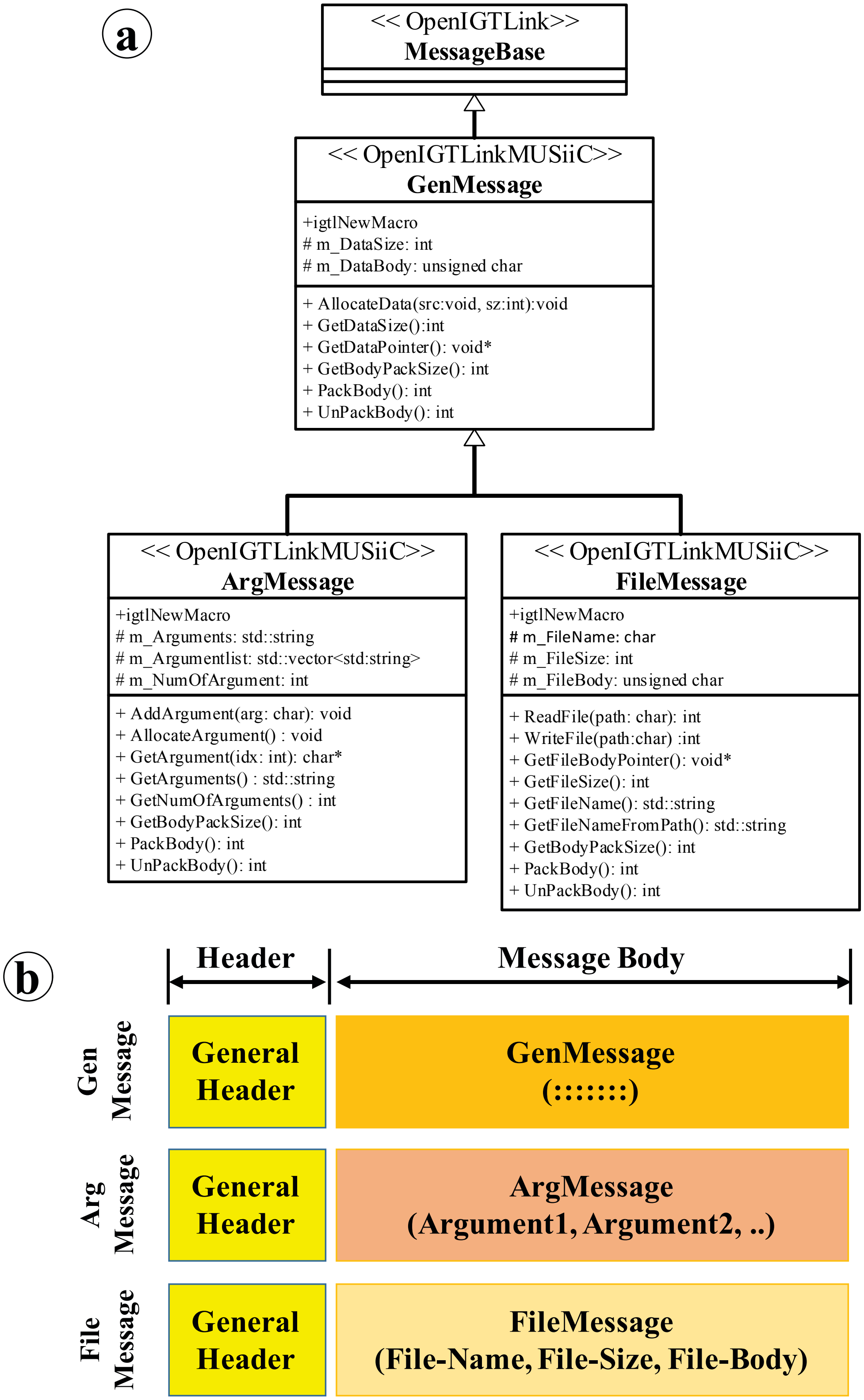


Figure 3.3. New message types *GenMessage*, *ArgMessage* and *FileMessage*: (a) class diagram, (b) message structure

FileMessage is used for communicating any file “as itself” in binary form between modules. Apart from *FileMessage*’s information about file name and file size, it is similar to *ArgMessage*; this message is also derived from *GenMessage* and has its own serialization and deserialization functions for files (see **Figure 3.3**). Example codes of packing and unpacking for *FileMessage* might look as follows:

Snippet 3.3. Packing *FileMessage* – Add a file using the function *ReadFile()*

```
const char* path = “c:/MUSiiCData/USBMode.igtl”
igtl::FileMessage::Pointer fMsg = igtl::FileMessage::New();
fMsg->ReadFile(path);
fMsg->Pack();
```

Snippet 3.4. Unpacking *FileMessage* – Extract a file using the functions *WriteFile()* and *GetFileName()*

```
fMsg->Unpack(1);
fMsg->WriteFile(fMsg->GetFileName());   //“c:/MUSiiCData/USBMode.igtl”
```

3.2.2. Real-time US Data-Computation in *OpenIGTLinkMUSiiC*

A significant advantage of US in IGT systems is its ability to provide intra-operative data such as B-mode images, US strain images, or photoacoustic images [30, 41, 50, 63, 81]. However, in general, US data processing is computationally expensive [22]. Therefore, a well-defined task abstract class that supports multithreaded programming is an essential part of a software framework for US research.

To fulfill these requirements, we built an active task object class, *MUSiiCTaskObject*, to run a task function with an independent thread, and two task abstract classes, *MUSiiCTaskAbstract* and *MUSiiCTaskInterfaceAbstract*, to manage multiple task objects efficiently. Also, we made a thread-safe data interface, *MUSiiCVector*, for transferring data between task objects or task abstract classes and a data interface manager, *MUSiiCVectorSet*. Both of these classes define “concurrent_pushback”, “concurrent_pop”, and “concurrent_get” functions that allow the data to be safely written, deleted, or read by multiple threads. **Figure 3.4** represents the Unified Modeling Language (UML) class diagrams for *MUSiiCTaskObject*, *MUSiiCTaskAbstract*, *MUSiiCTaskInterfaceAbstract*, *MUSiiCVector*, and *MUSiiC-VectorSet*.

Snippet 3.5. The function type of *MUSiiCTaskFunction*/*MUSiiCCallbackFunction*

```
/// The function type for MUSiiCTaskFunction/MUSiiCCallbackFunction
/// Input parameters
///          - command: Represents a changed status or any command
///          - taskInfo: Describe the information of a task
///          - ptr: The pointer of caller
///          - msg: A pointer of message of OpenIGTLink or OpenIGTLinkMUSiiC
///          - data1: A pointer of user defined data
///          - data2: A pointer of user defined data
/// Output parameter
///          return an integer value
///          - 1: failed
///          ≥ 0: Successes
typedef int (*MUSiiCTaskFtn) (int command, int taskinfo, void* ptr,
                              igtl::MessageBase::Pointer msg,
                              void* data1, void* data2)
```

The *MUSiiCTaskObject* class has a Multithread class pointer from the OpenIGTLink library that supports independent threads on multiple operating systems. In this class, the main task function is declared as a virtual function. This means that we can implement our own specific task function in any subclasses of this class. The task function of *MUSiiCTaskObject* follows the function type of *MUSiiCTaskFunction* as shown in **Snippet 3.5**. Multiple input parameters are defined in these function types, allowing us to delicately control a task function. Moreover, we can build a task sequence by adding local or global functions of *MUSiiCTaskFunction* type to *MUSiiCTaskObject*.

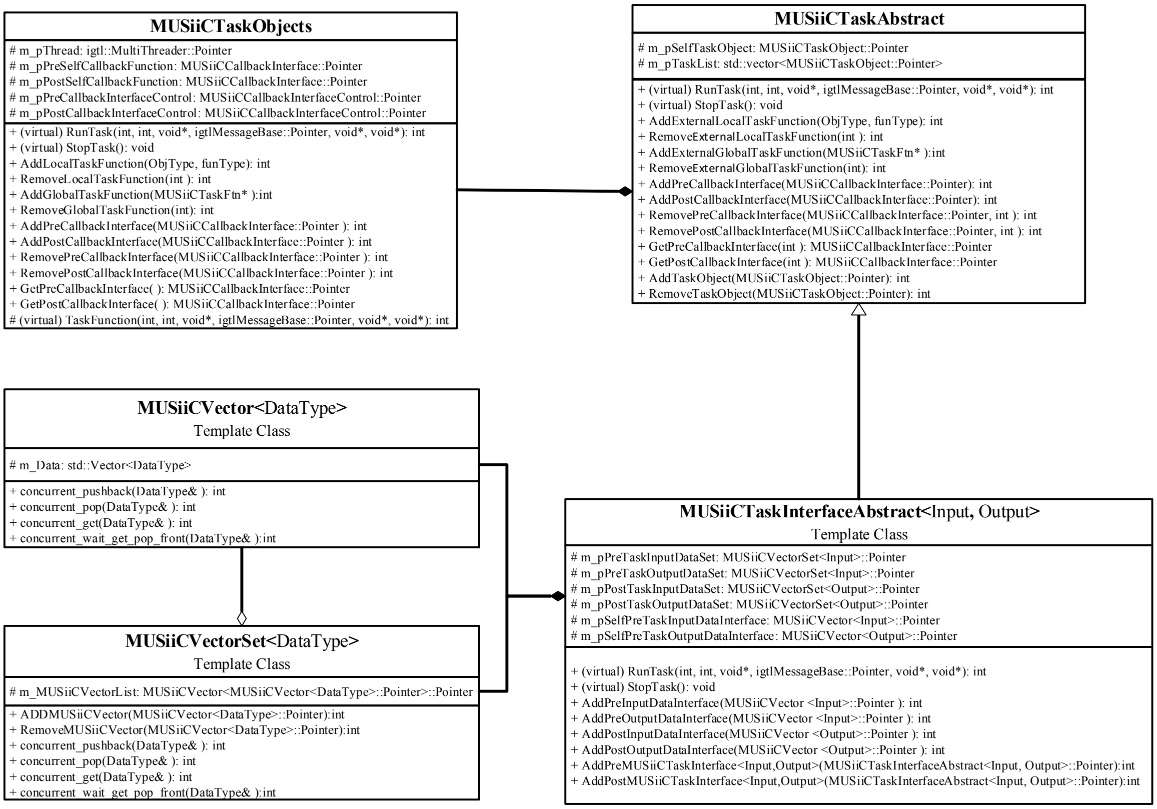

MUSiiCTaskObjects
# m_pThread: igtl::MultiThreader::Pointer
# m_pPreSelfCallbackFunction: MUSiiCCallbackInterface::Pointer
# m_pPostSelfCallbackFunction: MUSiiCCallbackInterface::Pointer
# m_pPreCallbackInterfaceControl: MUSiiCCallbackInterfaceControl::Pointer
# m_pPostCallbackInterfaceControl: MUSiiCCallbackInterfaceControl::Pointer
+ (virtual) RunTask(int, int, void*, igtlMessageBase::Pointer, void*, void*): int
+ (virtual) StopTask(): void
+ AddLocalTaskFunction(ObjType, funType): int
+ RemoveLocalTaskFunction(int): int
+ AddGlobalTaskFunction(MUSiiCTaskFtn*):int
+ RemoveGlobalTaskFunction(int): int
+ AddPreCallbackInterface(MUSiiCCallbackInterface::Pointer): int
+ AddPostCallbackInterface(MUSiiCCallbackInterface::Pointer): int
+ RemovePreCallbackInterface(MUSiiCCallbackInterface::Pointer): int
+ RemovePostCallbackInterface(MUSiiCCallbackInterface::Pointer): int
+ GetPreCallbackInterface( ): MUSiiCCallbackInterface::Pointer
+ GetPostCallbackInterface( ): MUSiiCCallbackInterface::Pointer
# (virtual) TaskFunction(int, int, void*, igtlMessageBase::Pointer, void*, void*): int
MUSiiCTaskAbstract
# m_pSelfTaskObject: MUSiiCTaskObject::Pointer
# m_pTaskList: std::vector<MUSiiCTaskObject::Pointer>
+ (virtual) RunTask(int, int, void*, igtlMessageBase::Pointer, void*, void*): int
+ (virtual) StopTask(): void
+ AddExternalLocalTaskFunction(ObjType, funType): int
+ RemoveExternalLocalTaskFunction(int): int
+ AddExternalGlobalTaskFunction(MUSiiCTaskFtn*):int
+ RemoveExternalGlobalTaskFunction(int): int
+ AddPreCallbackInterface(MUSiiCCallbackInterface::Pointer): int
+ AddPostCallbackInterface(MUSiiCCallbackInterface::Pointer): int
+ RemovePreCallbackInterface(MUSiiCCallbackInterface::Pointer, int): int
+ RemovePostCallbackInterface(MUSiiCCallbackInterface::Pointer, int): int
+ GetPreCallbackInterface(int): MUSiiCCallbackInterface::Pointer
+ GetPostCallbackInterface(int): MUSiiCCallbackInterface::Pointer
+ AddTaskObject(MUSiiCTaskObject::Pointer): int
+ RemoveTaskObject(MUSiiCTaskObject::Pointer): int
MUSiiCVector<DataType>
Template Class
# m_Data: std::Vector<DataType>
+ concurrent_pushback(DataType&): int
+ concurrent_pop(DataType&): int
+ concurrent_get(DataType&): int
+ concurrent_wait_get_pop_front(DataType&):int
MUSiiCVectorSet<DataType>
Template Class
# m_MUSiiCVectorList: MUSiiCVector<MUSiiCVector<DataType>::Pointer>::Pointer
+ ADDMUSiiCVector(MUSiiCVector<DataType>::Pointer):int
+ RemoveMUSiiCVector(MUSiiCVector<DataType>::Pointer):int
+ concurrent_pushback(DataType&): int
+ concurrent_pop(DataType&): int
+ concurrent_get(DataType&): int
+ concurrent_wait_get_pop_front(DataType&):int
MUSiiCTaskInterfaceAbstract<Input, Output>
Template Class
# m_pPreTaskInputDataSet: MUSiiCVectorSet<Input>::Pointer
# m_pPreTaskOutputDataSet: MUSiiCVectorSet<Input>::Pointer
# m_pPostTaskInputDataSet: MUSiiCVectorSet<Output>::Pointer
# m_pPostTaskOutputDataSet: MUSiiCVectorSet<Output>::Pointer
# m_pSelfPreTaskInputDataInterface: MUSiiCVector<Input>::Pointer
# m_pSelfPreTaskOutputDataInterface: MUSiiCVector<Output>::Pointer
+ (virtual) RunTask(int, int, void*, igtlMessageBase::Pointer, void*, void*): int
+ (virtual) StopTask(): void
+ AddPreInputDataInterface(MUSiiCVector <Input>::Pointer): int
+ AddPreOutputDataInterface(MUSiiCVector <Input>::Pointer): int
+ AddPostInputDataInterface(MUSiiCVector <Output>::Pointer): int
+ AddPostOutputDataInterface(MUSiiCVector <Output>::Pointer): int
+ AddPreMUSiiCTaskInterface<Input,Output>(MUSiiCTaskInterfaceAbstract<Input, Output>::Pointer):int
+ AddPostMUSiiCTaskInterface<Input,Output>(MUSiiCTaskInterfaceAbstract<Input, Output>::Pointer):int

Figure 3.4. UML class diagrams of *MUSiiCTaskObjects*, *MUSiiCTaskAbstract*, *MUSiiCTaskInterfaceAbstract*, *MUSiiCVector*, and *MUSiiCVectorSet*

Although we can run a task thread independently using an instance of *MUSiiCTaskObject*, multiple task threads are needed to improve the performance of US data processing. *MUSiiCTaskAbstract* is designed to address this requirement. In **Figure 3.4**, *MUSiiCTaskAbstract* has a container variable and several task control functions to control both its inherent task-function as a member variable of *MUSiiCTaskObject* and additional multiple *MUSiiCTaskObject* instances. *MUSiiCTaskInterfaceAbstract* is a derived class of *MUSiiCTaskAbstract*, which is designed for thread-safe data communication between *MUSiiCTaskInterfaceAbstract* objects. We designed *MUSiiCVector*, a concurrent data interface, based on the mutual exclusion method and *MUSiiCVectorSet*, a management class for this data interface. Both of these classes are based on the template design pattern and will allow any kind of data type to be transferred efficiently in our data interface.

Figure 3.5 shows the flexibility in our software framework to support a number of different multitasking patterns. The combination of *MUSiiCTaskObject*, *MUSiiCTaskAbstract* and *MUSiiCTaskInterfaceAbstract* allow us to implement the cases shown in **Figure 3.5(a)–(d)** respectively. These cases are the combinations of single or multiple threads and a single task or a sequence of tasks. Moreover, *MUSiiCTaskInterfaceAbstract* has instances of *MUSiiCVectorSet*, the provided data interface, and *MUSiiCVector*, the required data interface, for data communication between task class objects. As seen in **Figure 3.4**, since *MUSiiCVectorSet* can control multiple *MUSiiCVector* objects concurrently, each task class can receive multiple data from different data providers. The result from a task class can also be broadcasted to multiple data consumers in parallel. **Figure 3.5(e)** shows the network of task classes that are based on *MUSiiCTask-*

InterfaceAbstract. Tasks 1 and 2 form a serial connection, while tasks 4, 5, and 6 are connected to task 2 in parallel. Task 2 and 3 demonstrate a serial connection between *MUSiiCTaskInterfaceAbstract* and any data provider or consumer that is not based on *MUSiiCTaskInterfaceAbstract*. This example shows that *MUSiiCTaskInterfaceAbstract* provides high flexibility in the connectivity and data communication between task classes.

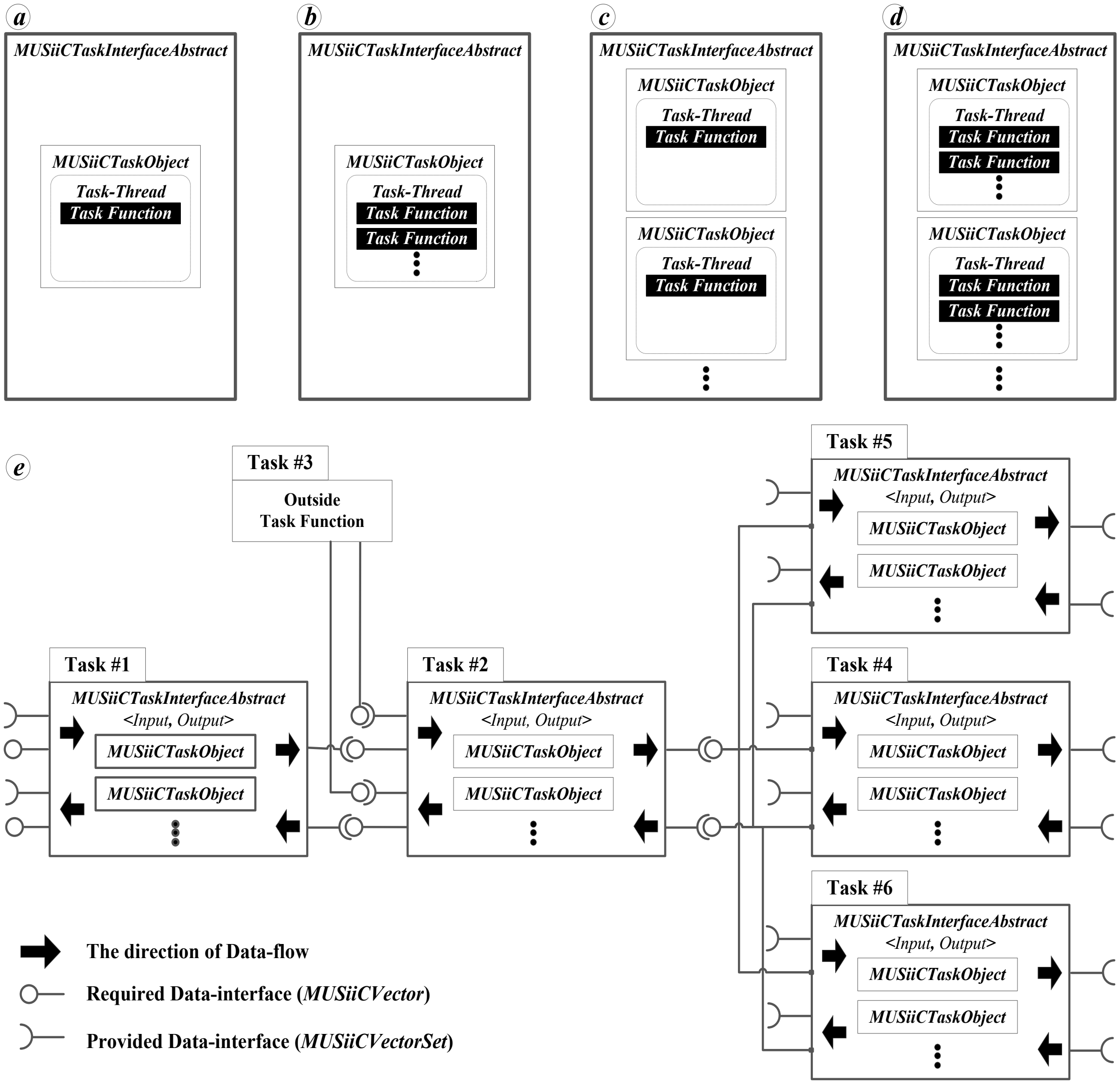


Figure 3.5. Examples of Multitasking pattern using *MUSiiCTaskObject* and *MUSiiCTaskInterfaceAbstract*. (a) Single thread of single task, (b) Single thread of task sequence, (c) Multiple threads of single task, (d) Multiple threads of task sequence, and (e) Network diagram of task classes based on *MUSiiCTaskInterfaceAbstract*.

Snippet 3.6 represents an example source code to establish the network diagram of task classes based on *MUSiiCTaskInterfaceAbstract* (see **Figure 3.5(e)**). First, a user defined task function (Task3) is declared as a global function (see **Snippet 3.6**: line 5 – 12) and five instances (Task1, Task2, Task4, Task5 and Task6) of *MUSiiCTask-InterfaceAbstract* class are created with the data type of IGTLMsg (see **Snippet 3.6**: line 15 – 24). And, the Task1 and the Task2 are connected as a serial connection by using the function of *AddPostMUSiiCTaskInterface* in Task1 (see **Snippet 3.6**: line 28). The user defined function (Task 3) replaces the original task function in Task2 with the function call *AddExternalGlobalTaskFunction* (see **Snippet 3.6**: line 31). Then, the parallel connection of Task4, Task5 and Task6 to Task2 is established with the *AddPostMUSiiCTaskInterface* (see **Snippet 3.6**: line 34 - 36).

Snippet 3.6. An example source code to establish the network diagram of task classes based on *MUSiiCTaskInterfaceAbstract*

```
 1  using namespace igtl;
 2  typedef MessageBase::Pointer  IGTLMsg;
 3
 4  /// Declare user-defined task function as global function
 5  int Task3(int numOfRun = 0, int taskInfo = 0,
 6                 void* ptr = NULL, igtl::MessageBase::Pointer msg = NULL,
 7                 void* data2 = NULL, void* data3 = NULL)
 8  {
 9          /// User-defined Task function
10           /// .......
11           return 1;
12  }
13
14  /// Create instances of MUSiiCTaskInterfaceAbstract
```

```
MUSiiCTaskInterfaceAbstract<IGTLMsg, IGTLMsg>::Pointer Task1
                              = MUSiiCTaskInterfaceAbstract<IGTLMsg, IGTLMsg>::New();
MUSiiCTaskInterfaceAbstract<IGTLMsg, IGTLMsg>::Pointer Task2
                             = MUSiiCTaskInterfaceAbstract<IGTLMsg, IGTLMsg>::New();
MUSiiCTaskInterfaceAbstract<IGTLMsg, IGTLMsg>::Pointer Task4
                             = MUSiiCTaskInterfaceAbstract<IGTLMsg, IGTLMsg>::New();
MUSiiCTaskInterfaceAbstract<IGTLMsg, IGTLMsg>::Pointer Task5
                             = MUSiiCTaskInterfaceAbstract<IGTLMsg, IGTLMsg>::New();
MUSiiCTaskInterfaceAbstract<IGTLMsg, IGTLMsg>::Pointer Task6
                             = MUSiiCTaskInterfaceAbstract<IGTLMsg, IGTLMsg>::New();

/// Establish connection of MUSiiCTaskInterface's instances
/// Connect the Task1 and Task2 with serial connection
Task1->AddPostMUSiiCTaskInterface<IGTLMsg, IGTLMsg>(Task2);

/// Add an external global task function to Task2
Task2->AddExternalGlobalTaskFunction(Task3);

/// Connect the Task4, Task5 and Task6 to Task2 in parallel
Task2->AddPostMUSiiCTaskInterface<IGTLMsg, IGTLMsg>(Task4);
Task2->AddPostMUSiiCTaskInterface<IGTLMsg, IGTLMsg>(Task5);
Task2->AddPostMUSiiCTaskInterface<IGTLMsg, IGTLMsg>(Task6);
```

3.2.3. Bidirectional communication mechanism in *OpenIGTLink-MUSiiC*

An interactive communication mechanism between the application level and task class component level is necessary to integrate US systems and IGT systems. **Figure 3.5** shows that asynchronous bidirectional communication between the task classes using the instances of *MUSiiCVector* and *MUSiiCVectorSet* is possible. We describe a synchronous bidirectional communication mechanism between task classes and an asynchronous communication method between applications.

3.2.3.1. Bidirectional Communication Mechanism at software class level.

We apply the observer software design pattern in *OpenIGTLinkMUSiiC* by adding instances of the *MUSiiCallbackInterface* and *MUSiiCCallbackInterfaceControl* classes in the *MUSiiCTaskObject* and *MUSiiCTaskAbstract* classes.

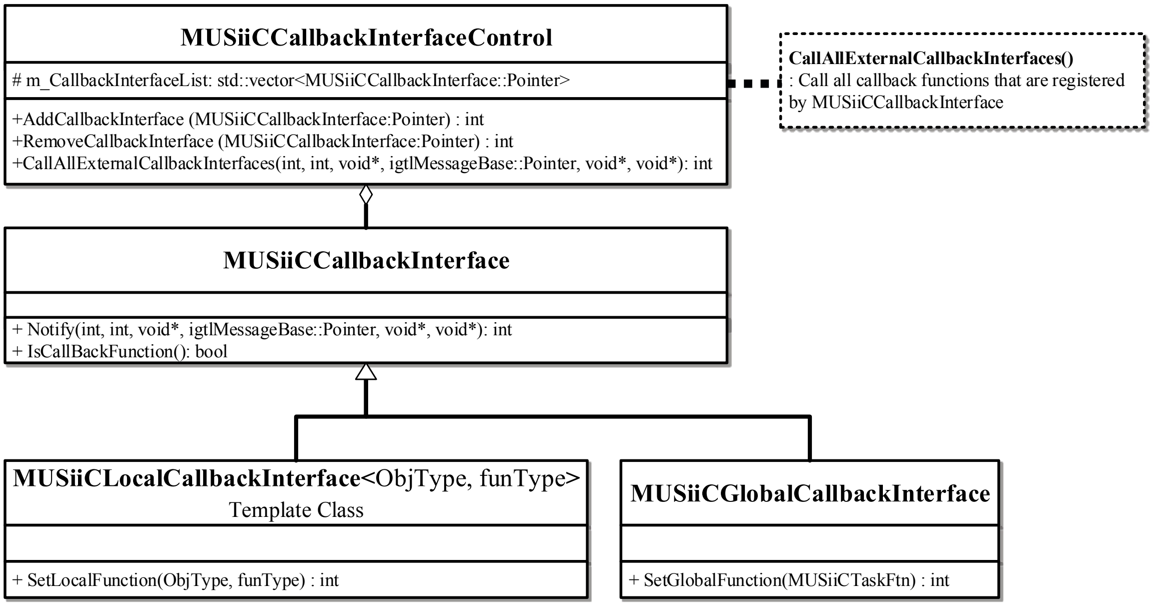

MUSiiCCallbackInterfaceControl
# m_CallbackInterfaceList: std::vector<MUSiiCCallbackInterface::Pointer>
+AddCallbackInterface (MUSiiCCallbackInterface:Pointer) : int
+RemoveCallbackInterface (MUSiiCCallbackInterface:Pointer) : int
+CallAllExternalCallbackInterfaces(int, int, void*, igtlMessageBase::Pointer, void*, void*): int
CallAllExternalCallbackInterfaces()
: Call all callback functions that are registered by MUSiiCCallbackInterface
MUSiiCCallbackInterface
+ Notify(int, int, void*, igtlMessageBase::Pointer, void*, void*): int
+ IsCallBackFunction(): bool
MUSiiCLocalCallbackInterface<ObjType, funType>
Template Class
+ SetLocalFunction(ObjType, funType) : int
MUSiiCGlobalCallbackInterface
+ SetGlobalFunction(MUSiiCTaskFtn) : int

Figure 3.6. UML class diagram of *MUSiiCCallbackInterface* and *MUSiiCCallbackInterface-Control*

Figure 3.6 represents the UML class diagram of *MUSiiCCallbackInterface*, *MUSiiCLocalCallbackInterface*, *MUSiiCGlobalCallbackInterface* and *MUSiiCCallbackInterfaceControl*. The *MUSiiCCallbackInterface* class provides basic observer functions: "Notify()," which sends notifications or messages to a pre-registered callback function, and "*IsCallBackFunction()*" which checks whether any callback function has been registered in this interface. As shown in **Figure 3.6**, this class has two child classes. *MUSiiCLocalCallbackInterface* is based on the template class design pattern and *MUSiiCGlobalCallbackInterface* can register any local or global function as a callback function of this interface class as long as the type of the function follows the type of *MUSiiCTaskFunction* (See **Snippet 3.5**). Moreover, *MUSiiCCallbackInterfaceControl* is designed to control multiple *MUSiiCCallbackInterface* objects efficiently. It has dedicated functions to add or remove a pre-defined instance of *MUSiiCCallbackInterface* to or from this class. All predefined callback functions registered in this class are notified or updated by "*CallAllExternalCallbackInterfaces*".

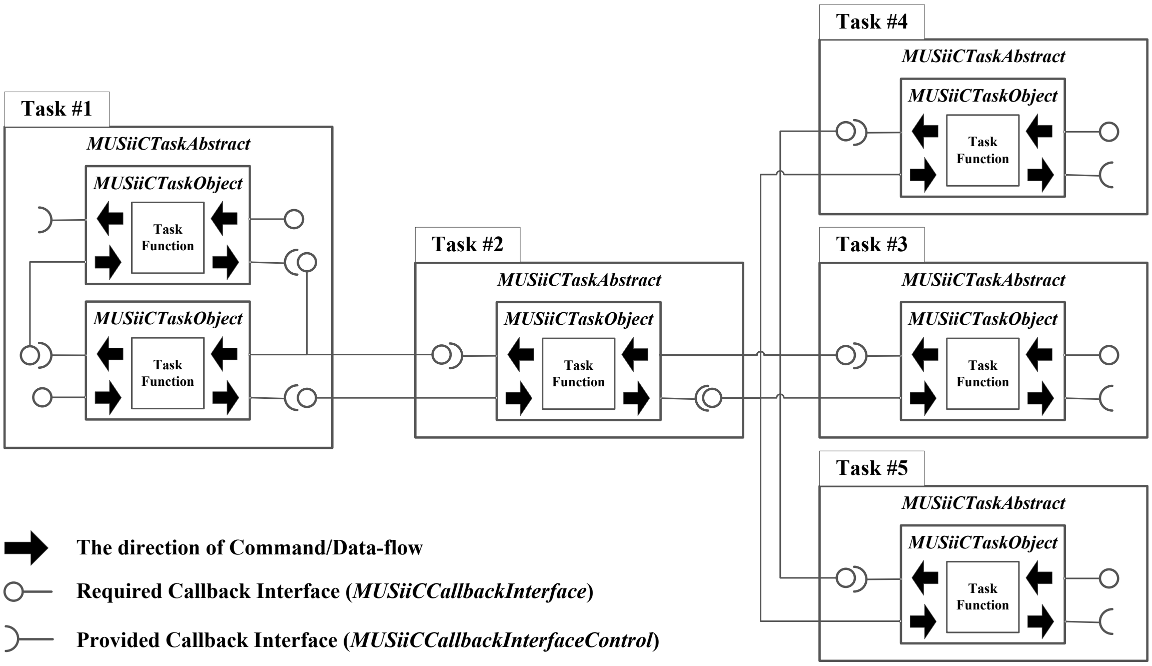


Figure 3.7. Network diagram of callback interfaces between task classes based on *MUSiiC-TaskAbstract*

As in **Figure 3.4**, *MUSiiCTaskObject* has instances of *MUSiiCCallbackInterface* and *MUSiiCCallbackInterfaceControl*, and *MUSiiCTaskAbstract* provides several functions allowing access to these instances. With this design, we can achieve a synchronous bidirectional communication based on the observer design pattern between the *MUSiiCTaskObject* instances in a single *MUSiiCTaskAbstract* class or in multiple *MUSiiCTaskAbstract* classes. A network diagram of callback interfaces between task classes based on this software design is shown in **Figure 3.7**. Although this network diagram is very similar to the network diagram in **Figure 3.5** (e), it demonstrates synchronous bidirectional communication using callback interfaces as opposed to asynchronous bidirectional communication using concurrent data interfaces. This shows that our software framework is capable of providing asynchronous and synchronous bidirectional communication at the software task class level.

Snippet 3.7 shows an example source code to establish the connection of task classes' callback interfaces based on *MUSiiCTaskAbstract* (see **Figure 3.7**). Similar to the establishment of network diagram of task classes, an instance of *MUSiiCTaskObject* (TaskObj) and pointers of *MUSiiCTaskAbstract* (Task1, Task2, Task3, Task4 and Task5) are created (see **Snippet 3.7:** line 2 - 9). And, the TaskObj instance is added to Task1 with the function of *AddTaskObject()* (see **Snippet 3.7:** line 12). Then, the callback interface connections of the TaskObj and the Task1's default TaskObject are established with the *MUSiiCTaskObject*'s functions (*AddPreCallbackInterface()* and *GetPostSelf-CallbackInterface()*) and the *MUSiiCTaskAbstract*'s functions (*GetPreSelfCallBack-Interface()* and *AddPostCallbackInterface()*) (see the **Snippet 3.7:** 15–16). Where, *-Pre-* and *-Post-* represent the interface connection to a prior and posterior instance of *MUSiiCTaskAbstract,* respectively. A serial and parallel callback interface connection between two instances of *MUSiiCTaskAbstract* can be created by the functions of

GetPreSelfCallbackInterface() and *AddPostCallback-Interface()*)(see **Snippet 3.7:** line 19 - 30).

Snippet 3.7. An example source code to establish the network diagram of task classes' callback interfaces based on *MUSiiCTaskAbstract*

```
 1  ///Create an instance of MUSiiCTaskObject
 2  MUSiiCTaskObject::Pointer TaskObj = MUSiiCTaskObject::New();
 3
 4  ///Create instance of MUSiiCTaskAbstract
 5  MUSiiCTaskAbstract::Pointer Task1 = MUSiiCTaskAbstract::New();
 6  MUSiiCTaskAbstract::Pointer Task2 = MUSiiCTaskAbstract::New();
 7  MUSiiCTaskAbstract::Pointer Task3 = MUSiiCTaskAbstract::New();
 8  MUSiiCTaskAbstract::Pointer Task4 = MUSiiCTaskAbstract::New();
 9  MUSiiCTaskAbstract::Pointer Task5 = MUSiiCTaskAbstract::New();
10
11  /// Add the TaskObj to the Task1
12  Task1->AddTaskObject(TaskObj);
13
14  /// Establish callback interface  connections of  the TaskObj and the default TaskObject  of Task1
15  TaskObj->AddPostCallbackInterface(Task1->GetPreSelfCallbackInterface());
16  Task1->AddPreCallbackInterface(TaskObj->GetPostSelfCallbackInterface());
17
18  /// Establish the connection of callback interfaces between Task1 and Task2 in serial
19  Task1->AddPostCallbackInterface(Task2->GetPreSelfCallbackInterface());
20  Task2->AddPreCallbackInterface(Task1->GetPostSelfCallbackInterface());
21
22  /// Establish the callback interface connections of Task2 to Task3, Task4 and Task5 in parallel
23  Task2->AddPostCallbackInterface(Task3->GetPreSelfCallbackInterface());
24  Task3->AddPreCallbackInterface(Task2->GetPostSelfCallbackInterface());
25
26  Task2->AddPostCallbackInterface(Task4->GetPreSelfCallbackInterface());
```

```
Task4->AddPreCallbackInterface(Task2->GetPostSelfCallbackInterface());

Task2->AddPostCallbackInterface(Task5->GetPreSelfCallbackInterface());
Task5->AddPreCallbackInterface(Task2->GetPostSelfCallbackInterface());
```

3.2.3.2. Bidirectional Communication Mechanism at application level

Since US IGT subsystems are often connected at the application level, software frameworks for these systems must support bidirectional communication. With the increasing availability of Ethernet in IGT systems, TCP/IP is a common communication mechanism. However, the TCP/IP socket provided by OpenIGTLink only supports synchronous communication. There is a need to improve this method and the performance of TCP/IP data transmission [47, 94].

We built custom TCP/IP network I/O classes (*MUSiiCTCPServer* and *MUSiiCTCPClient*) based on I/O completion ports [47, 95] for the Windows operating system. We intend to implement asynchronous TCP/IP network I/O classes using epoll [96, 97] and Kqueue [98] for Linux and Mac OS X operating systems, respectively, in the near future. They provide an efficient threading model for handling multiple asynchronous I/O requests in a program and for supporting bidirectional communication between multiple clients at the application level. The block diagram for our network classes is shown in **Figure 3.8**. Each of these classes has three instances of *MUSiiCTaskObject* to run independent tasks. The *MUSiiCTCPServer* class has Listening-Task, Data-Sending, and Data-Receiving *MUSiiCTaskObject* instances. The *MUSiiCTCPClient* class has Creating-client socket, Data-Sending, and Data-Receiving *MUSiiCTaskObject* instances. The task thread of Listening-Task in *MUSiiCTCPServer* provides multi-client connections and the Creating-client socket task thread in *MUSiiCTCPClient* can create multiple client sockets in a single *MUSiiCTCPClient* instance. The data communication of these classes is based on the *igtlMessageBase* data type, which is also the parent of all message data types in OpenIGTLink [45]. This allows our network classes to send or receive any kind of message type defined in OpenIGTLink. As shown in **Figure 3.8**, required (*MUSiiCCallbackInterface* or *MUSiiCVector*) and provided (*MUSiiCCallbackInterfaceControl* or *MUSiiCVectorSet*) interfaces allow us to send and receive data asynchronously with these network classes. In addition, executable modules with these network classes are capable of asynchronous bidirectional TCP/IP data communication with subsystems that use message types supported in OpenIGTLink, such as 3D Slicer or other tracking devices [47, 72, 74, 80].

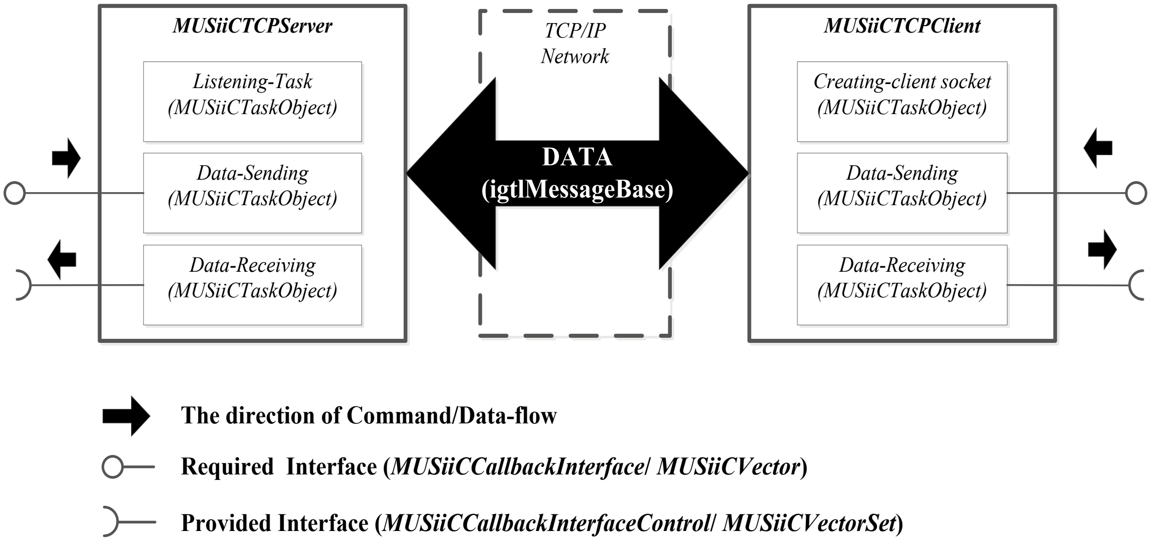

MUSiiCTCPServer
Listening-Task (MUSiiCTaskObject)
Data-Sending (MUSiiCTaskObject)
Data-Receiving (MUSiiCTaskObject)
TCP/IP Network
DATA (igtlMessageBase)
MUSiiCTCPClient
Creating-client socket (MUSiiCTaskObject)
Data-Sending (MUSiiCTaskObject)
Data-Receiving (MUSiiCTaskObject)
The direction of Command/Data-flow
Required Interface (MUSiiCCallbackInterface/ MUSiiCVector)
Provided Interface (MUSiiCCallbackInterfaceControl/ MUSiiCVectorSet)

Figure 3.8. Block diagram of *MUSiiCTCPServer* and *MUSiiCTCPClient*

Simple example code snippets of *MUSiiCTCPServer* are presented in **Snippet 3.8 and 3.9.** The *MUSiiCTCPServer* supports two methods to transfer data: explicit and implicit transmission based on the usage of an instance of *MUSiiCTCPServer* class and a connection of *MUSiiCTaskInterfaceAbstract* and *MUSiiCTCPServer,* respectively. **Snippet 3.8** describes the explicit method. The function *PutIGTLMessage()* of *MUSiiCTCPServer* will broadcast a given OpenIGTLink compatible message (e.g. USMessage) to multiple connected clients.

Snippet 3.8. An example source code about how to use the *MUSiiCTCPServer*.

```
using namespace igtl;
/// Create an instance of MUSiiCTCPServer
MUSiiCTCPServer::Pointer pServer = MUSiiCTCPServer::New();

/// Open a TCP/IP server socket with a user-defined port number
pServer->CreateServer(18994);

/// Collecting or processing data
USMessage::Pointer us = USMessage::New();
  //// … Collecting/processing US data

/// Send US data to connected clients
pServer->PutIGTLMessage(static_cast<MessageBase::Pointer>(us));

/// Close TCP/IP server socket
pServer->CloseServer();
```

The implicit method is represented in **Snippet 3.9.** The instance of *MUSiiCTaskInterfaceAbstract* (Task1) and the instance of *MUSiiCTCPServer* (pServer) are connected with the function *AddPostMUSiiCTaskInterface*() (see **Figure 3.5** and **Snippet 3.6**). Then, a data stream generated in Task1 transfers implicitly to the pointer of pServer for sending it to the TCP/IP network.

Snippet 3.9. An example source code about the usage of *MUSiiCTCPServer* with an instance of *MUSiiCTaskInterfaceAbstract.*

```
using namespace igtl;
typedef MessageBase::Pointer  IGTLMsg;

/// Create an instance of MUSiiCTaskInterfaceAbstract
MUSiiCTaskInterfaceAbstract<IGTLMsg, IGTLMsg>::Pointer Task1
                              = MUSiiCTaskInterfaceAbstract<IGTLMsg, IGTLMsg>::New();

/// Create an instance of MUSiiCTCPServer
MUSiiCTCPServer::Pointer pServer = MUSiiCTCPServer::New();

/// Connect the Task1 and the instance of MUSiiCTCPServer
Task1->AddPostMUSiiCTaskInterface<IGTLMsg, IGTLMsg>(pServer);

/// Open a TCP/IP server socket with a user-defined port number
pServer->CreateServer(18994);

/// Processing data in Task1 and sending it through TCP/IP network
Task1->RunTask();

/// Stop the data processing of Task1
Task1->StopTask();

/// Close the TCP/IP server socket
pServer->CloseServer();
```

Similar to *MUSiiCTCPServer*, *MUSiiCTCPClient* provides two data transmission modes: an implicit data transfer with establishing a connection between *MUSiiCTaskInterfaceAbstract* and *MUSiiCTCPClient* (see **Snippet 3.6** and **3.10**) and an explicit transmission using the callback interface of *MUSiiCTCPClient* (see **Snippet 3.7** and **3.11)**. **Snippet 3.10** shows an example source code about the implicit data transfer of

MUSiiCTCPClient. The instance of *MUSiiCTaskInterfaceAbstract* (Task1) is linked with the instance of *MUSiiCTCPClient* (pClient) using the function *AddPostMUSiiCTaskInterface()* (see line 12). And, the pClient is connected to a server socket with a given ip address and port (see line 18). Then, Task1 can get a received data from the pClient implicitly.

Snippet 3.10. An example source code about the implicit data transfer of *MUSiiCTCPClient*

```
 1  using namespace igtl;
 2  typedef MessageBase::Pointer  IGTLMsg;
 3
 4  /// Create an instance of MUSiiCTaskInterfaceAbstract
 5  MUSiiCTaskInterfaceAbstract<IGTLMsg, IGTLMsg>::Pointer Task1
 6                           = MUSiiCTaskInterfaceAbstract<IGTLMsg, IGTLMsg>::New();
 7
 8  /// Create an instance of MUSiiCTCPClient
 9  MUSiiCTCPClient::Pointer pClient = MUSiiCTCPClient::New();
10
11  /// Connect the pClient and the Task1
12  pClient ->AddPostMUSiiCTaskInterface<IGTLMsg, IGTLMsg>(Task1);
13
14  /// Run the task-thread of Task1
15  Task1->RunTask();
16
17  /// Connect to a sever socket with a given ip address and port
18  pClient ->ConnectToHost('127.0.0.1', 18994);
19
20  /// Stop the data processing of Task1
21  Task1->StopTask();
22
23  /// Close the TCP/IP client socket
24  pClient ->CloseAllClients();
```

A sample source code about the explicit data transmission of *MUSiiCTCPClient* is represented in Snippet 3.11. This method is based on the callback interface of *MUSiiCTCPClient*. First, a user-defined callback function following the function type of *MUSiiCCallbackFunction* (see **Snippet 3.5** and **Snippet 3.11:** line 4 - 11) is declared to handle a received data. And, the callback function (RxData) is added to the pClient with the function *AddExternalGlobalOutputCallbackFunction()*. When the pClient is connected to a server socket, the task thread of the pClient calls the callback function (RxData) and transmits a received message of OpenIGTLink or *OpenIGTLinkMUSiiC* to the RxData function with the parameter of a MessageBase's smart pointer (msg) (see **Snippet 3.11:** line 4 – 11).

Snippet 3.11. A sample source code about the explicit data transmission of *MUSiiCTCPClient*

```
 1  using namespace igtl;
 2
 3  /// Declare a user-defined callback function
 4  int RxData (int numOfRun = 0, int taskInfo = 0,
 5                  void* ptr = NULL, MessageBase::Pointer msg = NULL,
 6                  void* data2 = NULL, void* data3 = NULL)
 7  {
 8          /// Processing a received data (msg)
 9           /// .......
10           return 1;
11  }
12
13  /// Create an instance of MUSiiCTCPClient
14  MUSiiCTCPClient::Pointer pClient = MUSiiCTCPClient::New();
15
16  /// Add the callback function to the pClient
```

```
pClient->AddExternalGlobalOutputCallbackFunction(RxData);

/// Connect to a sever socket with a given ip address and port
pClient ->ConnectToHost('127.0.0.1', 18994);

/// Close the TCP/IP client socket
pClient ->CloseAllClients();
```

3.2.4. Real-time multiple Data Synchronization

US IGT systems can be composed of many subsystems including a robotic system, tracking devices, and medical imaging systems. Real-time synchronization between multiple data sources is therefore necessary to integrate the US system with other IGT subsystems.

We built a specific task class, *MUSiiCSync*, to solve this requirement. Our synchronization method is closest data combining based on the timestamp of each data. The block diagram of *MUSiiCSync* is shown in **Figure 3.9** (**a**). There are two task objects, Data-Collect and Data-Combine, in this class. The Data-Collect task thread receives multiple data from different data sources through the TCP/IP network and generates a *MUSiiCSyncDataSet* instance designed to contain a reference data and multiple data of other types during the time of data collection, **Tc** (see **Figure 3.9 (b)** and **Snippet 3.12**). Generally, **Tc** is two divided by the frame rate of the reference data in seconds. The Data-Combine task thread finds the data from each data group with the closest timestamp to the timestamp of the reference data. If *TrackingDataMessage* and *ImageMessage* data of OpenIGTLink are present in the instance of *MUSiiCSyncData*, this task thread updates the tracking information of *ImageMessage* with the information from *TrackingDataMessage*. At this time, we can apply additional information, such as the calibration information of the US transducer. The two task objects (Data-Collect and Data-Combine) share a concurrent data interface (*MUSiiCVector*) with the data type of *MUSiiCSyncDataSet* for data communication between those two independent task threads (see **Snippet 3.12**).

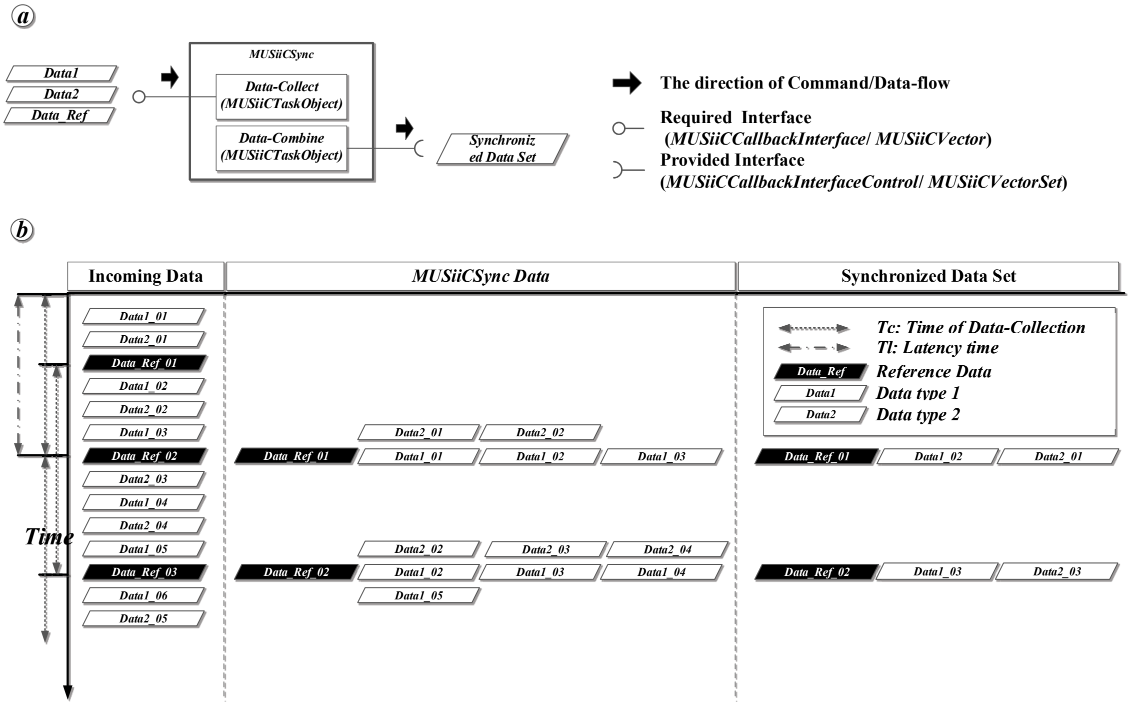


Figure 3.9. (a)Block diagram and (b) Timeline of *MUSiiCSync*

There is a latency time (**Tl**) in *MUSiiCSync* to generate the first set of synchronized data. The set of synchronized data from *MUSiiCSync* will be delayed by at most **Tc**. In general, US data is the reference data in *MUSiiCSync*, and the frame rate is 30 frames per second. In this case, **Tc** will be 60 ms and **Tl** will be less than 60 ms. We feel that this delay is acceptable in US IGT systems. The set of synchronized data from

MUSiiCSync can be sent to another task class through an instance of *MUSiiC-CallbackInterface* or *MUSiiCVector*. In this case, the data can be saved to the local hard disk using the *MUSiiCFileIO* class or sent to the TCP/IP network with *MUSiiC-TCPClient*.

Snippet 3.12. The concise source code of *MUSiiCSync*

```
using namespace igtl;
/// Declare the data structure of MUSiiCSyncDataelement
class MUSiiCSyncDataElement {
    …….
    std::map<double, Message::Pointer>  m_SyncDataElement;
}

/// Declare the data structure of MUSiiCSyncData
class MUSiiCSyncData {
    …….
    std::map<double, MUSiiCSyncDataElement::Pointer>  m_SyncData;
}

/// Declare the data structure of MUSiiCSyncDataSet
class MUSiiCSyncDataSet {
   …….
   MUSiiCSyncData::Pointer  m_pSyncDataSet;
   MessageBase::Pointer     m_RefMsg;

/// Declare the class of MUSiiCSync
class MUSiiCSync{
   …….
   /// Implement the task function of SyncDataCollect() for the task object of Data-Collect
   int SyncDataCollect{
       /// Receiving multiple data streams from MUSiiCTCPClient or other
```

```
            MUSiiCTaskInterfaceAbstract instances
         /// Generate an instance of MUSiiCSyncDataSet with the received data
          MUSiiCSyncDataSet::Pointer syncDataSet = MUSiiCSyncDataSet::New();
           ……
          /// Send it to the function of SyncDataCombine
          m_pPostCollectorDataIF->concurrent_pushback(syncDataSet);
      }

    /// Implement the task function of SyncDataCombine () for the task object of Data-
         Combine
   int SyncDataCombine{
        /// Get an instance of MUSiiCSyncDataSet from the task object of Data-Collect
        MUSiiCSyncDataSet::Pointer syncDataSet;
        m_pPostCollectorDataIF->concurrent_wait_get_pop_fron(syncDataSet);
       /// Find closest IGTL Messages related to a reference msg
       ……
       /// Combine and synchronize multiple closest IGTL Messages
       …...
      /// Transfer a synchronized data set to MUSiiCTCPClient or other
          MUSiiCTaskInterfaceAbstract classes
      PostProcessMUSiiCSync();
   }
    ………
   MUSiiCVector<MUSiiCSyncDataSet::Pointer>::Pointer   pPostCollectorDataIF;
 }
```

3.3. *MUSiiC* Modules

Our *MUSiiC Modules* are composed of several specialized executable modules. Each module has its own task, and they communicate via TCP/IP networking. Each module has its own TCP/IP server socket, client socket, or both. Furthermore, communication in the *MUSiiC Toolkit* is bidirectional. In this case, a server socket provides data to other client modules. Frequently, a server socket will provide the same data to multiple client modules, e.g. for navigated laparoscopic ultrasound systems [30]. For this reason, the functionality for multiple-clients to connect to a single server socket is necessary. As we mentioned, this functionality is implemented in the *MUiiCTCPServer* class.

3.3.1. RF-Server

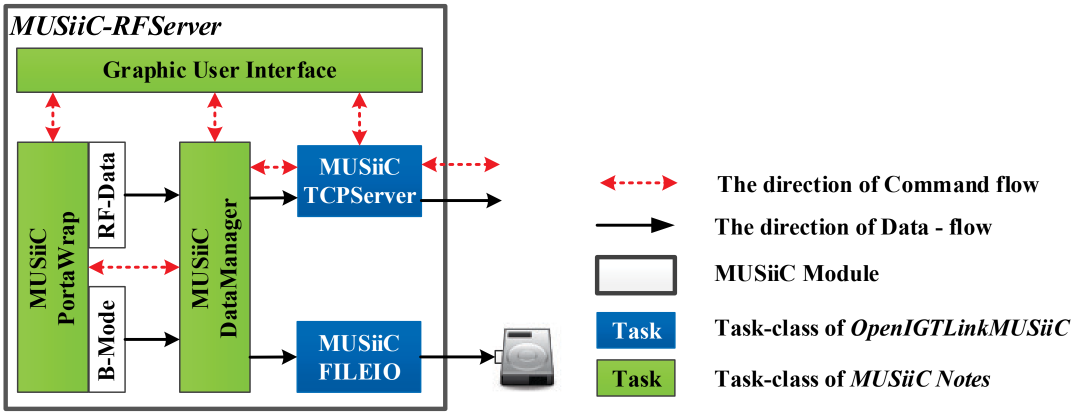


Figure 3.10. Block diagram of RF-Server

One elementary function of ultrasound research is the interface to the ultrasound machine. Although some manufacturer’s systems allow access to US RF data directly, it is not offered in real-time, but rather provided offline. Moreover, US RF data and B-mode image streams are needed for real-time monitoring of acoustic ablation therapy. Therefore, we developed a software module, *RF-Server*, for collecting US RF data and B-Mode image stream with Porta SDK (Ultrasonix) [54]. This module allows high-performance ultrasound data collection with Field-programmable gate array (FPGA)-based filtering (three-band finite impulse response (FIR) filter for depth-dependent image quality improvement) and real-time B-mode image generation simultaneously (refer to **Figure 3.10**). As the diagram indicates, *RF-Server* has two sets of *MUSiiCVector* and *MUSiiCTCPServer* classes of *OpenIGTLinkMUSiiC* [47, 72] to send ultrasound RF data and B-Mode image streams to other client programs through TCP/IP network. Also, we are upgrading the *RF-Server* to support bidirectional communication between other *MUSiiC Modules* as **Figure 3.10** indicates.

3.3.2. Tracker-Server

For any US-IGT system requiring pose information for US transducers, surgical instruments, or targeted-tissue in real-time, an interface with tracking devices is necessary. Although there are several different tracking methods, such as optical/infrared (IR), electromagnetic (EM), or visual camera image-based tracking, the basic result of tracking is a pose associated with the tracked marker or sensor. Moreover, several messages from the OpenIGTLink library (e.g. *TrackingDataMessage*, *ImageMessage*, and *PositionMessage*) [45] support the protocol for communicating tracking information. **Figure 3.11** represents the block diagram of Tracker Server. *MUSiiCEMTrackerWrap* class is customized to collect tracking information from electromagnetic tracking devices in real-time. If there are connected client modules, the data will be sent to the modules simultaneously.

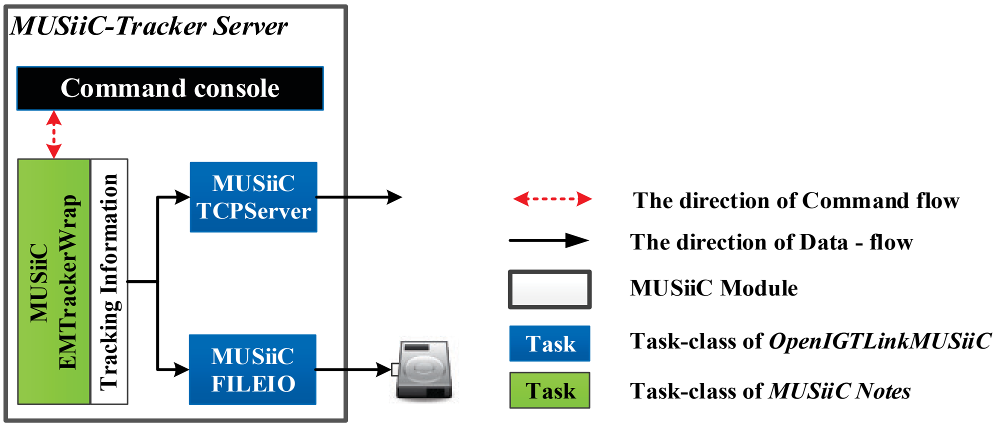


Figure 3.11. Block diagram of *Tracker-Server*

3.3.3. B-Mode Module

The flowchart of *B-Mode* module appears in **Figure 3.12**. US RF data is received by *MUSiiCTCPClient* via TCP/IP network. This module also has *MUSiiCTCPClient*, *MUSiiCVector*, and *MUSiiCTCPServer* classes.

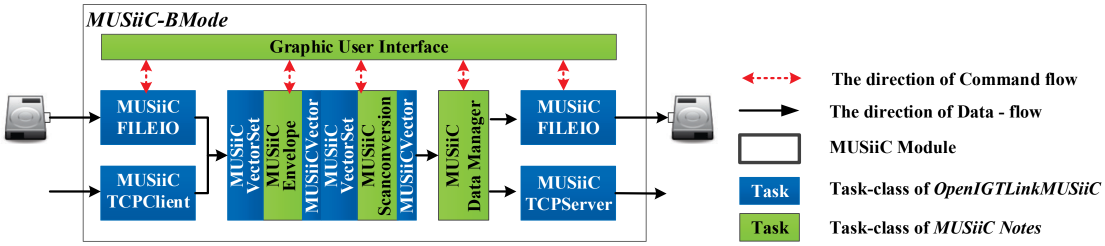


Figure 3.12. Block diagram of *B-Mode* module

Although the *B-Mode* and other *MUSiiC* modules are on the same workstation, TCP/IP network communication is used between these two modules for Inter-Process Communication (IPC) [87, 88]. Afterwards, two active classes, *MUSiiCEnvelope* and *MUSiiCScanConversion*, get data from *MUSiiCVector* and generate a B-Mode image. Those two classes (*MUSiiCEnvelope* and *MUSiiCScanConversion*) are connected with concurrent data interfaces based on instances of *MUSiiCVector* and *MUSiiCVectorSet* (see **Figure 3.5** and **Snippet 3.6**). The output B-Mode image will be packed into an OpenIGTLink *ImageMessage* for communicating with existing image-guidance systems such as *3D Slicer* [44, 80], and then sent to another client program to be displayed.

3.3.4. Real-time Elasticity module[1]

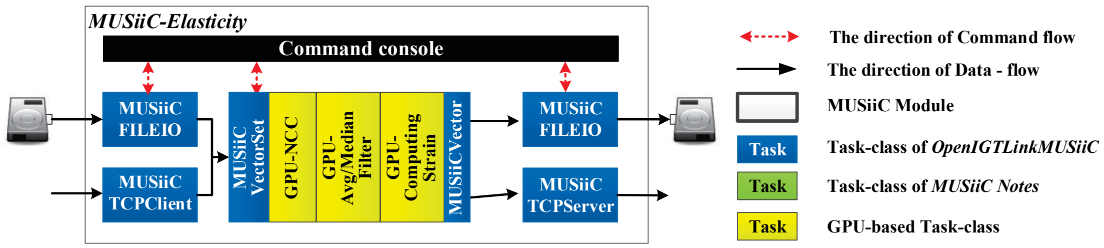


Figure 3.13. Block diagram of NCC-EI module

Ultrasound elasticity image (EI) represents the stiffness or strain images of soft tissue. Appropriate algorithms include windowed pattern matching using normalized cross-correlation (NCC) [22, 63], dynamic programming (DP) on sample intensities, and analytic methods on DP results (AM/DP) to achieve sub-sample displacement estimation [99]. However, these techniques are computationally expensive. Thus, we implemented our elasticity imaging algorithms on a GPU-EI-Engine based on Graphics processing unit (GPU) programming technique [22] as shown in **Figure 3.13**. The *Elasticity Image*

[1] Nishikant Deshmukh implemented our elasticity imaging algorithms (NCC, DP and AM/DP) on Graphics processing unit (GPU) programming technique [22].

module receives an US RF data stream from *RF-Server* via TCP/IP network, generates elastography streams from consecutive RF data frames, and then sends out the results over TCP/IP.

3.3.5. Quality-based Frame chooser[2]

For some image streams, it is possible to associate quality metrics with individual image frames. For example, in the real-time elasticity module, each frame (*EIMessage*) carries an average cross-correlation (ACC) value in its header [47, 94], denoting the mean quality of the matched pattern across the frame.

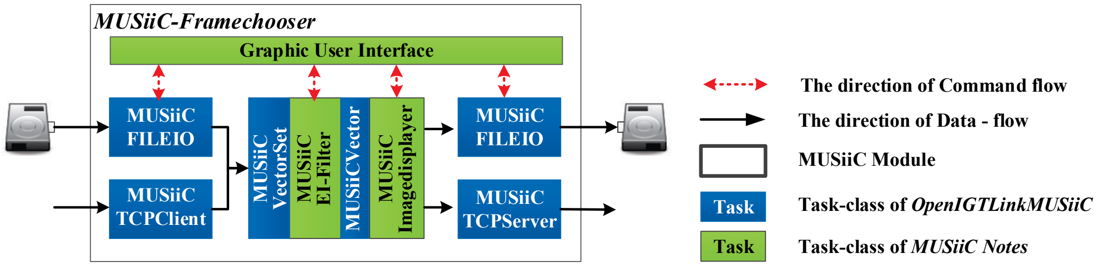


Figure 3.14. Block diagram of *Framechooser* module

[2] Dr. Philipp J. Stolka developed the *Framechooser* module for the project of 3D Elastography Guided Laparoscopic Surgery [24,94].

The quality-based Frame chooser module (**Figure 3.14**) allows the filtering of incoming *EIMessage* [47, 72] streams, either synchronously based on their associated quality value or a user-defined quality threshold before immediate re-sending.

3.3.6. ImageViewer Module

As stated in the previous section, photoacoustic/ultrasound B-Mode image streams are packed in an OpenIGTLink *ImageMessage* [45]. While *3D-Slicer* can display an *ImageMessage* [80], it does not maintain a continuous network connection. Thus, we built the *ImageViewer* module to display ultrasound B-mode, elastography or photoacoustic image streams in real-time. Using *MUSiiCTCPClient* and *MUSiiCVector* classes, this module can receive consecutive images packed with ImageMessage or its compatible message (USMessage or EIMessage) stream as in **Figure 3.15**.

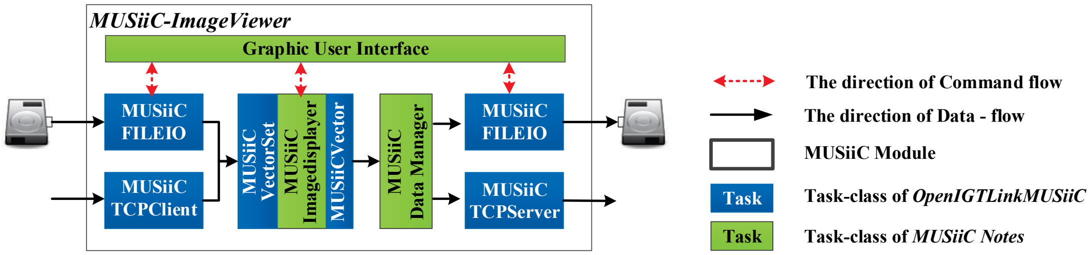


Figure 3.15. Block diagram of *ImageViewer* module

Moreover, a rendering function of *ImageMessage* and basic image processing functions, such as zooming in or out, and adjusting contrast and brightness of the image, are implemented in the *MUSiiCImageDisplay* class.

3.3.7. Data synchronization module (*MUSiiC-Sync*)

The main function of *MUSiiC-Sync* is to receive multiple types of OpenIGTLink and *OpenIGTLinkMUSiiC* messages from various data acquisition modules (e.g., *MUSiiCRFServer* and *MUSiiCTrackerServer*) and to synchronize these messages using their respective timestamps. This module also loads a precomputed US calibration for the transducer in use from the local hard disk, and applies this calibration to received *TrackingDataMessages* [45] to get the pose information of the 2D RF frames.

Figure 3.16 represents the block diagram for the *MUSiiC-Sync* module. The *MUSiiCTCPClient* class provides TCP/IP connections to the TCP/IP server sockets of multiple data acquisition modules. For example, the *MUSiiC-Sync* module can receive data streams of *USMessages* and *TrackingDataMessages* from *MUSiiCRFServer* and *MUSiiCTrackerServer*, respectively. The *MUSiiCSync* task-class of this module has two specific task-threads, *DataCollect* and *DataCombine*, to improve the efficiency of data synchronization. The *DataCollect* task-thread generates a *MUSiiCSyncData* instance with incoming reference data and other additional data. The *DataCombine* task-thread finds the additional data with the closest timestamp to the reference data timestamp, and combines them. In this step, these task-threads generate a spatially-tracked 2D RF frame by updating the tracking information of the *USMessage* (2D frame) with the selected *TrackingDataMessage* (pose information of EM tracked sensor) and precomputed calibration information [93] of the US transducer.

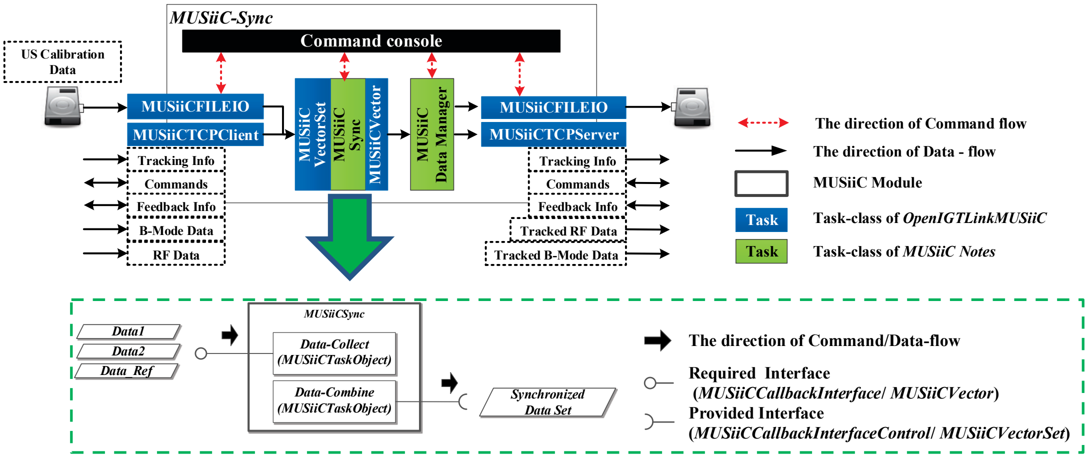


Figure 3.16. Block diagram of *MUSiiC-Sync* module

3.4. Conclusion

A real-time medical ultrasound imaging and interventional component framework for advanced interventional US imaging research is introduced in this chapter. The software component framework was designed and implemented with network-distributed modular properties to improve software framework flexibility, an open interface to establish communication with existing IGT systems or external software, and real-time communication and processing to provide real-time visual feedback information based on a US image.

To fulfill the design specifications of an open interface and network-distributed modularity, the *OpenIGTLinkMUSiiC* software library provides new OpenIGTLink compatible message types: *USMessage* and *EIMessage* for advanced interventional US imaging research, and *GenMessage*, *ArgMessage*, and *FileMessage* for communicating text-based control parameters or arguments to other software modules. In addition, OpenIGTLinkMUSiiC supports thread-safe concurrent data interfaces (*MUSiiCVector* and *MUSiiCVectorSet*), and customized multi-threaded task classes (*MUSiiCTaskObject*, *MUSiiCTaskAbstract*, and *MUSiiCTaskInterfaceAbstract*) for real-time computation. Moreover, custom-built callback function interfaces (*MUSiiCCallbackInterface* and *MUSiiCCallbackInterfaceControl*), and network classes (*MUSiiCTCPServer* and *MUSiiCTCPClient*) are provided by *OpenIGTLinkMUSiiC* to support real-time bidirectional communication of network-distributed modules and multi-threaded task classes.

MUSiiC modules provides a collection of executable and communicating software modules for interventional US imaging research, which enables us to disassemble the monolithic processing structure embedded in existing US systems into a flexible advanced interventional US image processing pipeline. In addition, the US image processing pipeline can provide low-level or intermediate US data for a vendor independent US imaging research.

Recapitulation of thesis contribution

- Designing and building the *OpenIGTLinkMUSiiC* library to provide the properties of network distributed modular architecture, real-time computation, and open-interface in our software framework.
- Designing and developing *MUSiiC* modules based on the *OpenIGTLinkMUSiiC* library, accomplished by implementing the US reconstruction process. A US data processing pipeline by connecting several *MUSiiC* modules enables the following:
    - A vendor independent and customized US imaging research by providing low-level and intermediate US data.
    - Easy reconfiguration of the US imaging system for several specific US imaging and interventional US imaging research.
    - Easy integration with existing OpenIGTLink compatible IGT systems.
- Designing and implementing the *MUSiiC-Sync* module to receive multiple messages of OpenIGTLink and *OpenIGTLinkMUSiiC* from different data sources and to synchronize the received messages with its respective time-stamps. This software module enables us to combine multiple data such as US image, tracking information or other sensor data in real-time for interventional US imaging research.

Chapter 4:

Use Cases of Interventional Ultrasound Elastography

4.1. Ultrasound Imaging Software Framework for Real-Time Thermal Monitoring of Acoustic Ablation Therapy

This work was accomplished with the aid of the following collaborators and was submitted under the following citation:

Hyun Jae Kang, Nishikant Deshmukh, Philipp J. Stolka, E. Clif Burdette, and Emad M. Boctor, "Ultrasound imaging software framework for real-time monitoring of acoustic ablation therapy," in SPIE Medical Imaging 2012, San Diego, CA/USA, 2012, p. 83201E.

Abstract

In this section, a new software framework for pre- and intraoperative real time monitoring of ablative therapy is presented. The software framework provides US B-mode images and elasticity images simultaneously in real-time. The position of the ablator's tip and a region of heated tissue are monitored in the B-Mode images, because the images represent tissue morphology. Furthermore, US elasticity images are used for finding the boundary and region of the tumor before ablation, and monitoring thermal dose in tissue during ablation. By providing B-Mode images and elasticity images concurrently, reliable information for monitoring thermal therapy can be offered.

4.1.1. Motivation

The concern with interstitial ablative therapy for the treatment of hepatic tumors has been growing. There are several methods in this therapy: chemical ablation, cryoablation [100], radiation therapy [101], and heat ablative therapies using different energy sources such as radiofrequency (RF) [102], laser, microwave [103] or focused ultrasound [104]. In spite of advances in these therapies, there are several technical challenges due to tissue deformation and target motion: localization of the tumor, monitoring of the ablator's tip and thermal dose in heated tissue. Moreover, current ablative systems are highly dependent on the operator's skill.

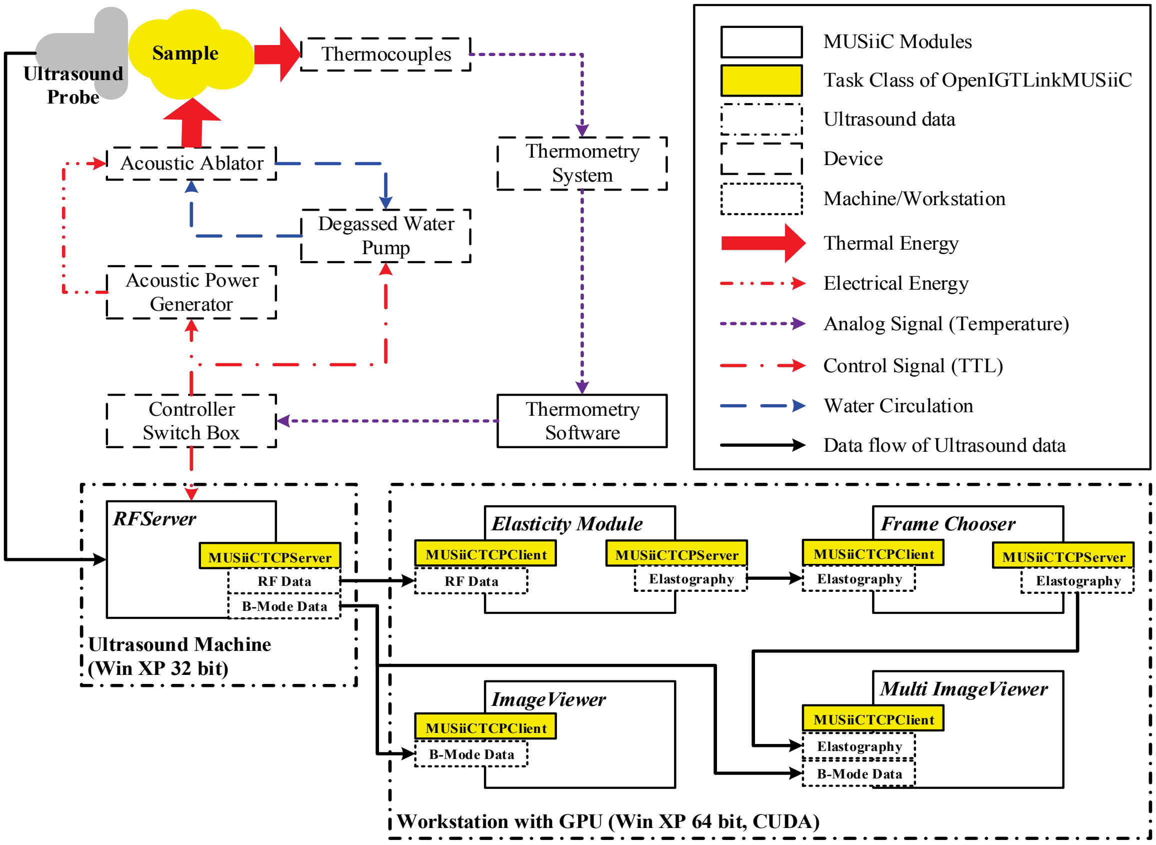


Figure 4.1. Block diagram of monitoring system for acoustic ablative therapy.

In previous work, ACUSITT, a steerable acoustic ablator for accurate targeting of its tip to the tumor area, has been developed [19]. However, real-time monitoring techniques for providing image feedback of the ablation tip positioning and the thermal dose deposited in the tissue by heating are still needed. In this research, we report a new ultrasound software framework to help provide a real time monitoring system for acoustic ablation therapy. **Figure 4.1** shows the block diagram of this new real-time monitoring system. The ultrasound imaging software framework provides a real-time ultrasound Brightness Mode (B-Mode) image and elastography simultaneously. B-Mode images represent tissue-morphology by collecting and displaying the intensity information of reflected ultrasound waves. Therefore this imaging modality is mainly used for diagnosis and monitoring. On a B-Mode image, the location of the ablation tip can be tracked in real-time by a needle detection algorithm. On the other hand, ultrasound elastography represents tissue stiffness and its Young's modulus [58]. Since a tumor's abnormal tissue has different stiffness compared to the normal surrounding tissue, its area and boundary can be recognized in an ultrasound elasticity image. Moreover, ultrasound thermal imaging, a heat-induced echo-strain image, is very similar to ultrasound elastography. During the ablation, the speed of sound in heated tissue is higher than non-heated tissue. By tracking shifts in the backscattered RF signals, a heat-induced echo-strain image is generated using a similar time-delay estimation algorithm as ultrasound elastography. In a heat-induced echo-strain image, low and high strains correspond to constant and higher temperature, respectively. Thus, thermal dose in heated tissue can be estimated by using an ultrasound elasticity image.

4.1.2. Methods

As seen in **Figure 4.1**, the software framework is composed of several specialized executable modules: *MUSiiC-RFServer*, *MUSiiC-Elasticity* [1] , *MUSiiC-ImageViewer*, *MUSiiC-MultiImageViewer* and *MUSiiC-Framechooser* [2] modules [41]. Ultrasound data such as RF data, elastography, and B-Mode images are communicated between software modules with an extended version of *OpenIGTLink* [45], called *OpenIGTLinkMUSiiC* [47]. For TCP/IP networking, all the modules have network classes of *OpenIGTLinkMUSiiC* that support multiple-client connections via TCP/IP network simultaneously [47, 72].

The ultrasound software framework and acoustic ablator system are synchronized by a Transistor-Transistor Logic (TTL) control signal of the controller switch box (see **Figure 4.1**). **Figure 4.2** shows the time-sequence diagram of ultrasound data acquisition and processing the data according to the TTL signal. Before ablation, the position of the

[1] Nishikant Deshmukh implemented real-time US elasticity module with the elasticity imaging algorithms based normalized cross correlation and Graphics processing unit (GPU) programming technique [22]

[2] Dr. Philipp J. Stolka developed the *Framechooser* module for the project of 3D Elastography Guided Laparoscopic Surgery [24,94].

ablator's tool tip is tracked on B-Mode image streams and the boundary and region of the tumor are identified on ultrasound elasticity images in real-time.

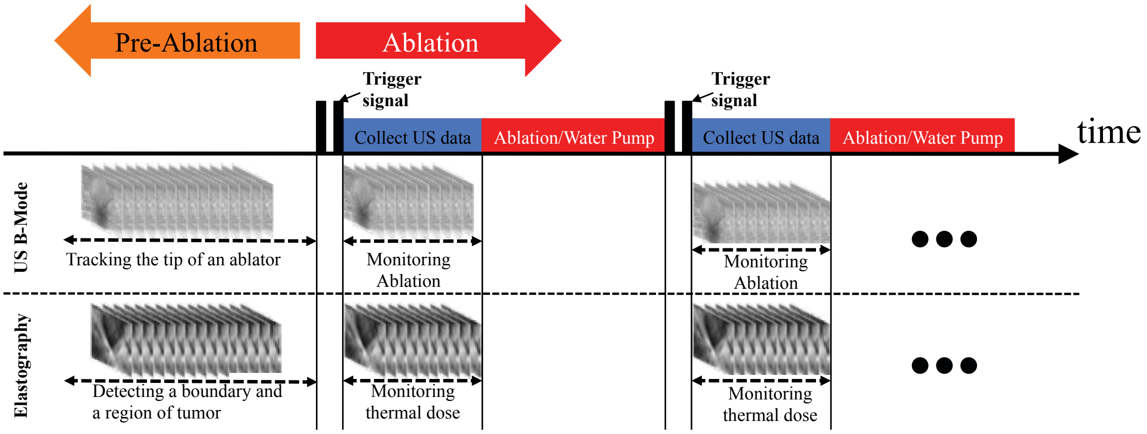


Figure 4.2. Time-sequence diagram for triggered monitoring of acoustic ablation.

During acoustic ablative therapy, the ultrasound software framework is externally triggered by a TTL control signal from our controller switch box for anti-synchronization (see **Figure 4.1**, **4.2**). An ultrasound B-Mode and an elastography stream are generated in the software framework simultaneously as N-frames of RF data are being collected in *MUSiiC-RFServer*. The *MUSiiC-Elasticity* module receives the RF data streams from *MUSiiC-RFServer* with the method of the explicit data transmission of *MUSiiCTCPClient* (see **Snippets 3.11**) as shown in **Snippets 4.1** and generates an ultrasound elastogram stream with the GPU-EI-Engine (see **Chapter 3**) in real time. Afterwards, this module broadcasts the computed ultrasound elastography images to the

TCP/IP network with an instance of *MUSiiCTCPServer* as *EIMessage* (see **Snippets 4.2**)[3]. This data can be used to monitor thermal dose in the heated tissue during acoustic ablation therapy [41].

Snippet 4.1. Receiving US RF data in *MUSiiC-Elasticity* module using a callback function of *MUSiiCTCPClient*

```
/// User defined callback function for the method of the explicit data transmission
    of MUSiiCTCPClient
int    ReceiveMsg (int numOfRun = 0, int taskInfo = 0, void* ptr = NULL,
        igtl::MessageBase::Pointer data1 = NULL, void* data2 = NULL,
       void* data3 = NULL)
  {
       /// Check a received data
       if(data1.IsNull())
          return 0;

       // Retrieve USMessage
       igtl::USMessage::Pointer USMsg =
       igtl::RetrieveMessage<igtl::USMessage::Pointer, igtl::USMessage>(data1);

       if(USMsg.IsNull())
          return 0;

       /// Get the size of the received US RF data
       USMsg->GetDimensions(size);
       USMsg->GetSpacing (spacing);

       /// Get the timestamp of received US RF data
       igtl::TimeStamp::Pointer ts = igtl::TimeStamp::New();
       USMsg->GetTimeStamp(ts);
```

[3] Thanks to Nishikant Deshmukh for providing his source code.

```

        /// Copy US RF data to a cuda compatible data
        char *tempdata;
        cuda_malloc_host ((void **)&tempdata, USMsg->GetImageSize());
        memcpy (tempdata, USMsg->GetScalarPointer(), USMsg->GetImageSize());

        /// Build a data structure for computing US elstogram
        new_data                        = new data_frame_queue;
        new_data->data                  = tempdata;
        new_data->height                = size [1];
        new_data->width                 = size [0];
        new_data->number_frames = size[2];
        new_data->itime                 = ts->GetTimeStamp();

        new_data->spacing[0]        = spacing[0];
        new_data->spacing[1]        = spacing[1];
        new_data->spacing[2]        = spacing[2];

        // Put additional US information in the data structure
        // Get transmit frequency of RF data
        new_data->fhr.txf             = USMsg.GetTx();
        // Get sampling frequency of RF data
        new_data->fhr.sf              = USMsg.GetTx();
        // Get a US probe ID of RF data
        new_data->fhr.probe         = USMsg.GetTx();

        // Put the built data structure to a thread of computing US elastogram
        in_queue.push (new_data);
        ....
    }
```

Snippet 4.2. Sending generated US elastogram data in *MUSiiC-Elasticity* module using an instance of *MUSiiCTCPServer*

```
int   int ncc_thread (ncc_parameters *ncc_p)
{
      / // Computing US elastogram with NCC method
      .....

      /// Packing a computed US elastography as EIMessage
```

```
    igtl::EIMessage::Pointer EIMsg = igtl::EIMessage::New();

    EIMsg->SetScalarTypeToUint8();
    EIMsg->SetDimensions(scale_width, scale_height, 1);
    EIMsg->SetDeviceType ("IMAGE");
    EIMsg->SetDeviceName ("EI_NCC");
    EIMsg->SetDimensions(scale_width, scale_height, 1);
    EIMsg->SetSpacing (ncc_p->spacing[0], ncc_p->spacing[1],
                       ncc_p->spacing[2]);
    EIMsg->AllocateScalars();
    memcpy (EIMsg->GetScalarPointer(), avg_scaled_out, scale_size);
    EIMsg->Pack();

    /// Broadcast the EIMessage to TCP/IP network
    ncc_p->pServer->PutIGTLMessage((igtl::MessageBase::Pointer)ImgMsg);

}
```

Figure 4.3 shows the system setup of our *ex-vivo* tissue experiment for the real-time monitoring of acoustic ablation therapy. In our experiment, we used turkey breast tissue as an *ex-vivo* phantom. For collecting ultrasound RF data and B-mode images, our system setup used a *SONIX-CEP* machine (UltraSonix Co. Vancouver, Canada) equipped with a 3D linear array transducer (model 4DL14-5/38). For real-time monitoring of thermal dose in a tissue, our controller switch box is connected to the ultrasound machine (*SONIX-CEP*) and the *Acoustic power generator* (Acoustic MedSystem Inc. Savoy, USA) with Bayonet Neil-Concelman (BNC) cables to provide the anti-synchronization that is based on the time-sequence (**Figure 4.2**) with TTL signals.

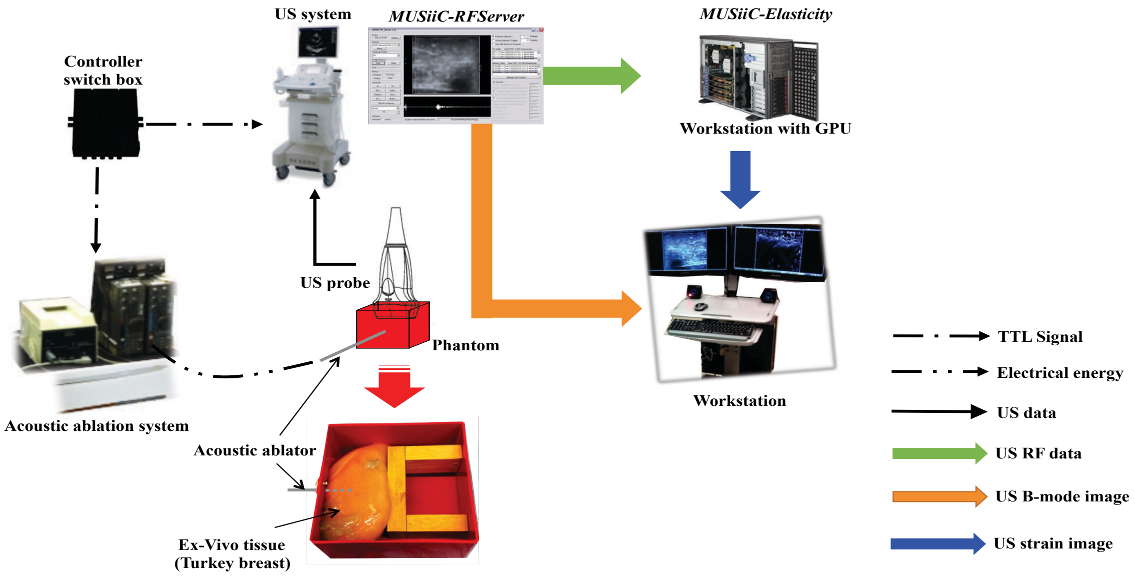


Figure 4.3. System setup for real-time monitoring of acoustic ablation Therapy.

4.1.3. Results

In our software framework, *MUSiiC-RFServer* collects ultrasound RF data and B-Mode images and provides the data to other client modules (*MUSiiC-Elasticity*, *MUSiiC-ImageViewer*, *MUSiiC-MultiImageViewer* and *MUSiiC-Framechooser* modules) in real time [41]. The performance of this program is important for real-time monitoring of the acoustic ablation system. To verify the data-collecting performance of *MUSiiC-RFServer*, we collected RF data and B-Mode image streams at different depths and with different sampling frequencies. Also, we tested this module with/without network connection. The result of our test is in **Table 4.1**. As shown in the table, the data-collecting performance depends on the depth of ultrasound data or image. But, there is little difference in the performance of *MUSiiC-RFServer* between different sampling frequencies. Moreover *MUSiiC-RFServer* has similar frame rates per second with/

without TCP/IP network connection, because it has an independent network class [47, 72].

Table 4.1. The data-collecting performance of *RF-Server* (fps)

Depth (cm)	Sampling Frequency 20 MHz			Sampling Frequency 40 MHz		
	RF	B	B+N	RF	B	B+N
3	70.922	70.922	70.922	71.429	70.922	64.103
4	58.140	58.480	58.140	58.140	58.140	58.140
5	49.020	49.261	49.261	49.261	49.261	49.020
6	42.735	42.553	42.735	42.735	42.553	42.553

- RF Collecting RF Data
- B Collecting B-Mode Image
- B+N Collecting B-Mode Image with TCP/IP network connection

Figure 4.4 shows gross a pathology image (a), an ultrasound B-Mode Image (b) and an ultrasound elasticity image (c). As seen in the histopathological image, we can see two lesions (L1 and L2) on both the ultrasound B-mode image and elastography. Moreover, strain results of ultrasound elastography correspond well with the gross pathology image.

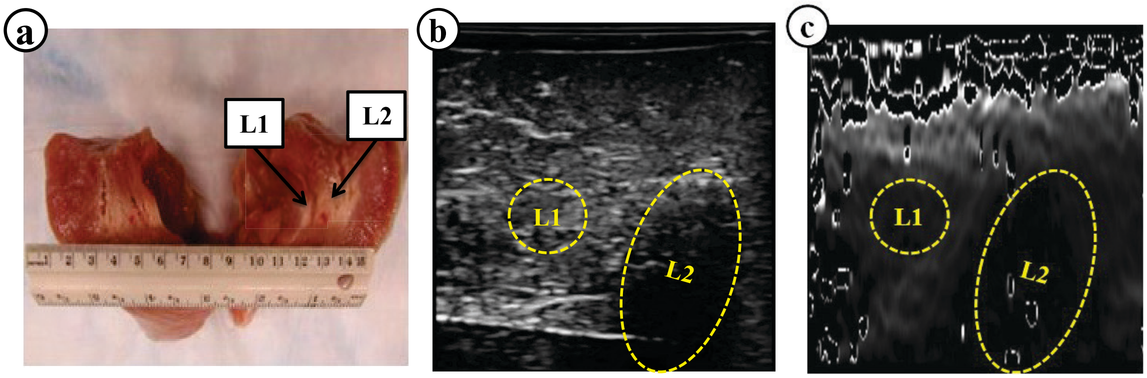


Figure 4.4. Intraoperative thermal monitoring of acoustic ablation therapy. (a) Gross pathology image, (b) Ultrasound B-mode image, (c) Ultrasound elasticity image.

4.1.4. Conclusion

We have developed a software framework which is composed of specialized executable modules: *MUSiiC-RFServer*, *MUSiiC-Elasticity*, *MUSiiC-ImageViewer*, *MUSiiC-MultiImageViewer* and *MUSiiC-Framechooser* modules. The software framework is based on a network distributed system and multithreaded architecture. That is, each module of the software framework has its own independent task-thread for improving its performance with multithreaded programming. Using our software framework, we can monitor the location of the ablator tip (with B-Mode images), tumor area (with US elastography), and distribution of thermal dose in tissue (with US elastography).

Our software framework is based on components of ultrasound data processing, therefore, we can expect that this software framework can be applied other ultrasound research [41, 47, 72].

4.2. System for Robot-Assisted Real-Time Laparoscopic Ultrasound Elastography

This work was achieved with the aid of the following collaborators and was submitted under the following citation:

Seth Billings [4], Nishikant Deshmukh, Hyun Jae Kang, Russell Taylor, and Emad M. Boctor, "System for robot-assisted real-time laparoscopic ultrasound elastography," in SPIE Medical Imaging 2012, San Diego, CA/USA, 2012, p. 83161W.

Abstract

In this section, we present a novel robot-assisted real-time US elastography system to generate consistent US strain images in real-time and to provide tactile information of the target tissue for minimally invasive robot-assisted interventions. Our system is composed of a daVinci surgical robot with an experimental software interface, a robotically articulated laparoscopic US probe, a clinical US system, and a real-time US elastography engine (*MUSiiC-Elasticity* module) implemented with the multithreading techniques of General-purpose computing on graphics processing units (GPGPU) [22, 23]. Generated US elastography and B-mode images are displayed in the daVinci console as picture-in-picture. To improve the image quality of US elastography images consistently, our proposed system minimizes its dependence on the operator by providing computer-assisted axial palpation motion. Preliminary tests of the system with an

[4] Seth Billings led this project and was the main contributor to integrate robot-assisted real-time laparoscopic ultrasound elastography [30].

elasticity phantom demonstrate the ability to differentiate simulated lesions of varied stiffness and clearly delineate lesion boundaries.

4.2.1. Motivation

Surgical robotic systems enhance minimally invasive surgical procedures by providing precise motion, high dexterity, and crisp stereo vision. However, current robotic operations cannot provide tactile information of the target tissue, such as stiffness, compared to open surgery.

Quasistatic elastography (strain imaging) proposed by Ophir et al. [58] estimates tissue stiffness based on the internal displacement of tissue caused by an internal or external force. Moreover, cancerous tissue is generally stiffer than surrounding normal tissue. Several research applications of US elastography are to detect malignant tissue in prostate, breast, pancreas, lymph nodes, and thyroid [105-109]. Consequently, US elastography has emerged as a very promising image guidance method for robot-assisted procedures that will further enhance the role of robots in interventions where precise knowledge of hidden anatomical features is crucial.

However, in quasistatic elastography, a force aligned with the US image's axial axis is required to generate high quality strain images consistently [110]. Since most US elastography algorithms assume there is only axial compression, the strain image's quality may degrade with compression from non-axial motion or probe rotation. Due to the variability in manual compression, image quality for free-hand US elastography is highly dependent on the operator's skill and experience.

To address the above limitations of surgical robotic systems and US strain imaging, a novel robot-assisted real-time US elastography system for minimally invasive robot-assisted interventions is proposed.

4.2.2. Methods

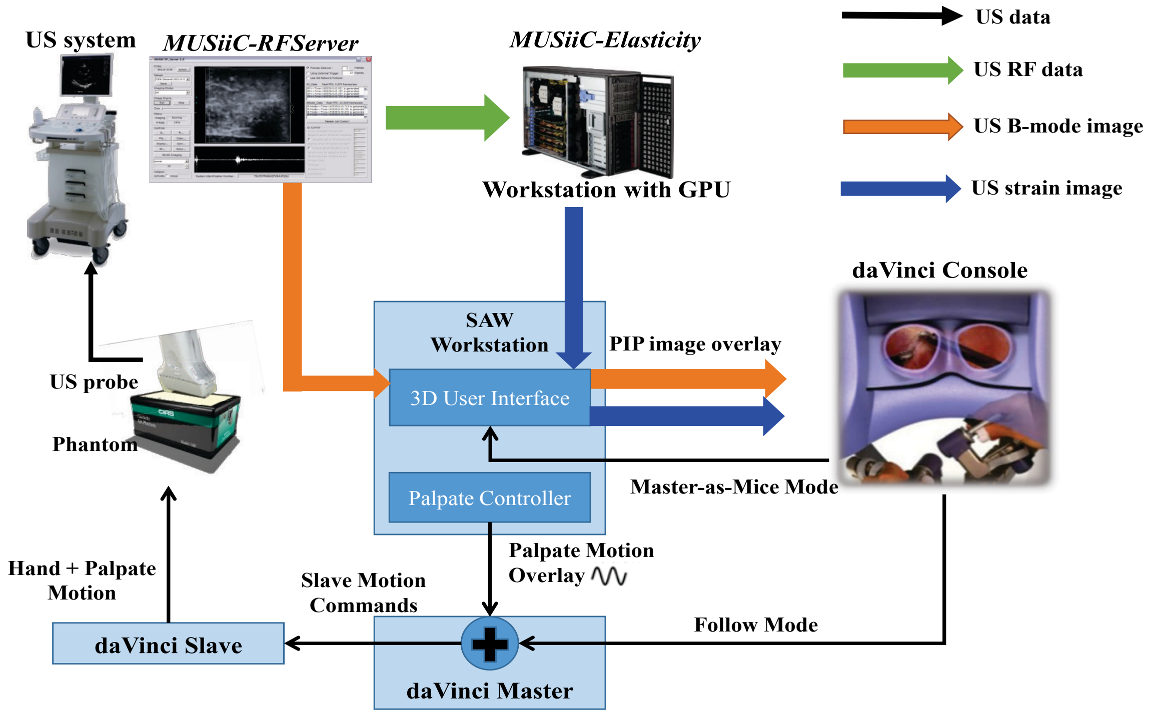


Figure 4.5. System setup of our robot-assisted system for real-time ultrasound elastography (Ref. [30])

Figure 4.5 shows the system setup of the proposed system for robot-assisted ultrasound elastography. The system is composed of the daVinci Surgical robot (Intuitive Surgical Inc., Sunnyvale, CA), a clinical US system with a customized laparoscopic US probe, and our *MUSiiC-RFServer* and *MUSiiC-Elasticity* modules [41] to generate a real-time US strain image stream. The system provides assistive control of robot motion by generating a sinusoidal palpation motion that is overlaid onto motion commands from the master manipulators (**Figure 4.5**). With this functionality of the system, the surgeon retains overall control of the ultrasound probe's position and orientation, while the system controls the tissue compression with the pose information of the ultrasound probe. Therefore, the elastography assistance mode provided by the system improves the image quality of US strain images.

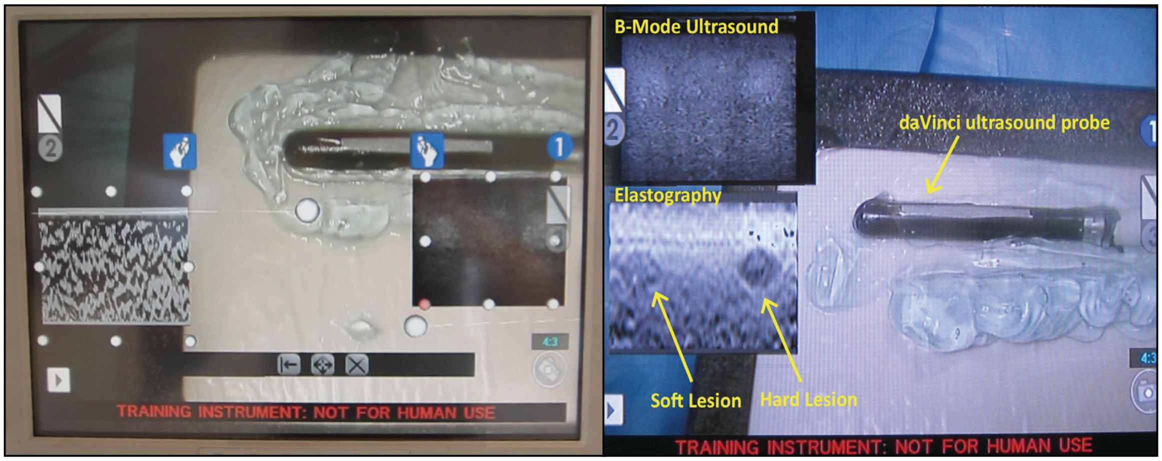
Computer integrated control of the surgical robot's motion is facilitated by an experimental *Read/Write* research API provided by the manufacturer. The daVinci robot manipulates a laparoscopic ultrasound probe customized to the form factor of a standard daVinci tool. The US probe is connected to a Sonix RP US system (Ultrasonix Co., Vancouver, Canada). *MUSiiC-RFServer* [41] collects both the US RF-data and B-mode image streams with the US system and transducer in real-time. The collected US RF-data stream is sent to the *MUSiiC-Elasticity* [5] module [41] to generate a real-time US strain image stream. Elastography images are streamed to a workstation of the daVinci robot system. In parallel, the workstation receives an US B-mode image stream from *MUSiiC-RFServer*. Sequentially, the workstation overlays the received image (US strain and B-mode image) streams in the daVinci console's stereo display as picture-in-picture (PIP).

[5] This module was developed by Nishikant Deshmukh by implementing the elasticity imaging algorithms based on NCC on the GPGPU architecture and utilizing *MUSiiC-TCPServer* and *MUSiiCTCPClient* for communication of US data streams [22].

The picture overlays enable the surgeon to observe the ultrasound and elastography image feeds in real-time without releasing control of the robot arms or diverting attention away from the task at hand.

This application used existing Surgical Assistant Workstation (SAW) [111-113] software wrapper classes that provide the ability to interface with the read-only and read-write research interfaces of the daVinci robot [114]. This allows us to superimpose an externally computed motion onto motion inputs from the master manipulators to create computer generated motion overlays for tissue palpation. Another function of SAW allows us to use a master arm controller effectively as a 3D mouse, which is called masters-as-mice mode. We use this mode in our system to build an interactive environment for the daVinci operator to control and interact with video overlays (see **Figure 4.6(a)**). The SAW software package is implemented on the central workstation of our system, which centralizes processing for user interaction with image overlays, acquisition of real-time image streams, and implementation of the control loop generating assistive motion for tissue palpation. The amplitude and frequency of the robot-assisted tissue palpation can be defined and updated in real-time by the user through a command terminal of the workstation [30].[6]

[6] Seth Billings developed a control software to generate and control a robot palpation motion, to receive US B-Mode and elastogram streams by utilizing *MUSiiCTCPClient* and to inject the image streams in the daVinci console's stereo display as PIP [30].


TRAINING INSTRUMENT: NOT FOR HUMAN USE
B-Mode Ultrasound
daVinci ultrasound probe
Elastography
Soft Lesion
Hard Lesion
TRAINING INSTRUMENT: NOT FOR HUMAN USE

(a) (b)

Figure 4.6. (a) Interactive menu environment for displaying image overlays in the daVinci console display, showing an active menu with two virtual mice corresponding to the left and right master manipulators and two picture-in-picture image overlays (picture taken of a patient side monitor). b) View of the daVinci console display during a test with an elasticity phantom; the elastography image overlay differentiates lesions of different stiffness. (Ref. [30])

4.2.3. Results

Our experimental tests of the system have been conducted with a CIRS Model 049 Elasticity QA Phantom (CIRS, Inc, Norfolk, USA), which has simulated lesions of different calibrated stiffness. **Figure 6(b)** represents a snap-shot of the daVinci display during phantom testing with real-time US elastography and B-mode image overlays controlled by our proposed system. The US elastography image in **Figure 6** shows clear differences in contrast between a soft and hard lesion, with the hard lesion appearing darker in the image. This distinction was invisible on the US B-mode. Moreover, the elastography image also provided clear boundaries of the lesions, which were not observable on the US B-mode image because of the lesion's isoechoic properties. These images were recorded with tissue palpation set to 1mm amplitude and 1Hz frequency.

4.2.4. Conclusion

A robot-assisted system for minimally invasive, real-time US elastography was proposed and investigated. An experimental study of the proposed system resulted in improved image quality of US elastography images with computer-assisted motion control compared to those generated by manual palpation motion. In addition, the elasticity information of the target tissue is provided to the surgical robotic system by supporting real-time US strain and B-mode image streams with *EIMessage* and *USMessage* of *OpenIGTLinkMUSiiC* [47].

4.3. On-line tracked Ultrasound Elastography

This work was accomplished with the support of the following collaborators and was submitted under the following citation:

Nishikant Deshmukh[7], Hyun Jae Kang, Seth Billings, Russell Taylor, Greg Hager, and Emad M. Boctor, "Elastography Using Multi-Stream GPU: An Application to Online Tracked Ultrasound Elastography, In-Vivo and the da Vinci Surgical System," PloS one, vol. 9, p. e115881, 2014.

Abstract

An US quasi-static elastography image represents the elasticity properties of the target tissue and is usually generated with the motion of a freehand US probe. However, this imaging method requires a uniform compression force aligned to the axial dimension of the US probe. Consequently, the quality of US elastography depends on the expertise of the operator. Moreover, reconstruction of an US elastography image from consecutive US RF frames demands high computational power. Therefore, developing and customizing US elastography imaging is not easy with a commercial clinical US system. To overcome these limitations, we propose a novel online tracked US elastography

[7] Nishikant Deshmukh led this study and is the main contributor who implemented the tracked ultrasound elastogram [110] algorithm on CUDA architecture and analyzed the performance of the software (O-TrUE) module and the quality of the generated US elastogram from the software module.

system to generate high quality US elastography images consistently by implementing the method of tracked US elastography on a GPU platform.

4.3.1. Motivation

US quasi-static elastography imaging is based on the displacements of US speckles of pre- and post-compression US RF frames. These displacements can be used to estimate elasticity properties of a target material [58]. Therefore, US elastography images are used to detect cancerous tissues or malignant tumors, which are stiffer than the surrounding normal tissue [107, 115]. Real-time thermal monitoring with this imaging modality has been researched by several research groups because ablated tissue has higher stiffness than regular tissue [19, 41, 116-118].

An US elastography image generated with the motion of a hand-held US transducer is referred to as free-hand elastography [119-121]. This type of US elastography does not require special hardware or a major change on an US system. Therefore, this method can be easily integrated with an existing US system [110]. However, the quality of freehand elastography imaging is dependent on an operator because this imaging technique usually requires a uniform compression force in the axial direction of the US transducer. Small lateral or out-of-plane motions can reduce the signal-to-noise ratio (SNR) or contrast-to-noise ratio (CNR) of the elastography image. Especially, a free-hand pure axial compression motion is not easy in a real clinical application because of the slippery or oblique surface of the tissue or uneven tissue deformation [63, 110].

To address the limitation of freehand elastography, Foroughi et al. proposed tracked US elastography (TrUE) that is based on spatially-tracked US RF data and utilizes the pose information of a RF frame to find pairs of RF frames to generate enhanced US elastography [110]. However, this method was implemented as an off-line method because of high computation power and data synchronization requirements for a real-time implementation [63].

On the other hand, real-time US elastography can provide real-time elasticity information of a target tissue to the user and IGT system. However, reconstruction of US elastography is computationally expensive. Consequently, almost all commercial clinical US systems implemented the method on their customized field programmable gate array (FPGA) or digital signal processor (DSP) [22]. These implementations do not allow for modifications or extensions such as an application of a novel US elastography algorithm. To solve this conventional limitation, our group proposed an US elastography implementation in a general-purpose graphic processing unit (GPGPU) with computer unified device architecture (CUDA) architecture [22, 63].

To address these challenges, we introduce a novel online tracked US elastography (O-TrUE) system to generate high quality US elastography in real-time. In this research, we improved our software engine of GPU elastography [22] with the technique of multi stream of CUDA architecture and multi threads to increase the computational performance. Then, we demonstrate the results of our phantom test to verify the O-TrUE [63] system.

4.3.2. Methods

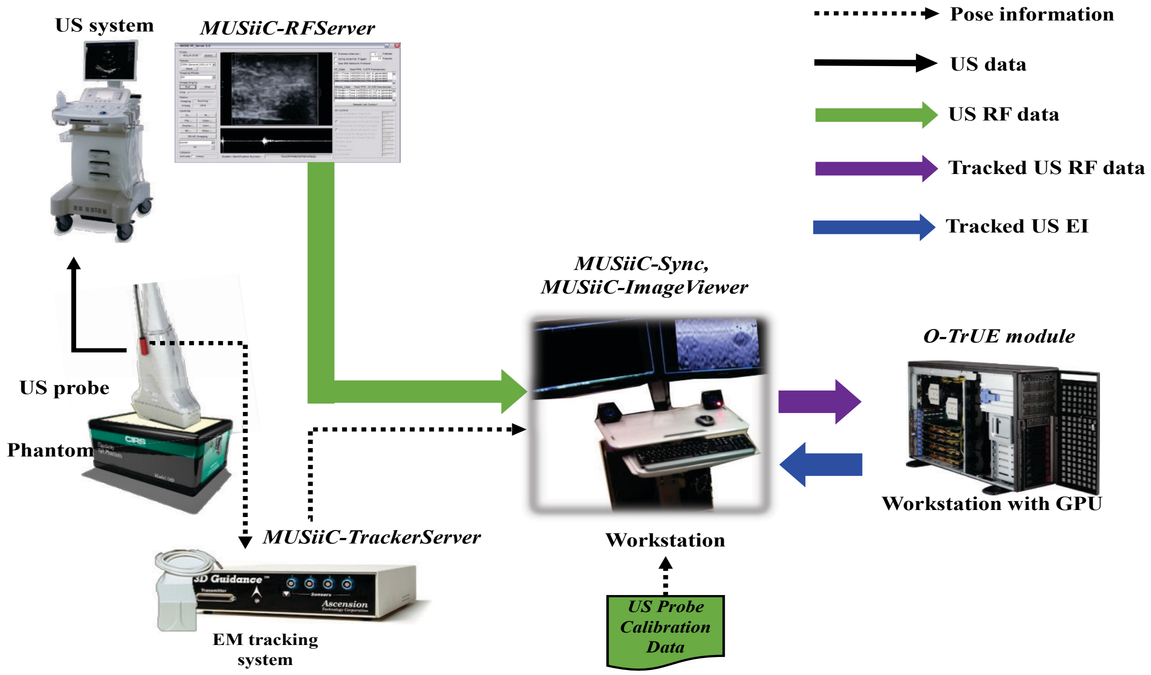


Figure 4.7. System setup for online tracked US elastography (O-TrUE)

Figure 4.7 shows a system setup for our online O-TrUE system. *MUSiiC-RFServer* and *MUSiiC-TrackerServer* acquire US RF data and pose information of an attached electromagnetic (EM) position sensor on the US transducer in real-time. Then, these two *MUSiiC* modules send the collected data streams to the *MUSiiC-Sync* module with their corresponding data collection time-stamps [72]. The *MUSiiC-Sync* module applies a pre-computed calibration to the received pose information of the US transducer to generate the pose transformation matrix of the US RF frame, and generates a spatially-tracked RF frame stream by synchronizing the received RF frame and updating its pose information. The O-TrUE [8] module receives the spatially-tracked RF frames and retrieves each RF frame’s synchronized spatial tracking information as shown in **Snippets 4.3**, and generates high quality elastography images. The *MUSiiC-ImageViewer* module [41] gets the generated US elastography image from the O-TrUE module to display [63].

Snippet 4.3. Receiving and retrieving US RF data with its synchronized tracking information in O-TrUE module using a callback function of *MUSiiCTCPClient*

```
/// User defined callback function for the method of
   the explicit data transmission of MUSiiCTCPClient
int    ReceiveMsg (int numOfRun = 0, int taskInfo = 0, void* ptr = NULL,
        igtl::MessageBase::Pointer data1 = NULL, void* data2 = NULL,
       void* data3 = NULL)
  {
       /// Check a received data
       if(data1.IsNull())
         return 0;

       // Retrieve USMessage
       igtl::USMessage::Pointer USMsg =
       igtl::RetrieveMessage<igtl::USMessage::Pointer, igtl::USMessage>(data1);

       if(USMsg.IsNull())
         return 0;

```

[8] The O-TrUE software module developed by Nishikant et al. by implementing the TrUE algorithm with the technique of multi stream of CUDA architecture on GPU framework is used to generate real-time tracked US elastogram [63].

```
/// Get the size of the received US RF data
USMsg->GetDimensions(size);
USMsg->GetSpacing (spacing);

/// Get the timestamp of received US RF data
igtl::TimeStamp::Pointer ts = igtl::TimeStamp::New();
USMsg->GetTimeStamp(ts);

/// Get synchronized tracking information of RF data
igtl::Matrix4x4 matrix;
USMsg->GetMatrix(matrix);

/// Copy US RF data to a cuda compatible data
char *tempdata;
cuda_malloc_host ((void **)&tempdata, USMsg->GetImageSize());
memcpy (tempdata,
          USMsg->GetScalarPointer(), USMsg->GetImageSize());

/// Build a data structure for computing US elstogram
new_data = new data_frame_queue;
new_data->data              = tempdata;
new_data->height            = size [1];
new_data->width             = size [0];
new_data->number_frames = size[2];
new_data->itime             = ts->GetTimeStamp();

new_data->spacing[0]        = spacing[0];
new_data->spacing[1]        = spacing[1];
new_data->spacing[2]        = spacing[2];

/// Put tracking information to the data structure
new_data->matrix            = matrix;

// Put additional US information in the data structure
// Get transmit frequency of RF data
new_data->fhr.txf           = USMsg.GetTx();
// Get sampling frequency of RF data
new_data->fhr.sf            = USMsg.GetTx();
// Get a US probe ID of RF data
new_data->fhr.probe         = USMsg.GetTx();
```

```
        // Put the built data structure to
        a thread of computing US elastogram
        in_queue.push (new_data);
        ....
}
```

For our phantom experiment, we used a CIRS Model 049 Elasticity QA Phantom (CIRS Inc. Norfolk, USA), which has a background medium with 25 kilopascal (kPa) of elasticity and four different lesions with different Young's modulus (8, 14, 45, and 80 kPa). Specifically, our experiment was performed at the 80 kPa lesion with free-hand palpation. Then, we generated two different types of US elastography images by using the conventional and TrUE methods, which are based respectively on consecutive RF frames and RF frame pairs filtered by their pose information, to evaluate the consistency of a generated US elastogram's quality. The consistency is calculated by a maximum value of normalized cross-correlation (NCC) between sub-regions of a reference and generated US elastogram as shown in equation 4.1 [63]. The sub-region (template window) of a reference image is smaller than the sub-region (target window) of the generated elastography images to compensate for different compression motions during free-hand palpation by searching for a maximum NCC value.

$$\alpha_i = \max_{u,v \in R}\left(\frac{\sum_{x,y}\left[f_{i+1}(x,y)-\bar{f}_{i+1}\right]\left[f_i(x-u,y-v)-\bar{f}_i\right]}{\left\{\sum_{x,y}\left[f_{i+1}(x,y)-\bar{f}_{i+1}\right]^2\sum_{x,y}\left[f_{i+1}(x-u,y-v)-\bar{f}_i\right]^2\right\}^{0.5}}\right) \tag{4.1}$$

Where, i is the sequence number of the generated US elastography image. x and y represent the position of a pixel in the target window of an *(i+1)-th* image and $f_i(x,y)$ denotes the intensity value of the pixel. *x-u* and *y-v* are a pixel location in the lateral and axial directions, respectively, within the template window of an *i-th* image. $\bar{f_i}$ is the mean value of the template window while $\bar{f}_{i+1}$ is the mean value of the target window under the area of the template windows shifted to (u,v) [63].

4.3.3. Results

Table 4.2 shows the performance of our US elastography implementation with different GPU technologies in frames per second (fps). In the table, *normal*, *single-stream*, and *multi-stream* indicate basic GPU implementations without GPU stream technique, using a single GPU stream, and utilization of multi GPU stream with multithreads of the central processing unit (CPU), respectively. Three different cases of a window size, maximum searching distance, and percentage of window's overlap for computation of NCC between consecutive RF frames were considered. US RF frames were collected with 128 RF lines, different scan depths (4, 5, and 6 cm), and 40 MHz of sampling rates. Then, the collected US RF frames were utilized in an offline method to evaluate the standalone performance.

The performance of *multi-stream* is higher than the value of *normal* and *single-stream* for all three cases and US scan depths. Moreover, this significant improvement in the performance of GPU implementation matches the data acquisition speed of *MUSiiC-RFServer* [41, 72]. Therefore, all collected RF frames in our *MUSiiC-RFServer* can be converted to US elastography images in real-time.

Table 4.2. The performance of our US elastography implementation in fps (Ref. [63])

Conditions of NCC computation			Technique of GPU implementation	Scan depth of RF frame		
				4 cm	5 cm	6 cm
Case 1	Window Size	10 pixel	***normal***	36.56	31.1	31.56
	Distance	2 mm	***single-stream***	36.31	29.01	30.76
	Overlap	98%	***multi-stream***	78.02	61.14	48.4
Case 2	Window Size	12 pixel	***normal***	34.74	31.98	30.62
	Distance	2 mm	***single-stream***	34.68	31.91	30.56
	Overlap	98%	***multi-stream***	72.07	56.43	45.55
Case 3	Window Size	14 pixel	***normal***	21.53	21.54	19.73
	Distance	4 mm	***single-stream***	21.76	21.6	19.84
	Overlap	98%	***multi-stream***	46.43	35.44	29.53

Figure 4.8 (a) shows a template window and a target window in the *i-th* and *(i+1)-th* US elastography images, respectively, of the generated US elastography images to apply to equation (4.1). The maximum coefficient values of NCC computed in 100 pairs of two consecutive US elastography image streams are presented in **Figure 4.8 (b).** Red and blue lines represent the computed maximum coefficients of NCC in US elastography image streams generated by the conventional and O-TrUE methods, respectively [63]. The O-TrUE method generates more consistent US elastography with higher coefficient values than the conventional method.

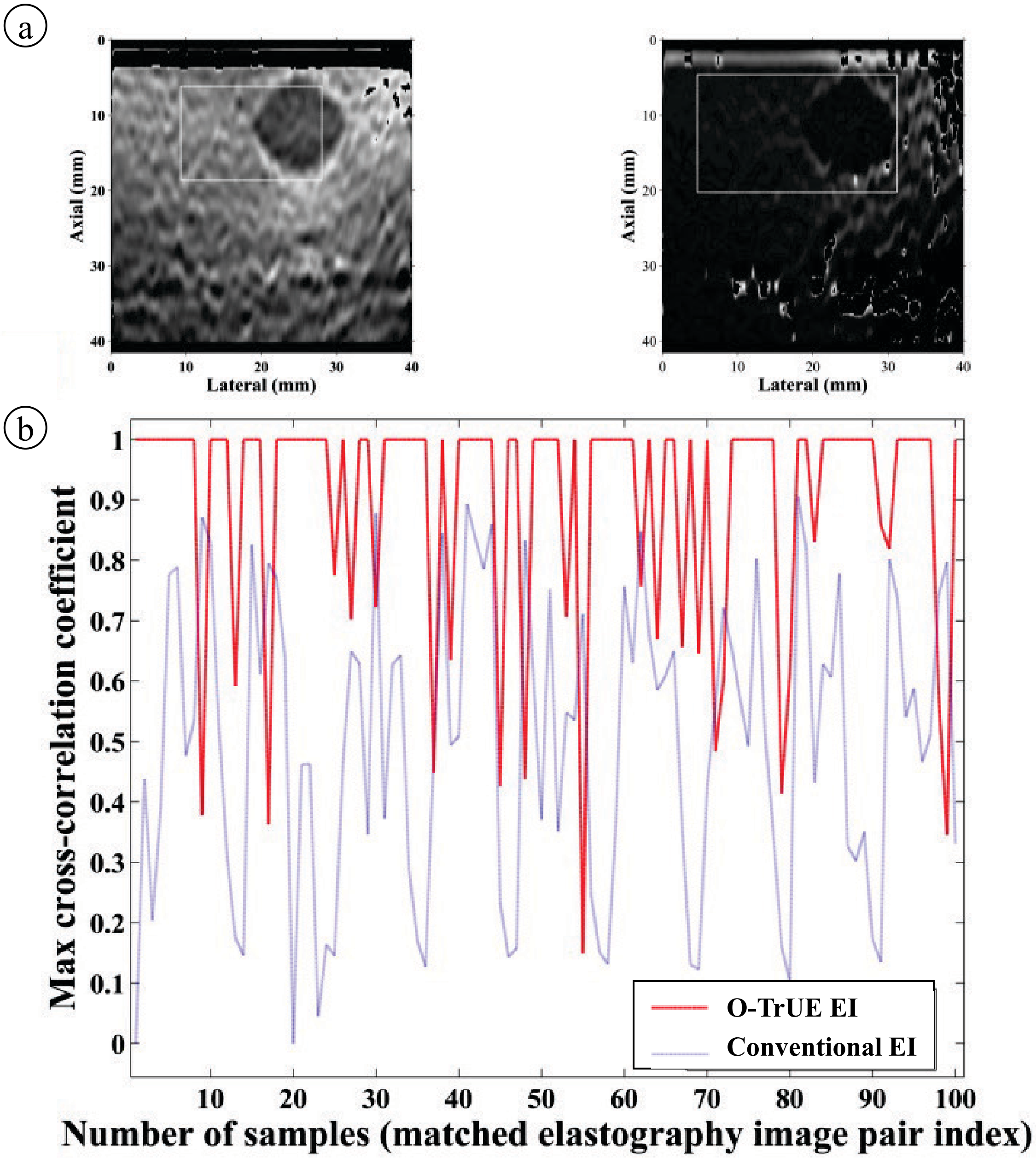


Figure 4.8. Comparison of the consistency of US elastography images generated by O-TrUE and the conventional method. (a) a template window and a target window in *i-th* and *(i+1)-th* US elastography images, respectively, (b) maximum coefficient values of NCC computed in 100 pairs of two consecutive US elastography image streams. Red and blue lines represent the computed maximum coefficients of NCC in US elastography image streams generated by the conventional and O-TrUE methods, respectively. (Ref. [63])

4.3.4. Conclusion

In this research, we demonstrated the versatility of our software framework by adding a real-time tracked US elastography, O-TrUE, software module. The performance of the O-TrUE software module was evaluated under different conditions with our framework. In addition, the quality consistence of generated US elastography images was evaluated based on the maximum NCC coefficient between a template and the target windows in subsequent US elastograms. Our system using the O-TrUE module generated a more consistent US elastogram image stream compared to conventional freehand US elastography imaging. Therefore, our proposed system can provide real-time US B-mode image streams and high quality US elastography image streams to the user or other IGT systems consistently.

4.4. Conclusion

Novel applications using interventional US elastography enabled by our software framework are introduced in this chapter. The first application is an US imaging software framework for real-time thermal monitoring of acoustic therapy. Although ablation therapy is a promising minimally invasive approach to remove cancerous tissue, this approach requires real-time ablation tool tracking and monitoring of thermal dose in heated tissue to improve its efficiency and success rate. To address these requirements, we integrated a real-time US imaging system and an existing acoustic ablation system. In pre-ablation, a real-time US B-Mode and elastography image stream provide the information of an ablation tool's position and a tumor's boundary, respectively. During ablation, the thermal dose in heated tissue can be estimated by the real-time US elastography image.

In addition, a system for robot-assisted real-time laparoscopic US elastography is presented. A vision system of a surgical robot only provides surface information of a target object. In parallel, an axial compression motion with respect to the US probe is required to generate a high quality US elastogram. To overcome these limitations of the surgical robot and US elastography imaging system, we integrated these two systems. The surgical robot generates an axial compression motion for good quality US elastography and the US imaging system provides the inner structure and tactile (stiffness) information of a target object to the surgical robot by real-time US B-mode and elastography image streams.

Finally, an online tracked US elastography system is discussed. In this research, we developed a spatially tracked US RF data acquisition system with *MUSiiC-RFServer*, *MUSiiC-TrackerServer*, and *MUSiiC-Sync* modules. Then, we utilized spatial pose information of a US probe and a pre-computed US probe calibration information to filter proper RF frame pairs from a consecutive RF frame stream for consistent high quality US elastography images. In addition, we implemented the existing algorithm of tracked US elastography to the GPU architecture with the multi-stream technique for real-time computation of tracked US elastography images.

Recapitulation of thesis contribution

- Integrating a real-time GPU based US elastography system with an interventional US thermal ablation system (Acoustic MedSystems TeraVisionTM), studying real-time thermal monitoring for acoustic ablation therapy, and validating the performance through a series of in vivo experiments.
- Developing spatially tracked US data acquisition system with *MUSiiC-RFServer*, *MUSiiC-TrackerServer* and *MUSiiC-Sync* modules. This system improves the quality of US elastography by implementing tracked US elastography with the provided spatial information.

Chapter 5:

Real-time photoacoustic imaging and interventional component framework

Acquisition of ultrasound (US) pre-beamformed radio-frequency (RF) data is essential in photoacoustic (PA) imaging research. Moreover, 3D PA imaging can provide volumetric information for a target of interest. However, existing 3D PA systems require specifically designed motion stages, an ultrasound scanner and a data acquisition system to collect 3D pre-beamformed RF data. These systems are incompatible with clinical ultrasound systems and are difficult to reconfigure and generalize to other PA research. To overcome these limitations, we propose a new software framework for PA imaging research and spatially-tracked pre-beamformed RF data acquisition with a conventional 2D ultrasound transducer and an external tracking device.

5.1. Introduction

Acquisition of ultrasound (US) pre-beamformed radio-frequency (RF) data is essential in advanced ultrasound imaging research such as adaptive beamforming [66, 122], synthetic aperture ultrasound imaging [123], and photoacoustic (PA) imaging [27, 35]. PA imaging is an emerging medical imaging modality that relies on the absorption of optical energy and the subsequent emission of an acoustic wave. PA images include optical and acoustic information of the target material, and provide high contrast and high resolution medical images. Due to these merits, PA imaging has been used in biomedical research for various structural and functional imaging applications [27, 64, 66, 124].

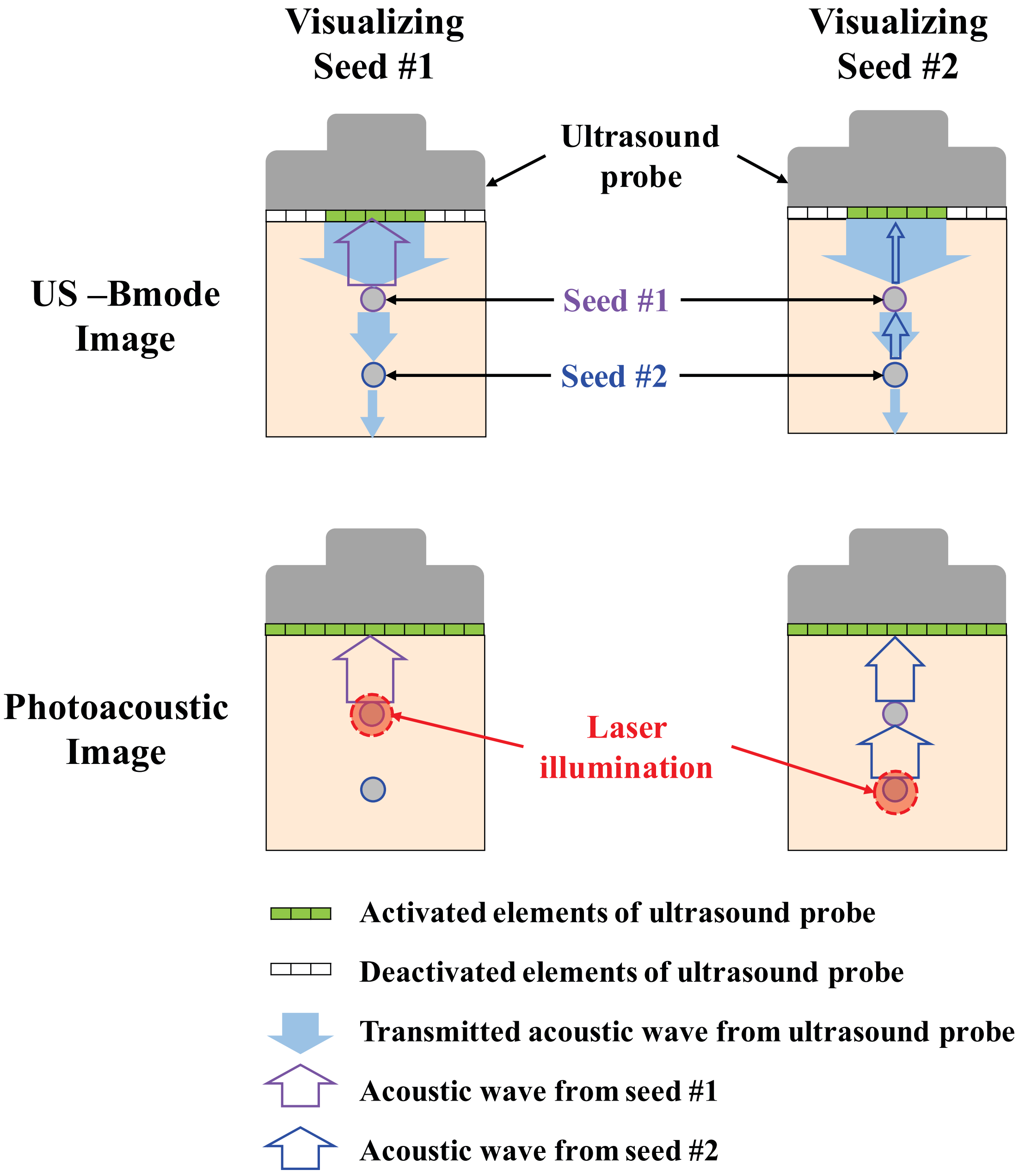

Visualizing
Seed #1
Visualizing
Seed #2
Ultrasound
probe
US –Bmode
Image
Seed #1
Seed #2
Photoacoustic
Image
Laser
illumination
Activated elements of ultrasound probe
Deactivated elements of ultrasound probe
Transmitted acoustic wave from ultrasound probe
Acoustic wave from seed #1
Acoustic wave from seed #2

Figure 5.1. Concept diagram representing the difference between UB echo B-mode and photoacoustic imaging

Figure 5.1 shows a concept diagram representing the difference between US echo B-mode and photoacoustic imaging when visualizing two small targets. In **Figure 5.1**, the width of each acoustic wave corresponds with its strength. The reconstruction of US echo B-mode images depends on the reflected acoustic waves from target objects (seed #1 and #2). PA images depend on acoustic waves emitted directly from the objects. Therefore, the acoustic wave travels half the distance in a PA image as compared to an echo B-mode image [69]. Consequently, the strength of the PA signal is generally higher than the signal strength of an US echo B-mode image. Also, the echo acoustic wave depends on the reflectivity and size of the target object, while the PA acoustic wave is affected by the optical and thermal properties of the target material [35]. Therefore, PA imaging enables detection and visualization of tiny objects or surgical metal implants such as brachytherapy seeds that are invisible on US echo B-mode images. In addition, PA imaging can provide enhanced tracking accuracy when tiny objects are tracked on a PA image with a focused laser beam [35, 64]. A change in PA signal can also be detected when there are changes in the temperature of the target material. Due to some of these properties, the desire to integrate PA imaging with existing IGT systems has been growing [124]. There have also been a considerable number of studies conducted on generating 3D PA volumes [35, 125-127].

However, existing PA systems are generally customized and independently built by each research group. Moreover, 3D PA systems require specifically designed motion stages, ultrasound scanner and data acquisition system to collect 3D pre-beamformed RF data. These systems are incompatible with conventional clinical ultrasound systems and are difficult to reconfigure and generalize to other PA research. In the case of 3D PA volumes acquired with a clinical 2D US transducer, each 2D PA frame requires its own spatial orientation and position information to reconstruct a 3D PA volume [49].

To overcome the limitation of existing PA systems and address the above considerations, we propose a software framework for spatially-tracked pre-beamformed RF data acquisition with a conventional 2D US transducer and an external tracking device. To fulfill these requirements, we upgraded our previous software framework [50, 72] by adding new *MUSiiC modules*: *MUSiiC-DAQServer*, *MUSiiC-Beamformer*, and *MUSiiC-OPO*.

5.2. *MUSiiC* modules for photoacoustic imaging research

5.2.1. *MUSiiC-DAQServer*

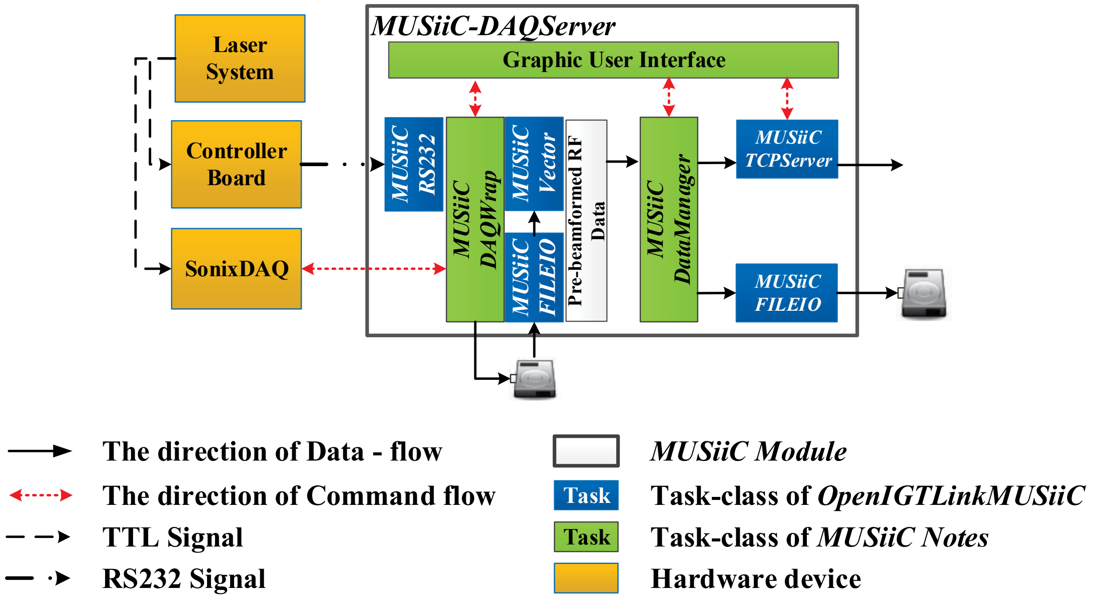


Figure 5.2. Block diagram of (a) *MUSiiC-DAQServer* module

MUSiiC-DAQServer [49, 50] was developed to collect 2D pre-beamformed RF data with a conventional 2D US transducer by using a SonixDAQ (Ultrasonix Co. Vancouver, Canada) device and the software development kit (SDK) from the manufacturer. As shown in **Figure 5.2**, this software module is composed of several software task-classes from *OpenIGTLinkMUSiiC* and *MUSiiC Notes*: *MUSiiCDAQWrap, MUSiiCRS232, MUSiiCTCPServer, and MUSiiC-FileIO* [47, 72].

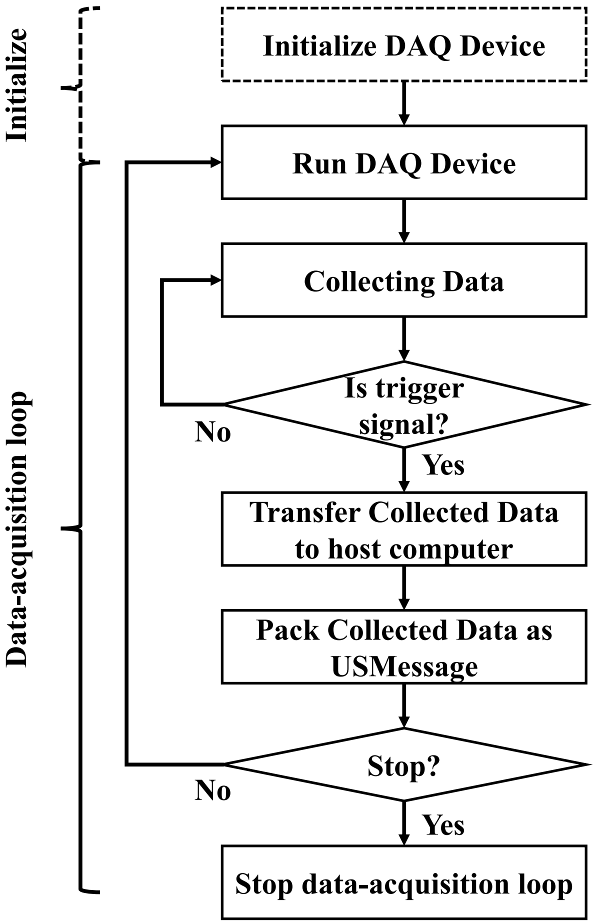


Figure 5.3. The data-acquisition flow of *MUSiiCDAQWrap* task-class

The *MUSiiCDAQWrap* [49, 50] software task-class is a customized task-class of *MUSiiC Notes* [72] based on the SDK to collect pre-beamformed RF data efficiently. The data acquisition process flow in the *MUSiiCDAQWrap* class is as follows: (1) initialize DAQ device, (2) run DAQ device, (3) collect Data, (4) transfer the data to the host computer, and (5) pack the data as a *USMessage* of *OpenIGTLinkMUSiiC* [47] (see **Figure 5.3**). Based on this process flow, the *MUSiiCDAQWrap* task-class supports two data-acquisition modes as shown in **Figure 5.4**: sequence mode and burst mode. In sequence mode (see **Figure 5.4 (a)**), a single 2D pre-beamformed RF frame is collected in each data-acquisition loop, while multiple frames are acquired in burst mode (see

Figure 5.4 (b)). Sequence and burst modes were designed for low and high laser source pulse repetition rates respectively.

One problem that we faced was that the SDK worked with a callback signal that only returned a binary status of the SonixDAQ (Ultrasonix Co. Vancouver, Canada) device memory (full or not). This functionality could not provide an accurate timestamp that indicated the acquisition time for each frame during burst mode data acquisition. To solve this problem, we built a controller board that converts the transistor-transistor logic (TTL) signal from the laser source to the RS-232C protocol to send to the host computer. The *MUSiiC-RS232* software task-class monitors for a RS-232C signal from the controller and measures the data-acquisition timestamp of each frame with an independent thread and a non-blocking IO mechanism. To minimize the communication time of the RS-232C protocol, we used a baud rate of 921600 bps and one character ("T") as the protocol message.

The collected data is packed as an *ImageMessage* of *OpenIGTLink* [45] or *USMessage* of *OpenIGTLinkMUSiiC* [47] with the US transducer information and the measured data-acquisition timestamp. Afterwards, the *MUSiiCTCPServer* task-class sends the packed message to other connected client programs, *MUSiiC-Sync* or *MUSiiCFileIO* task-class, which saves the message to the local hard disk.

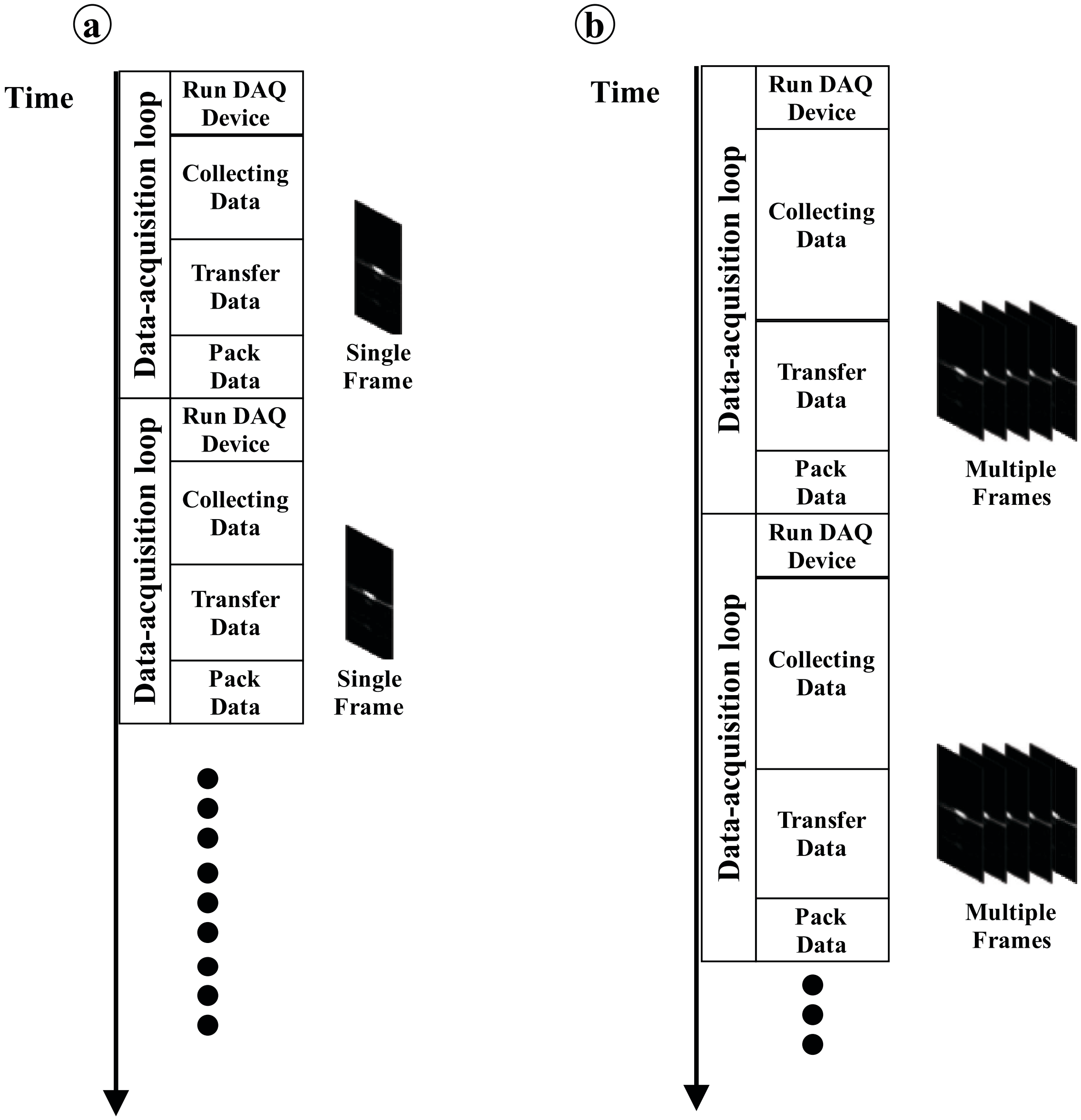


Figure 5.4. Two data acquisition modes of *MUSiiCDAQWrap* task-class (a) sequence data-acquisition, and (b) burst data-acquisition mode

5.2.2. *MUSiiC-Beamformer*

The first step of building PA images or general US B-Mode images from pre-beamformed RF data is converting the data into beamformed RF data. In our software architecture, a MUSiiC-*Beamformer* module performs this conversion. **Figure 5.5** shows a block diagram of the module. There are four functional classes in this module: *MUSiiCTCPClient*, *MUSiiCVector*, *MUSiiCBeamforming*, and *MUSiiCTCPServer* [47, 72]. The *MUSiiCTCPClient, MUSiiCBeamforming*, and *MUSiiCTCPServer* classes are active classes that have their own independent task-threads as mentioned before. To communicate data between these active classes, two instances of *MUSiiCVector* [72] are present in the *Beamformer* module.

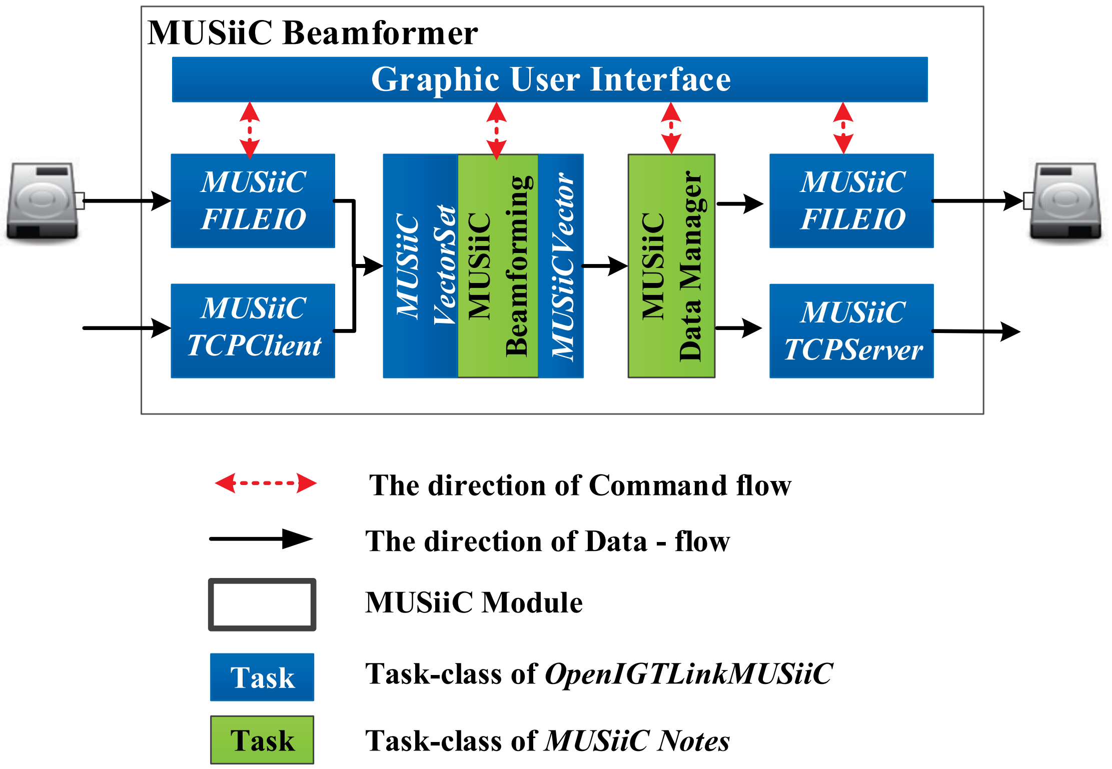


Figure 5.5. Block diagram of *MUSiiC-Beamformer* module

The delay-and-sum algorithm [126] is implemented in the *MUSiiCBeamforming* class to beamform a pre-beamformed RF data. The *MUSiiCTCPClient* class receives the pre-beamformed RF data stream from the *MUSiiC-DAQServer* via TCP/IP network. Then, the received data is converted to beamformed RF data in the *MUSiiCBeamForming* class. The output data is then transferred to the *MUSiiC B-Mode* module of the *MUSiiC Toolkit* [72, 94] to create US B-mode images in the *MUSiiCTCPServer* class.

5.2.3. *MUSiiC-OPO*

Different wavelengths of pulsed laser light [35, 64] will generate different PA waves dependent on the target material's optical absorption spectrum. These differences can be detected with PA imaging, and they can be used to characterize the target material [124]. Therefore, it is necessary to be able to control the wavelength of the laser source for PA imaging research. To achieve this requirement, we developed the *MUSiiC-OPO* module based on an Optical parametric oscillator (OPO) system embedded in a Q-switch Nd:YAG laser system (Quantel, Les Ulis, France) and its associated API.

Figure 5.6 shows the block diagram of *MUSiiC-OPO* module, which is composed of software active task-classes (*MUSiiCFileIO*, *MUSiiCOPOWrap*, *MUSiiCTCPClient*, and *MUSiiCDataManager*) and a concurrent data container (*MUSiiCVector*) of *MUSiiC Notes* [72]. *MUSiiCOPOWrap* is a customized software task-class of *MUSiiC Notes* based on the API to control the laser's wavelength efficiently. *MUSiiCOPOWrap* controls the laser system and the OPO system with bidirectional RS232 communication based on its supported protocols. The *MUSiiC-OPO* module is mainly a system-control module and not a data-acquisition module. This module's *MUSiiCTCPClient* receives PA channel frames from the *MUSiiC-DAQServer* [49, 50] module through TCP/IP communication. Then, one *MUSiiCFileIO* of this module saves the received PA channel data and the laser wavelength to a local hard disk for offline data processing. Another *MUSiiCFileIO* reads a configuration file of the OPO system to initialize the hardware and its control parameters.

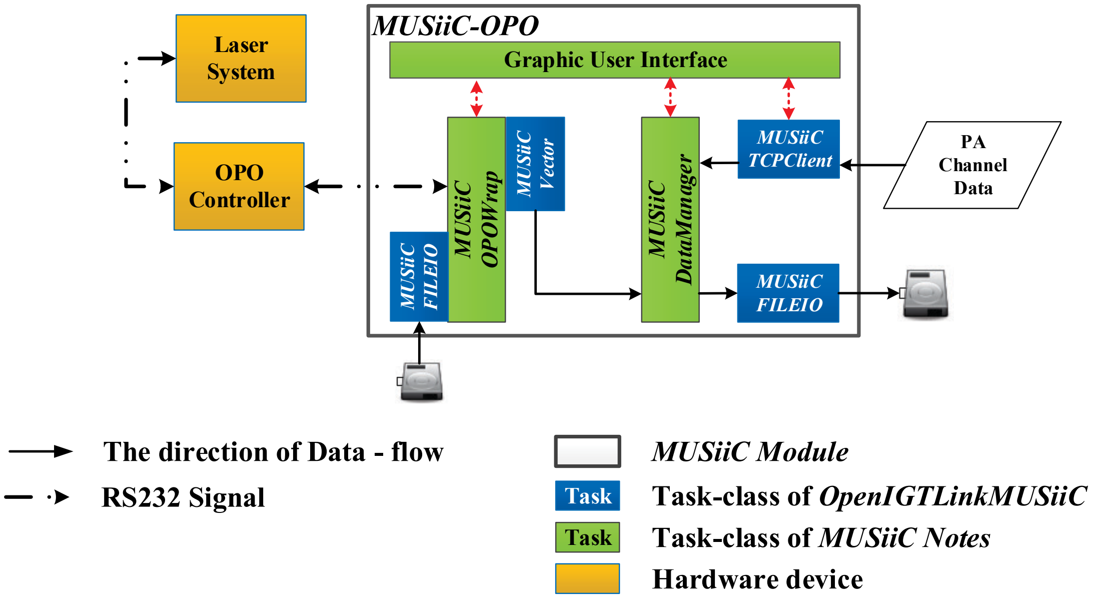


Figure 5.6. Block diagram of *MUSiiC-OPO* module

5.3. Spatially-tracked pre-beamformed RF data acquisition with a freehand clinical ultrasound transducer

This work was accomplished with the aid of the following collaborators and was submitted under the following citation:

Hyun Jae Kang, Xiaoyu Guo, Reza Zahiri Azar, Alexis Cheng, and Emad M. Boctor, "Software framework for spatially tracked pre-beamformed RF data acquisition with a freehand clinical ultrasound transducer," presented at the SPIE Medical Imaging, 2014.

As mentioned, spatially-tracked pre-beamformed RF data can provide pose information for each PA frame and volumetric information of a target of interest [49]. Therefore, a system for data collection is important for 3D PA imaging research including spatial compounding of PA image to improve its image quality and a reconstruction of 3D PA volume, and an integration of 3D PA images with existing IGT systems. In addition, compatibility with conventional US systems is required for the configurability and generality of the system. In this section, we propose a system for spatially-tracked pre-beamformed RF data acquisition with a conventional 2D US transducer and an external tracking device.

Figure 5.7 shows our system components and work-flow for collecting spatially-tracked 2D pre-beamformed data generated by PA effects. A Q-switch Nd:YAG laser (Quantel, Les Ulis, France) is operated at multiple wavelengths controlled by its OPO system to generate a PA signal. Our *MUSiiC-DAQServer* and SonixDAQ (Ultrasonix Co. Vancouver, Canada) device collect the generated PA signal with a handheld L14-5W/38 US probe (Ultrasonix Co. Vancouver, Canada) and Sonix-CEP US (Ultrasonix Co. Vancouver, Canada) system.

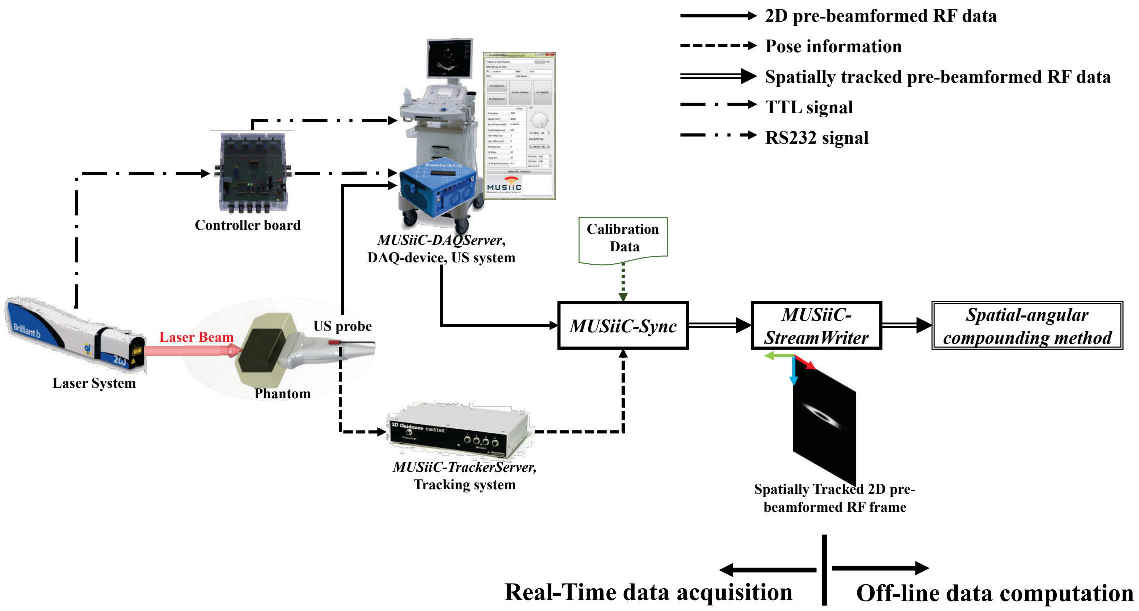


Figure 5.7. System components and data flow for acquiring spatially-tracked pre-beamformed RF data with a freehand clinical ultrasound transducer

Our controller board [1] received TTL trigger signals from the laser system, converted them to the predefined RS232 protocol and sent the protocol to *MUSiiC-DAQServer* to collect PA frames and to measure the data-acquisition timestamp of each frame. During data acquisition, pose information of the EM sensor was recorded with a medSAFE EM tracking system (Ascension Technology Co. Milton, USA) and *MUSiiC-TrackerServer* [49]. The *MUSiiC-Sync* [72] module generated spatially-tracked 2D pre-beamformed RF frames from the collected 2D pre-beamformed RF frames, pose information of the EM sensor and the precomputed calibration information of the probe. The spatially-tracked

[1] Xiaoyu Guo designed and implemented the controller board.

2D frames were then saved to a local hard disk with *MUSiiC-StreamWriter* [49] as *USMessage* format because this data structure can contain PA image data, its synchronized tracking information and additional information of the ultrasound probe. *MUSiiC* modules (*MUSiiC-DAQServer, MUSiiC-TrackerServer*, *MUSiiC-Sync,* and *MUSiiCStreamWriter*) in this system are connected by TCP/IP connections.

In addition, we wrote Matlab^{TM} scripts as part of *OpenIGTLinkMUSiiC* to load the saved spatially-tracked 3D PA frames and retrieve all of the information required for additional off-line data processing as shown in **Snippets 5.1**.

Snippet 5.1. *OpenIGTLinkMUSiiC* Matlab^{TM} script (ReadSingleIGTLFile) to load a saved *USMessage*

```
*************** An example of ReadSingleIGTLFile ******************
        file_path  = '/Data/Syn_PA_Channeldata.itgl'
        USMessage = ReadSingleIGTLFile file_path)

*************** Input/ Output of ReadSingleIGTLFile ******************
Input:
        file_path                           // a filepath of a saved IGTLFile
Output:
        USMessage
                 - IGTLHeader
                   |     - version          // Version of IGTLink Message
                   |     - type             // Message type
                   |     - device           // Device name
                   |     - ts               // Time stamp
                   |     - bodysize         // Body size of IGTLink Message
                   |     - crc              // CRC
                   - Body
                        - h                 // The Header of ImageMessage
                         |  - version       // Version of ImageMessage
                         |  - type          // Type of Image Data
                         |  - scalar Type   // Data type of Image Data
                         |  - endian        // Endian of this Message
                         |  - coordinate    // Coordinate system
|  - dim               // Dimension of Image Data
|  - sub_off           // Sub-Offset of Image Data
|  - sub_dim           // Sub-dimension of Image Data
|  - matrix            // Transformation matrix of Image Data
|  - spacing           // Spacing values of Image Data
- data                 // US Data (US-BMode, US RF data,
                          and US channel data)
- uh                   // The Header of USMessage
|  - TYPE              // Data Type
|  - txf               // transmit frequency
|  - sf                // sampling frequency
|  - dr                // frame rate
|  - ls                // Line density
|  - sa                // Steering Angle
|  - Probe             // ID of ultrasound Probe
|  - EA                // Extension Angle
|  - elements          // The number of elements in the probe
|  - pitch             // The spacing between elements of the probe
|  - radius            // The curvature of the probe
|  - probe_angle       // The field of view of t
|  - tx_offset         // For phased array, tx_offset
|  - Motor_Radius      // Motor Radius for 3D Ultrasound Data
|  - Frames            // Total Number of RF-Frames of 3D
|  - Frame_Index       // The index of RF-Frame in 3D
|  - Focus_Spacing     // The spacing between focus points
|  - Focus_Depth       // The depth of first focus point
|  - Extra_int32_1     // TBD
|  - Extra_int32_2     // TBD
|  - Extra_int32_3     // TBD
|  - Extra_int32_4     // TBD
|  - Extra_int32_5     // TBD
|  - Motor_Dir         // The Direction of 2D fram in 3D Volume
|  - Focus_Count       // The number of Focus point
|  - Extra_int8_1      // TBD
|  - Extra_int8_2      // TBD
|  - Extra_int8_3      // TBD
```

Our proposed system using an external tracking device allows for arbitrary and unrestricted US probe motion and provides six degree of freedom spatial-tracking information. Spatially-tracked 2D PA frames with a conventional handheld 2D US probe are collected efficiently with the system. Moreover, 3D PA volumes can be reconstructed from the collected 2D PA frames using their spatial tracking information, without the need for expensive motion stages or custom US scanners. Also, the spatially-tracked 2D PA frames can be used to generate spatial-angular compounded PA images to generate an enhanced PA image as discussed in Chapter 7.

Table 5.1. Process time of *MUSiiCDAQWrap* class

Data acquisition steps of *MUSiiCDAQWrap*		**Processing time (ms)**
Initialize	(1) Initialize DAQ device	586.11
Data-acquisition loop	(2) Run DAQ device	19.97
	(3) Collecting Data	100
	(4) Transfer the collected data to host computer	149.14
	(5) Pack the collected data as *USMessage*	0.42

MUSiiCDAQWrap is a core task-class to collect pre-beamformed RF data inside the *MUSiiC-DAQServer* [49, 50] module. **Table 5.1** shows the processing time for each data-acquisition step of this class under the following conditions: 10 Hz laser pulse repetition frequency, sequence data acquisition mode, 0.5 megabytes (MB) frame size and 100 sample frames. Since the TTL signal generated at every laser pulse from the laser machine activates the ‘Collecting Data’ step of the *MUSiiCDAQWrap* [49, 50] software task-class, the processing time of this step is 100 milliseconds (ms) as shown in **Table 5.1**. **Table 5.2** represents the performance of our proposed system. Although the *MUSiiC-TrackerServer* collected pose information with high speed (121.65 frame per second (fps)), the overall performance is limited by the performance of the *MUSiiC-DAQServer*.

Table 5.2. The performance of data acquisition of software framework

MUSiiC Modules	Performance (fps)
MUSiiC-DAQServer	3.75
MUSiiC-TrackerServer	121.65
MUSiiC-Sync	3.75
Overall	**3.75**

As shown in **Table 5.2**, the overall performance of our system is limited by the slowest module (e.g. *MUSiiC-DAQServer*). The *MUSiiC-Sync* [72] module set the 2D pre-beamformed RF frames as the reference data and the spatial tracking information of the EM sensor as additional data. In this case, the *Data-Combine* task-thread of the *MUSiiC-Sync* task-class [72] found the tracking information that had the closest timestamp to each 2D PA RF frame and generated spatially-tracked 2D PA RF frames using the calibration information of the US probe. It is generally advantageous to set the data stream with the slower frame rate as the reference data, as it is more likely for a frame with a similar timestamp to be found from the data stream with a higher frame rate. Although the performance of data collection is dependent on the pulse repetition of the laser machine, we can improve the performance with the burst data collection mode of *MUSiiC-DAQServer* and high-speed laser machines [49].

Moreover, the software modules of our proposed system are based on *OpenIGT-LinkMUSiiC* [47, 72] with the concept of network distributed modules and can simultaneously support multiple-client connections via TCP/IP network. In addition, the collected 2D pre-beamformed RF frames are compatible with other *MUSiiC* modules such as *MUSiiC-Beamformer, MUSiiC-BMode* and *MUSiiC-ImageViewer* modules [41, 50]. These aspects of our system allow us to easily reconfigure and generalize our system to other PA or US research.

5.4. Conclusion

In PA imaging, the generated acoustic waves from an object illuminated by a light source travel half the distance as compared to a pulse-echo US imaging such as US B-Mode or elastography. Consequently, the strength of the PA signal is generally higher than the signal strength of a pulse-echo US image. In addition, PA imaging provides the optical and thermal properties of the target object. It can also provide accurate pose information of tiny objects with a focused light source.

However, a US channel data acquisition system, control software for light sources, and a customized US beamforming algorithm are required for PA imaging. To address these requirements, we added additional software modules to our software framework: *MUSiiC-DAQServer* module to collect US pre-beamformed RF data in real-time, *MUSiiC-OPO* module to control a laser system and the wavelength of the laser-beam, and *MUSiiC-Beamformer* module to convert the collected US channel data to post-beamformed RF data with a delay-and-sum (DAS) algorithm.

In addition, we developed a real-time spatially tracked PA image acquisition system by integrating the new *MUSiiC-DAQServer* software module with existing *MUSiiC-TrackerServer* and *MUSiiC-Sync* software modules. The spatially tracked PA images provide the pose information for each PA frame and volumetric information of a target object. Moreover, the collected data is based on the *ImageMessage* of *OpenIGTLink* or *USMessage* of *OpenIGTLinkMUSiiC*, which is compatible with other *MUSiiC* modules such as *MUSiiC-Beamformer* and *MUSiiC-BMode* modules for reconstructing real-time PA images, and other *OpenIGTLink* compatible IGT software such as 3D-Slicer.

Recapitulation of thesis contribution
• Developing real-time spatially tracked PA image acquisition system for interventional PA surgical system with a real-time US channel data acquisition module (*MUSiiC-DAQ*), a spatial tracking software module (*MUSiiC-TrackerServer*) with electromagnetic tracking system, and *MUSiiC-Sync* software module to synchronize acquired 2D PA image stream and the tracking data stream in real-time. This PA image collecting system enables 3D PA imaging research and interventional PA imaging research.

Chapter 6:

Applications of Interventional Photoacoustic Imaging

6.1. Software Framework of Real-time Pre-beamformed RF Data Acquisition with an Ultrasound Research Scanner for brachytherapy seeds visualization

This work was accomplished with the aid of the following collaborators and was submitted under the following citation:

Hyun Jae Kang, Nathanael Kuo, Xiaoyu Guo, Danny Song, Jin Kang, and Emad M. Boctor, "Software framework of a real-time pre-beamformed RF data acquisition of an ultrasound research scanner," presented at the SPIE Medical Imaging, 2012.

Abstract

Acquisition of pre-beamformed data is essential in advanced imaging research studies such as adaptive beamforming, synthetic aperture imaging, and photoacoustic imaging (PAI). Ultrasonix Co. has developed such a data acquisition device for pre-beamformed data known as the SONIX-DAQ, but data can only be downloaded and processed offline rather than streamed in real-time. In Chapter 5, we added additional software modules such as *MUSiiC-DAQServer*, *MUSiiC-Beamformer*, and *MUSiiC-OPO* to our software framework for supporting various PAI researches.

The localization of brachytherapy seeds is a key step for prostate brachytherapy. However, conventional US B-Mode imaging is insufficient for visualizing seeds due to

noise artifacts. To address this challenge, we utilized PA imaging to visualize and localize the brachytherapy seeds with our software framework.

6.1.1. Motivation

Acquisition of pre-beamformed data is essential in advanced imaging research studies such as adaptive beamforming, synthetic aperture imaging, and PAI. Specifically, PAI for brachytherapy seed localization is also being investigated by several research groups [25, 37, 38, 128, 129].

PAI is a hybrid medical imaging modality that exploits the properties of light and ultrasound (US) by scanning the distribution of optical absorption in material, which is based on the PA effect that is the conversion of light waves to acoustic waves through optical absorption, localized thermal excitation, and localized pressure transient of tissue. PAI has been used in biomedical imaging areas such as various structural and functional imaging applications due to its high contrast and resolution.

In PAI technology, acoustic waves are generated simultaneously in the entire volume of irradiated tissue by a laser beam. To analyze the distribution of optical absorption in tissue using acoustic waves generated from the PA effect, the acoustic waves should be recorded on every element of an ultrasound transducer array simultaneously [34, 35, 64]. That is, pre-beamformed RF data is necessary to create a PA image. Recently, a SONIX-DAQ device (Ultrasonix Co.) allowing an access of pre-beamformed data was developed [54]. However, the SONIX-DAQ device and the provided control software from the manufacturer cannot provide pre-beamformed radio frequency (RF) data including the generated PA acoustic wave, in real time.

To address this current limitation, we propose a software framework for real-time pre-beamformed RF data acquisition with a clinical US transducer. Although here we apply the software specifically to PAI of brachytherapy seeds, it can also be used more generally in any application requiring pre-beamformed RF data [49, 50].

6.1.2. Methods

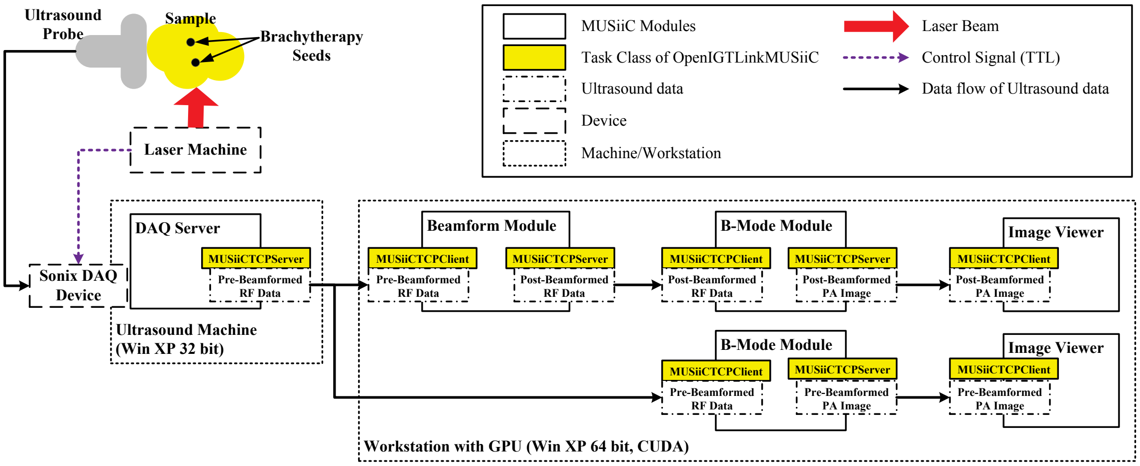


Figure 6.1. Block diagram of real-time photoacoustic imaging system, showing pre-beamformed and post-beamformed PA images.

Figure 6.1 shows an overview of our system for acquiring PA images in real-time. Our system can be divided into two groups: US imaging system and laser system. These two systems are synchronized by a Transistor-Transistor Logic (TTL) control signal from the laser system. The software framework of the ultrasound imaging part is based on *MUSiiC* modules and *OpenIGTLinkMUSiiC* [47, 72, 94].

The software framework of the photoacoustic imaging system is composed of multiple modules from the *MUSiiCToolkit*: *MUSiiC-DAQServer*, *MUSiiC-Beamformer*, *MUSiiC-BMode* and *MUSiiC-ImageViewer*. These *MUSiiC* modules are based on network-distribution systems and multithreaded architecture with communication accomplished using *OpenIGTLinkMUSiiC* [47] message types.

Figure 6.2 shows the block diagram of our phantom experiment. The goal of our experiment is to generate post-beamformed PA images and pre-beamformed PA images simultaneously. Acquired pre-beamformed RF data with *MUSiiC-DAQServer* is sent to two data-processing pipelines: the post- and the pre-beamformed PA imaging pipelines. The *Beamformer* module in the post-beamformed PAI pipeline converts pre-beamformed RF data to post-beamformed RF data with an aperture size of 32 using a standard delay-and-sum algorithm. Then, the *MUSiiC-BMode* module of both pipelines generates US B-mode images from RF data [49, 50].

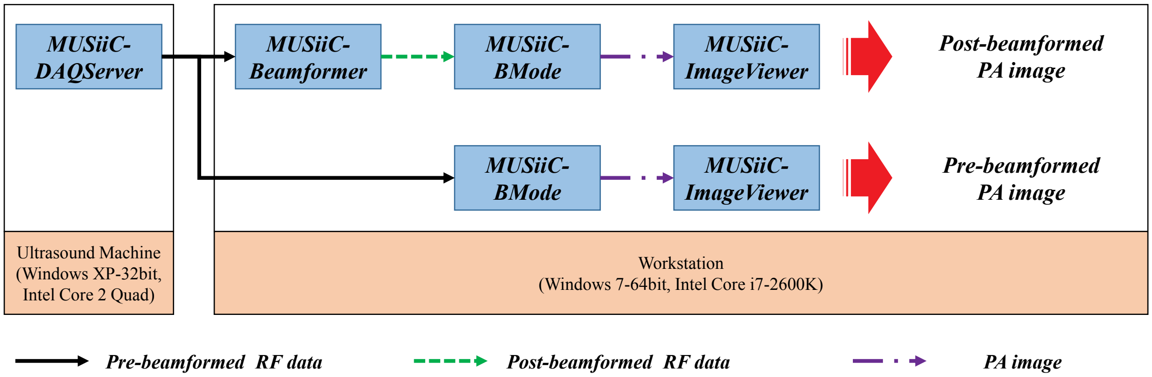


Figure 6.2. Block diagram of phantom experiment

Figure 6.3 shows the system setup for a real-time pre- / post-beamformed PAI experiment. For collecting pre-beamformed RF data, we used the ultrasound machine

SONIX-CEP (Ultrasonix Co.) and the data acquisition device SONIX-DAQ (Ultrasonix Co.), as was mentioned above. Also, the laser system Brilliant Nd:YAG (QUANTEL.), was used to generate the photoacoustic effect in the plastic phantom. The data-acquisition device (SONIX-DAQ) and the laser system are synchronized by a TTL control signal from the controller box of the laser system. The brachytherapy seed that is implanted into the plastic phantom is a decayed Palladium-103 seed (Theragenics Co.) encapsulated in a titanium cylindrical shell 0.8 mm in diameter and 4.5 mm in long.

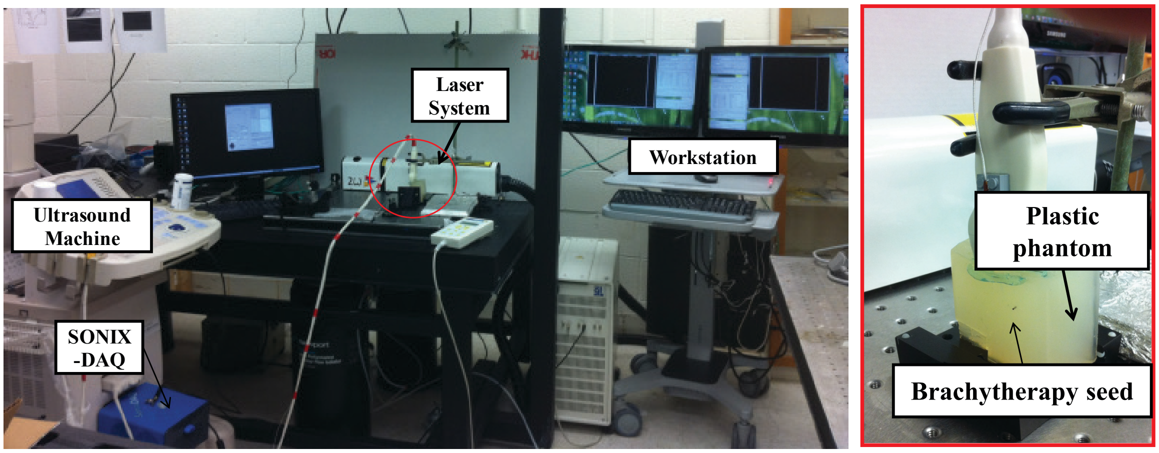


Figure 6.3. System setup of phantom experiment

6.1.3. Results

In our phantom experiment, the *MUSiiC-DAQServer* module has been tested on the SONIX-CEP (Windows XP-32 bit, Intel Core 2 Quad), and the *MUSiiC-BeamFormer*, *MUSiiC-BMode*, and *MUSiiC-ImageViewer* modules were executed on a Pentium 4 PC workstation (Windows XP 64bit Intel i7). The *MUSiiC-DAQServer* collected and sent out pre-beamformed RF data with around 3 frames per second (fps), and then the *MUSiiC-BeamFormer* and *MUSiiC-BMode* modules generated PA images from the acquired pre-beamformed RF data. Therefore, the overall performance (fps) of our system is around 3 fps. This performance depends on the depth of a PA image.

Figure 6.4 shows the pre- / post-beamformed PA images that are generated simultaneously from our real-time PAI system. In the PA images, the lower white spot represents the brachytherapy seed in the plastic phantom. The upper white line is an echo off the phantom wall. The location of the lower white spot corresponds with the position of the brachytherapy seed in the plastic phantom.

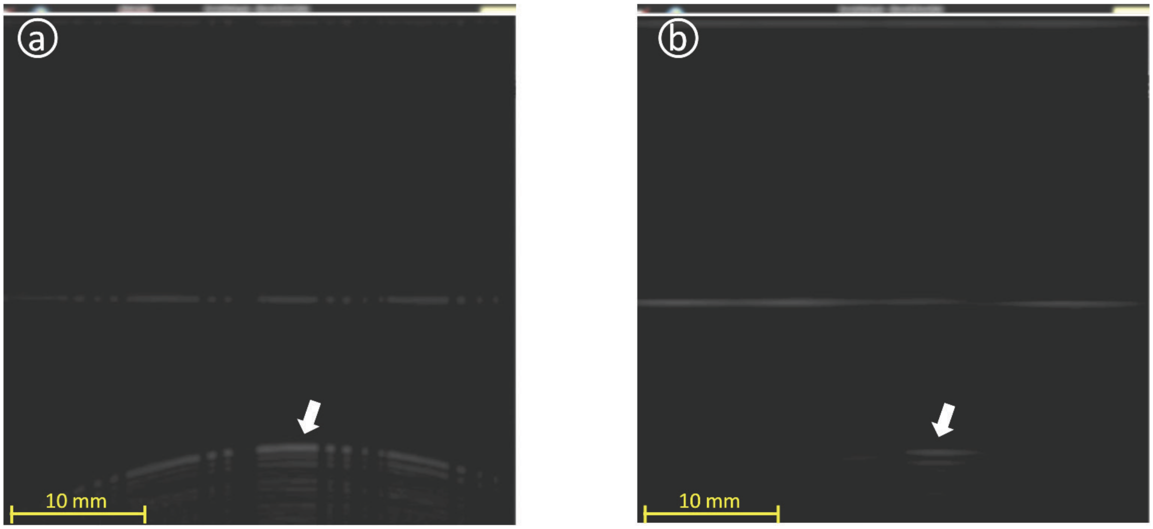


Figure 6.4. Photoacoustic images (width: 38 mm, height 38.5 mm) of a single brachytherapy seed (white arrow): (a) pre-beamformed photoacoustic image, (b) post-beamformed photoacoustic image

6.1.4. Conclusion

In this research, we developed a software framework that is composed of specialized executable programs for a real-time PAI system. The software framework enables the collection of pre-beamformed RF data and generates PA images from the collected data in real-time. Moreover, multiple programs related with data-processing and visualization modules (*MUSiiC-BeamFormer*, *MUSiiC-BMode*, and *MUSiiC-ImageViewer*) can be connected to the US data provider (*MUSiiC-DAQServer* module) due to the functionality of the *MUSiiCTCPServer* [47, 72], which supports multiple-client connections. By implementing several different beamforming algorithms on the *MUSiiC-Beamformer* module of each program set, different PA images are easily generated and compared with each other in real-time. In addition, we demonstrated a PAI of brachytherapy seeds with our proposed software framework in this research, but the software framework can be easily reconfigured and applied to any application that requires the streaming of pre-beamformed RF data.

6.2. Needle visualization using photoacoustic effect

This work was achieved with the support of the following collaborators and was submitted under the following citation:

Hyun Jae Kang, Xiaoyu Guo, Alexis Cheng, Michael Choti, and Emad M. Boctor, "Needle visualization using photoacoustic effect," in SPIE BiOS, 2015.

Abstract

We investigated a novel needle visualization using the PA effect to enhance needle-tip tracking. An optical fiber and laser source are used to generate acoustic waves inside the needle with the PA effect. Acoustic waves are generated along the needle. Some acoustic energy leaks into the surrounding material. The leakage of acoustic waves is captured by a conventional US transducer and US channel data collection system. Then, the collected data are converted to a PA image. The needle-tip can be visualized more clearly in this PA image than a general US brightness mode image.

6.2.1. Motivation

Medical US imaging is frequently used for interventional tool tracking in the operation room since it provides real-time data visualization, is mobile, and is harmless for the patient and the operator. Moreover, needle visualization and localization are very important for the safety and success of needle biopsy surgery or thermal ablation surgery. However, the exact localization of a needle-tip is difficult to achieve because the needle usually generates noise artifacts such as refraction and shadowing on the US B-mode image [51, 130, 131].

To address these limitations, several approaches including the beam steering method, vision tracking techniques, optical spatial tracking devices and EM spatial tracking devices have been investigated [130-137]. The basic principle of the beam steering method is to search for a transmitted US beam direction that optimizes the reflection from the shaft of a needle [130]. This method has been developed and validated by several research groups, and is implemented in a commercial US system (SonoSite Inc.) [138]. However, this approach has been proven to be effective only when a needle is in-plane of the US B-mode image with small insertion angle [51]. Optical spatial tracking methods estimate the position of a needle's tip by using attached optical markers at the needle, a stereo camera system to compute the position of the optical makers, and pre-computed calibration data representing the relationship between the optical marker and the needle's tip. Then, an estimated needle shape is overlaid on the US B-mode images in real-time [131]. However, this method has several challenges to overcome, including tracking error from needle bending, complexity of calibration, and clear line-of-sight from the camera to the optical markers [51]. On the other hand, needle visualization and estimation based on EM tracking devices have been implemented in several commercial US machines (GE LOGIQ E9, Ultrasonix GPS, etc.) due to the small sensor size, no camera line-of-sight requirement, and the capability of off-plane detection [132]. Nonetheless, this method has a limitation of tracking accuracy affected by ferromagnetic environments [51].

Recently, an approach of needle guidance with a biopsy needle including an optical fiber to deliver the laser source and PA effects was introduced [36]. This method enables the estimation of a needle's position and characterization of a target tissue. However, in this method, multiple PA frames are required to retrieve the shape of the needle. To overcome this limitation, we propose a novel needle visualization method using a single PA image and an image-processing algorithm.

6.2.2. Methods

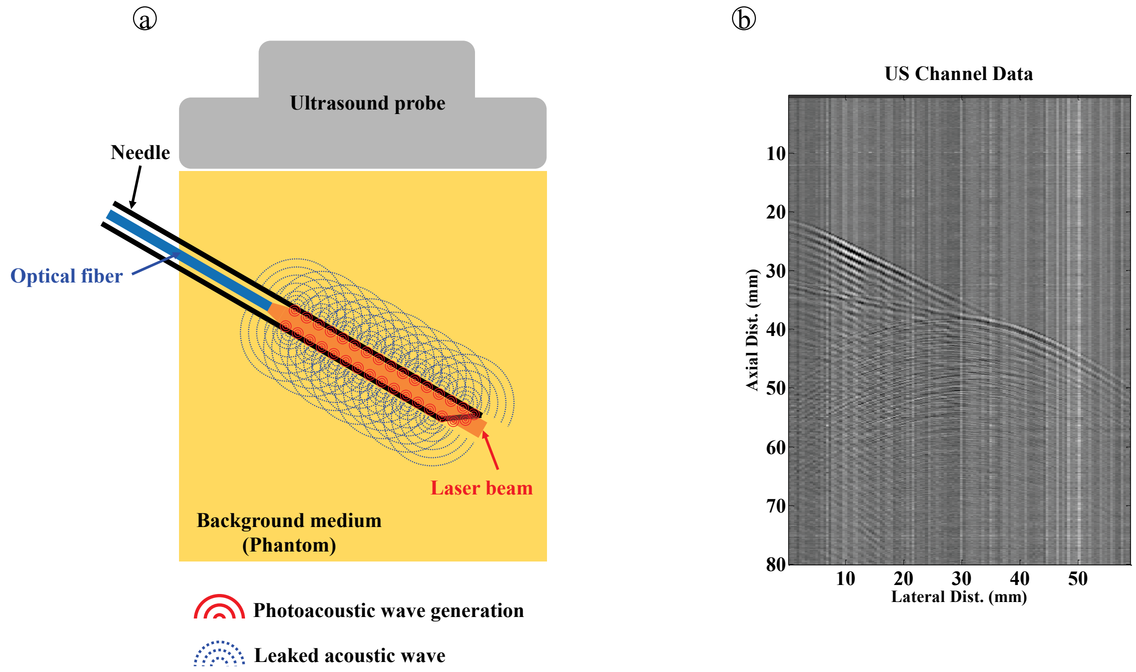


Figure 6.5. (a) Concept diagram of needle visualization using PA effects, and (b) US channel data represents leaked acoustic waves

Figure 6.5 (a) represents the concept diagram of needle visualization using the PA effect. In this research, an optical fiber and laser source are used to generate acoustic waves inside a needle with the PA effect. The PA phenomenon converts light energy to acoustic waves through optical absorption, localized thermal excitation, and localized pressure transient of a target material. Contrary to piezoelectric materials, this method does not require high driving voltage to generate acoustic pulses. Acoustic waves are generated at the needle's inner surface with the emission of the laser from the optical fiber, as shown in **Figure 6.5 (a)**. Some acoustic energy leaks into the surrounding material. A needle's shape can be estimated with the leaked acoustic waves captured by a conventional US transducer and US channel data collection system.

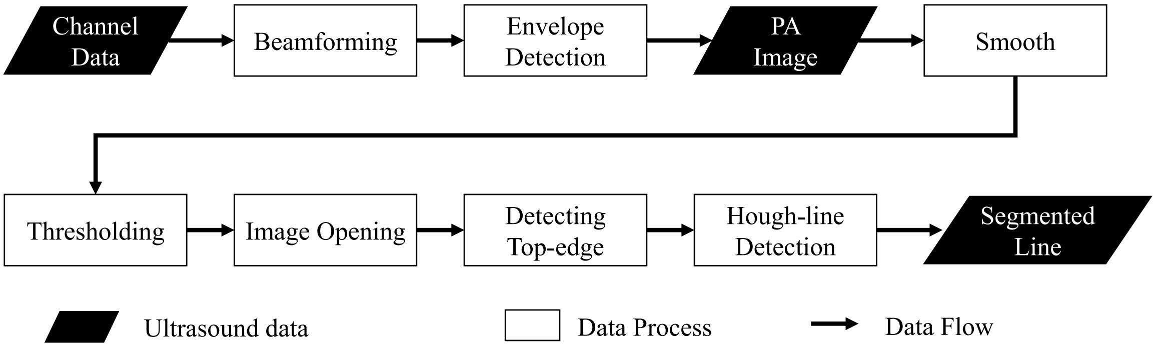


Figure 6.6. Flowchart of image processing pipeline for needle visualization with PA effect

However, a single frame of PA channel data is not enough to retrieve the shape of an inserted needle due to the presence of significant artifacts, as seen in **Figure 6.5 (b)**. Therefore, we suggest the following image-processing pipeline (see **Figure 6.6**). A 2D linear fast Fourier transformation (FFT) reconstruction method (kspaceLineRecon) of the K-wave toolkit [139] and an envelope detection algorithm based on Hilbert transform [55] are applied to the collected channel data to generate a PA image. Then, the PA image is smoothed by applying a Savitzky-Golay finite impulse response (FIR) smoothing filter [140] on the column-wise PA data. A column-wise thresholding algorithm based on Otsu's method [141] binarizes the smoothed PA image. After that, small objects on the binarized PA image are removed by an image-opening morphological operation [142]. Finally, top-edge detection and Hough-line detection algorithms [143] estimate the upper shape of the needle.

Figure 6.7 shows the system components and workflow for needle visualization with the PA effect. A Q-switched Nd:YAG laser (Quantel) operating with a wavelength of 1064 nm and a peak energy of 5.33 mJ, and a multimode optical fiber (THORLABS, Inc.) with a numerical aperture of 0.39 and a core diameter of 1000 um generated PA waves at the inside surface of an inserted needle (16G size) in our customized phantom (10 % of gelatin and 1% of titanium oxide).

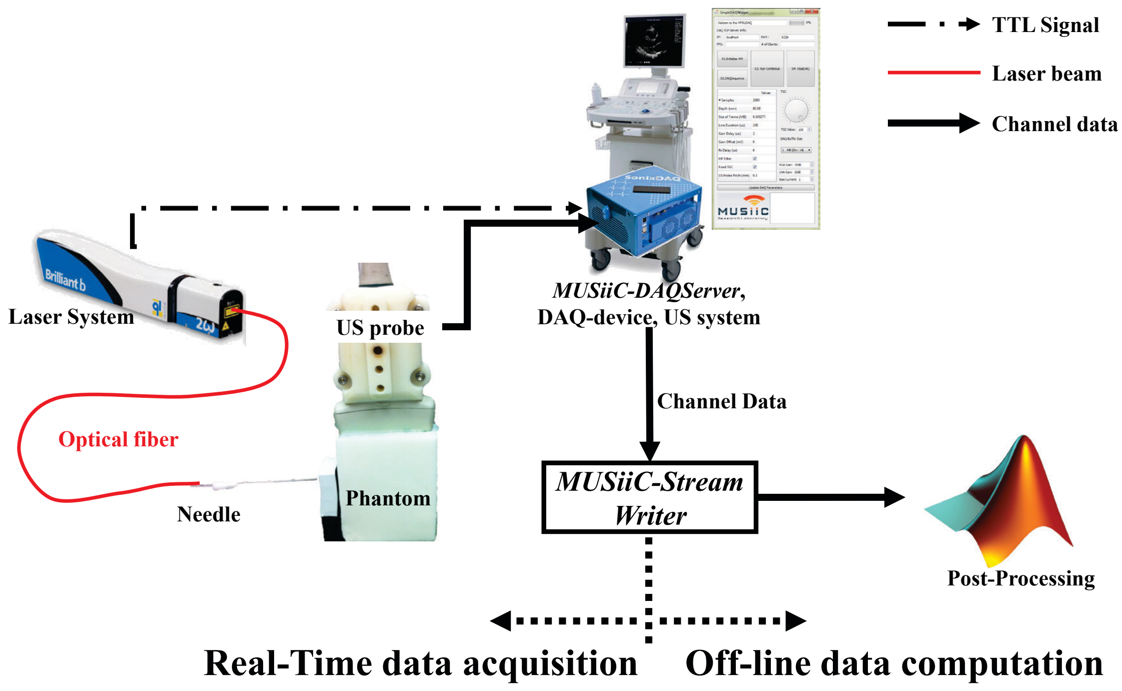


Figure 6.7. System components and workflow for a needle visualization with PA effects

A Sonix-CEP (Ultrasonix Co.) US system, a L14-5W/60 US Probe (Ultrasonix Co.), and a SonixDAQ device (Ultrasonix Co.) captured the generated PA waves and sent them to our *MUSiiC* modules [47, 49, 50, 72, 94], which contain a *MUSiiC-DAQServer* [49, 50] to acquire PA channel data frames in real-time and measures the time stamp of each acquired PA frame, and *MUSiiC-StreamWriter* software module to save the collected PA channel data to the local hard disk. Transistor-transistor logic (TTL) trigger signals from the laser system activate the SonixDAQ device and *MUSiiC-DAQServer* to begin collecting channel data. The proposed image-processing pipeline was applied off-line.

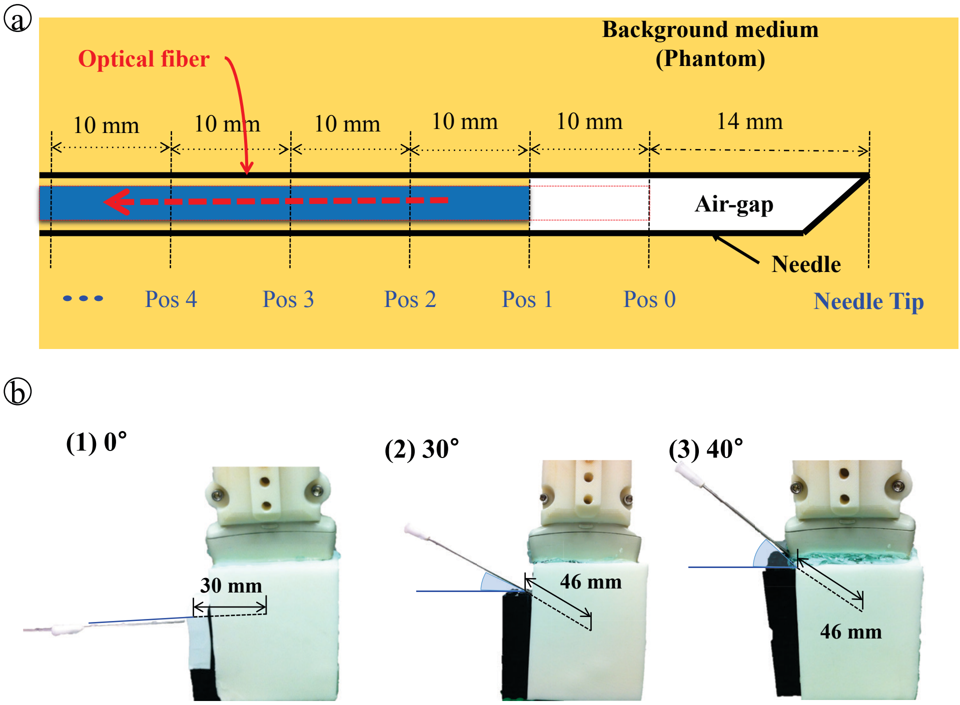


Figure 6.8. Experimental setup: (a) different positions of the optical fiber inside the needle, (b) three different insertion angles of needle: (0°, 30°, and 40°)

The optical fiber was initially positioned at the tip of the needle to prevent particles of the phantom from entering the needle during insertion. After needle insertion, the optical fiber was moved back 14 mm from the tip of the needle to generate an inner air-gap at the needle between the optical fiber and the phantom, as shown in the **Figure 6.8 (a)**. The optical fiber was pulled back with a step size of 10 mm for each data collection to determine the optimal position of the fiber inside the needle. Moreover, we investigated different insertion angles (0°, 30°, and 40°) to figure out the range of detectable insertion angles of the needle. Because the needle tip's position can be recognized with the generated PA waves from the surrounding phantom material at the needle's tip, we compared an estimated insertion angle of the needle to the insertion angles that we tried for evaluating our proposed method.

6.2.3. Results

Figure 6.9 shows PA images acquired at different optical fiber positions inside the needle with 40° of insertion angle. The size of each image is 60 mm and 80 mm in the lateral and axial dimensions, respectively. A tile of each image represents the position of the optical fiber inside the needle. The red arrow of each PA image indicates the PA wave created from the surrounding inner surface of the needle at the optical fiber's position.

In the case of 40° of insertion angle, the inserted length of the needle was 46 mm (see **Figure 6.8 (b)**). The optical fiber is within the US transducer scanning area in the images in the first row, while it is outside of the scanning area in the second and third rows.

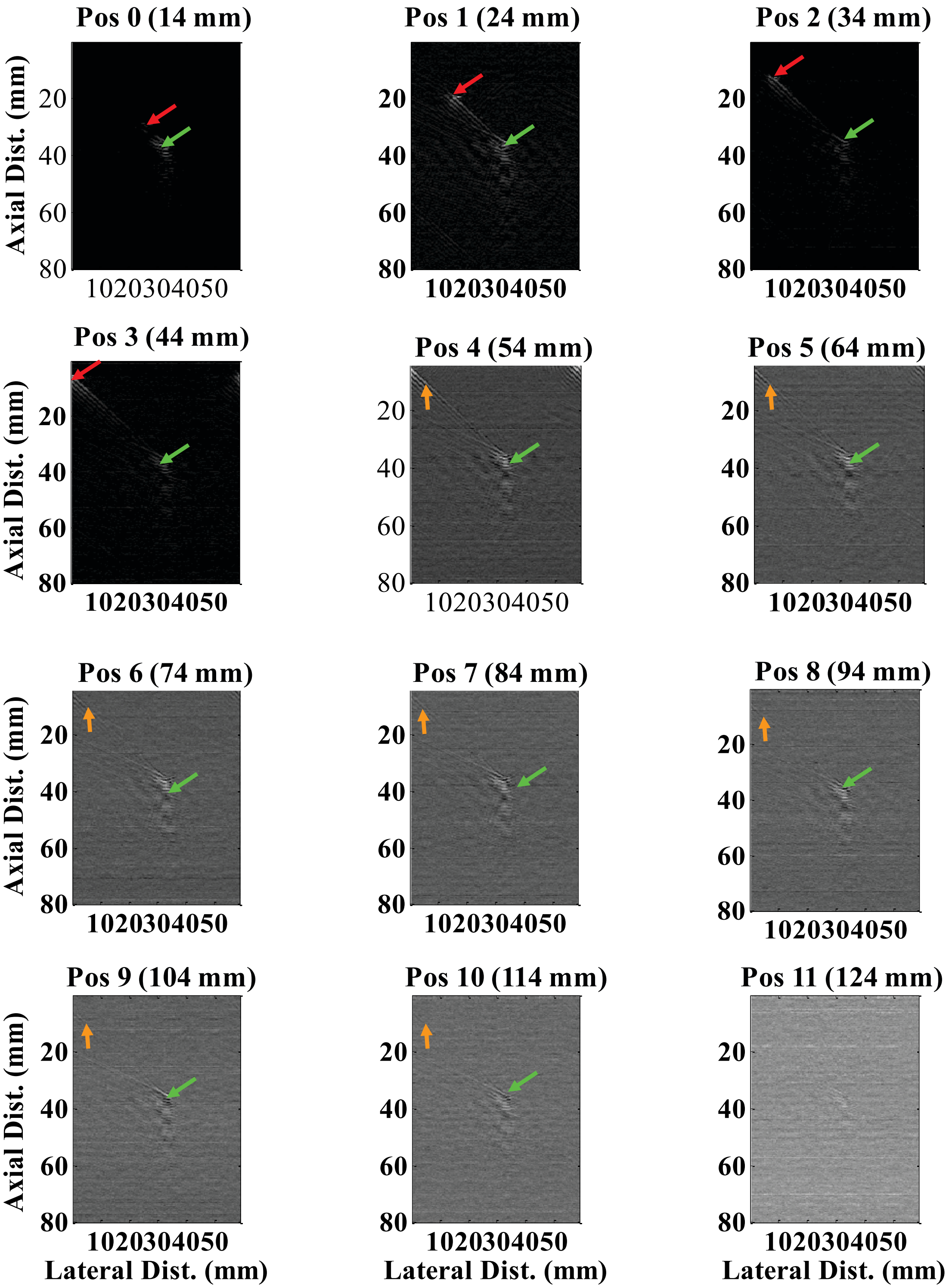


Figure 6.9. PA images acquired at different optical fiber positions inside the needle with 40° of insertion angle. A red and orange arrow of each PA image indicates generated and leaked PA waves, respectively, from inner surface of the needle at optical fiber's position. A green arrow of each image represents generated PA wave generated at surrounding phantom material at the needle's tip.

Although the optical fiber was outside the PA scanning area, the traveling laser beam generated a PA wave (indicated by an orange arrow in each figure) with low signal intensity, and the leaked acoustic waves were captured by the US probe. Therefore, the images in the first row have higher contrast compared to the images in the second and third rows. On the other hand, the green arrow of each PA image represents the PA wave generated from the surrounding phantom material at the needle tip. Intensities of the PA waves indicated by green arrows are inversely proportional to the distance between the optical fiber and the needle tip, as shown in **Figure 6.9**.

Figure 6.10 shows the intermediate results of the proposed image-processing pipeline (see **Figure 6.6**). Each row indicates an insertion angle of the needle, and PA images in each row were acquired with 44 mm of distance between the optical fiber and the needle's tip. Each column shows a step in the image-processing pipeline with its corresponding intermediate image. Blue lines in the third column and yellow lines in the fourth column indicate the results of the Hough-line detection algorithm [55] and our line estimation, respectively. As shown in the third column of the figure, a weak PA signal representing the upper surface of the needle could be retrieved by applying a column-wise thresholding method instead of whole image binarizing with a threshold value. In addition, the step of detecting the upper edge of the needle was also computed with a column-wise and top-down searching methods. Finally, we estimated an insertion angle of the needle with the computed yellow lines in the seventh column. However, the proposed needle detection method for estimating each insertion angle only succeeded in the case of the following optical fiber positions: 24 mm, 34 mm, and 44 mm away from the needle tip (see **Figure 6.8**).

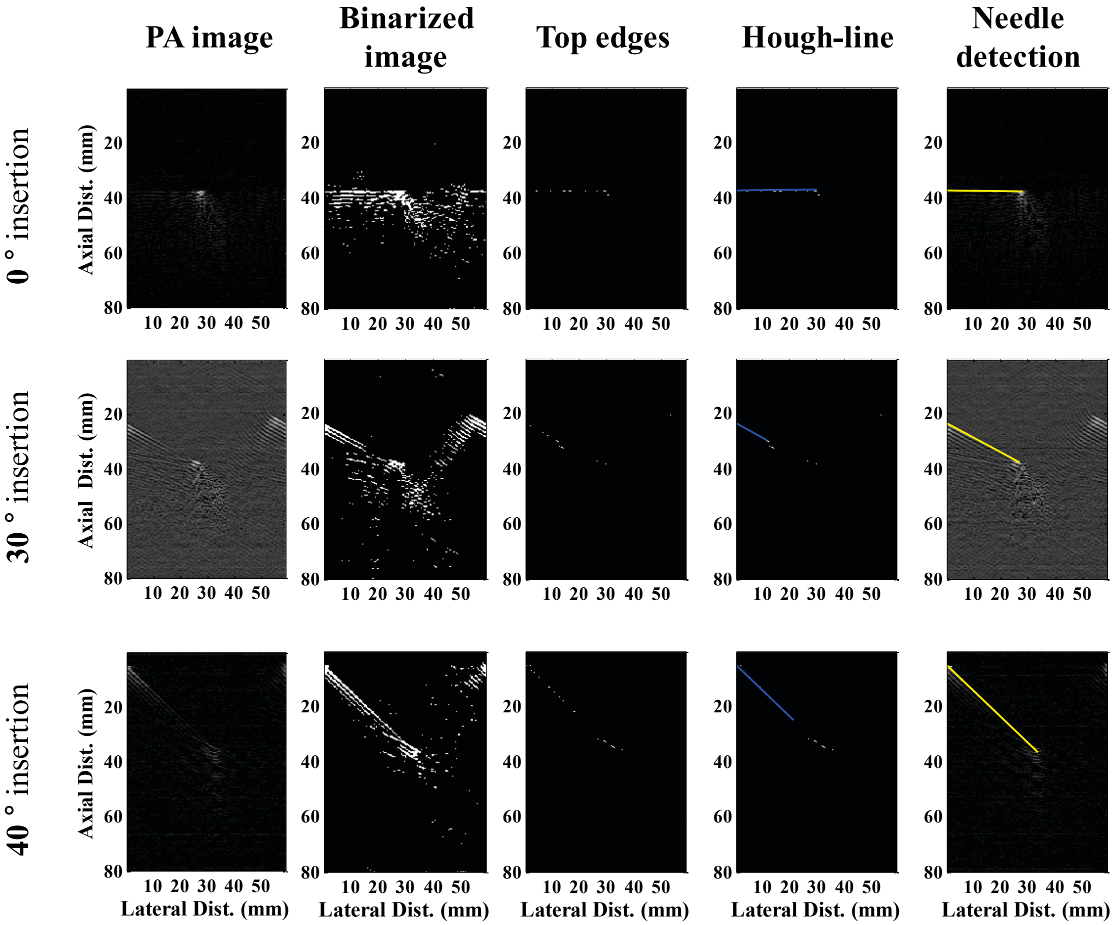


Figure 6.10. The intermediate results pipeline of the proposed image processing. Each row indicates insertion angle of the needle (0º, 30º, and 40º) and each column shows a step of image processing with a corresponding intermediate image of the pipeline.

Table 6.1 shows the results. In the case of 0° and 30° of needle insertion, estimated insertion angles have good results with 0.72º and 1.97º errors, respectively, at the 44 mm of distance between the optical fiber and the tip of the needle. The error of the computed insertion angle for 40° is 1.80° when the optical fiber is 34mm away from the needle tip.

Table 6.1. Estimated insertion angle of the needle according to position of the optical fiber

Insertion angle of the needle	Distance between the optical fiber and the needle's tip		
	24 mm	34 mm	44 mm
0º	2.26º	1.40º	0.72º
30º	28.03º	27.80º	28.72º
40º	45.66º	41.80º	43.56º

6.2.4. Conclusion

In this research, we proposed a new approach of needle visualization to estimate the insertion angle of a needle and visualize the inserted needle on a PA image. The method used only one single PA image generated with a needle including an optical fiber. To find the optimal position of the optical fiber inside the needle and detect the insertion angle of the needle, we collected multiple PA frames with different optical fiber positions with three different insertion angles (0º, 30º, and 40º). Then, the suggested image-processing pipeline was applied to the collected PA frames. Finally, we evaluated our approach by computing the error of each computed insertion angle.

Our method only succeeded when the optical fiber was located inside of the US probe's scanning area. However, the proposed approach could estimate the needle insertion angle for three different insertion angles with the optical fiber inside the PA imaging area. The errors of the computed needle insertion angle were 0.72º, 1.28º, and 1.80º for 0º, 30º, and 40º of needle insertion, respectively. In addition, the optimal position of the optical fiber inside the needle was 34 mm, 34 mm, and 44 mm away from the needle tip for 0º, 30º, and 40º of the insertion angle, respectively. Considering the alignment between US probe and the phantom and an unavoidable small motion during the experiment, these errors could be acceptable.

Since the proposed method uses a single PA image for estimating the needle's position and insertion angle, this approach can provide real-time needle position information for clinical applications such as needle biopsy surgery or thermal ablation surgery. Therefore, we expect this method can be integrated with a PA image-guided therapy system.

6.3. Image guided assistance system for Photoacoustic imaging

This work was accomplished with the aid of the following collaborators and was submitted under the following citation:

Sungmin Kim[1], Hyun Jae Kang, Alexis Cheng, Muyinatu A. Lediju Bell, E. Boctor, and Peter Kazanzides, "Photoacoustic image guidance for robot-assisted skull base surgery," in Robotics and Automation (ICRA), 2015 IEEE International Conference on, 2015, pp. 592-597.

Abstract

Recently, photoacoustic imaging for interventional guidance of endonasal surgery has been studied. The goal of the research is to use PA images to visualize blood vessels surrounded by bone that are invisible in US B-mode images. Aligning the US probe, the light source, and the target is essential to acquire a proper PA image including the target. To address this challenge, we proposed an image-guided assistance system for PA imaging. The system is composed of an US imaging system, a spatial tracking system, and a PA navigation software module. The data acquisition system collects the US image stream and spatial tracking information of the US probe, the patient frame, and the surgical instrument/laser in real-time, and sends the collected data to the PA

[1] Sungmin Kim led this research and was main contributor who integrated and tested the photoacoustic image guidance for robot-assisted skull base surgery [33].

navigation module to provide visual guidance. In addition, we investigated the accuracy of the proposed system with a phantom experiment.

6.3.1. Motivation

Transnasal transphenoidal surgeries (TSS) are minimally invasive procedures to remove pituitary tumors with a real-time video stream from a flexible endoscope, and surgical instruments such as a drill and a curette for sphenoidal bone removal and tumor resection, respectively [144, 145]. However, an endoscopic system supplies only the visual information of a target object's surface features, not its inner anatomical structures. Blood vessels hidden behind the sphenoidal bone cannot be visualized on the endoscopic image [146]. Moreover, image streams acquired by the endoscopic system are not enough to provide accurate localization of the surgical instruments. A surgical navigation system can provide the information of inner anatomical structure with pre-/intra-operative medical imaging modalities (x-ray, CT, MRI, or ultrasound) and accurate surgical instrument pose information with optical or electromagnetic spatial tracking systems [146-149].

Among the medical imaging modalities used with surgical navigation systems, clinical ultrasound (US) B-mode images are frequently used in the operation room due to its real-time imaging, ease-of-use, non-ionizing radiation, and mobility. In addition, US B-mode images can provide the surface and sub-surface anatomical information of a target object with 2D/3D images. However, US B-mode imaging has a limitation of visualizing an anatomic target object behind high acoustic impedance material such as bone because of the acoustic impedance mismatch [150-152].

To address the limitations of US B-mode images, Bell et al. proposed and investigated the approach of transcranial photoacoustic (PA) imaging for interventional guidance of endonasal surgeries [151]. Their results demonstrated the feasibility of PA image-guidance by detecting anatomical targets in a PA image based on a geometrical arrangement of the laser beam, an anatomical target, and a US transducer. However, the geometric alignment is not easy without pose information of the three components. To address this challenge, we propose an image-guided assistance system for PA imaging with a spatially-tracked US probe, and the pose information of an anatomical target and laser beam. To evaluate our proposed system, we performed a phantom experiment.

6.3.2. Methods

Figure 6.11 shows components and corresponding coordinate frames of our proposed system, where, T_A^B denotes the transformation from the coordinate frame of B to A, and P_C represents a position based on the coordinate of C. An optical tracking system (OT) collects the pose information of an optical fiber holder (T_{OT}^{OH}), US transducer (T_{OT}^{UT}), and a dynamic reference frame (T_{OT}^{DRF}) in real-time. The position of the target object (TO) is defined in the coordinates of the phantom (PT). For a real clinical case, the coordinate system of 3D volume data from a pre-operative medical imaging modality such as CT or MRI can be used instead of PT. A transformation from DRF to PT (T_{DRF}^{PT}) is computed by a paired-point registration with the paired-positions of fiducial markers in the PA and OT coordinate systems. Therefore, the position of the target in the coordinate frame ($P(TO)_{DRF}$) can be estimated by equation 6.1.

$$P(TO)_{DRF} = T_{DRF}^{PT}\, P(TO)_{PT} \tag{6.1}$$

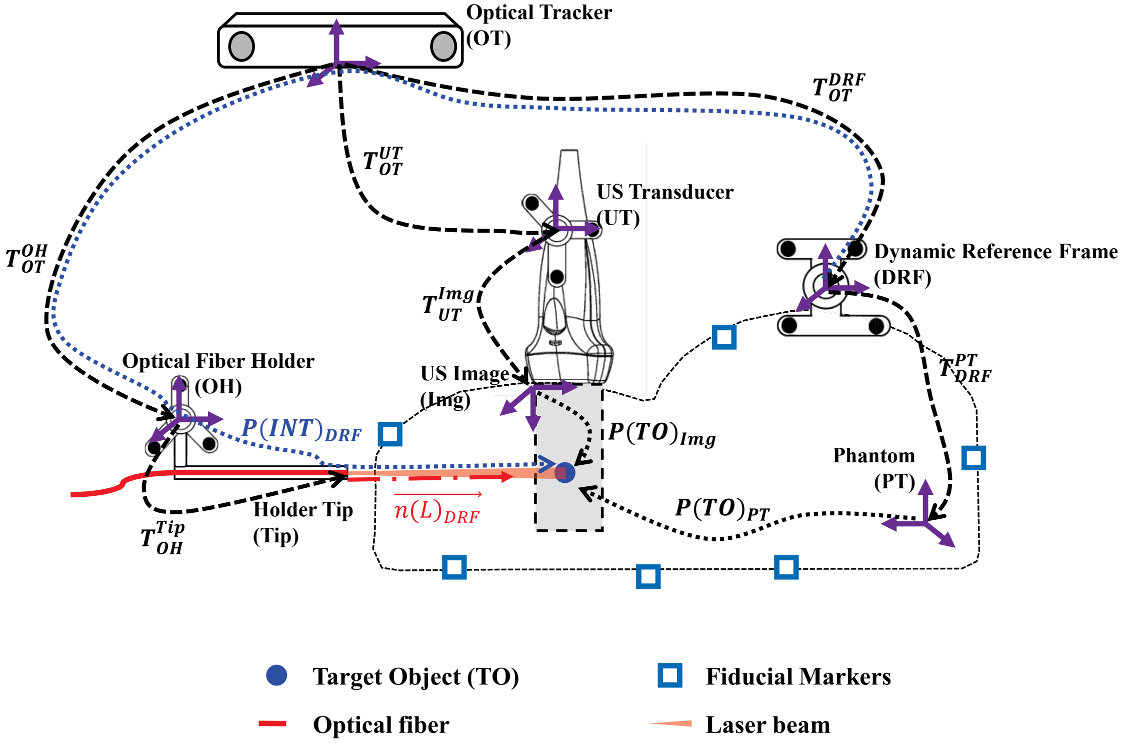


Figure 6.11. Components and corresponding coordinate frames of image-guided assistance system for PA imaging (Ref. [33])

The transformation from US probe to US image frame (T_{UT}^{Img}) is pre-computed with an US calibration method [153]. Consequently, the US image coordinate system with respect to the DRF coordinate system (T_{DRF}^{Img}) can be expressed by the following equation.

$$T_{DRF}^{Img} = (T_{OT}^{DRF})^{-1} T_{OT}^{UT}\, T_{UT}^{Img} \tag{6.2}$$

Where, $(T_{OT}^{DRF})^{-1}$ is the inverse transformation of T_{OT}^{DRF}. The first goal of our system is to provide guidance by enabling the image plane of the T_{DRF}^{Img} coordinate system to include a target point $(P(TO)_{DRF})$.

The transformation from optical fiber holder to the holder's tip (T_{DRF}^{Img}) can be estimated by a standard pivot calibration method [154]. Subsequently, the holder tip's coordinate frame based on the DRF coordinate frame can be represented as equation 6.3.

$$T_{DRF}^{Tip} = (T_{OT}^{DRF})^{-1} T_{OT}^{OH}\, T_{OH}^{Tip} \tag{6.3}$$

Then, the line direction vector $(\overrightarrow{n(L)_{DRF}})$ of the laser beam with respect to the DRF coordinate system can be calculated with geometric information of the optical fiber holder. Subsequently, an intersection (INT) point of the central axis of the laser beam with a US image plane $(P(INT)_{DRF})$ can be estimated with the line direction vector $(\overrightarrow{n(L)_{DRF}})$ and the US image (T_{DRF}^{Img}) coordinate frame. Another goal of our assistant system is aligning the intersection point with the target point $(P(TO)_{DRF})$.

Figure 6.12 illustrates the system setup and the workflow for our phantom experiment. A Q-switched Nd:YAG laser (Quantel) operated at a wavelength of 532 nm, and a multimode optical fiber (THORLABS, Inc.,) with a numerical aperture of 0.39 and a core diameter of 1000 um deliver the laser beam to our custom-built phantom for generating the PA effect. A Sonix-CEP (Ultrasonix Co.) US system with a L14-5W/38 linear US Probe (Ultrasonix Co.) and *MUSiiC-RFServer* [41, 47, 72, 94] collected real-time US B-mode data and sent to our PA navigation software module with *ImageMessage* of *OpenIGTLink* [45] via the implemented *MUSiiCTCPServer* [47, 94].

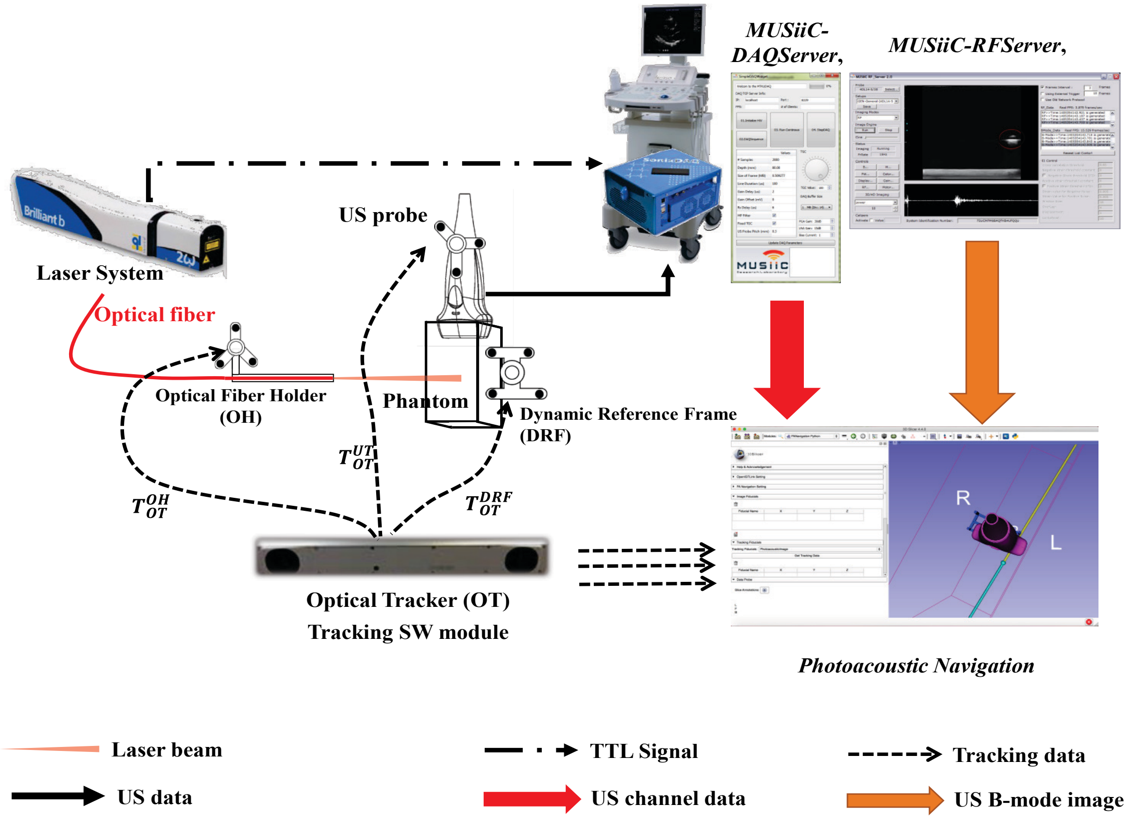


Figure 6.12. System setup and workflow for a phantom experiment

In addition, a SonixDAQ device (Ultrasonix Co.) and *MUSiiC-DAQServer* [49, 50] captured the generated PA acoustic wave with the US probe, and transferred the collected data to the PA navigation module with *USMessage* of *OpenIGTLinkMUSiiC* [47] and *MUSiiCTCPServer* [47, 72] in real-time. PA data acquisition was controlled by a transistor-transistor logic (TTL) signal from the laser system. During US data collection, an optical tracking system (Atracsys LLC.) and a customized tracking software module acquired the pose information of T_{OT}^{OH}, T_{OT}^{UT}, and T_{OT}^{DRF}, and then sent it to the PA navigation module with a *TrackingMessage* of OpenIGTLink [45] in real-time. The tracking software module was implemented with an application program interface (API) provided by the manufacturer, the cisst library [76], and the surgical assistant workstation (SAW) component [111-113].

The PA navigation module is a loadable module in 3D Slicer [44, 80], which is written in Python. This module updates and calculates the pose information of $P(TO)_{DRF}$, T_{DRF}^{Tip}, and $P(INT)_{DRF}$ with received tracking information of T_{OT}^{OH}, T_{OT}^{UT}, and T_{OT}^{DRF} through the *OpenIGTLink* interface module (*OpenIGTLinkIF*) [155] of 3D Slicer, and pre-computed calibration information (T_{DRF}^{PT}, T_{UT}^{Img}, and T_{OH}^{Tip}) in real-time. In addition, the software module provides visual guidance by using received US images (B-Mode or PA channel frame) via *OpenIGTLinkIF* and the computed pose information of $P(TO)_{DRF}$, T_{DRF}^{Tip}, and $P(INT)_{DRF}$.

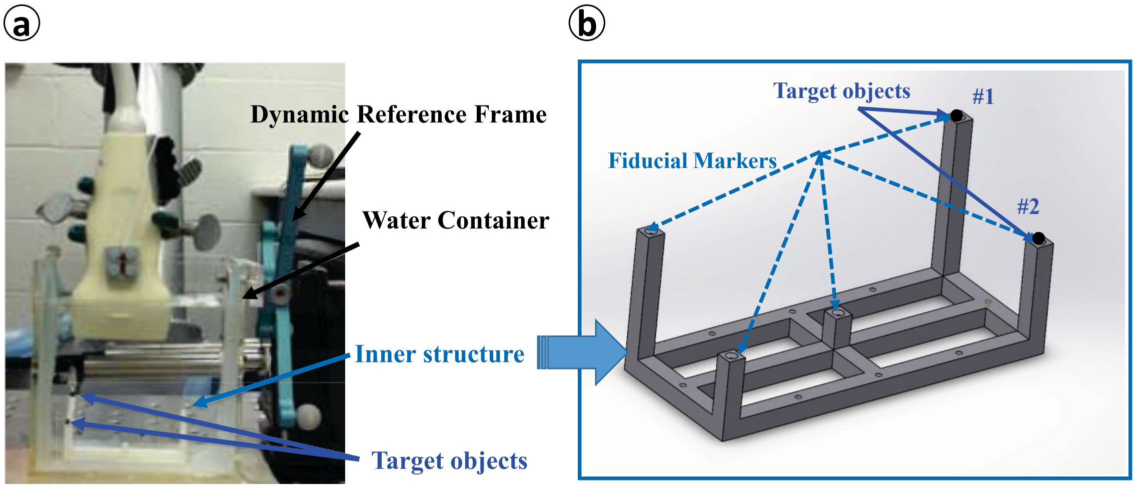


Figure 6.13. (a) Phantom setup for an evaluation experiment of the image-guided assistant system, and (b) the 3D model of the phantom (Ref. [33])

Our custom-built phantom (see **Figure 6.13** (**b**)) and a DRF were fixed in and on, respectively, a transparent water tank to compensate for unexpected motion of the optical tracking system or phantom as shown in **Figure 6.13** (**a**). The phantom was composed of five square pillars with different heights (see **Figure 6.13** (**b**)). In addition, there was a hemispherical concavity on each pillar. These were used as fiducial markers to estimate the transformation from DRF to the phantom coordinate (T_{DRF}^{PT}) and also to hold two target objects that are black rubber spheres with 2.4 mm diameter to generate the PA effect easily. The phantom coordinate frame was based on its 3D model coordinates. During the experiment, US B-mode and PA channel data were collected at the same target positions separately.

6.3.3. Results

Figure 6.14 shows the snapshot images of the visual guidance provided by the PA navigation software module with the real-time PA channel frame, the optical fiber holder tip, the estimated laser beam and the 3D model of our phantom. In the figure, the PA signal in the PA frame, the target object of the 3D model and the laser beam are close to each other with small error.

Figure 6.15 illustrates the acquired US B-mode, PA, and the overlaid image with each target point. The delay-and-sum beamforming algorithm with an aperture size of 33 and a predefined speed of sound (1540 m/sec) was applied to the collected PA channel frame to reconstruct PA images [68, 156, 157]. The size of each image is 38 mm and 40 mm in the lateral and axial direction, respectively. The red circle in each image indicates the corresponding target.

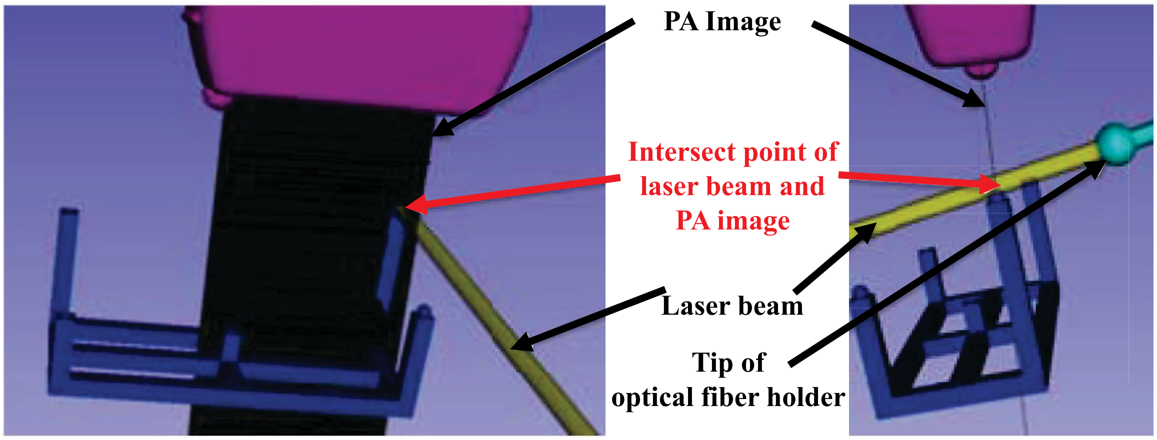

PA Image
Intersect point of laser beam and PA image
Laser beam
Tip of optical fiber holder

Figure 6.14. Visual guidance provided by PA navigation software module with real-time PA channel frame, the tip of the optical fiber holder, estimated laser beam and the 3D model of the phantom (Ref. [33]). Note that this visualization shows the laser beam as a cylinder, whereas in reality the laser beam will have a conical shape due to light diffusion in tissue.

For a more detailed evaluation of the visual guidance's accuracy, we compared the position of the PA signal and the laser beam's intersection point with the plane of the PA image to the position of the target object. The position of the PA signal was defined as the location with the highest intensity value. All positions were computed based on the DRF coordinate frame and represented in the PA image axes. **Table 6.2** shows the errors of this comparison with mm units. Since the intersection point and the PA signal are both estimated on the PA imaging plane, the elevation error of those points are the same.

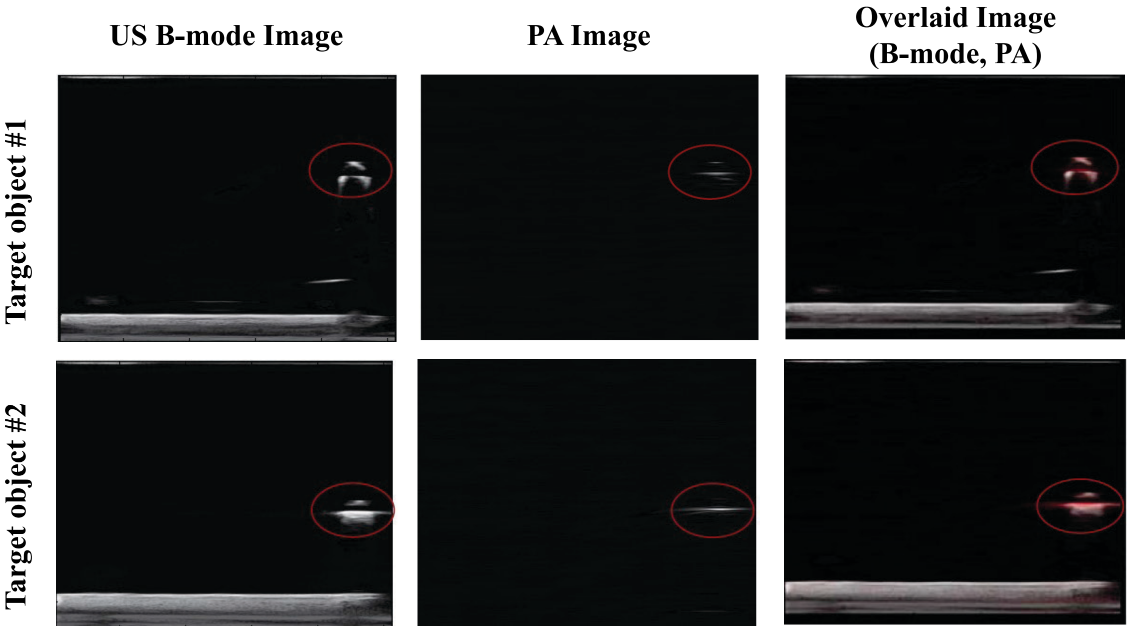


Figure 6.15. Acquired US B-mode, PA, the overlaid image from each target point

Table 6.2. The computed position errors of PA signal and intersection point of the laser beam and PA image's plane with respect to the position of target objects with mm units (Ref. [33])

Target Object	Intersection Point			PA signal		
	Lateral	Axial	Elevational	Lateral	Axial	Elevational
#1	0.12	0.32	0.079	0.7	-1.99	0.079
#2	-0.92	-0.16	1.332	0.11	-2.74	1.332

6.3.4. Conclusion

In this research, we introduced an image-guided assistance system for PA imaging and demonstrated its effectiveness for aligning the laser beam and the US probe to a target point in a phantom experiment. The system utilized real-time spatial-tracking information representing the pose of the laser beam, US probe, and the target object, and a 3D model of the target object to provide visual guidance. In addition, the PA navigation module of the system supports a real-time visual feedback by displaying a real-time US image overlaid with the estimated position of the US probe, laser beam, and the target point.

The accuracy of the provided visual guidance was investigated by computing position errors of an estimated intersection point of the laser beam, US image's plane and a detected PA signal in relation with the position of a target object. The results are represented in Table 2. Specifically, the PA signal position errors in the axial dimension are larger than those of the intersection point. This result can be caused by several factors such as US probe calibration errors (T_{UT}^{Img}), tracking system errors, synchronization of the PA image with its corresponding tracking information, and a mismatch in the speed of sound for the beamforming algorithm. We need more investigation to reduce these position errors of the PA signal. Except for the axial position error of the PA signal, our system can provide an accuracy of 1 mm for visual guidance.

Since our image-guided assistance system was designed for PA imaging, we can apply the system to clinical applications such as transcranial PA imaging for interventional guidance of endonasal surgery [151] and brachytherapy seed localization with PA images [25, 26, 50]. Moreover, the proposed system already includes IGT system technologies such as spatial-tracking, transformation of pose information to a target coordinate system, paired-point registration between pre-defined 3D volume data and real-time collected pose information, overlaying of images and 3D objects, and a network interface for exchanging data between different software modules. Therefore, this system can be easily integrated with existing IGT systems.

6.4. Conclusion

This chapter introduced novel applications of interventional PA imaging achieved by our framework. The first application is visualization of brachytherapy seeds with PA imaging. The pose of the brachytherapy seed with respect to the prostate is important in prostate brachytherapy. However, conventional US B-Mode imaging is insufficient for visualizing seeds due to noise artifacts. To address this challenge, we utilized PA imaging to visualize and localize a brachytherapy seed. PA waves generated by an individual seed by laser-beam targeting were captured with a clinical US imaging system and a *MUSiiC-DAQServer* module. Consequently, the captured US channel data was converted to a PA image with the *MUSiiC-Beamformer* and *MUSiiC-BMode* modules.

The second application is needle visualization using the PA effect. Needle visualization and localization are important in minimally invasive surgical procedures such as needle biopsies and ablation therapies. However, conventional US B-Mode imaging is not enough for the exact localization of a needle-tip due to its noise artifacts such as acoustic refraction and shadowing. To overcome these limitations, we investigated a new method of needle visualization using a single PA image, a needle with an attached optical fiber and an image-processing algorithm. The main idea of our method is to utilize the leaked acoustic waves to retrieve the needle's shape. To find the optimal position of the optical fiber inside the needle, we changed the position of the optical fiber for every data collection. We also tried three different insertion angles to determine a range of detectable needle insertion angles. To evaluate our proposed method, we compared the estimated needle insertion angle to the measured insertion angle.

Another application is an image-guided assistance system for PA imaging. Aligning a US probe, a target point of the light source, and a target object is essential to acquire a proper PA imaging including the target. However, this alignment is difficult without any guidance. To address this problem, we built an image-guided assistance system for PA imaging with pose information of the US probe, the optical fiber holder indicating the target point of the light source, and the target object. The assistance system is composed of *MUSiiC-DAQServer*, *MUSiiC-RFServer*, an external tracking software connected with an optical tracking system, and a custom-built 3D Slicer module. *MUSiiC-DAQServer* and *MUSiiC-RFServer* send real-time US channel data and B-Mode image stream, respectively, to the 3D Slicer module. The custom-built 3D Slicer module provides visual guidance to align the US probe, the laser-beam and the target object with the pose information collected by the external tracking software. In this research, we demonstrated the compatibility of our software framework with existing IGT software.

Recapitulation of thesis contribution
• Study of the localization of prostate brachytherapy seeds with a clinical ultrasound system and the PA effect, and demonstrating the feasibility of target localization with PA images.
• Proposing and inventing a novel needle detection method with a single PA image, a needle with an attached optical fiber, and a customized needle detection algorithm.

Chapter 7:

Spatial angular compounding of photoacoustic images

This work was accomplished with the aid of the following collaborators and is in preparation for submission.

Hyun Jae Kang, Muyinatu A Lediju Bell[1], Xiaoyu Guo, and Emad M. Boctor

Abstract

Photoacoustic (PA) images utilize pulsed lasers and ultrasound transducers to visualize targets with higher optical absorption than the surrounding medium. However, they are susceptible to acoustic clutter and background noise artifacts that obfuscate structures of interest. We investigated three spatial-angular compounding methods to improve PA image quality, implemented by combining multiple images acquired as an ultrasound probe was rotated about the elevational axis with the laser beam and target fixed. Compounding with conventional averaging was based on the pose information of each PA image, while compounding with weighted- and selective-averaging utilized both the pose and image content information. Weighted-average compounding enhanced PA images with the least distortion of signal size, particularly when there were large (i.e. 2.5

[1] Thank to Muyinatu A Lediju Bell for her help with editing and offering suggestions for the improvement of this section.

mm and 7°) perturbations from this initial probe position. Selective-averaging offered the best improvement in image quality with up to 181, 1665, and 1568 times higher contrast, CNR, and SNR, respectively, compared to the mean values of individual PA images. The proposed spatial compounding methods have significant potential to enhance image quality in multiple photoacoustic applications.

7.1. Introduction

Photoacoustic (PA) imaging has achieved expansive growth in potential applications, clinical utility, and equipment configurations within the past decade. It is based on the photoacoustic effect, excited through localized light transmission, absorption, thermal expansion, and a resulting pressure transient whose amplitude relies on the optical, thermal, and mechanical properties of the target material [27, 34, 158]. The resulting photoacoustic image represents optical absorption differences in the target region and has potential to detect breast cancer [159], identify atherosclerotic plaques [160], monitor thermal therapy [124], and localize medical implants such as brachytherapy seeds [25, 26, 37, 68]. However, similar to ultrasound (US) imaging imaging [161-163], PA images suffer from noise artifacts such as acoustic clutter or reverberation that reduce image quality - e.g., contrast, signal-to-noise ratio (SNR), and contrast-to-noise ratio (CNR) [67, 70, 161-164]. In addition, PA images have poor contrast, SNR and CNR when a low-energy laser source such as a laser diode is used to generate the photoacoustic effect [165].

Signal averaging is commonly used to improve the SNR of PA images when the ultrasound transducer, light source, and target are fixed in the same position [70, 165]. However, this type of averaging reduces frame rates for real-time imaging applications and has limited ability to reduce statistically dependent background noise [90, 166]. In addition, this approach may not be suitable for the clinical environment when the ultrasound probe is hand-held as subtle motions are difficult to avoid. Thus, averaging multiple PA images from different hand-held scans without motion compensation degrades the temporal and spatial resolution of images [167].

To overcome the limitations of frame averaging, advanced methods that parallel advances in US have been investigated. Adaptive photoacoustic beamforming methods, similar to adaptive US beamforming [168], were investigated to improve the lateral resolution and quality of PA images [169]. However, they suffer from suboptimal performance when the SNR is low [170] and unusual artifacts caused by the non-linear and data-dependent processing methods. In addition, motion-based approaches were implemented to reduce artifacts in PA [164, 171] and US images [172] requiring deformation of the target relative to the probe, which is not always feasible. Unlike previous methods which rely on signal amplitudes, the short-lag spatial coherence (SLSC) beamformer, which was originally developed for US [173], creates images based on spatial coherence, and it triples the effective penetration depth in photoacoustic images with no frame averaging required [67, 174]. It may also be weighted by amplitude-based images to reduce clutter and provide spectroscopic information [175]. Yet, SLSC is not sufficient for reducing coherent noise artifacts. As an alternative to these advanced methods, Pan et al. [176] and Mitcham et al. [177] enhanced the contrast of PA images by altering implanted targets to increase optical absorption, but this approach is not suitable for non-invasive clinical applications. Spatial-angular compounding of PA images may be implemented with free hand motion to overcome many of the stated limitations with existing clutter reduction approaches by reducing noise artifacts that vary with different scan directions. The concept is similar to spatial compounding methods for improving the quality of US B-mode [163, 178-180], quantitative US [181, 182] and US strain [161, 183, 184] images. We previously demonstrated that the main advantage of the free-hand approach, compared to conventional averaging, is the inclusion of images acquired with large spatial and angular deviations from an initial image with better preservation of the signal resolution [167]. To implement the technique, an external spatial tracking device such as an electromagnetic (EM) position sensor is used to simultaneously record the spatial position and orientation (i.e. pose) information of the ultrasound probe and acquire PA images [49]. This pose information is used to filter images in similar planes, and the filtered images are combined with user-defined thresholds to form a single compounded PA image.

In this paper, we propose novel compounding methods that automatically select images for spatial registration and rely on image content information to reduce tracking error. We compare the image quality of these compounded PA images with a more conventional compounding method and discuss possible clinical applications. To the authors' knowledge, this is the first study to integrate spatial tracking and compounding of photoacoustic images acquired with a handheld probe to improve image quality.

7.2. Methods

7.2.1. Frame Selection

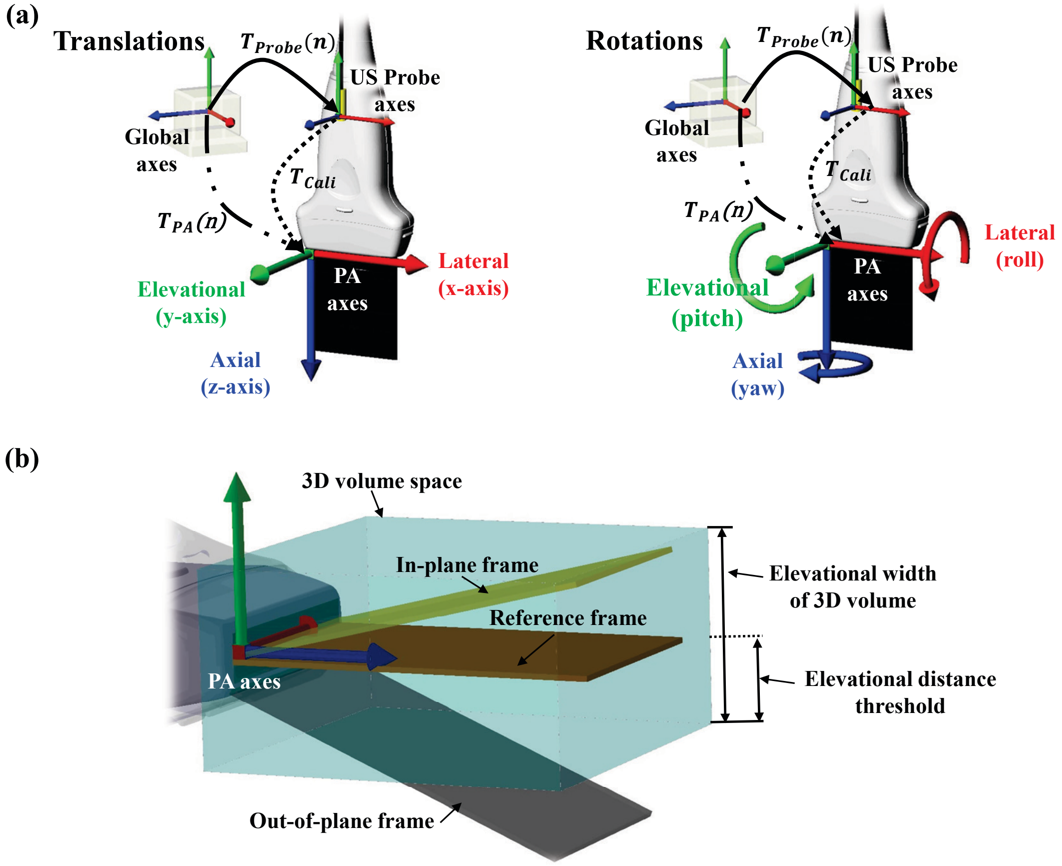


Figure 7.1. (a) Coordinate systems of spatially tracked PA images and (b) definitions of in-plane and out-of-plane images based on the user-defined elevational distance thresholds

To select frame *n* for spatial-angular compounding, an external tracking system records the US probe pose (T_{Probe}) with each image acquisition, and this pose can be converted to the PA image pose (T_{PA}) through a pre-computed calibration transformation (T_{Cali}) [93], as illustrated in **Figure 7.1** (a). T_{PA} may then be registered to the reference frame (i.e. the first frame) in an acquired image sequence. This relative pose information ($T_{PA,Rel}$) can be used to select in-plane images and reject out-of-plane images. The relationship between the relative transformations of selected and reference frames is described by the following equations:

$$T_{PA}(n) = T_{Probe}(n) \cdot T_{Cali} \tag{7.1}$$

$$T_{PA,Rel}(n) = T_{PA}(n)^{-1} \cdot \; T_{PA,Ref} \tag{7.2}$$

where $T_{PA}(n)^{-1}$ is the inverse matrix of the pose of the selected frame and $T_{PA,Ref}$ represents the pose information of the reference frame.

Figure 7.1(b) illustrates how an in-plane or out-of-plane image was defined with the relative elevational distance threshold. If the maximum elevational distance between a PA frame being considered and the reference image was smaller than this threshold, the frame was sorted as being in-plane, and it was used for the compounding operation. The second threshold value was the relative elevational rotation angle, as the recorded PA signal and noise regions are dependent on both translations and rotations of the ultrasound transducer. Images outside of these two threshold values were considered as out-of-plane images and rejected from the compounding operation.

7.2.2. Compounding PA images

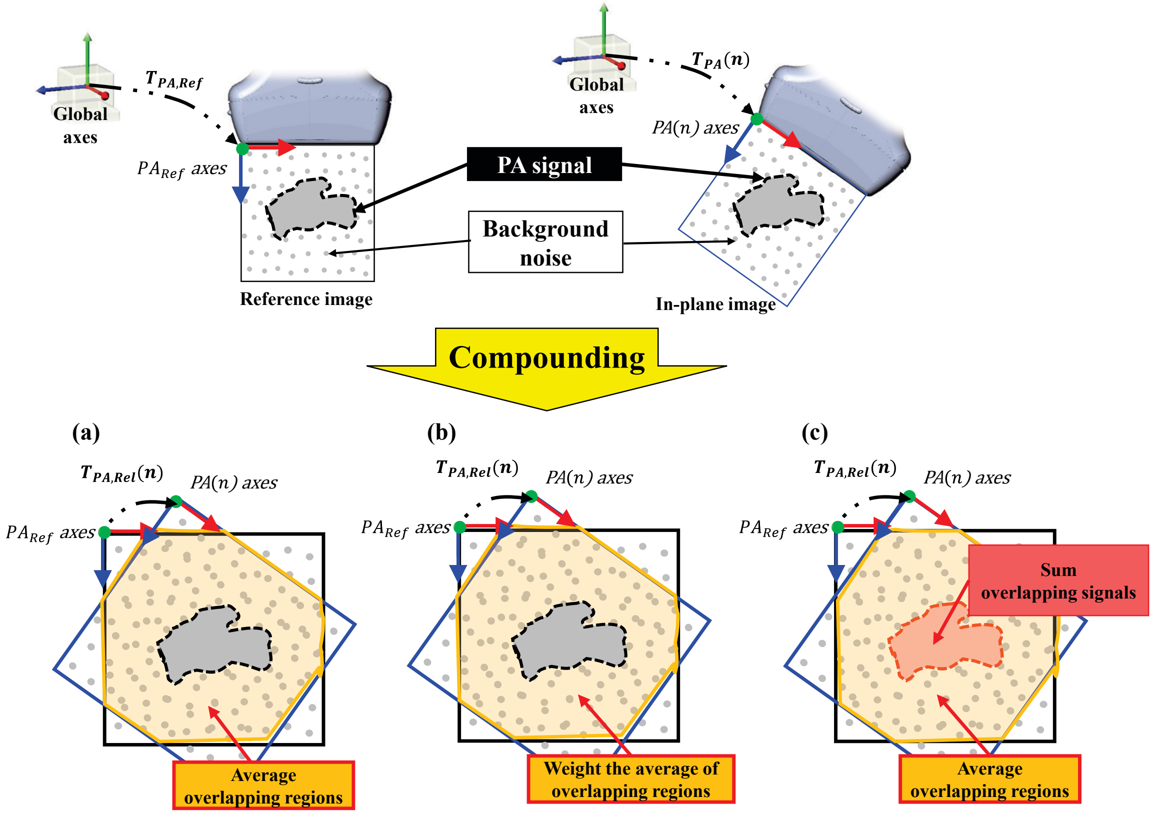


Figure 7.2. Schematic illustrations of spatial-angular compounding methods: (a) average, (b) weighted-average and (c) selective-average

The **Figure 7.2** shows schematic illustrations of compounding with (a) averaging, (b) weighted-averaging, and (c) selective-averaging. These three compounding methods are based on spatial registration of the lateral and axial pixel location of a compounded image with indices i_c and j_c, respectively, defined as:

$$\begin{bmatrix} i_c \\ j_c \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} \frac{1}{s_{ic}} & 0 & 0 & 0 \\ 0 & \frac{1}{s_{ic}} & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \left(T_{PA,\mathrm{Re}l}(n)^{-1} \cdot \begin{bmatrix} s_{is}(n) & 0 & 0 & 0 \\ 0 & s_{js}(n) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} i_s(n) \\ j_s(n) \\ 0 \\ 1 \end{bmatrix} \right) \qquad (7.3)$$

where $i_s(n)$ and $j_s(n)$ represent the lateral and axial pixel locations, respectively, in one of the selected in-plane images, and $T_{PA,Ref}(n)^{-1}$ is the inverse matrix of the relative transformation of the selected frame. The lateral and axial pixel spacings of the compounded image are represented by s_{ic} and s_{jc}, respectively, while those of the selected in-plane image are represented by $s_{is}(n)$ and $s_{js}(n)$, respectively. All four pixel spacings have units of *mm*/*pixel*.

Figure 7.2(a) describes the conventional compounding operator, which can be expressed as the following equation:

$$Img_{comp}(i_c, j_c) = \frac{1}{N} Img_{ref}(i_c, j_c) + \frac{1}{N} \sum_{n=2}^{N} Img_{sel}(i_s(n), j_s(n)) \qquad (7.4)$$

where Img_{comp}, Img_{ref} and Img_{sel} are the signal intensities of a compounded image, reference image, and the selected in-plane image, respectively, and N is the total number of compounded images. Conventional free-hand compounding uses the relative pose information of each PA image and relies on an external tracking system. However, the typical tracking accuracy of optical or EM tracking systems (0.20 – 0.3 *mm*) [185] causes accumulation of errors in the spatial registration between a reference and selected images, and therefore generates distorted PA signals in compounded images [167]. To overcome this challenge, we propose weighted-average compounding, as illustrated in **Figure 7.2(b)**. The weight factor ($w_{i_c,j_c}(n)$) represents a ratio of normalized difference between signal intensities in a reference frame (Img_{ref}) and a selected PA frame (Img_{sel}) :

$$w_{i_c,j_c}(n) = 1 - \left|\frac{Img_{ref}(i_c,j_c) - Img_{sel}(i_s(n),j_s(n))}{\max\left(Img_{ref}(i_c,j_c), Img_{sel}\left(i_s(n),j_s(n)\right)\right)}\right| \tag{7.5}$$

This weight factor is high when signals overlap and low when they do not, and it modulates the intensities of signals in the selected image, prior to compounding as described by the following equation:

$$Img_{comp}(i_c,j_c) = \frac{1}{N}Img_{ref}(i_c,j_c) + \frac{1}{N}\sum_{n=2}^{N}\left\{Img_{sel}(i_s(n),j_s(n)) \cdot w_{i_c,j_c}(n)\right\} \tag{7.6}$$

Selective-average compounding was additionally designed to overcome limitations with the accumulation of tracking errors, particularly when there are large variations in electronic background noise. It identifies and sums overlapping regions of the PA signal with higher amplitudes than a pre-defined threshold, as illustrated in **Figure 7.2(c)**. The pre-defined threshold value is calculated by an iterative confidence interval of the background noise of PA images. The noise ($\mathcal{N}$) and signal ($\mathcal{S}$) are first sorted by a temporary threshold value ($\mathcal{T}'$) as described by:

$$Img_{sel}(i_s, j_s) \rightarrow \begin{cases} \mathcal{N}(\iota), & if\ Img_{sel}(i_s, j_s) \leq \mathcal{T}'(\iota) \\ \mathcal{S}(\iota), & if\ Img_{sel}(i_s, j_s) > \mathcal{T}'(\iota) \end{cases} \tag{7.7}$$

where, ι is the iteration number and $\mathcal{T}'$ represents a temporary threshold. An initial (ι = 1) threshold value $\mathcal{T}'(1)$ is the standard deviation of the PA image data. The updated temporary threshold value $\mathcal{T}'(\iota + 1)$is the Rayleigh inverse cumulative distribution [186] computed with a standard deviation of noise data ($\sigma_{\mathcal{N}(\iota)}$) and a confidence coefficient ($\mathcal{C}$) defined as 0:99999:

$$\mathcal{T}'(\iota + 1) = \sqrt{-2\sigma_{\mathcal{N}(\iota)}{}^2 \cdot \log_{10}(1 - \mathcal{C})} \tag{7.8}$$

An error threshold $(\mathcal{T}'_{err}(\iota + 1))$ is then calculated based on $\mathcal{T}'(\iota + 1)$ and $\mathcal{T}'(\iota)$as described in the following equation:

$$\mathcal{T}'_{err}(\iota + 1) = \frac{\mathcal{T}'(\iota + 1) - \mathcal{T}'(\iota)}{\mathcal{T}'(\iota)} \tag{7.9}$$

When $\mathcal{T}'_{err}(\iota + 1) \geq 0.0001$, the process (Eq. 7.7 – 7.9) is repeated with PA data thresholded by $\mathcal{T}'(\iota + 1)$. Otherwise, a final threshold value ($\mathcal{T}$) is calculated with the average, S_μ, and minimum, S_{min}, values of the signal data as described by:

$$\mathcal{T} = \frac{S_\mu(\iota + 1) - S_{min}(\iota + 1)}{2} \tag{7.10}$$

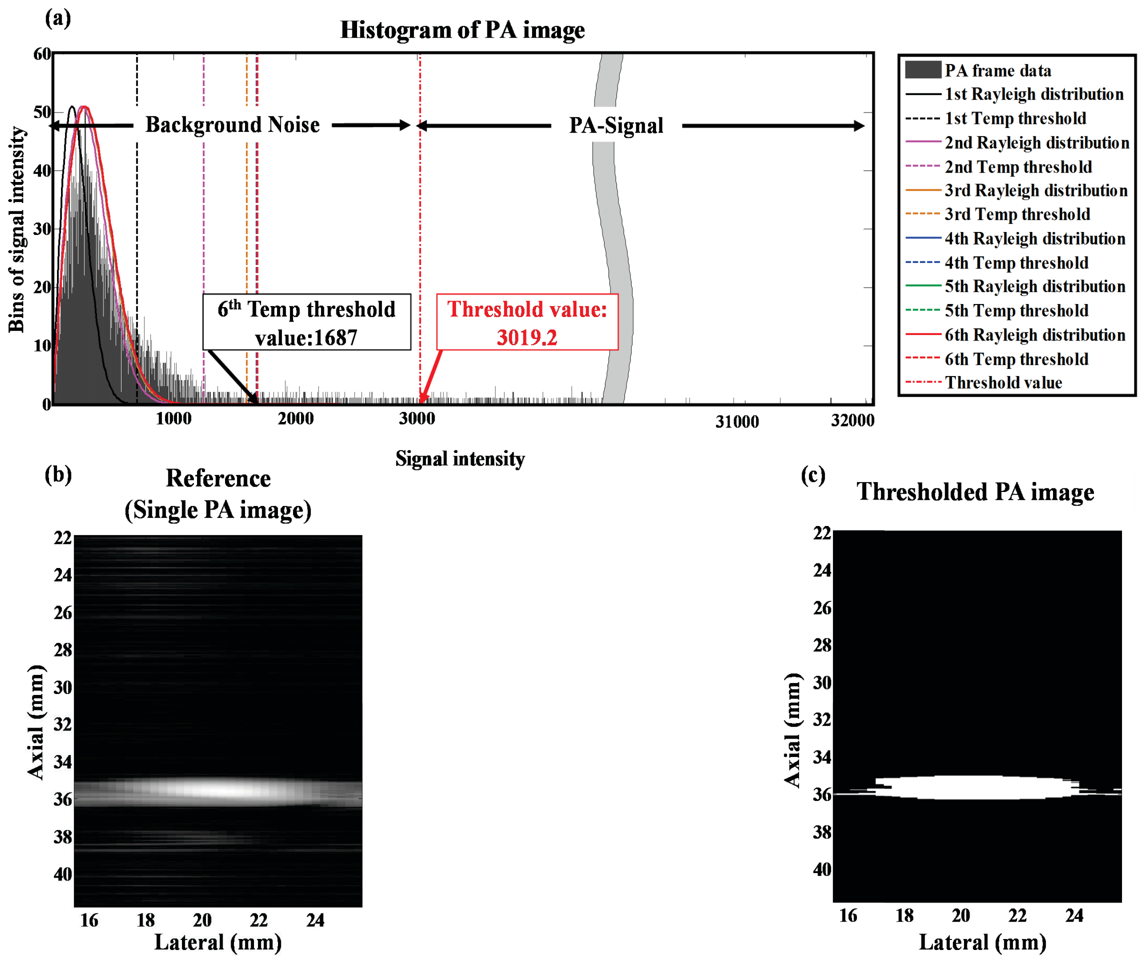


Figure 7.3. The result of user-defined threshold values using an iterative confidence interval of noise for selective-averaging: (a) histogram of values representing the confidence level of noise data (Eq.8) and the corresponding Rayleigh distribution of each iteration computation; (b) PA image from which the initial histogram was derived; and (c) corresponding thresholded PA image with the threshold value ($T = 3019.2$) computed with the Eq.10, where white indicates signal and black indicates noise.

This thresholding iteration process is illustrated in **Figure 7.3**, and it was applied to all selected PA frames. A compounded image with selective-averaging is computed by summation of the signal intensities of the reference and compounding frames as described by:

$$Img_{comp}(i_c, j_c) = Img_{ref}(i_c, j_c) + \sum_{n=2}^{M} Img_{sel}(i_s(n), j_s(n)) \quad (7.11)$$

where, M represents the number of overlapping signals at the position $Img_{comp}(i_c, j_c)$. The remaining overlapping regions are compounded using Eq. 7.4.

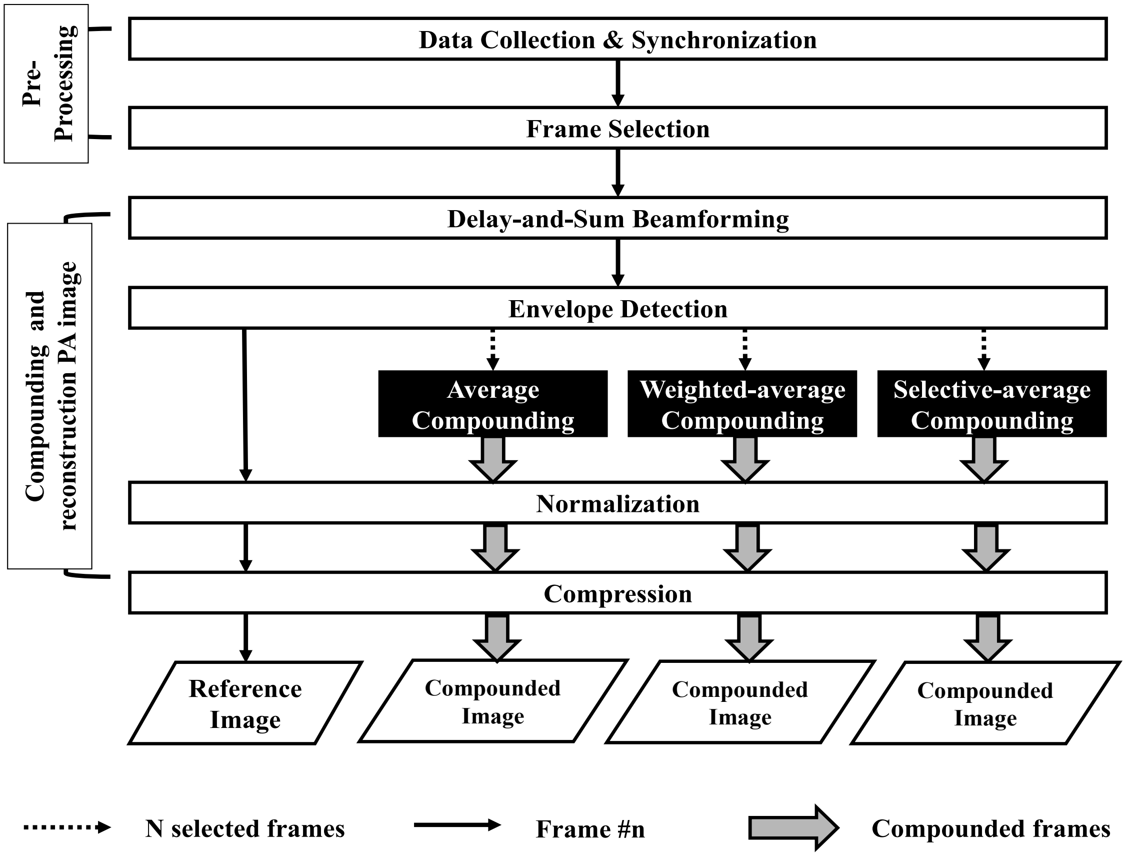


Figure 7.4. Flowchart for spatial angular compounding of PA images reconstruction.

Figure 7.4 illustrates the data flowchart for generating reference and compounded PA images. Note that the three compounding operators (average, weighted-average and selective-average) were applied to envelope detected data and not to pre- or post-beamformed radio frequency (RF) data as in our previous publication [167], because the tracking accuracy is not suitable to account for the phase sensitivity of the RF data.

7.2.3. Experimental procedure

A Q-A Q-switched Nd:YAG laser (Quantel, Les Ulis, France) operated at a wavelength of 1064 nm irradiated a plastisol phantom embedded with a black rectangular region to generate PA signals. The laser beam was fixed relative to the phantom. The US equipment consists of a SonixDAQ (Ultrasonix Co., Vancouver Canada) device, a hand-held L14-5W/38 (Ultrasonix Co., Vancouver, Canada) US probe and Sonix-CEP (Ultrasonix Co., Vancouver, Canada) system. Pose information was recorded with a medSAFE (Ascension Technology Co., Milton, USA) EM tracking system. The hand-held US probe was rotated about its elevation axis with minimal rotations about the axial and lateral axes. Transistor-transistor logic (TTL) trigger signals from the laser system were converted to a RS232 protocol with our custom-built controller board[2], and sent to our *MUSiiC* toolkit software [47, 72, 94, 167], which contains a *MUSiiC-DAQServer* [49, 50] to acquire channel data frames in real-time and measure the data acquisition timestamp of each PA frame, a spatial tracking software module (*MUSiiC-TrackerServer* [49]) to acquire EM tracking information, a *MUSiiCSync* [49, 72] software module to

[2] Xiaoyu Guo designed and developed the controller board.

synchronize simultaneously acquired 2D PA frame and pose information, and a *MUSiiC-Stream Writer* software module to save spatially-tracked 2D frames to a local hard disk. Spatial-angular compounding methods were then applied off-line. The relationship between the system and software components is illustrated in **Figure 7.5**.

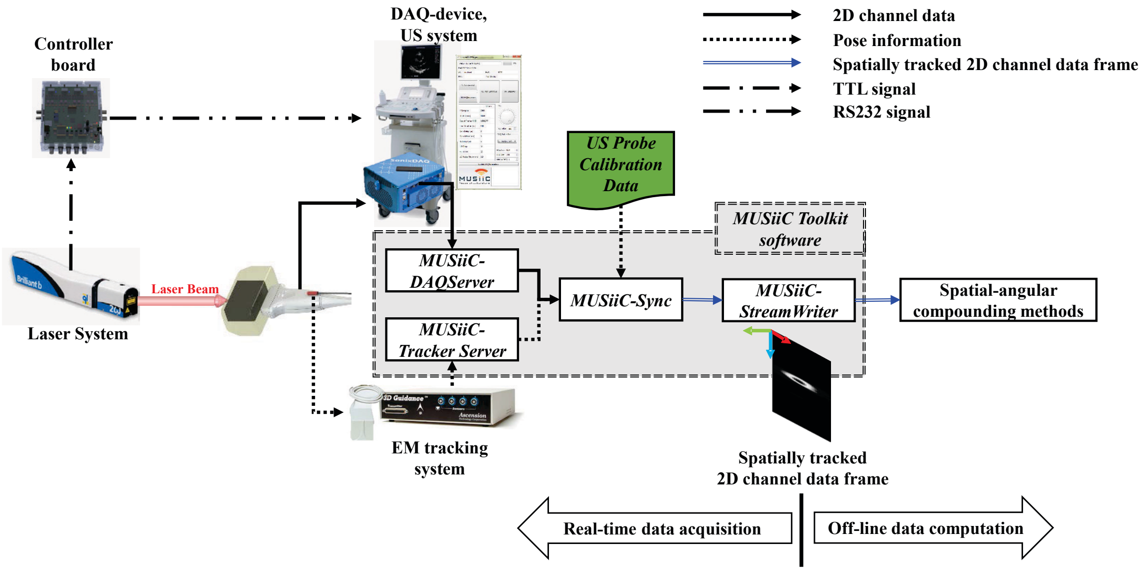


Figure 7.5. System components and workflow for spatial-angular compounding of PA images[3]

To determine optimal threshold values for frame-selection, the relative elevational distance threshold was varied from 0.5 *mm* to 2.5 *mm* in 0.5 *mm* increments with the relative elevational rotation threshold fixed at values ranging from 0.25º to 7.00º in 0.25º increments.

[3] Xiaoyu Guo designed and set up an optics system for our phantom experiment.

To evaluate the quality of compounded PA images, contrast, CNR, and SNR were computed with normalized envelope-detected data as follows:

$$Contrast = \frac{\mu_s - \mu_n}{\mu_n} \tag{7.12}$$

$$CNR = \frac{|\mu_s - \mu_n|}{\sigma_n} \tag{7.13}$$

$$SNR = \frac{\mu_s}{\sigma_n} \tag{7.14}$$

where, μ_s represents the average value of the image intensities in the selected PA signal region of interest (ROI) surrounding the maximum signal intensity and μ_n and σ_n represent the average and standard deviation, respectively, of image intensities in the background noise ROI. The sizes of the ROIs were fixed to 4.2 mm in the lateral dimension and 1.0 mm in the axial dimension. The noise ROI was located above the signal ROI, and the distance between two ROIs was fixed to 0 mm in the lateral dimension and 7.7 mm in the axial dimension. In addition, the lateral and axial full width at half maximum (FWHM) of PA signal were measured to provide comparative measurements of resolution and thereby determine the similarity of compounded signals compared to the reference PA signal.

7.3. Results

Figure 7.6(**a**) and (**b**) show the relative elevational pose of the tracked PA images. Each point represents one relative elevational pose, and the connector line shows the time trajectory from the reference frame (i.e. the first acquired image). The red points indicate PA frames that were selected based on the user-defined thresholds.

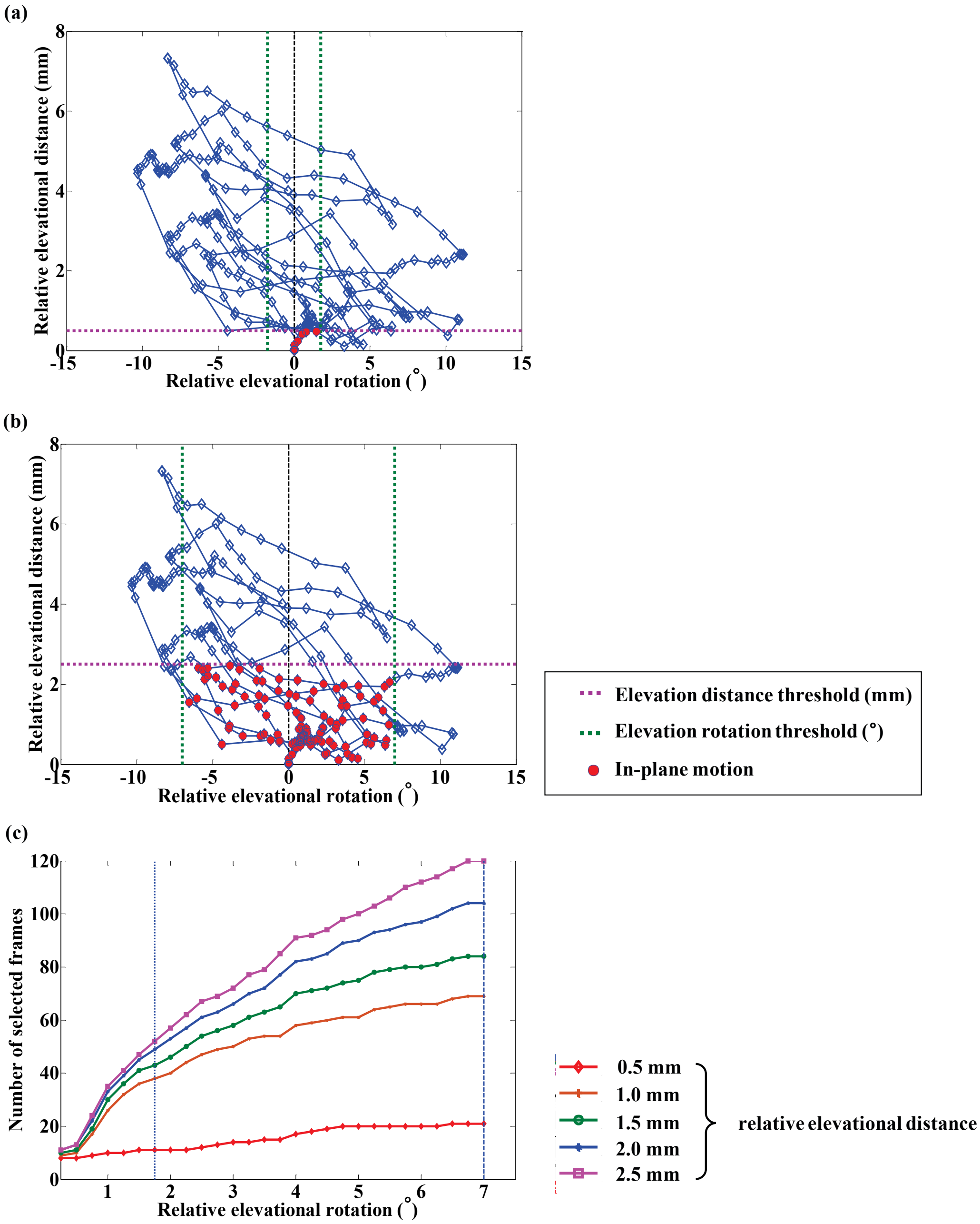


Figure 7.6. Selected frames as function of user-defined threshold values: (a) 1.75° relative elevational rotation and 0.5 *mm* relative elevational distance, and (b) 7.0° relative elevational rotation and 2.5 *mm* relative elevational distance, where the red dots indicate selected frames given the defined threshold values. (c) The number of selected frames as function of user-defined threshold values, with vertical lines corresponding to the elevational rotation thresholds illustrated in (a) and (b), respectively

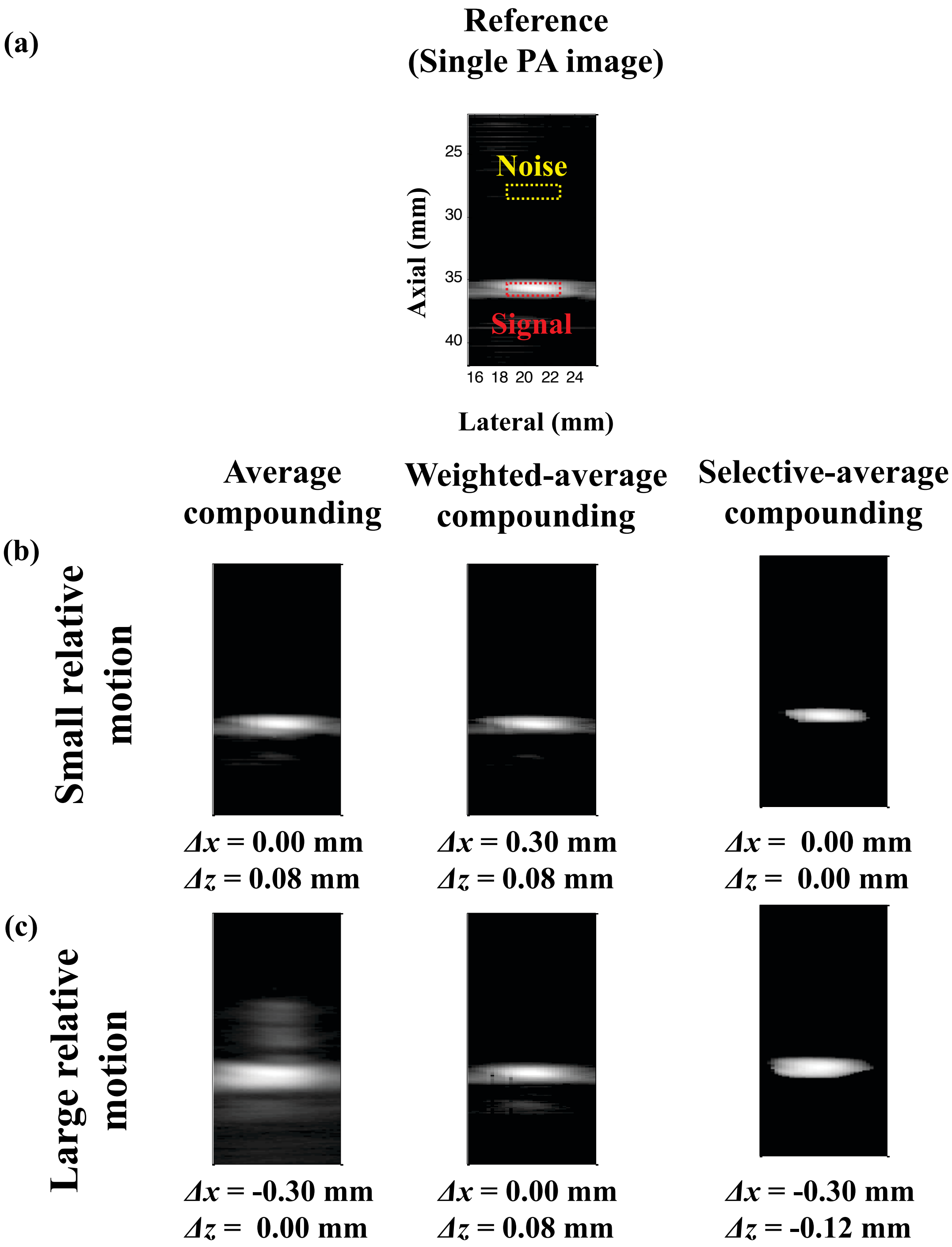


Figure 7.7. (a) Single PA reference image and compounded images with (b) relative elevational rotation and relative elevational distance threshold of 1.75° and 0.5 *mm*, respectively (i.e. small relative motion) and (c) relative elevational rotation and relative elevational distance threshold of 7.0° and 2.5 *mm*, respectively (i.e. large relative motion). The signal shift (Δx, Δz) between the reference PA image and compounded PA image are reported below each compounded PA image. Each compounded image is taken from the same axial and lateral distances reported in (a). All images shown with 40 dB dynamic range.

In **Figure 7.6**(**a**), 11 PA frames were selected with relative elevational rotation and elevational distance thresholds of 1.75º and 0.5 mm, respectively. These thresholds represent minimal deviations about the reference image. A total of 120 PA frames were allowed by the frame-selection parameters displayed in **Figure 7.6**(**b**), where the relative elevational rotation and distance thresholds were 7.0º and 2.5 mm, respectively. Note that increasing the threshold values increases the number of images that are compounded as shown in **Figure 7.6**(**c**).

Figure 7.7 displays reference and compounded PA images based on the selected frames illustrated in **Figure 7.6**(**a**) and (**b**). Shifts (i.e. Δx and Δz) between maximum signal intensity in the reference and compounded images are reported below each image defined relative to the axes shown in **Figure 7.1**(**a**). With small relative motion, the selective-average compounded image has the least signal shift as shown in **Figure 7.7**(**b**). With large relative motion, weighted-average compounding generated the smallest overall signal shift (**Figure 7.7**(**c**)). In addition to these signal shifts, the shape and distribution of compounded PA signals were distorted when compared to the reference image, particularly for large relative motion, with the greatest signal preservation obtained with weighted-average compounding.

Figure 7.8 shows the image quality metrics of PA images compounded with the three methods. The abscissa of each plot represents relative elevational rotation (º), while the legend indicates the relative elevational distance (*mm*) for selecting in-plane frames to be compounded. The two vertical lines of each plot indicate 1.75º and 7.00º of relative elevational rotation, which corresponds to the thresholds chosen to display the images in **Figure 7.7**. The solid horizontal lines indicate the mean values of all collected frames with shaded error bars showing ± one standard deviation. These values for contrast, CNR, SNR, lateral FWHM, and axial FWHM were 14 ± 23, 30 ± 48, 32 ± 48, 3.75 ± 0.35 *mm* and 0.93 ± 0.94 *mm*, respectively. Note that each plot is limited to 0, so negative values are not displayed.

Compounded PA images with weighted-averaging (second column of **Figure 7.8**) had a up to 1.47 times higher contrast compared to conventional compounded PA images (first column of **Figure 7.8**). However, the conventional compounded PA images had up to 5 times higher CNR and SNR compared to the weighted-average compounded PA images.

Contrast decreased over the first few elevational rotations (0.75º – 1.25º) with the larger relative elevational distances (1.0 *mm* – 2.5 *mm*), likely because of the registration errors or the increased misalignment between the laser and probe, indicating the importance of frame-selection for average and weighted-average compounding. This dependence is removed with selective-averaging. Compounding with selective-averaging generated images with up to 181, 1665 and 1568 times higher contrast, CNR and SNR, respectively, than the corresponding mean values of all collected PA frames. This method produced the best contrast, CNR and SNR when compared to images compounded with averaging and weighted-averaging. The lateral and axial FWHM (**Figure 7.8**(**d**) and (**e**), respectively) of conventional compounded PA signals generally increased as the relative elevational rotation and distance increased. Unlike compounding with conventional averaging, the lateral and axial FWHM were mostly within one standard deviation of all acquired frames when compounding with weighted- and selective-averaging.

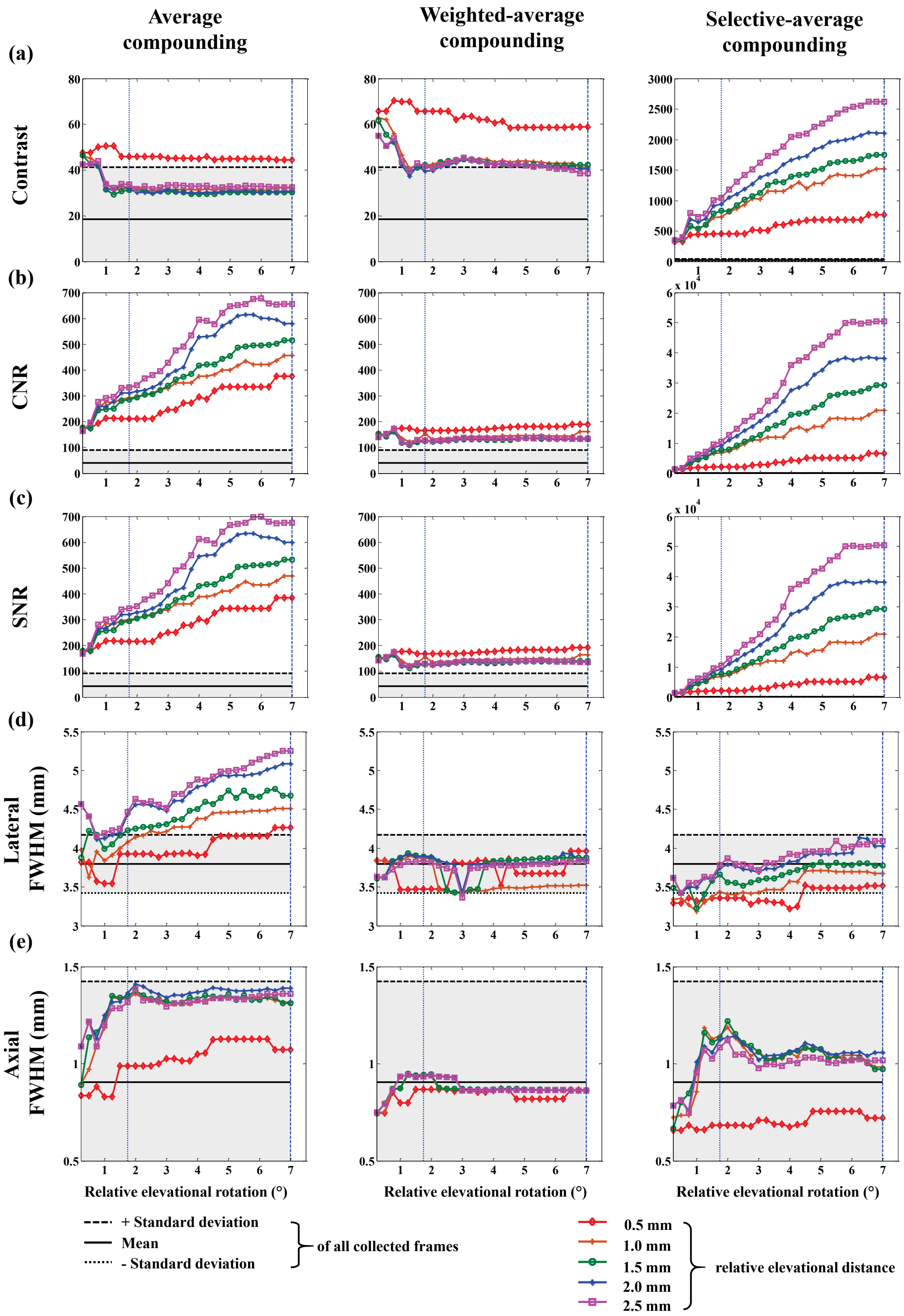


Figure 7.8. Image quality metrics of (a) CNR, (b) SNR (c) Contrast (d) lateral FWHM and (e) axial FWHM of PA images compounded with the threshold values indicated in the ordinate axis and the legend. Shaded error bars show ± one standard deviation from mean. Vertical lines correspond to the large and small relative motion shown in **Figure 7.7**.

Figure 7.9 shows the image quality of compounded PA images as a function of the number of selected frames for all previously reported thresholds. The measured values for average and weighted-average compounding are associated with the left ordinate, while the right ordinate represents the values for selective-average compounding. The solid horizontal lines in each plot represent the mean values of all collected frames with shaded error bars spanning ± one standard deviation (values are associated with left ordinate).

The image quality of compounded PA images with selective-averaging was highly dependent on the number of selected frames. In particular, contrast, CNR, and SNR ranged 323 – 2622, 1188 – 50494 and 1191 – 50513, respectively, as the number of selected frames increased. Compounding with averaging and weighted-averaging produced higher contrast when the number of frames was ≤ 24 (i.e. 2.5 *mm* and 0.75º elevational distance and rotation thresholds, respectively), with minimal change in contrast when more frames were included. The CNR and SNR generally increased with the number of frames for compounding with conventional and selective averaging, while they were relatively constant when compounding with weighted averaging.

The lateral FWHM (**Figure 7.9**(**d**)) of images compounded with conventional averaging and selective averaging linearly increased when the number of frames was ≥ 32 (i.e. 1.0 *mm* and 1.25º elevational distance and rotation thresholds, respectively) with minimal change observed for weighted-averaging over this range. Nonetheless, note that conventional averaging causes the largest degradation of lateral resolution as the number of frames increases.

The axial FWHM (**Figure 7.9**(**e**)) of images compounded with conventional and selective-averaging increased and then remained relatively constant as the number of frames increased, while that of weighted-averaging remained relatively constant for the majority of compounded images. Despite these trends for axial FWHM, all frames are within one standard deviation of the mean of all collected frames.

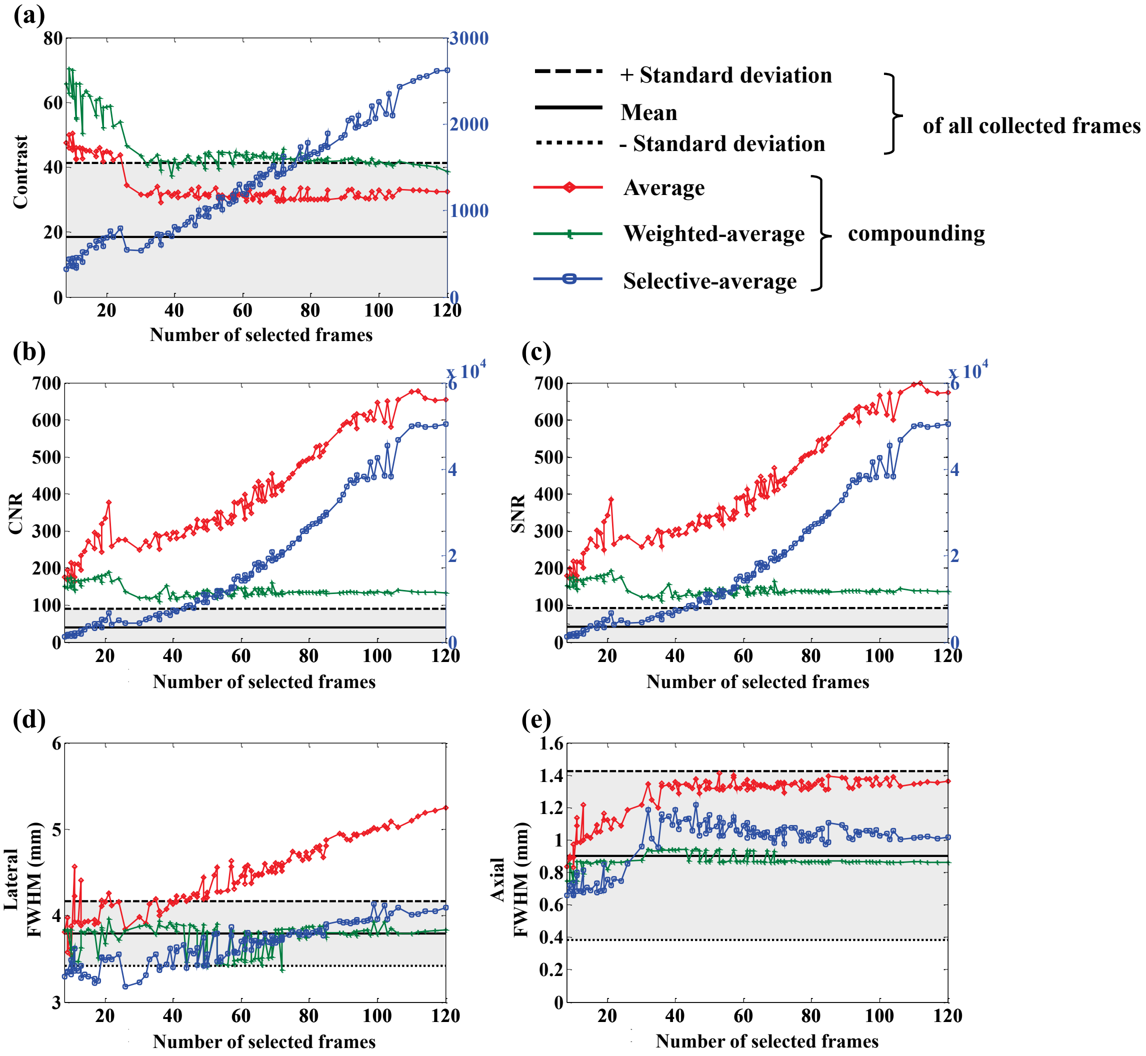


Figure 7.9. (a) CNR, (b) SNR (c) Contrast (d) lateral FWHM, and (e) axial FWHM of compounded PA images as a function of the number of selected frames.

7.4. Discussion

We investigated three compounding methods that require spatial tracking of free-hand PA images to improve image quality. Contrary to US imaging, where the success of spatial compounding relies on the combination of fully decorrelated speckle patterns achieved with large relative translations [16], the background noise of PA images rapidly decorrelates with minimal perturbation from an initial probe position. Thus improved contrast, CNR, and SNR were achieved with as little as 0.5 *mm* and 0.75° frame separation, as shown in **Figure 7.8**.

Compounding with conventional averaging is straightforward and relies solely on the tracking information. The main limitation with this method is signal blurring caused by the accumulation of tracking errors with large free hand motion as shown in **Figure 7.7**. Thus, this method is most useful when probe motion is restricted to minimal translational and rotational perturbations, which is possible with robotic assistance or electronic beam steering implemented with a relatively stationary probe.

Compounding with weighted- and selective-averaging utilizes tracking information along with the intensity values in each image to overcome limitations with tracking accuracy and provides improved quality when PA signals have significant overlap. Weighted-average compounding considers the error in spatial registration by providing a high weighting when there is signal overlap and a lower weighting when there is a mismatch due to registration errors. Consequently, the image quality of weighted-average compounded images was less sensitive to the elevational rotation thresholds as shown in **Figure 7.8**(**a**) – (**c**). In addition, this method generated compounded PA signals that were similar in size to the reference PA signal, independent of the number of frames, and insensitive to large motion, unlike PA signals compounded with averaging and selective-averaging (**Figure 7.9**(**d**) and (**e**)).

Selective-averaging provides orders of magnitude higher contrast, CNR, and SNR because the intensity difference between compounded signals and noise is increased by a multiple of the number of compounded frames. As selective-averaging depends on the separation of signal and noise, resolution could be improved with alternative separation methods (e.g. Otsu method [141], balanced histogram thresholding [187]), although the method herein might be sufficient as it provides a resolution within one standard deviation of the mean of all selected frames. PA images compounded with selective-averaging is favorable given the significantly higher contrast, CNR, and SNR for small and large relative motions. Its performance increased with more frames, which is particularly advantageous for lasers with low energies and high pulse repetition frequencies (e.g. pulsed laser diodes).

Examples of potential applications for selective- and weighted-averaging, which are most useful when the noise appears in a different location with each viewing angle, include removing reverberation clutter caused by closely spaced targets (e.g. brachytheraphy seeds), removing diffuse clutter while maintaining diffuse signals in molecular imaging, and rotating around entire objects to acquire images as in breast or small animal imaging. We expect that the presented compounding methods can be implemented in real-time with the parallel computation capability of graphics processing units (GPUs). These and related applications will be the focus of our future investigations.

7.5. Conclusion

We demonstrated spatial-angular compounding of PA images using conventional averaging, weighted-averaging, and selective-averaging. An experimental study of these three compounding methods revealed improved contrast, CNR, and SNR with each method compared to the corresponding mean values of individual images. Average compounding enhanced PA images with small relative motion required to preserve the size of signals. Both weighted- and selective-averaging produced signal sizes that were within one standard deviation of all PA signals acquired, yet those of weighted-averaging were most similar to the mean of all frames. In addition, weighted-averaging was the least sensitive to large motions and the number of selected frames, while selective-averaging offered the greatest improvements in contrast, CNR, and SNR. These three compounding methods have unique clinical advantages and promise to enhance PA images in photoacoustic imaging applications that range from detecting breast cancer to localizing metal implants and removing clutter from molecular and pre-clinical images.

Recapitulation of thesis contribution
• Inventing and investigating three freehand spatial-angular compounding algorithms (averaging, weighted-averaging, and selective-averaging) of a PA image by utilizing spatial registration techniques and the contents of a PA image, and a frame selection method of PA images to enhance certain image quality metrics: contrast, SNR, CNR, and axial/lateral resolutions.

Chapter 8:

Conclusions and Future work

8.1. Conclusions

In this dissertation, we proposed and implemented a real-time medical ultrasound (US) imaging and interventional component framework that is composed of *Open-IGTLinkMUSiiC*, *MUSiiC Notes* and *MUSiiC* modules for vendor independent advanced US imaging research such as US elastography, photoacoustic (PA) imaging, and applications of advanced interventional US imaging. The design goals of the proposed software framework include network-distributed modularity to enhance the flexibility of the framework by supporting easy reconfiguration of the US imaging system for specific interventional US imaging research, real-time computation and communication to generate advanced US images and exchange control parameters among software modules in real-time, and a well-defined open interface to improve the extensibility of the software framework by providing communication protocols and backward-compatibility for interoperation between existing software modules and third-party software components.

We applied our proposed software framework to several advanced interventional US imaging research applications including US thermal monitoring [41], real-time laparoscopic US elastography with the da Vinci surgical robot [30], localization of brachytherapy seeds with PA images [25, 26, 50], and spatially-tracked PA imaging [49] as shown in **Figure 8.1**. In addition, we proposed and investigated new interventional

PA imaging methods with the software framework, including free-hand spatial angular compounding of PA images [167], and needle visualization using a single PA image [188].

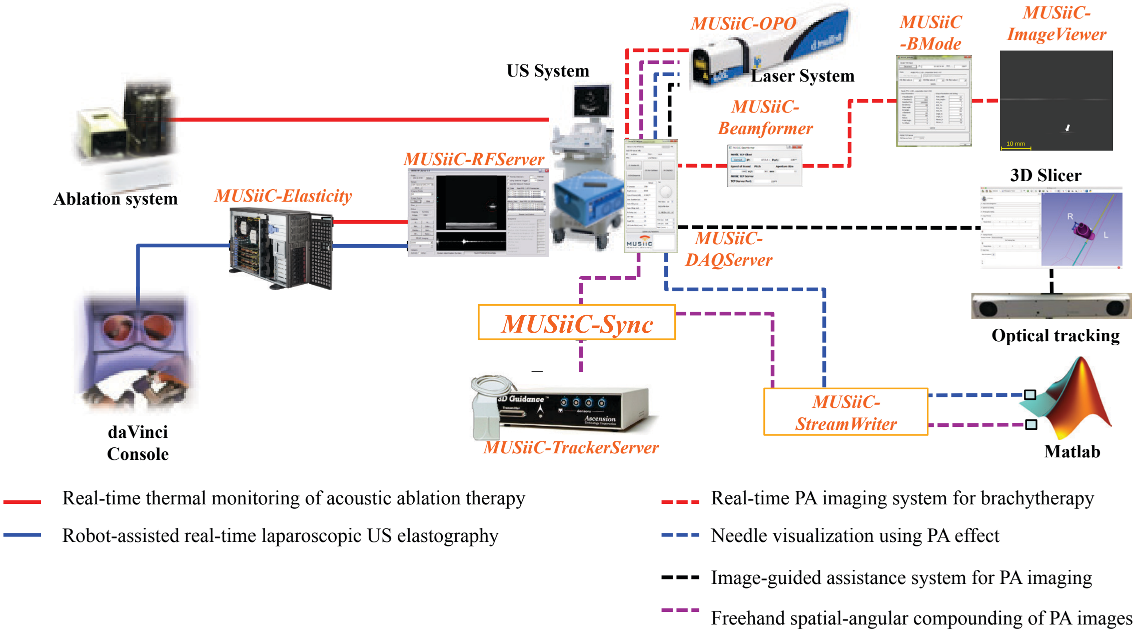

MUSiiC-OPO
MUSiiC -BMode
MUSiiC-ImageViewer
US System
Laser System
MUSiiC-Beamformer
MUSiiC-RFServer
Ablation system
MUSiiC-Elasticity
3D Slicer
MUSiiC-DAQServer
MUSiiC-Sync
Optical tracking
MUSiiC-StreamWriter
Matlab
daVinci Console
MUSiiC-TrackerServer
Real-time thermal monitoring of acoustic ablation therapy
Robot-assisted real-time laparoscopic US elastography
Real-time PA imaging system for brachytherapy
Needle visualization using PA effect
Image-guided assistance system for PA imaging
Freehand spatial-angular compounding of PA images

Figure 8.1. Advanced interventional US imaging research applications enabled by our proposed software framework.

We demonstrated the flexibility, real-time computation, and open-interface of the software framework through performing experimental tests for several applications. Therefore, our software framework can be utilized for other advanced interventional US imaging research such as robot-assisted synthetic aperture US imaging and robotically tracked PA computed tomography, and clinical applications including detecting breast cancer and localizing metal implants.

8.2. Future work

For future work, we proposed a publish/subscribe connection mechanism of *OpenIGTLinkMUSiiC* to support the communication between network distributed *MUSiiC* modules. The proposed method supports the communication of loosely coupled and asynchronous data streaming among *MUSiiC* modules, and a task-service communication between *MUSiiCPublisher* and *MUSiiCSubscriber* modules. In addition, the proposed scheme provides two modes of communication: with and without a message broker.

In general, middleware supports communication and input/output interfaces that are independent of operating systems for network-distributed modules. *OpenIGTLink-MUSiiC* can work as the middleware inside our framework by providing a loosely coupled publish/subscribe messaging communication interface among *MUSiiC* modules with *OpenIGTLink* messages and additional new messages provided by *OpenIGTLink-MUSiiC*. Publish/subscribe communication of *OpenIGTLinkMUSiiC* can be achieved by the components of *MUSiiCPublisher*, *MUSiiCSubscriber*, and *MUSiiCBroker*. But, the use of the message broker, *MUSiiCBroker*, in *OpenIGTLinkMUSiiC* is optional. That is, each publisher (*MUSiiCPublisher*) and subscriber (*MUSiiCSubscriber*) in *OpenIGTLinkMUSiiC* shares communication information about each other via IP multicast and unicast.

MUSiiCPublisher and *MUSiiCSubscriber* exchange communication information, *MTKPubSubInfo*, about each *MUSiiC* module to find each other, publish/subscribe a data stream, and establish a service connection with IP multicast and unicast. For IP multicast communication, *MUSiiCPublisher* and *MUSiiCSubscriber* use a predefined multicast address and port number.

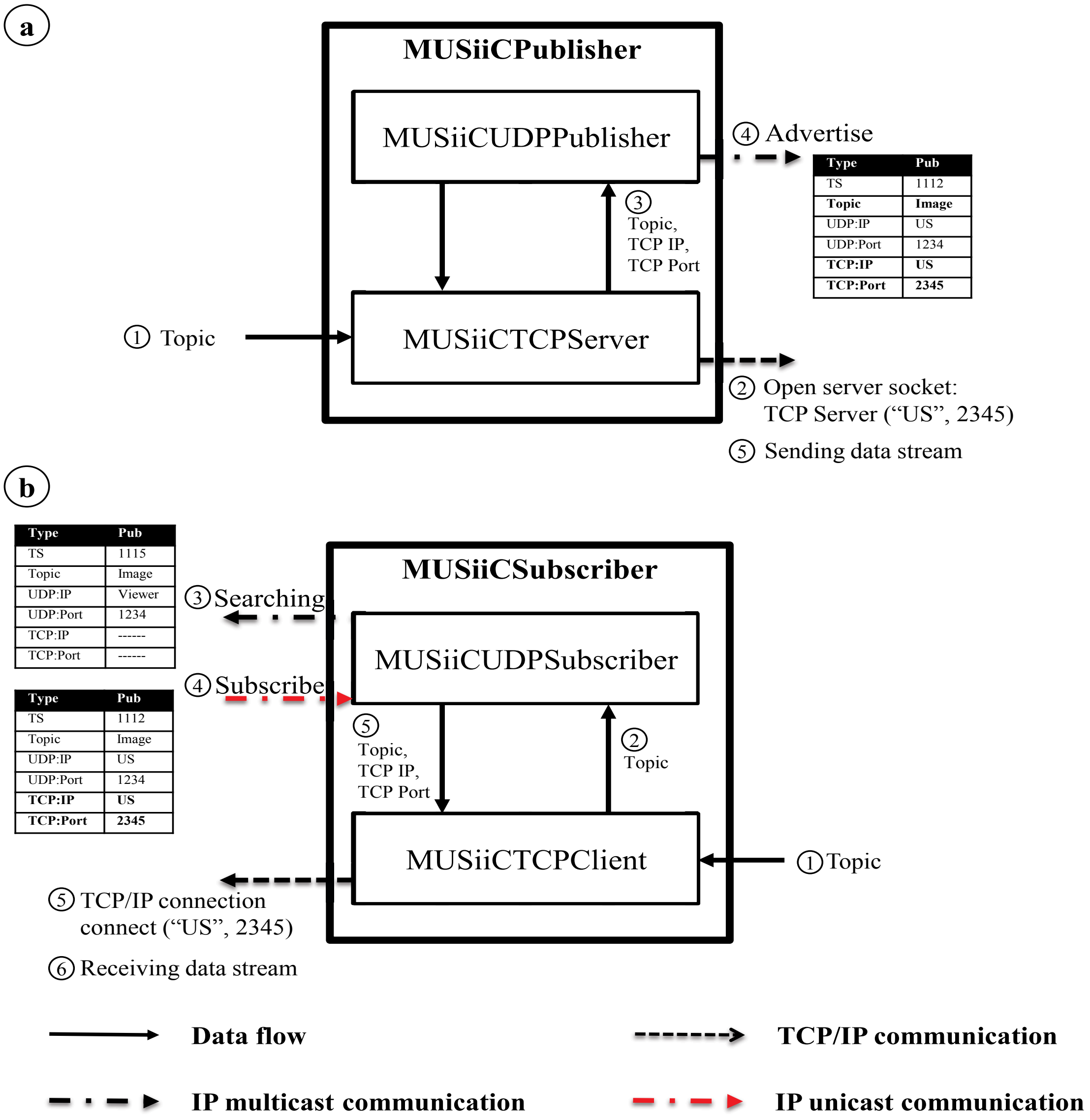
Table 8.1 shows the data fields of *MTKPubSubInfo*. The first field represents the data stream type of desired communication: publication data stream (Pub), subscription data stream (Sub), publication task service (PubTask), subscription task service (SubTask), and error messages (Err). The second data field is the timestamp of this message. A topic of providing or requesting data stream or a name of the task service is also included. *MTKPubSubInfo* has a *MUSiiC* module's UDP IP and port number for unicast communication, and a service TCP IP and port number to exchange data or messages.

Table 8.1. Data fields of *MTKPubSubInfo*

Type		Pub/Sub/PubTask/SubTask/Err
TS		Timestamp of MUSiiCPubSubInfo
Topic/Service		The name of Topic/Service
UDP	**IP**	Module's UDP URL
	Port	Module's UDP Port
TCP	**IP**	TCP URL of a providing data stream
	Port	TCP Port of a providing data stream

Figure 8.2 represents the block diagrams for *MUSiiCPublisher* and *MUSiiC-Subscriber*. *MUSiiCUDPPublisher* and *MUSiiCUDPSubscriber* are implemented in *MUSiiCPublisher* and *MUSiiCSubscriber*, respectively, to exchange *MTKPubSubInfos* of other *MUSiiC* modules using IP multicast or unicast. For bidirectional data streaming

between publisher and subscriber modules, *MUSiiCPublisher* and *MUSiiCSubscriber* include *MUSiiCTCPServer* and *MUSiiCTCPClient*, respectively.



Type	Pub
TS	1112
Topic	**Image**
UDP:IP	US
UDP:Port	1234
TCP:IP	**US**
TCP:Port	**2345**

Type	Pub
TS	1115
Topic	Image
UDP:IP	Viewer
UDP:Port	1234
TCP:IP	------
TCP:Port	------

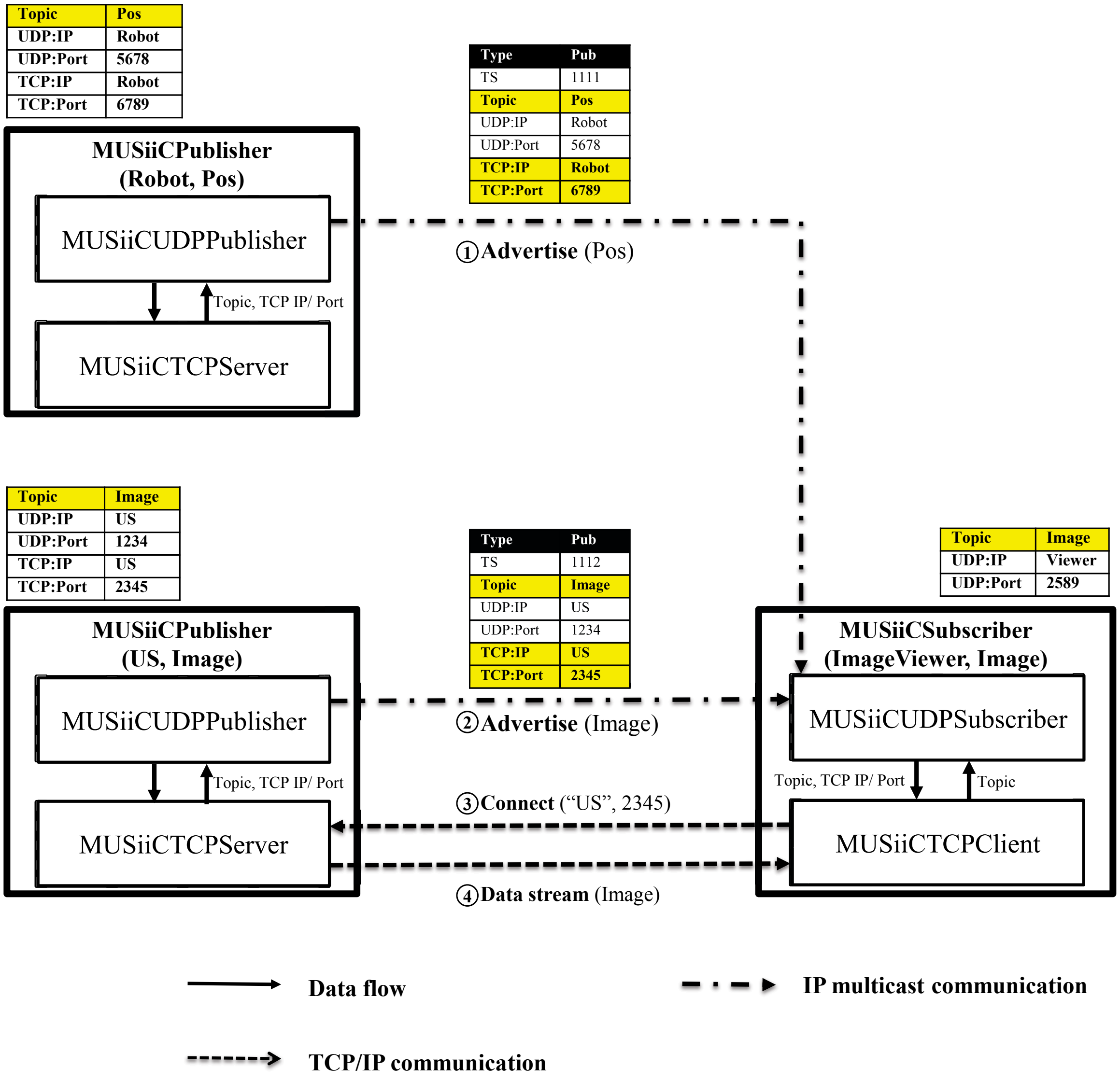
Type	Pub
TS	1112
Topic	Image
UDP:IP	US
UDP:Port	1234
TCP:IP	**US**
TCP:Port	**2345**

Figure 8.2. Block diagram for (a) *MUSiiCPublisher* and (b) *MUSiiCSubscriber*

MUSiiCPublisher broadcasts a topic of the publishing data stream and provides the data stream to connected *MUSiiCSubscribers*. In *MUSiiCPublisher*, *MUSiiCTCP-Server* creates a TCP/IP server socket with a topic of the publishing data stream, and

MUSiiCUDPPublisher generates an *MTKPubSubInfo* structure to advertise it to other *MUSiiC* modules with IP multicast (see **Figure 8.2 (a)**). On the other hand, as shown in **Figure 8.2 (b)**, *MUSiiCSubscriber* generates a subscribing *MTKPubSubInfo* with a topic of the desired data stream and searches for a *MUSiiCPublisher* with the topic on a local network. When *MUSiiCSubscriber* receives an *MTKPubSubInfo* from a *MUSiiCPublisher* that has the same topic, a client TCP socket tries to connect the TCP server socket of the *MUSiiCPublisher* with the embedded information in the *MTKPubSubInfo*.

As we mentioned previously, the publish/subscribe communication of *OpenIGTLink-MUSiiC* supports two modes of communication: with and without the message broker, *MUSiiCBroker*. *OpenIGTLinkMUSiiC*'s publish/subscribe sequence without a message broker depends on the order of *MUSiiCPublisher* and *MUSiiCSubscriber*. **Figure 8.3** shows the sequence of establishing data communication for the order of *MUSiiCSubscriber-MUSiiCPublisher*. A *MUSiiCSubscriber* (ImageViewer) module is running to visualize an ultrasound (US) B-mode image stream on a local network with the topic of 'Image'. Then, two *MUSiiCPublisher* modules (Robot and US) are executed to acquire robot pose information and ultrasound B-mode images from the US machine, respectively, and advertise their *MTKPubSubInfo* with the topics of "Pos" and "Image", respectively. Subsequently, the ImageViewer accepts only the *MTKPubSubInfo* message that includes the same topic from the US module. Then, the ImageViewer establishes a direct TCP/IP connection with the US module with the TCP/IP and port number of the accepted message to get the B-mode data stream.



Topic	Pos
UDP:IP	Robot
UDP:Port	5678
TCP:IP	Robot
TCP:Port	6789

Type	Pub
TS	1111
Topic	Pos
UDP:IP	Robot
UDP:Port	5678
TCP:IP	Robot
TCP:Port	6789

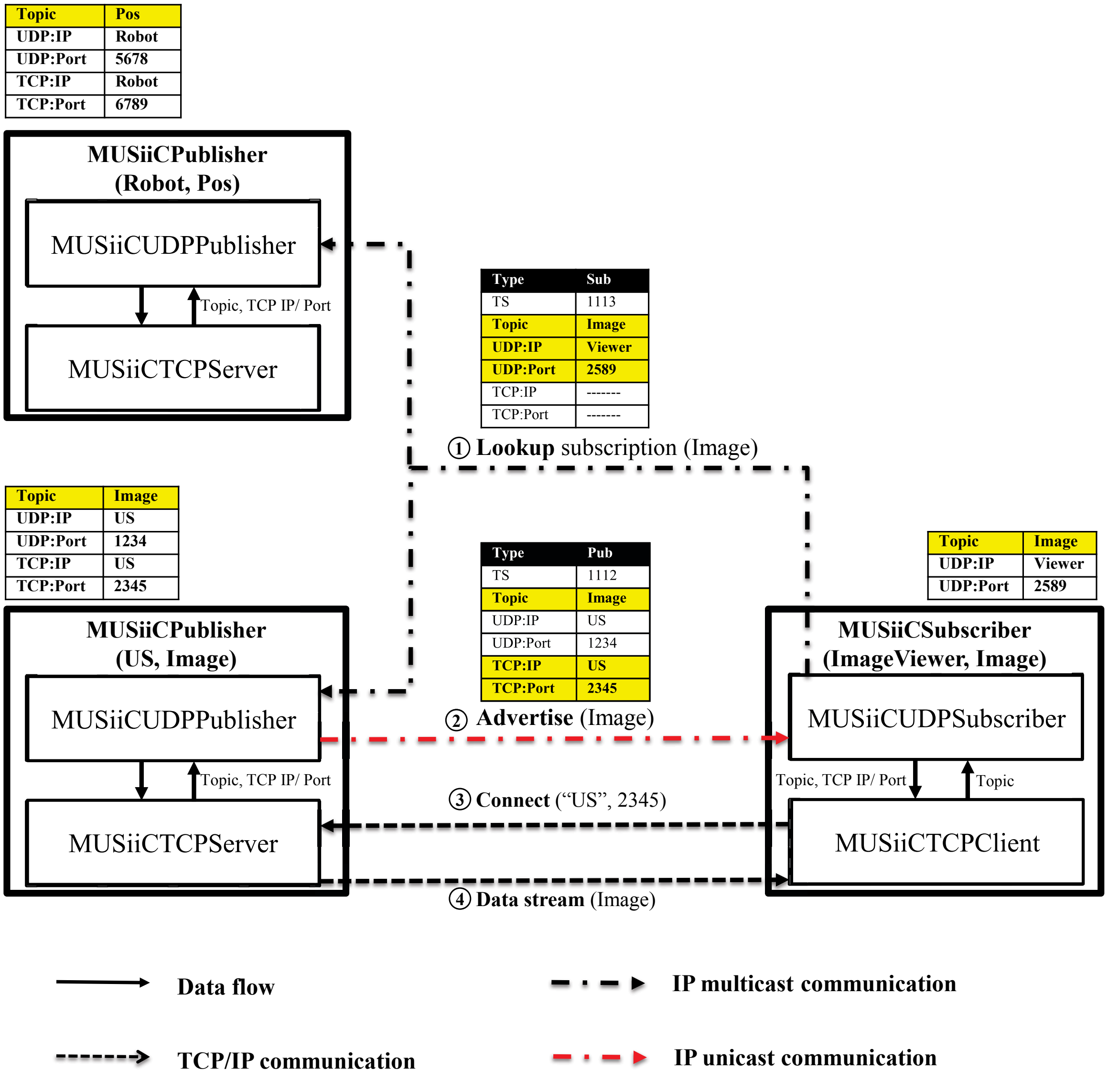
Topic	Image
UDP:IP	US
UDP:Port	1234
TCP:IP	US
TCP:Port	2345

Type	Pub
TS	1112
Topic	Image
UDP:IP	US
UDP:Port	1234
TCP:IP	US
TCP:Port	2345

Topic	Image
UDP:IP	Viewer
UDP:Port	2589

Figure 8.3. Diagram of the communication sequence of *OpenIGTLinkMUSiiC* publish/subscribe data stream with the order of *MUSiiCSubscriber-MUSiiCPublisher* and without a message broker

On the other hand, **Figure 8.4** represents the communication sequence of the order of *MUSiiCPublisher-MUSiiCSubscriber*. Two *MUSiiCPublisher* modules (Robot and US) are running and collecting robot pose information and US B-mode images. Then, a *MUSiiCSubscriber* module (ImageViewer) is executed and searches for a *MUSiiCPublisher* module with its *MTKPubSubInfo* that includes the topic of the desired data stream (Image). Although Robot and US modules receive the subscribing *MTKPubSubInfo* from ImageViewer, only the US module sends its publishing *MTKPubSubInfo* that includes the TCP/IP address and port number of the US B-mode image stream to the ImageViewer with the unicast method. Finally, the ImageViewer can connect to the US module with the provided *MTKPub-SubInfo* and get the B-Mode image stream.



Topic	Pos
UDP:IP	Robot
UDP:Port	5678
TCP:IP	Robot
TCP:Port	6789

Type	Sub
TS	1113
Topic	Image
UDP:IP	Viewer
UDP:Port	2589
TCP:IP	-------
TCP:Port	-------

Topic	Image
UDP:IP	US
UDP:Port	1234
TCP:IP	US
TCP:Port	2345

Type	Pub
TS	1112
Topic	Image
UDP:IP	US
UDP:Port	1234
TCP:IP	US
TCP:Port	2345

Topic	Image
UDP:IP	Viewer
UDP:Port	2589

Figure 8.4. Diagram of the communication sequence of *OpenIGTLinkMUSiiC* publish/subscribe data stream with the order of *MUSiiCPublisher-MUSiiCSubscriber* and without a message broker

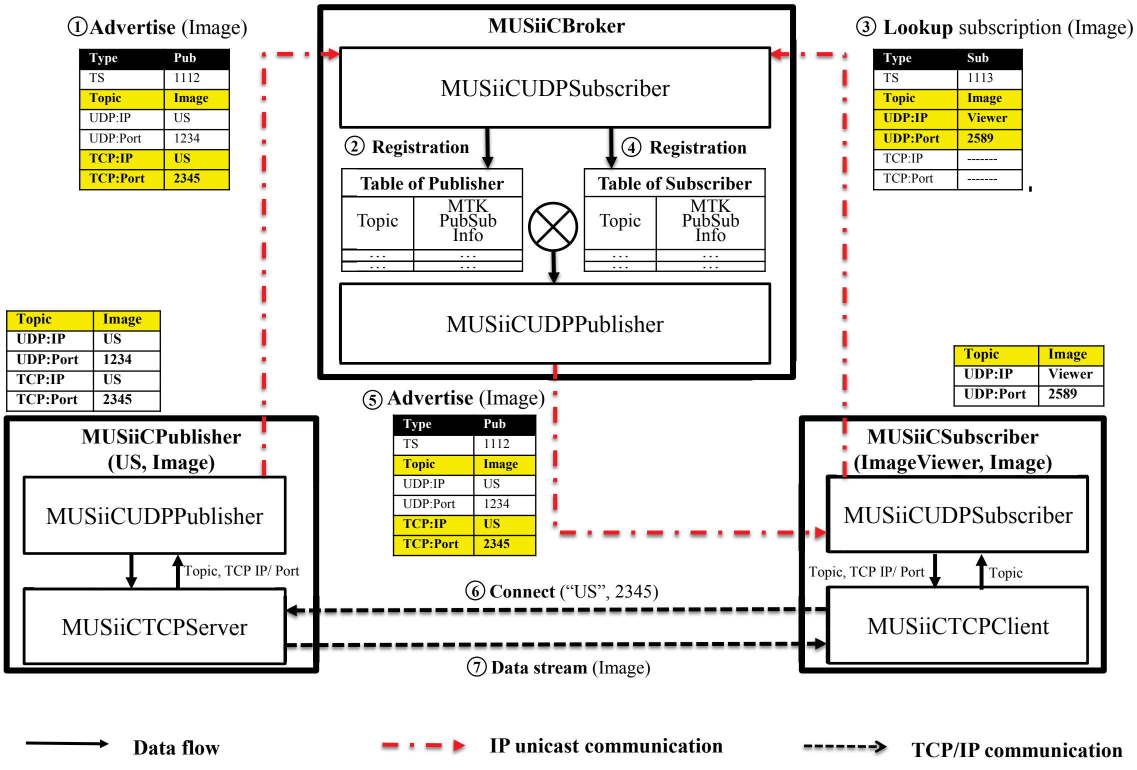


Figure 8.5. Diagram of the communication sequence of *OpenIGTLinkMUSiiC* publish/subscribe data stream with a message broker (*MUSiiCBroker*)

OpenIGTLinkMUSiiC can also provide the communication with a message broker, *MUSiiCBroker*. **Figure 8.5** shows the sequence of the communication. *MUSiiCBroker* enables *MUSiiC* modules to know the existence and communication information of each other. *MUSiiCBroker* is composed of *MUSiiCUDPSubscriber* and *MUSiiCUDPPublisher* to receive and send out *MTKPubSubInfo* messages, respectively, and has tables of publisher and subscriber to manage the information of *MUSiiC* modules. In this communication scheme, *MUSiiCPublisher* (US) and *MUSiiCSubscriber* (ImageViewer) modules connect to the *MUSiiC-Broker* with a predefined IP address and port number for IP unicast, and send their publishing and subscribing *MTKPubSubInfo* messages, respectively, to the *MUSiiCBroker*. Then, the *MUSiiCBroker* puts the received *MTKPubSubInfo* messages into the publisher and subscriber tables separately, and finds a pair of *MTKPubSubInfo* messages that have the same topic (Image). Finally, the *MUSiiCBroker* provides the US module communication information to the ImageViewer module to establish the data stream communication.

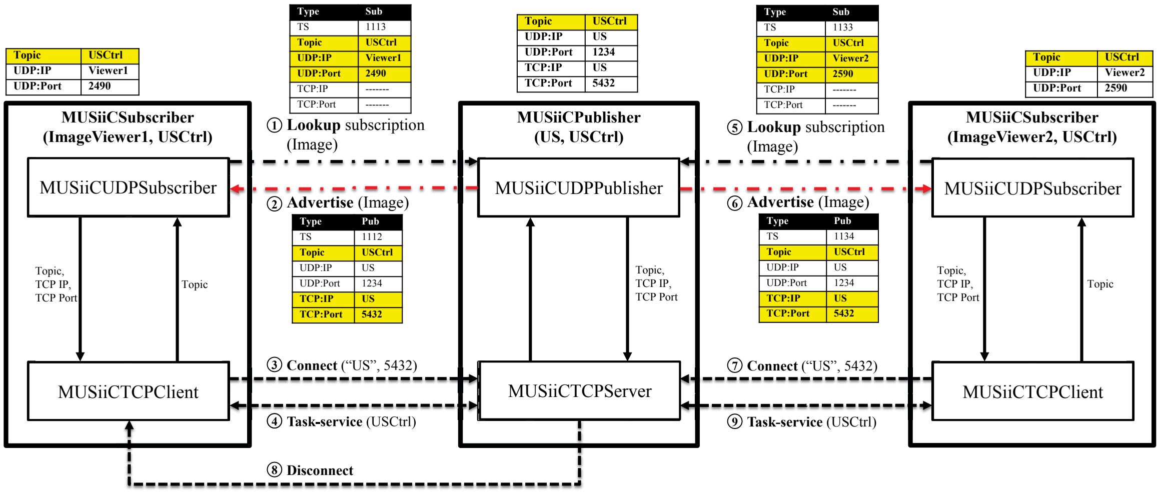


Figure 8.6. Diagram of the communication sequence of *OpenIGTLinkMUSiiC* publish/subscribe task-service

In addition, *OpenIGTLinkMUSiiC* can also provide a service connection, called task-service, between *MUSiiCPublisher* and *MUSiiCSubscriber* modules with the data types of communication (PubTask or SubTask, see Table 8.1) as shown in **Figure 8.6.** Each task can be identified by a specific name of a service. Consequently, a *MUSiiCSubscriber* module can control or change the control parameters of a *MUSiiCPublisher* module with the bidirectional communication functionality of *MUSiiC-TCPServer* and *MUSiiCTCPClient*. For example, the *MUSiiCSubscriber* module (ImageViewer1) can change the depth of acquiring US B-mode image with the task-service (USCtrl) connection to the *MUSiiCPublisher* module (US) as illustrated in

Figure 8.6. This task-service communication sequence is similar to the data streaming sequence and can be established with or without a *MUSiiCBroker*. However, the task-service communication scheme of *OpenIGTLinkMUSiiC* only allows one-to-one connection. For instance, the US module disconnects its connection with ImageViewer1 when ImageViewer2 connects to the US module for task-service connection as shown in **Figure 8.6**.

Appendix:

OpenIGTLinkMUSiiC tutorial

A.1. How to build the library

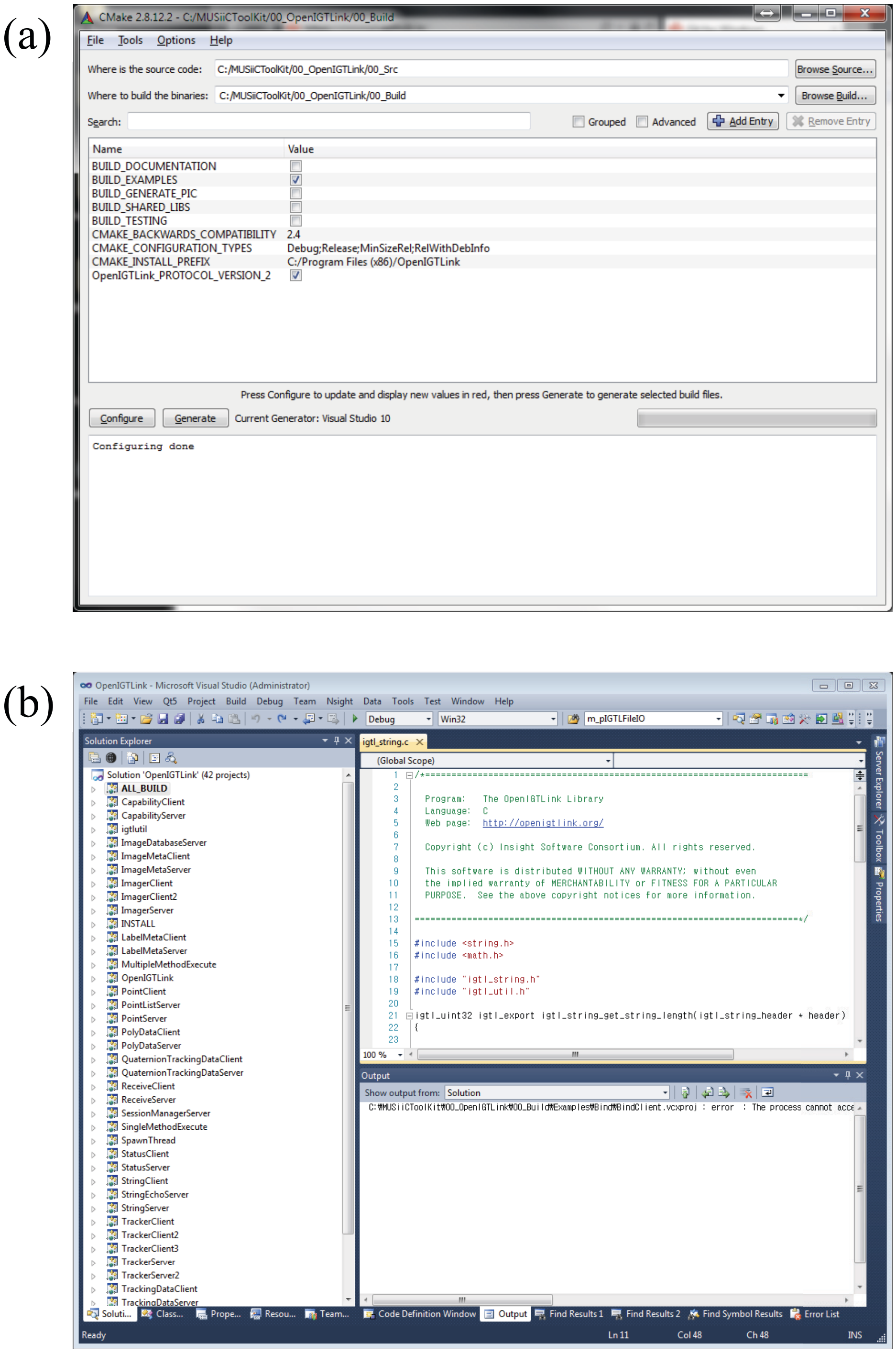
In this section, instructions on how to build the *OpenIGTLinkMUSiiC* library on the Windows operating system are presented. Building the OpenIGTLink library is a pre-requisite because the *OpenIGTLinkMUSiiC* library is based on the library of OpenIGTLink.

First, download and build the source code of OpenIGTLink library with the explanation from the following webpage: http://openigtlink.org/library/build.html[1] as shown in **Figure A.1**. Then, obtain the source of *OpenIGTLinkMUSiiC*[2] library from the Github website (https://github.com/MUSiiCLab/OpenIGTLinkMUSiiC.git) using a Git client program (e.g., tortoisegit, https://code.google.com/p/tortoisegit/). Similar to the building method of OpenIGTLink, the *OpenIGTLinkMUSiiC* library is configured and built with CMake (http://www.cmake.org/download/) and a C/C++ compiler of Microsoft Visual C++ (https://www.visualstudio.com/en-US/products/visual-studio-express-vs)

[1] Please follow the instructions for building on a Windows system in the webpage: http://openigtlink.org/library/build.html.

[2] We will post updates and news of *OpenIGTLinkMUSiiC* on the MUSiiC lab webpage: https://musiic.lcsr.jhu.edu/Research/MUSiiC_Toolkit.

with a generated solution file ('OpenIGTLinkMUSiiC.sln' in the building directory) and the build directory of OpenIGTLink library (see **Figure A.2**).



(a)

CMake 2.8.12.2 - C:/MUSiiCToolKit/00_OpenIGTLink/00_Build

Where is the source code: C:/MUSiiCToolKit/00_OpenIGTLink/00_Src

Where to build the binaries: C:/MUSiiCToolKit/00_OpenIGTLink/00_Build

Name	Value
BUILD_DOCUMENTATION	□
BUILD_EXAMPLES	☑
BUILD_GENERATE_PIC	□
BUILD_SHARED_LIBS	□
BUILD_TESTING	□
CMAKE_BACKWARDS_COMPATIBILITY	2.4
CMAKE_CONFIGURATION_TYPES	Debug;Release;MinSizeRel;RelWithDebInfo
CMAKE_INSTALL_PREFIX	C:/Program Files (x86)/OpenIGTLink
OpenIGTLink_PROTOCOL_VERSION_2	☑

Press Configure to update and display new values in red, then press Generate to generate selected build files.

Current Generator: Visual Studio 10

```
Configuring done
```

(b)

OpenIGTLink - Microsoft Visual Studio (Administrator)

Solution 'OpenIGTLink' (42 projects)

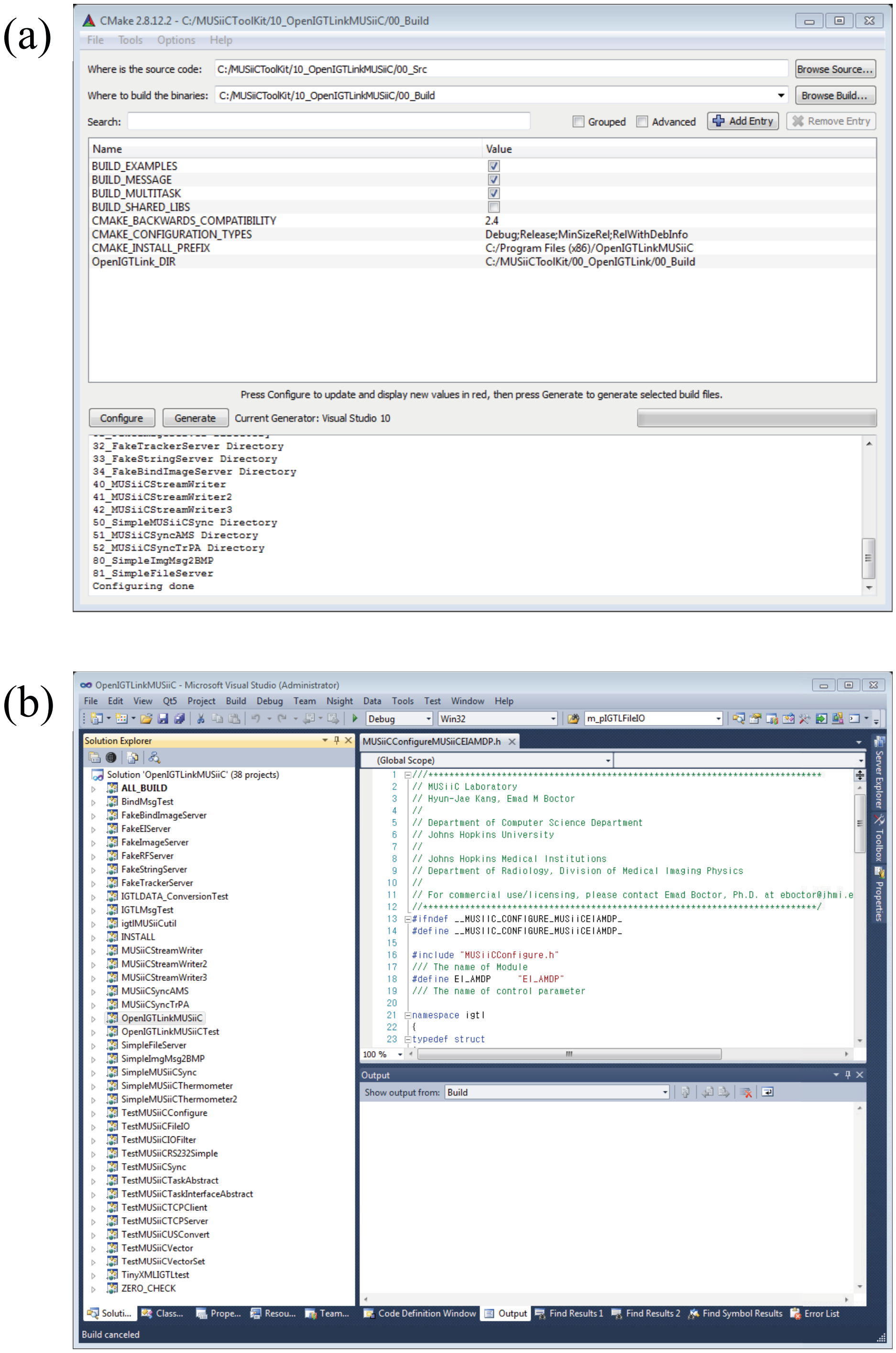
- **ALL_BUILD**
- CapabilityClient
- CapabilityServer
- igtlutil
- ImageDatabaseServer
- ImageMetaClient
- ImageMetaServer
- ImagerClient
- ImagerClient2
- ImagerServer
- INSTALL
- LabelMetaClient
- LabelMetaServer
- MultipleMethodExecute
- OpenIGTLink
- PointClient
- PointListServer
- PointServer
- PolyDataClient
- PolyDataServer
- QuaternionTrackingDataClient
- QuaternionTrackingDataServer
- ReceiveClient
- ReceiveServer
- SessionManagerServer
- SingleMethodExecute
- SpawnThread
- StatusClient
- StatusServer
- StringClient
- StringEchoServer
- StringServer
- TrackerClient
- TrackerClient2
- TrackerClient3
- TrackerServer
- TrackerServer2
- TrackingDataClient
- TrackingDataServer

```
/*=========================================================================

  Program:   The OpenIGTLink Library
  Language:  C
  Web page:  http://openigtlink.org/

  Copyright (c) Insight Software Consortium. All rights reserved.

  This software is distributed WITHOUT ANY WARRANTY; without even
  the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR
  PURPOSE.  See the above copyright notices for more information.

=========================================================================*/

#include <string.h>
#include <math.h>

#include "igtl_string.h"
#include "igtl_util.h"

igtl_uint32 igtl_export igtl_string_get_string_length(igtl_string_header * header)
{
```

Show output from: Solution

```
C:₩MUSiiCToolKit₩00_OpenIGTLink₩00_Build₩Examples₩Bind₩BindClient.vcxproj : error  : The process cannot acce
```

Figure A.1. Building OpenIGTlink library with CMake and Microsoft Visual Studio: (a) the screen of CMake to configure and generate a project file of OpenIGTLink and (b) the screen of Visual Studio C++ to build the library of OpenIGTLink with the project of 'ALL_BUILD'



(a)

CMake 2.8.12.2 - C:/MUSiiCToolKit/10_OpenIGTLinkMUSiiC/00_Build

Where is the source code: C:/MUSiiCToolKit/10_OpenIGTLinkMUSiiC/00_Src

Where to build the binaries: C:/MUSiiCToolKit/10_OpenIGTLinkMUSiiC/00_Build

Search: □ Grouped □ Advanced

Name	Value
BUILD_EXAMPLES	☑
BUILD_MESSAGE	☑
BUILD_MULTITASK	☑
BUILD_SHARED_LIBS	□
CMAKE_BACKWARDS_COMPATIBILITY	2.4
CMAKE_CONFIGURATION_TYPES	Debug;Release;MinSizeRel;RelWithDebInfo
CMAKE_INSTALL_PREFIX	C:/Program Files (x86)/OpenIGTLinkMUSiiC
OpenIGTLink_DIR	C:/MUSiiCToolKit/00_OpenIGTLink/00_Build

Press Configure to update and display new values in red, then press Generate to generate selected build files.

Current Generator: Visual Studio 10

```
32_FakeTrackerServer Directory
33_FakeStringServer Directory
34_FakeBindImageServer Directory
40_MUSiiCStreamWriter
41_MUSiiCStreamWriter2
42_MUSiiCStreamWriter3
50_SimpleMUSiiCSync Directory
51_MUSiiCSyncAMS Directory
52_MUSiiCSyncTrPA Directory
80_SimpleImgMsg2BMP
81_SimpleFileServer
Configuring done
```

(b)

OpenIGTLinkMUSiiC - Microsoft Visual Studio (Administrator)

Debug | Win32 | m_pIGTLFileIO

Solution Explorer

Solution 'OpenIGTLinkMUSiiC' (38 projects)
- ALL_BUILD
- BindMsgTest
- FakeBindImageServer
- FakeEIServer
- FakeImageServer
- FakeRFServer
- FakeStringServer
- FakeTrackerServer
- IGTLDATA_ConversionTest
- IGTLMsgTest
- igtlMUSiiCutil
- INSTALL
- MUSiiCStreamWriter
- MUSiiCStreamWriter2
- MUSiiCStreamWriter3
- MUSiiCSyncAMS
- MUSiiCSyncTrPA
- OpenIGTLinkMUSiiC
- OpenIGTLinkMUSiiCTest
- SimpleFileServer
- SimpleImgMsg2BMP
- SimpleMUSiiCSync
- SimpleMUSiiCThermometer
- SimpleMUSiiCThermometer2
- TestMUSiiCConfigure
- TestMUSiiCFileIO
- TestMUSiiCIOFilter
- TestMUSiiCRS232Simple
- TestMUSiiCSync
- TestMUSiiCTaskAbstract
- TestMUSiiCTaskInterfaceAbstract
- TestMUSiiCTCPClient
- TestMUSiiCTCPServer
- TestMUSiiCUSConvert
- TestMUSiiCVector
- TestMUSiiCVectorSet
- TinyXMLIGTLtest
- ZERO_CHECK

MUSiiCConfigureMUSiiCEIAMDP.h

(Global Scope)

```
///*************************************************************************
// MUSiiC Laboratory
// Hyun-Jae Kang, Emad M Boctor
//
// Department of Computer Science Department
// Johns Hopkins University
//
// Johns Hopkins Medical Institutions
// Department of Radiology, Division of Medical Imaging Physics
//
// For commercial use/licensing, please contact Emad Boctor, Ph.D. at eboctor@jhmi.e
//************************************************************************/
#ifndef __MUSIIC_CONFIGURE_MUSiiCEIAMDP_
#define __MUSIIC_CONFIGURE_MUSiiCEIAMDP_

#include "MUSiiCConfigure.h"
/// The name of Module
#define EI_AMDP     "EI_AMDP"
/// The name of control parameter

namespace igtl
{
typedef struct
```

Output

Show output from: Build

Build canceled

Figure A.2. Building *OpenIGTLinkMUSiiC* library with CMake and Microsoft Visual Studio: (a) the screen of CMake to configure and generate a project file (**'OpenIGTLinkMUSiiC.sln'**) of *OpenIGTLinkMUSiiC* with the build directory of OpenIGTLink and (b) the screen of Visual Studio C++ to build the library of *OpenIGTLinkMUSiiC* with the project of 'ALL_BUILD'

A.2. *SimpleFileStreamWriter* with *OpenIGTLinkMUSiiC*

For an off-line ultrasound medical imaging research, we need to save data streams generated from various data source modules (e.g. *RF-Server*, *Tracker-Server* and etc.) or data processing modules such as *B-Mode* module or *MUSiiC-Sync* to the local hard drive. To address this requirement, we introduce a *SimpleFileStreamWriter* with the OpenIGTLink and *OpenIGTLinkMUSiiC* libraries, which receive an IGTL data stream from an IGTL data source program and saves the data stream to local data storage. **Snippet A.1** and **A.2** show the CMake file of this project and the whole source code.

Snippet A.1. The CMake file of the *SimpleFileStreamWriter*

```
#/*******************************************************
# Copyright
# MUSiiC Laboratory
# Hyun Jae Kang,Emad M Boctor
#
# Department of Computer Science
# Johns Hopkins University

# For commercial use/licensing, please contact Emad Boctor, Ph.D. at eboctor@jhmi.edu.
# (C) Copyright [2010-2014] Johns Hopkins University (JHU), All Rights Reserved.
# *******************************************************
PROJECT(SimpleFileStreamWriter)

cmake_minimum_required(VERSION 2.4)
if(COMMAND cmake_policy)
  cmake_policy(SET CMP0003 NEW)
endif(COMMAND cmake_policy)

```

```
find_package(OpenIGTLink REQUIRED)

find_package(OpenIGTLinkMUSiiC REQUIRED)

include(${OpenIGTLinkMUSiiC_USE_FILE})

# Setting LINK_LIBS
SET( LINK_LIBS ${LINK_LIBS} ${OpenIGTLink_LIBRARIES} )
SET( LINK_LIBS ${LINK_LIBS} ${OpenIGTLinkMUSiiC_LIBRARIES} )

ADD_EXECUTABLE(SimpleFileStreamWriter SimpleFileStreamWriter.cxx)
TARGET_LINK_LIBRARIES(SimpleFileStreamWriter ${LINK_LIBS})
```

SimpleFileStreamWriter requires two components, *OpenIGTLinkMUSiiC*: *MUSiiCTCPClient* and *MUSiiCIGTLFileIO,* to receive an IGTL Message stream through the TCP/IP network and to save the received message stream to a local hard disk, respectively. First, two instances (pClient and pIgtlFileIO) are created at Snippet A.2: line 56 and 59, respectively. Then the two instances are connected with the function of *AddPostMUSiiCTaskInterface*() (see Snippet A.2 : line 62 and Snippet 3.6). When pClient is connected to a data server program (see Snippet A.2: line 75), pClient passes the received data stream to the pIgtlFileIO. Simultaneously, the pIgtlFileIO dumps the passed data stream to a local storage with a separate thread activated by the WriteStreamData()(see Snippet A.2: line 72) function.

Snippet A.2. The source file (**SimpleFileStreamWriter.cxx)** of the *SimpleFileStreamWriter*

```
1  ///**************************************************
2  // MUSiiC Laboratory
3  // Hyun Jae Kang, Emad M Boctor
```

```
//
// Department of Computer Science
// Johns Hopkins University
//
// For commercial use/licensing, please contact Emad Boctor, Ph.D. at eboctor@jhmi.edu.
// (C) Copyright [2010-2014] Johns Hopkins University (JHU), All Rights Reserved.
//**************************************************
#include <MUSiiCIGTLFileIO.h>
#include <MUSiiCTCPClient.h>
#include <direct.h>

/// Main function
int main (int argc, char* argv[])
{
    /// Get a Current Time with String
    std::string CurrentTime = igtl::GetCurrentTimeString();
    /// Parsing Arguments
    char*      hostname;
    int        port;
    char*      pre_fix_filename;
    std::string DirName;

    if(argc == 4)
    {
        hostname           = argv[1];
        port               = atoi(argv[2]);
        pre_fix_filename   = argv[3];
        DirName = CurrentTime + std::string("_MTK_");
    }
    else if(argc == 5 )
    {
        hostname           = argv[1];
        port               = atoi(argv[2]);
        pre_fix_filename   = argv[3];
        std::string tempDir(argv[4]);
        DirName = CurrentTime + std::string("_MTK_") + tempDir;
    }
    Else
    {
        /// If not correct, print usage
        std::cerr << "   <hostname> : IP or host name"                      << std::endl;
        std::cerr << "   <port>     : Port # (18944 in Slicer default)"     << std::endl;
        std::cerr << "   prefix  of filename : "                            << std::endl;
        std::cerr << "   postfix of Folder : "                              << std::endl;
        exit(0);
    }

    /// Create a directory with the user defined folder name (DirName)
    mkdir(DirName.c_str());
    std::string DirPath = std::string("./") + DirName + std::string("/");

    /// Create a instance of MUSiiCTCPClient (pClient)
    igtl::MUSiiCTCPClient::Pointer pClient = igtl::MUSiiCTCPClient::New();

    /// Create a instance of MUSiiCFileIO (pIgtlFileIO)
    igtl::MUSiiCIGTLMsgFileIO::Pointer pIgtlFileIO =
                                        igtl::MUSiiCIGTLMsgFileIO::New();

    /// Establish a task connection between pClient and pIgtlFileIO
    pClient->AddPostMUSiiCTaskInterface<igtl::MessageBase::Pointer,
                                        igtl::MessageBase::Pointer>(pIgtlFileIO);

    /// Set a prefix of filename with user input (argv[3])
    std::string filename(pre_fix_filename);

    /// Set a target file path
    filename = DirPath+pre_fix_filename;
    pIgtlFileIO->SetUseIGTLFileIndex(true);

    /// Run a task thread of MUSiiCFileIO for saving IGTLMessage stream
    pIgtlFileIO->WriteStreamData(filename);

    /// Connect to a server socket
    pClient->ConnectToHost(hostname, port);

    /// Print out user message
    printf("Prees q for quit this program\n");
    char sel;
    for(;;)
    {
        scanf("%c", &sel);
        if( sel == 'q')
            break;
    }

    /// Stop the task thread of MUSiiCFileIO
    pIgtlFileIO->WriteStreamDataStop();
    /// Close the client socket
    pClient->CloseClientSocket();

    return 0;
}
```

A.3. *SimpleFileServer* with *OpenIGTLinkMUSiiC*

Contrary to the *SimpleFileStreamWriter*, a module providing a real-time data stream from data saved on local storage is needed to develop and debug real-time data processing software modules. In this section, we present a *SimpleFileServer* with *OpenIGTLink-MUSiiC* library, which loads IGTL files saved by the *SimpleFileStreamWriter* and sends them out to TCP/IP network. The CMake file and the source code of the *SimpleFileServer* are presented in **Snippet A.3** and **A.4.**

Snippet A.3. The CMake file of the *SimpleFileServer*

```
#/***********************************************************
# Copyright
# MUSiiC Laboratory
# Hyun Jae Kang,Emad M Boctor
#
# Department of Computer Science
# Johns Hopkins University
```

```
#
# For commercial use/licensing, please contact Emad Boctor, Ph.D. at eboctor@jhmi.edu.
# (C) Copyright [2010-2014] Johns Hopkins University (JHU), All Rights Reserved.
# ****************************************************
PROJECT(SimpleFileServer)

cmake_minimum_required(VERSION 2.4)
if(COMMAND cmake_policy)
  cmake_policy(SET CMP0003 NEW)
endif(COMMAND cmake_policy)

find_package(OpenIGTLink REQUIRED)

find_package(OpenIGTLinkMUSiiC REQUIRED)

include(${OpenIGTLinkMUSiiC_USE_FILE})

# Setting LINK_LIBS
SET( LINK_LIBS ${LINK_LIBS} ${OpenIGTLink_LIBRARIES} )
SET( LINK_LIBS ${LINK_LIBS} ${OpenIGTLinkMUSiiC_LIBRARIES} )

ADD_EXECUTABLE(SimpleFileServer SimpleFileServer.cxx)
TARGET_LINK_LIBRARIES(SimpleFileServer ${LINK_LIBS})
```

First, saved files are loaded with the function of *LoadIGTLFiles*() and a user-defined file path. In the *LoadIGTLFiles*() function, an instance of *MUSiiCIGTLMsg-FileIO* reads the IGTL file, retrieves its original message type and converts it into a igtl::MessageBase smart pointer (see Snippet A.4: line 73). The loaded files are saved in the global memory buffer (g_MemoryBuffer, see Snippet A.4: line 75). After the file loading is finished successfully (see Snippet A.4: line 131), an instance of *MUSiiCTCPServer* (g_pServer) is created (see Snippet A.4: line 137) and the instance creates a server socket with a user-defined port number (see Snippet A.4: line 140) to send out the loaded IGTL Message stream. When a user input (‘s’) is detected, the g_pServer sends out the IGTL Messages from the g_MemoryBuffer to the TCP/IP network with the function of *SendLoadedIGTLMessage*() at a pre-defined frame rate (see Snippet A.4: line 84 - 98). The user can change the frame rate with the user input of ‘f’ and the function of *SetFrameRate*()(see Snippet A.4: line 157, 158 and 45 -51).

Snippet A.4. The source file (**SimpleFileServer.cxx)** of the *SimpleFileServer*

```
 1  ///************************************************
 2  // MUSiiC Laboratory
 3  // Hyun Jae Kang, Emad M Boctor
 4  //
 5  // Department of Computer Science Department
 6  // Johns Hopkins University
 7  //
 8  // Johns Hopkins Medical Institutions
 9  // Department of Radiology, Division of Medical Imaging Physics
10  //
11  // For commercial use/licensing, please contact Emad Boctor, Ph.D. at eboctor@jhmi.edu.
12  // (C) Copyright [2010-2014] Johns Hopkins University (JHU), All Rights Reserved.
13  //*******************************************************/
14  #include <MUSiiCIGTLFileIO.h>
15  #include <MUSiiCTCPServer.h>
16  #include <MUSiiCIGTLUtil.h>
17  #include <direct.h>
18  #include <stdio.h>
19  #include <stdlib.h>
20  #include <io.h>
21  
22  /// Declare global variables
23  int                                          g_index;
24  int                                          gFrameRate;
25  std::vector<igtl::MessageBase::Pointer>      g_MemoryBuffer;
26  igtl::MUSiiCTCPServer::Pointer               g_pServer;
27  
```

```
/// Print out a console menu
void PrintMenu()
{
   ////////////////////////////////
    std::cout << std::endl;
    std::cout << "*********************************" << std::endl;
    std::cout << "****** How to use ***************" << std::endl;
    std::cout << "01. Show Menu          : Press 'm'" << std::endl;
    std::cout << "02. Quit this program : Press 'q'" << std::endl;
    std::cout << "03. Send Data          : Press 's'" << std::endl;
    std::cout << "05. Set Frame Rate    : Press 'f'" << std::endl;
    std::cout << "*********************************" << std::endl;
    std::cout << std::endl;
    /////////////////////////////
}

/// Function for setting the frame rate of sending data
void SetFrameRate()
{
    std::cout << "Enter frmae rates per second" << std::endl;
    int fps;
    scanf("%d", &fps);
    gFrameRate = 1000/fps;
}

/// Loading IGTL files from the user-defined file path
int LoadIGTLFiles(std::string path)
{
    igtl::MUSiiCIGTLMsgFileIO::Pointer pFileIO =
                                              igtl::MUSiiCIGTLMsgFileIO::New();

    struct _finddata_t c_file;
    intptr_t hFile;

    _chdir(path.c_str());

    if((hFile = _findfirst("*.igtl", &c_file)) == -1L)
        return false;
    else
    {
        do{
            /// Loading *.igtl file from the user-defined directory
            std::string filename, num;
            std::string pathF;
            filename = c_file.name;
            printf("File name is %s\n", c_file.name);
            igtl::MessageBase::Pointer msg = pFileIO->ReadSingleFile(filename.c_str());

            g_MemoryBuffer.push_back(msg);
            g_index++;
        }while(_findnext(hFile, &c_file) ==0);
        _findclose(hFile);
    }
    return g_index;
}

/// Send the loaded IGTL files to the TCP/IP network
int SendLoadedIGTLMessage()
{
    if(g_pServer->IsConnectedClients())
    {
        int sz(g_MemoryBuffer.size());
        for (int i=0; i< sz ; i++)
        {
            g_pServer->PutIGTLMessage(g_MemoryBuffer[i]);
            igtl::Sleep(gFrameRate);
            std::cout << "Sending Message" << i << std::endl;
        }
        return sz;
    }
    return 0;
}

/// Main function
int main(int argc, char** argv)
{

    /// Declare local variables
    int port;
    std::string src_Path;
    char c;
    int r(0);
    bool        bTask(false);
    gFrameRate  = 1;

    /// Parsing Arguments
    if (argc == 3) // check number of arguments
    {
        port      = atoi(argv[1]);
        src_Path = argv[2];

        /// Initialize the global memory buffer
        g_index = 0;
        g_MemoryBuffer.clear();
    }
    else
    {
        // If not correct, print usage
        std::cerr << "Usage::   SimpleFileServer.exe server_port Src_Dir "  << std::endl;
        std::cerr << "Example:: SimpleFileServer.exe 1234 C:/IGTLData/ "  << std::endl;
        return 0;
    }

    /// Loading files from a user-defined file path
    if(LoadIGTLFiles(src_Path)>0)
    {
        /// Debugging Message
        std::cout << g_MemoryBuffer.size() << " Files are loaded ^^ " << std::endl;

        /// Creat a instnace of MUSiiCTCPServer
        g_pServer = igtl::MUSiiCTCPServer::New();

        /// Creater server Socket
        if(g_pServer->CreateServer(port)>0)
            bTask = true;

        /// Print a console menu
        PrintMenu();

        /// Run the main process
        while(bTask)
        {
            if(igtl::kbhit())
            {
                c = getchar();
                if(c == 'q')
                {
                    bTask = false;
                    break;
                }
                else if(c == 'f')
                    SetFrameRate();
                else if(c == 'm')
                    PrintMenu();
                else if(c == 's')
                    SendLoadedIGTLMessage();
            }
        }
        /// Close MUSiiCServerSocket
         g_pServer->CloseServer();
        g_MemoryBuffer.clear();
    }
    else
        std::cout  << " No File are loaded ^^ " << std::endl;
return 0;
}
```

Bibliography

[1] K. Cleary and T. M. Peters, "Image-guided interventions: technology review and clinical applications," *Annual review of biomedical engineering,* vol. 12, pp. 119-142, 2010.

[2] E. H. Burrows, "Pioneers and early years. A history of British radiology," 1986.

[3] J. Cox and R. C. Kirkpatrick, *The New Photography: with Report of a Case in which a Bullet was Photographed in the Leg*, 1984.

[4] Compston, Alastair. "The structure and functions of the cerebellum examined by a new method. By Sir Victor Horsley, FRS, FRCS and RH Clarke, MA, MB. Brain 1908: 31; 45–124." Brain 130.6 (2007): 1449-1452.

[5] T. Peters, J. Clark, B. Pike, M. Drangova, and A. Olivier, "Stereotactic surgical planning with magnetic resonance imaging, digital subtraction angiography and computed tomography," *Stereotactic and Functional Neurosurgery,* vol. 50, pp. 33-38, 1987.

[6] T. Peters, J. Clark, G. Pike, C. Henri, L. Collins, D. Leksell*, et al.*, "Stereotactic neurosurgery planning on a personal-computer-based work station," *Journal of digital imaging,* vol. 2, pp. 75-81, 1989.

[7] E. M. Friets, J. W. Strohbehn, J. F. Hatch, and D. W. Roberts, "A frameless stereotaxic operating microscope for neurosurgery," *Biomedical Engineering, IEEE Transactions on,* vol. 36, pp. 608-617, 1989.

[8] D. W. Roberts, J. W. Strohbehn, J. F. Hatch, W. Murray, and H. Kettenberger, "A frameless stereotaxic integration of computerized tomographic imaging and the operating microscope," *Journal of neurosurgery,* vol. 65, pp. 545-549, 1986.

[9] R. L. Galloway Jr, J. T. Lewis, R. J. Maciunas, and C. A. Edwards II, "Image display and surgical visualization in interactive image-guided neurosurgery," *Optical Engineering,* vol. 32, pp. 1955-1962, 1993.

[10] R. H. Taylor, B. D. Mittelstadt, H. A. Paul, W. Hanson, P. Kazanzides, J. F. Zuhars*, et al.*, "An image-directed robotic system for precise orthopaedic surgery," *Robotics and Automation, IEEE Transactions on,* vol. 10, pp. 261-275, 1994.

[11] E. J. Hazan, "Computer-assisted Orthopaedic Surgery: A new paradigm," *Techniques in Orthopaedics,* vol. 18, pp. 221-229, 2003.

[12] S. L. Delp, D. S. Stulberg, B. Davies, F. Picard, and F. Leitner, "Computer assisted knee replacement," *Clinical orthopaedics and related research,* vol. 354, pp. 49-56, 1998.

[13] P. Marhofer and V. W. Chan, "Ultrasound-guided regional anesthesia: current concepts and future trends," *Anesthesia & Analgesia,* vol. 104, pp. 1265-1269, 2007.

[14] S. S. Liu, J. E. Ngeow, and J. T. YaDeau, "Ultrasound-guided regional anesthesia and analgesia: a qualitative systematic review," *Regional anesthesia and pain medicine,* vol. 34, pp. 47-59, 2009.

[15] A. Kumar and A. Chuan, "Ultrasound guided vascular access: efficacy and safety," *Best Practice & Research Clinical Anaesthesiology,* vol. 23, pp. 299-311, 2009.

[16] C. Harvey, J. Pilcher, J. Richenberg, U. Patel, and F. Frauscher, "Applications of transrectal ultrasound in prostate cancer," 2014.

[17] A. Potthoff, M. Gebel, and K. Rifai, "Diagnostic and interventional abdominal ultrasonography," *Der Internist,* vol. 53, pp. 261-270, 2012.

[18] R. H. Taylor, *Computer-integrated surgery: technology and clinical applications*: Mit Press, 1996.

[19] E. M. Boctor, P. Stolka, H. J. Kang, C. Clarke, C. Rucker, J. Croom*, et al.*, "Precisely shaped acoustic ablation of tumors utilizing steerable needle and 3D ultrasound image guidance," in *SPIE Medical Imaging 2010*, San Diego, CA/USA, 2010.

[20] J. Boisvert, D. Gobbi, S. Vikal, R. Rohling, G. Fichtinger, and P. Abolmaesumi, "An open-source solution for interactive acquisition, processing and transfer of interventional ultrasound images," presented at the MICCAI 2008, International Workshop on System and Architectures for Computer Assisted Interventions, 2008.

[21] P. Foroughi, C. Csoma, H. Rivaz, G. Fichtinger, R. Zellars, G. Hager*, et al.*, "Multi-modality fusion of CT, 3D ultrasound, and tracked strain images for breast irradiation planning," in *SPIE Medical Imaging 2009*, Lake Buena Vista, FL/USA, 2009, p. 72651B.

[22] N. Deshmukh, H. Rivaz, and E. Boctor, "GPU-based elasticity imaging algorithms," in *MICCAI-GRID 2009 - International Conference on Medical Image Computing and Computer Assisted Intervention*, London/UK, 2009.

[23] N. Deshmukh, H. Rivaz, P. Stolka, H. Kang, G. Hager, M. Alaf*, et al.*, "Real-time GPU-based analytic minimization/dynamic programming elastography," 2010.

[24] P. J. Stolka, M. Keil, G. Sakas, E. McVeigh, M. E. Allaf, R. H. Taylor*, et al.*, "A 3D-elastography-guided system for laparoscopic partial nephrectomies," 2010, p. 76251I.

[25] N. Kuo, H. J. Kang, T. DeJournett, J. Spicer, and E. Boctor, "Photoacoustic imaging of prostate brachytherapy seeds in ex vivo prostate," 2011, p. 796409.

[26] N. Kuo, H. J. Kang, D. Y. Song, J. U. Kang, and E. M. Boctor, "Real-time photoacoustic imaging of prostate brachytherapy seeds using a clinical ultrasound system," *Journal of biomedical optics,* vol. 17, pp. 066005-1-066005-7, 2012.

[27] X. Wang, J. B. Fowlkes, J. M. Cannata, C. Hu, and P. L. Carson, "Photoacoustic Imaging With a Commercial Ultrasound System and a Custom Probe," *Ultrasound in medicine & biology,* vol. 37, pp. 484-492, 2011.

[28] J. L. S. Su, B. Wang, and S. Y. Emelianov, "Photoacoustic imaging of coronary artery stents," *Opt. Express,* vol. 17, pp. 19894-19901, 2009.

[29] E. Boctor, G. Fichtinger, A. Yeung, M. Awad, R. Taylor, and M. Choti, "Robotic strain imaging for monitoring thermal ablation of liver," *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2004,* pp. 81-88, 2004.

[30] S. Billings, N. Deshmukh, H. J. Kang, R. Taylor, and E. M. Boctor, "System for robot-assisted real-time laparoscopic ultrasound elastography," in *SPIE Medical Imaging 2012*, San Diego, CA/USA, 2012, p. 83161W.

[31] E. M. Boctor, R. J. Webster, M. A. Choti, R. H. Taylor, and G. Fichtinger, "Robotically assisted ablative treatment guided by freehand 3D ultrasound," 2004, pp. 503-508.

[32] E. M. Boctor, R. H. Taylor, G. Fichtinger, and M. A. Choti, "Robotically assisted intraoperative ultrasound with application to ablative therapy of liver cancer," 2003, pp. 281-291.

[33] S. Kim, H. J. Kang, A. Cheng, M. A. Lediju Bell, E. Boctor, and P. Kazanzides, "Photoacoustic image guidance for robot-assisted skull base surgery," in *Robotics and Automation (ICRA), 2015 IEEE International Conference on*, 2015, pp. 592-597.

[34] E. De Montigny, "Photoacoustic Tomography: Principles and applications," ed: OCIS, 2011.

[35] C. Li and L. V. Wang, "Photoacoustic tomography and sensing in biomedicine," *Physics in medicine and biology,* vol. 54, p. R59, 2009.

[36] D. Piras, C. Grijsen, P. Schütte, W. Steenbergen, and S. Manohar, "Photoacoustic needle: minimally invasive guidance to biopsy," *Journal of biomedical optics,* vol. 18, pp. 070502-070502, 2013.

[37] J. L. Su, R. R. Bouchard, A. B. Karpiouk, J. D. Hazle, and S. Y. Emelianov, "Photoacoustic imaging of prostate brachytherapy seeds," *Biomedical optics express,* vol. 2, pp. 2243-2254, 2011.

[38] J. Su, A. Karpiouk, B. Wang, and S. Emelianov, "Photoacoustic imaging of clinical metal needles in tissue," *Journal of biomedical optics,* vol. 15, pp. 021309-021309-6, 2010.

[39] E. M. Boctor, A. Viswanathan, S. Pieper, M. A. Choti, R. H. Taylor, R. Kikinis*, et al.*, "CISUS: an integrated 3D ultrasound system for IGT using a modular tracking API," 2004, p. 27.

[40] A. Enquobahrie, P. Cheng, K. Gary, L. Ibanez, D. Gobbi, F. Lindseth*, et al.*, "The image-guided surgery toolkit IGSTK: an open source C++ software toolkit," *Journal of Digital Imaging,* vol. 20, pp. 21-33, 2007.

[41] H. J. Kang, N. P. Deshmukh, P. Stolka, E. C. Burdette, and E. M. Boctor, "Ultrasound imaging software framework for real-time monitoring of acoustic ablation therapy," in *SPIE Medical Imaging 2012*, San Diego, CA/USA, 2012, p. 83201E.

[42] A. Lasso, T. Heffter, A. Rankin, C. Pinter, T. Ungi, and G. Fichtinger, "PLUS: open-source toolkit for ultrasound-guided intervention systems," 2014.

[43] M. J. Clarkson, G. Zombori, S. Thompson, J. Totz, Y. Song, M. Espak*, et al.*, "The NifTK software platform for image-guided interventions: platform overview and NiftyLink messaging," *International journal of computer assisted radiology and surgery,* pp. 1-16, 2014.

[44] S. Pieper, M. Halle, and R. Kikinis, "3D Slicer," in *Biomedical Imaging: Nano to Macro, 2004. IEEE International Symposium on*, 2004, pp. 632-635.

[45] J. Tokuda, G. S. Fischer, X. Papademetris, Z. Yaniv, L. Ibanez, P. Cheng*, et al.*, "OpenIGTLink: an open network protocol for image-guided therapy environment," *The International Journal of Medical Robotics and Computer Assisted Surgery,* vol. 5, pp. 423-434, 2009.

[46] *Qt SDK*. Available: https://http://www.qt.io/

[47] H. J. Kang, P. J. Stolka, and E. M. Boctor, "OpenITGLinkMUSiiC: A Standard Communications Protocol for Advanced Ultrasound Research," presented at the

MICCAI 2011, International Workshop on System and Architectures for Computer Assisted Interventions, 2011.

[48] E. Boctor, N. Deshmukh, M. Ayad, C. Clarke, K. Dickie, and E. Burdette, "Three-Dimensional Heat-induced Echo-Strain Imaging for Monitoring Interstitial High Intensity Ablation," *The International Society for Optical Engineering (SPIE) on Medical Imaging,* vol. 7265, 2009.

[49] H. J. Kang, X. Guo, R. Z. Azar, A. Cheng, and E. M. Boctor, "Software framework for spatially tracked pre-beamformed RF data acquisition with a freehand clinical ultrasound transducer," presented at the SPIE Medical Imaging, 2014.

[50] H. J. Kang, N. Kuo, X. Guo, D. Song, J. U. Kang, and E. M. Boctor, "Software framework of a real-time pre-beamformed RF data acquisition of an ultrasound research scanner," presented at the SPIE Medical Imaging, 2012.

[51] X. Guo, H.-J. Kang, R. Etienne-Cummings, and E. M. Boctor, "Active Ultrasound Pattern Injection System (AUSPIS) for Interventional Tool Guidance," *PloS one,* vol. 9, p. e104262, 2014.

[52] M. Ali, D. Magee, and U. Dasgupta, "Signal processing overview of ultrasound systems for medical imaging," *Texas Instruments, White Paper SPRAB12,* 2008.

[53] J. Baun, "Physical principles of general and vascular sonography," *San Francisco, Calif: California Publishing Co,* 2004.

[54] *Ultrasonix Wiki*. Available: http://www.ultrasonix.com/wikisonix/index.php/Main_Page

[55] M. A. Hassan and Y. M. Kadah, "Digital Signal Processing Methodologies for Conventional Digital Medical Ultrasound Imaging System," *American Journal of Biomedical Engineering,* vol. 3, pp. 14-30, 2013.

[56] R. E. Daigle, "Ultrasound scan conversion and memory system," ed: Google Patents, 1984.

[57] G. Paladini, "Real-time scan conversion and rendering of ultrasound data," ed: Google Patents, 2003.

[58] J. Ophir, I. Cespedes, H. Ponnekanti, Y. Yazdi, and X. Li, "Elastography: a quantitative method for imaging the elasticity of biological tissues," *Ultrasonic imaging,* vol. 13, pp. 111-134, 1991.

[59] R. M. Lerner, S. Huang, and K. J. Parker, "“Sonoelasticity” images derived from ultrasound signals in mechanically vibrated tissues," *Ultrasound in medicine & biology,* vol. 16, pp. 231-239, 1990.

[60] K. Nightingale, M. S. Soo, R. Nightingale, and G. Trahey, "Acoustic radiation force impulse imaging: in vivo demonstration of clinical feasibility," *Ultrasound in medicine & biology,* vol. 28, pp. 227-235, 2002.

[61] E. Turgay, S. Salcudean, and R. Rohling, "Identifying the mechanical properties of tissue by ultrasound strain imaging," *Ultrasound in medicine & biology,* vol. 32, pp. 221-235, 2006.

[62] E. Konofagou, "Quo vadis elasticity imaging?," *Ultrasonics,* vol. 42, pp. 331-336, 2004.

[63] N. P. Deshmukh, H. J. Kang, S. D. Billings, R. H. Taylor, G. D. Hager, and E. M. Boctor, "Elastography Using Multi-Stream GPU: An Application to Online

Tracked Ultrasound Elastography, In-Vivo and the da Vinci Surgical System," *PloS one,* vol. 9, p. e115881, 2014.

[64] L. V. Wang, *Photoacoustic imaging and spectroscopy*: CRC press, 2009.

[65] S. Park, S. Mallidi, A. B. Karpiouk, S. Aglyamov, and S. Y. Emelianov, "Photoacoustic imaging using array transducer," in *Biomedical Optics (BiOS) 2007*, 2007, pp. 643714-643714-7.

[66] J.-F. Synnevag, A. Austeng, and S. Holm, "Adaptive beamforming applied to medical ultrasound imaging," *Ultrasonics, Ferroelectrics and Frequency Control, IEEE Transactions on,* vol. 54, pp. 1606-1613, 2007.

[67] M. A. Lediju Bell, N. Kuo, D. Y. Song, and E. M. Boctor, "Short-lag spatial coherence beamforming of photoacoustic images for enhanced visualization of prostate brachytherapy seeds," *Biomedical optics express,* vol. 4, pp. 1964--1977, 2013.

[68] M. A. L. Bell, N. P. Kuo, D. Y. Song, J. Kang, and E. M. Boctor, "In vivo photoacoustic imaging of prostate brachytherapy seeds," in *SPIE BiOS*, 2014, pp. 894348-894348-10.

[69] T. Harrison and R. J. Zemp, "Applying ultrasound beamformers to photoacoustic imaging," in *Ultrasonics Symposium (IUS), 2011 IEEE International*, 2011, pp. 2357-2359.

[70] B. Pourebrahimi, S. Yoon, D. Dopsa, and M. C. Kolios, "Improving the quality of photoacoustic images using the short-lag spatial coherence imaging technique," in *SPIE BiOS*, 2013, pp. 85813Y-85813Y-6.

[71] K. P. Köstli, M. Frenz, H. Bebie, and H. P. Weber, "Temporal backward projection of optoacoustic pressure transients using Fourier transform methods," *Physics in medicine and biology,* vol. 46, p. 1863, 2001.

[72] C. A. Kang H.J. , Boctor E., "MUSiiC ToolKit 2.0: Bidirectional Real-time Software Framework for Advanced Interventional Ultrasound Research.," *MICCAI 2012, International Workshop on System and Architectures for Computer Assisted Interventions,* 2012.

[73] K. Chinzei and J. Tokuda, "Extension to OpenIGTLink; Smart Socket Connection, XML as Message, Logging, and One-to-multi Relaying," *MIDAS Journal,* 2009.

[74] *Open IGT Link Protocol (Ver. 2)*
Available: http://www.na-mic.org/Wiki/index.php/OpenIGTLink/ProtocolV2

[75] cisst libraries and Surgical Assistant Workstation (SAW)
Available: https://github.com/jhu-cisst/cisst/wiki

[76] A. Deguet, R. Kumar, R. Taylor, and P. Kazanzides, "The cisst libraries for computer assisted intervention systems," *The MIDAS Journal-Systems and Architectures for Computer Assisted Interventions (MICCAI 2008 Workshop),* 2008.

[77] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs*, et al.*, "ROS: an open-source Robot Operating System," in *ICRA workshop on open source software*, 2009, p. 5.

[78] *ROS WIKI*. Available: http://wiki.ros.org/

[79] E. M. Boctor, M. A. Choti, E. C. Burdette, and R. J. Webster III, "Three-dimensional ultrasound-guided robotic needle placement: an experimental evaluation," *The International Journal of Medical Robotics and Computer Assisted Surgery,* vol. 4, pp. 180-191, 2008.

[80] *Slicer*. Available: http://www.slicer.org/

[81] H. Sen, N. Deshmukh, R. Goldman, P. Kazanzides, R. H. Taylor, E. Boctor*, et al.*, "Enabling technologies for natural orifice transluminal endoscopic surgery (NOTES) using robotically guided elasticity imaging," in *SPIE Medical Imaging 2012*, San Diego, CA/USA, 2012, p. 83161Y.

[82] T. Wilson, J. Zagzebski, T. Varghese, Q. Chen, and M. Rao, "The Ultrasonix 500RP: A commercial ultrasound research interface," *Ultrasonics, Ferroelectrics and Frequency Control, IEEE Transactions on,* vol. 53, pp. 1772-1782, 2006.

[83] M. Ashfaq, S. S. Brunke, J. J. Dahl, H. Ermert, C. Hansen, and M. F. Insana, "An ultrasound research interface for a clinical system," *Ultrasonics, Ferroelectrics and Frequency Control, IEEE Transactions on,* vol. 53, pp. 1759-1771, 2006.

[84] V. Shamdasani, U. Bae, S. Sikdar, Y. M. Yoo, K. Karadayi, R. Managuli*, et al.*, "Research interface on a programmable ultrasound scanner," *Ultrasonics,* vol. 48, pp. 159-168, 2008.

[85] L. Mo, D. DeBusschere, G. McLaughlin, D. Napolitano, W. Bai, K. Fowkes*, et al.*, "Compact ultrasound scanner with simultaneous parallel channel data acquisition capabilities," in *Ultrasonics Symposium, 2008. IUS 2008. IEEE*, 2008, pp. 1342-1345.

[86] C. M. Fabian, K. N. Ballu, J. A. Hossack, T. N. Blalock, and W. F. Walker, "Development of a parallel acquisition system for ultrasound research," in *Medical Imaging 2001*, 2001, pp. 54-62.

[87] M. Crovella, R. Bianchini, T. LeBlanc, E. Markatos, and R. Wisniewski, "Using communication-to-computation ratio in parallel program design and performance prediction," in *Parallel and Distributed Processing, 1992. Proceedings of the Fourth IEEE Symposium on*, 1992, pp. 238-245.

[88] W. R. Stevens, B. Fenner, and A. M. Rudoff, *UNIX network programming* vol. 1: Addison-Wesley Professional, 2004.

[89] T. G. Mattson, B. A. Sanders, and B. L. Massingill, *Patterns for parallel programming*: Pearson Education, 2004.

[90] P.-C. Li and M. O'Donnell, "Elevational spatial compounding," *Ultrasonic imaging,* vol. 16, pp. 176--189, 1994.

[91] *The Visualization Toolkit (VTK)*. Available: http://www.vtk.org

[92] *Ascension Technology Coperation*. Available: http://www.ascensiontech.com

[93] Z. Yaniv, P. Foroughi, H.-J. Kang, and E. Boctor, "Ultrasound calibration framework for the image-guided surgery toolkit (IGSTK)," presented at the SPIE Medical Imaging, 2011.

[94] P. J. Stolka, H. J. Kang, and E. M. Boctor, "The MUSiiC Toolkit: Modular Real-Time Toolkit for Advanced Ultrasound Research," presented at the MICCAI 2010, International Workshop on System and Architectures for Computer Assisted Interventions, 2010.

[95] *Writing Windows NT Server Applications in MFC Using I/O Completion Ports*. Available: https://msdn.microsoft.com/en-us/library/ms810436.aspx

[96] *Epoll*. Available: http://linux.die.net/man/4/epoll

[97] R. Love, *Linux System Programming: Talking Directly to the Kernel and C Library*: " O'Reilly Media, Inc.", 2013.

[98] *Kequeu*. Available:https://developer.apple.com/library/mac/documentation/Darwin/Reference/ManPages/man2/kqueue.2.html

[99] H. Rivaz, P. Foroughi, I. Fleming, R. Zellars, E. Boctor, and G. Hager, "Tracked regularized ultrasound elastography for targeting breast radiotherapy," in *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2009*, ed: Springer, 2009, pp. 507-515.

[100] B. S. Kuszyk, J. K. Boitnott, M. A. Choti, D. A. Bluemke, S. Sheth, C. A. Magee*, et al.*, "Local Tumor Recurrence Following Hepatic Cryoablation: Radiologic-histopathologic Correlation in a Rabbit Model 1," *Radiology,* vol. 217, pp. 477-486, 2000.

[101] L. G. Koniaris, D. Y. Chan, C. Magee, S. B. Solomon, J. H. Anderson, D. O. Smith*, et al.*, "Focal hepatic ablation using interstitial photon radiation energy," *Journal of the American College of Surgeons,* vol. 191, pp. 164-174, 2000.

[102] M. A. Choti, "Surgical management of hepatocellular carcinoma: resection and ablation," *Journal of vascular and interventional radiology,* vol. 13, pp. S197-S203, 2002.

[103] N. Izumi, Y. Asahina, O. Noguchi, M. Uchihara, N. Kanazawa, J. Itakura*, et al.*, "Risk factors for distant recurrence of hepatocellular carcinoma in the liver after complete coagulation by microwave or radiofrequency ablation," *Cancer,* vol. 91, pp. 949-956, 2001.

[104] G. Ter Haar, "Ultrasound focal beam surgery," *Ultrasound in medicine & biology,* vol. 21, pp. 1089-1100, 1995.

[105] K. Kamoi, K. Okihara, A. Ochiai, O. Ukimura, Y. Mizutani, A. Kawauchi*, et al.*, "The utility of transrectal real-time elastography in the diagnosis of prostate cancer," *Ultrasound in medicine & biology,* vol. 34, pp. 1025-1032, 2008.

[106] S. Tan, H. Teh, J. Mancer, and W. Poh, "Improving B mode ultrasound evaluation of breast lesions with real-time ultrasound elastography—a clinical approach," *The Breast,* vol. 17, pp. 252-257, 2008.

[107] Y. Hong, X. Liu, Z. Li, X. Zhang, M. Chen, and Z. Luo, "Real-time ultrasound elastography in the differential diagnosis of benign and malignant thyroid nodules," *Journal of Ultrasound in Medicine,* vol. 28, pp. 861-867, 2009.

[108] A. Săftoiu, P. Vilmann, H. Hassan, and F. Gorunescu, "Analysis of endoscopic ultrasound elastography used for characterisation and differentiation of benign and malignant lymph nodes," *Ultraschall in der Medizin-European Journal of Ultrasound,* vol. 27, pp. 535-542, 2006.

[109] A. Săftoiu, P. Vilmann, F. Gorunescu, D. I. Gheonea, M. Gorunescu, T. Ciurea*, et al.*, "Neural network analysis of dynamic sequences of EUS elastography used for the differential diagnosis of chronic pancreatitis and pancreatic cancer," *Gastrointestinal endoscopy,* vol. 68, pp. 1086-1094, 2008.

[110] P. Foroughi, H. Rivaz, I. Fleming, G. Hager, and E. Boctor, "Tracked ultrasound elastography (TrUE)," *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2010,* pp. 9-16, 2010.

[111] M. Jung, T. Xia, A. Deguet, R. Kumar, R. Taylor, and P. Kazanzides, "A surgical assistant workstation (saw) application for teleoperated surgical robot system," *The MIDAS Journal-Systems and Architectures for Computer Assisted Interventions,* 2009.

[112] Vagvolgyi, B., DiMaio, S., Deguet, A., Kazanzides, P., Kumar, R., Hasser, C., Taylor, R., "The Surgical Assistant Workstation," *MICCAI Workshop on Systems and Arch. for Computer Assisted Interventions*, 2008

[113] P. Kazanzides, S. DiMaio, A. Deguet, B. Vagvolgyi, M. Balicki, C. Schneider*, et al.*, "The surgical assistant workstation (saw) in minimally-invasive surgery and microsurgery," in *MICCAI 2010, International Workshop on System and Architectures for Computer Assisted Interventions*, 2010.

[114] S. DiMaio and C. Hasser, "The da Vinci research interface," in *MICCAI Workshop on Systems and Arch. for Computer Assisted Interventions, Midas Journal*, 2008.

[115] C. L. De Korte, G. Pasterkamp, A. F. Van Der Steen, H. A. Woutman, and N. Bom, "Characterization of plaque components with intravascular ultrasound elastography in human femoral and coronary arteries in vitro," *Circulation,* vol. 102, pp. 617-623, 2000.

[116] S. Huang, K. Kim, R. S. Witte, R. Olafsson, and M. O Donnell, "Inducing and imaging thermal strain using a single ultrasound linear array," *IEEE Transactions on Ultrasonics Ferroelectrics and Frequency Control,* vol. 54, p. 1718, 2007.

[117] R. Souchon, G. Bouchoux, E. Maciejko, C. Lafon, D. Cathignol, M. Bertrand*, et al.*, "Monitoring the formation of thermal lesions with heat-induced echo-strain imaging: A feasibility study," *Ultrasound in medicine & biology,* vol. 31, pp. 251-259, 2005.

[118] J. Shah, S. R. Aglyamov, K. Sokolov, T. E. Milner, and S. Y. Emelianov, "4J-2 Ultrasound-Based Thermal and Elasticity Imaging to Assist Photothermal Cancer Therapy-Preliminary Study," in *Ultrasonics Symposium, 2006. IEEE*, 2006, pp. 1029-1032.

[119] K. M. Hiltawsky, M. Krüger, C. Starke, L. Heuser, H. Ermert, and A. Jensen, "Freehand ultrasound elastography of breast lesions: clinical results," *Ultrasound in medicine & biology,* vol. 27, pp. 1461-1469, 2001.

[120] Q.-L. Zhu, Y.-X. Jiang, J.-B. Liu, H. Liu, Q. Sun, Q. Dai*, et al.*, "Real-time ultrasound elastography: its potential role in assessment of breast lesions," *Ultrasound in medicine & biology,* vol. 34, pp. 1232-1238, 2008.

[121] J. E. Lindop, G. M. Treece, A. H. Gee, and R. W. Prager, "3D elastography using freehand ultrasound," *Ultrasound in medicine & biology,* vol. 32, pp. 529-545, 2006.

[122] M. Sasso and C. Cohen-Bacrie, "Medical ultrasound imaging using the fully adaptive beamformer," in *ICASSP (2)*, 2005, pp. 489-492.

[123] J. A. Jensen, S. I. Nikolov, K. L. Gammelmark, and M. H. Pedersen, "Synthetic aperture ultrasound imaging," *Ultrasonics,* vol. 44, pp. e5-e15, 2006.

[124] J. Shah, L. Ma, K. Sokolov, K. Johnston, T. Milner, S. Y. Emelianov*, et al.*, "Photoacoustic imaging and temperature measurement for photothermal cancer therapy," *Journal of biomedical optics,* vol. 13, pp. 034024-034024-9, 2008.

[125] L. G. Montilla, R. Olafsson, D. R. Bauer, and R. S. Witte, "Real-time photoacoustic and ultrasound imaging: a simple solution for clinical ultrasound systems with linear arrays," *Physics in medicine and biology,* vol. 58, p. N1, 2013.

[126] R. J. Zemp, L. Song, R. Bitton, K. K. Shung, and L. V. Wang, "Realtime photoacoustic microscopy in vivo with a 30-MHz ultrasound array transducer," *Optics express,* vol. 16, pp. 7915-7928, 2008.

[127] J. J. Carson, M. Roumeliotis, G. Chaudhary, R. Z. Stodilka, and M. A. Anastasio, "3D photoacoustic imaging," in *Photonics North 2010*, 2010, pp. 775009-775009-9.

[128] T. Harrison and R. J. Zemp, "Photoacoustic imaging of brachytherapy seeds using a channel-domain ultrasound array system," in *SPIE BiOS*, 2011, pp. 78990H-78990H-7.

[129] E. Boctor, "Prostate brachytherapy seed localization using combined photoacoustic and ultrasound imaging," *Medical Imaging,* pp. 7629-29, 2010.

[130] S. Cheung and R. Rohling, "Enhancement of needle visibility in ultrasound-guided percutaneous procedures," *Ultrasound in medicine & biology,* vol. 30, pp. 617-624, 2004.

[131] C. Chan, F. Lam, and R. Rohling, "A needle tracking device for ultrasound guided percutaneous procedures," *Ultrasound in medicine & biology,* vol. 31, pp. 1469-1483, 2005.

[132] J. Krücker, S. Xu, N. Glossop, A. Viswanathan, J. Borgert, H. Schulz*, et al.*, "Electromagnetic tracking for thermal ablation and biopsy guidance: clinical evaluation of spatial accuracy," *Journal of Vascular and Interventional Radiology,* vol. 18, pp. 1141-1150, 2007.

[133] F. Simonetti, "A guided wave technique for needle biopsy under ultrasound guidance," in *SPIE Medical Imaging*, 2009, pp. 726118-726118-8.

[134] C. Merdest and P. D. Wolf, "Locating a catheter transducer in a three-dimensional ultrasound imaging field," *Biomedical Engineering, IEEE Transactions on,* vol. 48, pp. 1444-1452, 2001.

[135] J. Mung, F. Vignon, and A. Jain, "A non-disruptive technology for robust 3D tool tracking for ultrasound-guided interventions," in *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2011*, ed: Springer, 2011, pp. 153-160.

[136] J. Stoll and P. Dupont, "Passive markers for ultrasound tracking of surgical instruments," in *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2005*, ed: Springer, 2005, pp. 41-48.

[137] D. Vilkomerson and D. Lyons, "A system for ultrasonic beacon-guidance of catheters and other minimally-invasive medical devices," *Ultrasonics, Ferroelectrics and Frequency Control, IEEE Transactions on,* vol. 44, pp. 27-35, 1997.

[138] *Advanced Needle Visulaization (SonoSite Inc.).*

Available: http://www.sonosite.com/needleguide

[139] B. E. Treeby and B. T. Cox, "k-Wave: MATLAB toolbox for the simulation and reconstruction of photoacoustic wave fields," *Journal of biomedical optics,* vol. 15, pp. 021314-021314-12, 2010.

[140] S. J. Orfanidis, *Introduction to signal processing*: Prentice-Hall, Inc., 1995.

[141] N. Otsu, "A threshold selection method from gray-level histograms," *Automatica,* vol. 11, pp. 23--27, 1975.

[142] E. Supriyanto, N. S. A. Zulkifli, M. M. Baigi, N. Humaimi, and B. Rosidi, "Abnormal tissue detection of breast ultrasound image using combination of morphological technique," in *Proceedings of the 15th WSEAS International Conference on Computers, Corfu Island, Greece*, 2011, pp. 234-239.

[143] Y. Zhou and Y.-P. Zheng, "Estimation of muscle fiber orientation in ultrasound images using revoting hough transform (RVHT)," *Ultrasound in medicine & biology,* vol. 34, pp. 1474-1481, 2008.

[144] R. L. Carrau, H. D. Jho, and Y. Ko, "Transnasal-Transsphenoidal Endoscopic Surgery of the Pituitary Gland," *The Laryngoscope,* vol. 106, pp. 914-918, 1996.

[145] H.-D. Jho, "Endoscopic transsphenoidal surgery," *Journal of neuro-oncology,* vol. 54, pp. 187-195, 2001.

[146] P. Cappabianca, L. M. Cavallo, A. Colao, and E. de Divitiis, "Surgical complications associated with the endoscopic endonasal transsphenoidal approach for pituitary adenomas," *Journal of neurosurgery,* vol. 97, pp. 293-298, 2002.

[147] T. Kawamata, H. Iseki, T. Shibasaki, and T. Hori, "Endoscopic augmented reality navigation system for endonasal transsphenoidal surgery to treat pituitary tumors: technical note," *Neurosurgery,* vol. 50, pp. 1393-1397, 2002.

[148] G. Lasio, P. Ferroli, G. Felisati, and G. Broggi, "Image-guided endoscopic transnasal removal of recurrent pituitary adenomas," *Neurosurgery,* vol. 51, pp. 132-137, 2002.

[149] C. B. Wilson and L. C. Dempsey, "Transsphenoidal microsurgical removal of 250 pituitary adenomas," *Journal of neurosurgery,* vol. 48, pp. 13-22, 1978.

[150] R. Park, T. Nyland, J. Lattimer, C. Miller, and J. Lebel, "B-Mode gray-scale ultrasound: imaging artifacts and interpretation principles," *Veterinary Radiology,* vol. 22, pp. 204-210, 1981.

[151] M. A. L. Bell, A. K. Ostrowski, P. Kazanzides, and E. Boctor, "Feasibility of transcranial photoacoustic imaging for interventional guidance of endonasal surgeries," in *SPIE BiOS*, 2014, pp. 894307-894307-7.

[152] F. j. J. Hulsmans, J. A. Castelijns, J. W. Reeders, and G. N. Tytgat, "Review of artifacts associated with transrectal ultrasound: understanding, recognition, and prevention of misinterpretation," *Journal of clinical ultrasound,* vol. 23, pp. 483-494, 1995.

[153] X. Guo, A. Cheng, H. K. Zhang, H.-J. Kang, R. Etienne-Cummings, and E. M. Boctor, "Active Echo: A New Paradigm for Ultrasound Calibration," in *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2014*, ed: Springer, 2014, pp. 397-404.

[154] S. Lavallee, P. Sautot, J. Troccaz, P. Cinquin, and P. Merloz, "Computer-assisted spine surgery: a technique for accurate transpedicular screw fixation using CT data and a 3-D optical localizer," *Computer Aided Surgery,* vol. 1, pp. 65-73, 1995.

[155] S. Tauscher, J. Tokuda, G. Schreiber, T. Neff, N. Hata, and T. Ortmaier, "OpenIGTLink interface for state control and visualisation of a robot for image-guided therapy systems," *International journal of computer assisted radiology and surgery,* pp. 1-8, 2014.

[156] C. Hoelen, F. De Mul, R. Pongers, and A. Dekker, "Three-dimensional photoacoustic imaging of blood vessels in tissue," *Optics letters,* vol. 23, pp. 648-650, 1998.

[157] C. G. Hoelen and F. F. de Mul, "Image reconstruction for photoacoustic scanning of tissue structures," *Applied Optics,* vol. 39, pp. 5872-5883, 2000.

[158] M. Xu and L. V. Wang, "Photoacoustic imaging in biomedicine," *Review of scientific instruments,* vol. 77, p. 041101, 2006.

[159] M. Heijblom, D. Piras, W. Xia, J. C. G. van Hespen, J. M. Klaase, F. M. van den Engh*, et al.*, "Visualizing breast cancer using the Twente photoacoustic mammoscope: What do we learn from twelve new patient measurements?," *Opt. Express,* vol. 20, pp. 11582--11597, May 2012.

[160] S. Sethuraman, J. H. Amirian, S. H. Litovsky, R. W. Smalling, and S. Y. Emelianov, "Spectroscopic intravascular photoacoustic imaging to differentiate atherosclerotic plaques," *Optics express,* vol. 16, pp. 3362--3367, 2008.

[161] T. Udomchai, Q. Chen, T. Varghese, J. A. Zagzebski, and E. L. Madsen, "Noise reduction using spatial-angular compounding for elastography," *Ultrasonics, Ferroelectrics and Frequency Control, IEEE Transactions on,* vol. 51, pp. 510--520, 2004.

[162] M. A. Lediju, M. J. Pihl, J. J. Dahl, and G. E. Trahey, "Quantitative assessment of the magnitude, impact and spatial extent of ultrasonic clutter," *Ultrasonic imaging,* vol. 30, pp. 151--168, 2008.

[163] G. Trahey, S. Smith, and O. von Ramm, "Speckle reduction in medical ultrasound via spatial compounding," presented at the Application of Optical Instrumentation in Medicine XIV and Picture Archiving and Communication Systems (PACS IV) for Medical Applications, 1986.

[164] M. Jaeger, D. Harris-Birtill, A. Gertsch, E. O'Flynn, and J. Bamber, "Deformation-compensated averaging for clutter reduction in epiphotoacoustic imaging in vivo," *Journal of Biomedical Optics,* vol. 17, pp. 0660071--0660078, 2012.

[165] T. J. Allen and P. C. Beard, "Dual wavelength laser diode excitation source for 2D photoacoustic imaging," presented at the Biomedical Optics (BiOS) 2007.

[166] V. Behar, D. Adam, and Z. Friedman, "A new method of spatial compounding imaging," *Ultrasonics,* vol. 41, pp. 377--384, 2003.

[167] H. J. Kang, M. A. L. Bell, X. Guo, R. H. Taylor, and E. M. Boctor, "Freehand spatial-angular compounding of photoacoustic images," presented at the SPIE BiOS, 2014.

[168] B. M. Asl and A. Mahloojifar, "Minimum variance beamforming combined with adaptive coherence weighting applied to medical ultrasound imaging," *Ultrasonics, Ferroelectrics and Frequency Control, IEEE Transactions on,* vol. 56, pp. 1923--1931, 2009.

[169] S. Park, A. B. Karpiouk, S. R. Aglyamov, and S. Y. Emelianov, "Adaptive beamforming for photoacoustic imaging," *Optics letters,* vol. 33, pp. 1291--1293, 2008.

[170] Y.-H. Wang and P.-C. Li, "SNR-dependent coherence-based adaptive imaging for high-frame-rate ultrasonic and photoacoustic imaging," *Ultrasonics, Ferroelectrics, and Frequency Control, IEEE Transactions on,* vol. 61, pp. 1419--1432, 2014.

[171] M. Jaeger, J. C. Bamber, and M. Frenz, "Clutter elimination for deep clinical optoacoustic imaging using localised vibration tagging (LOVIT)," *Photoacoustics,* vol. 1, pp. 19--29, 2013.

[172] M. A. Lediju, M. J. Pihl, S. J. Hsu, J. J. Dahl, C. M. Gallippi, and G. E. Trahey, "A motion-based approach to abdominal clutter reduction," *Ultrasonics, Ferroelectrics and Frequency Control, IEEE Transactions on,* vol. 56, pp. 2437--2449, 2009.

[173] M. A. Lediju, G. E. Trahey, B. C. Byram, and J. J. Dahl, "Short-lag spatial coherence of backscattered echoes: Imaging characteristics," *Ultrasonics, Ferroelectrics and Frequency Control, IEEE Transactions on,* vol. 58, pp. 1377--1388, 2011.

[174] M. A. L. Bell, X. Guo, H. J. Kang, and E. Boctor, "Improved contrast in laser-diode-based photoacoustic images with short-lag spatial coherence beamforming," presented at the Ultrasonics Symposium (IUS), 2014 IEEE International, 2014.

[175] E. J. Alles, M. Jaeger, and J. C. Bamber, "Photoacoustic clutter reduction using short-lag spatial coherence weighted imaging," presented at the Ultrasonics Symposium (IUS), 2014 IEEE International, 2014.

[176] L. Pan, A. Baghani, R. Rohling, P. Abolmaesumi, S. Salcudean, and S. Tang, "Improving photoacoustic imaging contrast of brachytherapy seeds," presented at the SPIE BiOS, 2013.

[177] T. Mitcham, K. Homan, W. Frey, Y.-S. Chen, S. Emelianov, J. Hazle*, et al.*, "Modulation of photoacoustic signal generation from metallic surfaces," *Journal of biomedical optics,* vol. 18, pp. 056008--056008, 2013.

[178] R. Rohling, A. Gee, and L. Berman, "Three-dimensional spatial compounding of ultrasound images," *Medical Image Analysis,* vol. 1, pp. 177--193, 1997.

[179] J. F. Krücker, C. R. Meyer, G. L. LeCarpentier, J. B. Fowlkes, and P. L. Carson, "3D spatial compounding of ultrasound images using image-based nonrigid registration," *Ultrasound in medicine \& biology,* vol. 26, pp. 1475--1488, 2000.

[180] R. R. Entrekin, B. A. Porter, H. H. Sillesen, A. D. Wong, P. L. Cooperberg, and C. H. Fix, "Real-time spatial compound imaging: application to breast, vascular, and musculoskeletal ultrasound," presented at the Seminars in ultrasound, CT and MRI, 2001.

[181] A. L. Gerig, T. Varghese, and J. A. Zagzebski, "Improved parametric imaging of scatterer size estimates using angular compounding," *Ultrasonics, Ferroelectrics and Frequency Control, IEEE Transactions on,* vol. 51, pp. 708--715, 2004.

[182] H. Tu, J. A. Zagzebski, A. L. Gerig, Q. Chen, E. L. Madsen, and T. J. Hall, "Optimization of angular and frequency compounding in ultrasonic attenuation estimations," *The Journal of the Acoustical Society of America,* vol. 117, pp. 3307--3318, 2005.

[183] M. Rao, Q. Chen, H. Shi, and T. Varghese, "Spatial-angular compounding for elastography using beam steering on linear array transducers," *Medical physics,* vol. 33, pp. 618--626, 2006.

[184] H. H. Hansen, R. G. Lopata, and C. L. de Korte, "Noninvasive carotid strain imaging using angular compounding at large beam steered angles: validation in vessel phantoms," *Medical Imaging, IEEE Transactions on,* vol. 28, pp. 872--880, 2009.

[185] T. Koivukangas, J. Katisko, and J. Koivukangas, "Technical accuracy of optical and the electromagnetic tracking systems," *SpringerPlus,* vol. 2, 2013.

[186] M. Siddiqui, "Statistical inference for Rayleigh distributions," *Journal of Research of the National Bureau of Standards, Sec. D,* vol. 68, 1964.

[187] A. dos Anjos and H. Shahbazkia, "Bi-Level Image Thresholding-A Fast Method.," presented at the BIOSIGNALS (2), 2008.

[188] H. J. Kang, X. Guo, A. Cheng, M. A. Choti, and E. M. Boctor, "Needle visualization using photoacoustic effect," in *SPIE BiOS*, 2015, pp. 93232Y-93232Y-7.

HYUN JAE KANG, PHD CANDIDATE

Department of Computer Science, Johns Hopkins University **hjkang@jhu.edu**
Baltimore, MD 21218

SUMMARY

- 10+ years of extensive and wide experience in research, development and engineering on medical imaging and surgical navigation systems
- Ph.D. research dedicated to developing real-time software component framework for advanced ultrasound researches (e.g. ultrasound elastography, photoacoustic imaging)
- Development leader for multiple surgical navigation systems
- Research interests include modular software framework, network-distributed system, medical ultrasound imaging reconstruction and analysis, medical data visualization, localization of surgical instruments and interventional surgical system

PROFESSIONAL EXPERIENCE

JOHNS HOPKINS UNIVERSITY, Department of Computer Science, Baltimore, MD **2008 – Present**

Research Assistant

Developed real-time medical ultrasound imaging and interventional component framework – modular software component framework which enables rapid development and system setup used for ultrasound imaging research.

- **OpenIGTLinkMUSiiC**: The extended version of OpenIGTLink, accomplished by adding new message types and well-defined bidirectional communication interface between components.
  - Proposed entire design concept of **OpenIGTLinkMUSiiC**.
  - Designed and implemented new message types (USMessage, EIMessage) to support real-time ultrasound-driven data streaming.
  - Designed and developed concurrent message interfaces and callback function interface to allow bidirectional message communication on multithreading process.
- **MUSiiC Note**: The collection of software components for ultrasound research, which enables the breakdown of the monolithic processing structure of existing ultrasound system into a collection of independent and communicating software modules.
  - Proposed entire design concept of **MUSiiC Note**.
  - Implemented real-time ultrasound data acquisition and network server system for real-time GPU based ultrasound elastography and robot-assisted real-time laparoscopic ultrasound elastography.
  - Proposed and implemented real-time multiple data synchronization to combine multiple OpenIGTLink messages with the timestamp of each message.
  - Developed spatially tracked ultrasound data acquisition system for ultrasound calibration and freehand ultrasound elastography for in-vivo application.
- **Precisely shaped acoustic ablation of tumor under 3D ultrasound image guidance** (Collaboration with Acoustic Medical system, Inc., University of California at San Francisco and Vanderbilt University): A true closed-loop system for steering, placement, guidance, percutaneous delivery of ultrasound thermal ablation tool, and on-line monitoring of the treatment of hepatic tumor.
  - Implemented sensorless freehand 3D ultrasound system, and validated its accuracy with phantom study.
  - Integrated real-time GPU based ultrasound elastography system with interventional ultrasound thermal ablation system (Acoustic MedSystems TheraVisionTM) for real-time thermal monitoring of acoustic ablation therapy, and validated the performance through a series of in vivo experiments.
- **Interventional photoacoustic surgical system (i-PASS):** Novel photoacoustic ultrasound imaging techniques for accurate, reliable, and efficient fusion of intraoperative ultrasound (IOUS) images with preoperative images, real time video, and other data to assist surgeons in minimally invasive surgery and other interventional procedures.
  - Developed real-time spatially tracked photoacoustic imaging system and network data server to localize prostate brachytherapy seeds using a clinical ultrasound system.
  - Invented 3 freehand spatial-angular compounding methods of photoacoustic images to enhance the contrast of photoacoustic image.
  - Developed 3D photoacoustic data acquisition system for concurrent photoacoustic markers for direct three-dimensional ultrasound to video registration.
  - Proposed and developed a technique of needle visualization using photoacoustic effect.

CYBERMED, INC., Seoul, Korea **2006 – 2008**

R&D Manager
Led a development team for **image-guided surgical navigation systems**, the first commercial surgical navigation systems in Korea. Managed / Directed entire development cycle: brainstorming, idea evaluation, software / hardware design, developing manufacturing processes, and managing OEM suppliers.

- **Image-guided surgical navigation systems**
  - **Neurosurgical navigation system (In2Vision^{TM}):**
    - Led 3 software engineers to implement surface registration method (ICP) and integrate Diffused Tensor Imaging (DTI) technique, including brainstorming, designing software architecture, implementation, and clinical test.
    - Engineering duties were evaluation idea, designing software architecture, and organizing clinical test.
  - **Image-free total knee replacement (TKR) navigation system (In2Knee^{TM}):**
    - Led 4 software engineers to develop user-interface software for In2Knee^{TM}.
    - Personal duties were designing and manufacturing surgical instruments (a fine-adjustable cutting block) for In2Knee^{TM}, coordinating OEM supplier collaboration, building system, and organizing clinical test.
  - **Dental implant surgical navigation system (In2Guide^{TM}):**
    - Led a team of three software engineers, starting from surveying the state of arts, building the concept of product, designing software architecture, and ending with implementation.
    - Engineering duties were evaluation idea, designing software architecture, and implementation.
- **A grant application of development of intravascular micro-robot development** (Accepted, Funded by the ministry of knowledge economy of Korea with the total amount of $ 20M, September 2007 – June 2014): To develop a micro-robot system to help cure blood vessel disease, such as Chronic Total Occlusions (CTO) and Thrombi within blood vessels.
  - Proposed and wrote the sub-grant proposal related to recognition of the pose of micro-robot in human body with real-time.
- **A grant application of development of medical treatment technology for blood vessel treatment by a micro-robot** (Accepted, Funded by the ministry of knowledge economy of Korea with the total amount of $3M, June 2008 – May 2013): To develop the common technology for robotic surgery system to improve the accuracy of surgery by combining image processing technology and the conventional surgical technology.
  - Proposed and wrote the sub-grant proposal related to image-guided surgical system.
- **Proposed and wrote 16 patents related to image guided surgical systems.**

CYBERMED, INC., Seoul, Korea **2004 – 2006**

Researcher
Performed research and development of **image-guided surgical navigation systems.**

- **Image-guided surgical navigation systems**
  - **Neurosurgical navigation system (In2Vision^{TM}):**
    - Developed and integrated the real-time video image interface of surgical microscope using a Matrox METEOR board and MIL library.
    - Developed and integrated the functional interface of surgical microscope (NC-4, Carl Zeiss Microscope) using RS232C protocol.
    - Developed and implemented a virtual-tip tracking function with the focal distance of surgical microscope (NC-4, Carl Zeiss Microscope).
    - Designed and fabricated neurosurgical navigation instruments.
  - **Spine surgical navigation system (In2Spine^{TM}):**
    - Designed and fabricated spine surgical navigation instruments.
    - Developed the real-time video image interface of C-arm using a Matrox METEOR board and MIL library.

KOREA INSTITUTE OF SCIENCE & TECHNOLOGY (KIST), Seoul, Korea **2002 – 2004**

Researcher
Researched and developed micromanipulations and wearable robot.

- **Sensor based micromanipulation system:**
  - **Gelatin-patterned cover glass for single bio-cell manipulation:** In-vitro cell culture scaffolds for patterned cell culturing.
    - Proposed and fabricated a gelatin patterned cover class for holding and culturing mouse bone marrow cell.
    - Measured of the property of the gelatin-patterned cover glass with Atomic Force Microscope (AFM).

KOREA INSTITUTE OF SCIENCE & TECHNOLOGY (KIST) (Continued)

- o **Single bio-cell manipulation:** A micro-robotic injector with cellular force sensing capabilities and a visual tracking function.
  - − Developed and integrated the vision-based micromanipulation of single bio-cell (mouse embryo).
  - − Developed and implemented auto-cell detection algorithm with pattern matching method.
  - − Designed and fabricated PVDF force sensor to measure cellular force.
- o **Flexible micro-assembly system:** A flexible micro-assembly system based on hybrid manipulation scheme with sensory feedback of vision and force.
  - − Developed and integrated a micromanipulation system based on vision and scaled teleoperation technique with force feedback for assembling optical components.
  - − Proposed and implemented an image processing algorithm of monitoring the pose of optical components in real-time.
  - − Implemented and integrated a haptic interface for force feedback-based micro-assembly.
  - − Developed the graphic user interface of the micromanipulation system.

- **Wearable robotic arm:** A 7-DOF wearable robotic arm with pneumatic actuators and teleopearation control
  - o Developed a 3D virtual wearable robotic arm that interacts with the wearable robotic arm.

SNU PRECISION CO. LTD. AND SEOUL NATIONAL UNIVERSITY, Seoul, Korea **2000 – 2002**

Researcher
Performed research and development of vision-based inspection system.

- **Vision-based inspection system of optical components**
  - o **Inspection system for pigtailed optical fiber:** A high-precision non-contact inspection system of pigtailed optical fiber.
    - − Proposed and developed an inspection algorithm based on image processing technique to detect the facet angle of pigtailed optical fiber.
    - − Developed and integrated the graphic user interface of the inspection system.
    - − Developed and integrated the video interface of optical microscope.
  - o **Inspection system for ferrules (optical component):** A high-precision non-contact automated inspection system of ferrules.
    - − Developed the control interface motorized x-y-z stage for automated inspection with RS232C protocols.

EDUCATION

Ph.D. Candidate, Department of Computer Science, Johns Hopkins University, Baltimore, MD, 2008 ~ Current

- − Thesis Topic: Real-time medical ultrasound imaging and interventional component framework for advanced ultrasound research

M.S.E., Department of Computer Science, Johns Hopkins University, Baltimore, MD, 2010

M.S., Mechanical Design & Production Engineering, Seoul National University, Seoul, Korea, 2002

- − Dissertation: Development of Inspection System for PigTail using Computer Vision with Optical Microscope

B.S., Mechanical Engineering, Kyungpook National University, Daegu, Korea, (GPA 4.04 / 4.3), 2000

HONORS AND AWARDS

Education Fellowship for excellent academic achievement, Woosung Foundation, Seoul, Korea, March 1997 ~ December 2001
Honor Student Scholarship, March 1996 ~ February 1997

TECHNICAL SKILLS

Programming Tools	C, C++, Microsoft Visual C++ (MFC), Qt SDK, JAVA, Python, CMake, OpenGL, DirectX, VTK, OpenCV, MATLAB, MIL (Matrox Image processing library), GHOST (Haptic device SDK), Ultrasonix SDK
Design Software	3D Max, SolidWorks, Pro/Engineer, AutoCAD
Simulation Tools	ANSYS, COMSOL, K-Wave
Ultrasound Machine	Ultrasonix CEP/RP/ Touch (Analogic Ultrasound.)
Ultrasound Channel Data Acquisition	Ultrasonix DAQ (Analogic Ultrasound
Laser System	Brilliant B (850 mJ) (Quantel)
Motion tracking system	Polaris Polaris Spectra/Vicra/Aurora (NDI, Inc.), 3D Guidance medSAFE (Ascension Technology Co.), Micron Tracker (Claron Technology Inc.)
Image Grab Board	Genesis, Meteor-MC, Meteor-Standard (Matrox Inc.)
Microscope	Inverted microscope IX-81 (Olympus Co.), MZ 12.5 (Leica)
Surgical Microscope	OPMI Neuro/NC 4 (Carl Zeiss AG),
Confocal Microscope	LSM 5 PASCAL (Carl Zeiss AG)
Atomic Force Microscope (AFM)	XE-100 (PSIA Inc.)
Haptic Device	PHANTOM-Haptic Device (SensAble Technologies, Inc.)
Micro/Nano Manipulation	MM3A-Nano-Manipulator (Kleindiek Nanotechnik GmbH), MP-285 micromanipulator (Shutter Inc.), NI-2 –biomanipulator (Eppendorf AG)

BIOLOGY RESEARCH

Experienced in culturing biological cells: zebrafish embryo, bone-marrow stem cell of mouse, cardiac cell of mouse, P-19 stem cell.

HYUN JAE KANG, PHD CANDIDATE

Department of Computer Science, Johns Hopkins University **hjkang@jhu.edu**
Baltimore, MD 21218

PATENTS AND PATENT APPLICATIONS

1. Emad M. Boctor, Behnoosh Tavakoli, **Hyun Jae Kang**, Xiaoyu Guo, Jin Kang, "SYSTEMS AND METHODS FOR REAL-TIME TRACKING OF PHOTOACOUSTIC SENSING", PCT International Patent Application - PCT/US2014/68650, Priority Claim(s): 61/911,586 filed 04 December 2013.
2. Sun Ha Paek, Yong Hoon Lim, **Hyun Jae Kang**, Se Ho Shin, Eun Ju Choi, Cheol Young Kim, "METHOD OF DETERMINING THE POSITION OF A DEEP BRAIN STIMULATION ELECTRODE", U.S.A Patent Application #2010324410 (published Dec. 23, 2010), Germany Patent #112008001642 (published Dec. 9, 2010), and Korean Patent# 1008975280000 issued May 07, 2009.
3. Cheol Young Kim, **Hyun Jae Kang**, Eun Ju Choi, Jae-ho Yoon, "MEDICAL INSTRUMENT USED IN MEDICAL NAVIGATION AND MEDICAL NAVIGATION METHOD," Korean Patent # 1009577270000 issued May 6, 2010.
4. Cheol Young Kim, **Hyun Jae Kang**, Se Ho Shin, "METHOD OF OPERATING A SURGICAL NAVIGATION SYSTEM," Korean Patent # 1009577130000, issued May 6, 2010.
5. Cheol Young Kim, **Hyun Jae Kang**, Jae-ho Yoon, "BONE CUTTING GUIDE USED IN A MEDICAL NAVIGATION SYSTEM," Korean Patent # 1009416140000, issued Feb 3, 2010.
6. Cheol Young Kim, **Hyun Jae Kang**, Eun Ju Choi, Se Ho Shin, "NAVIGATION METHOD IN BONE ABLATION SURGERY," Korean Patent # 1009416120000, issued Feb 3, 2010
7. Cheol Young Kim, Hyungmin Kim**,** **Hyun Jae Kang**, "UNIVERSAL TRACKER ADAPTER FOR SURGICAL NAVIGATION SYSTEM," Korean Patent # 1007458410000, issued July 27, 2007.
8. Cheol Young Kim, Hyungmin Kim, **Hyun Jae Kang**, "LASER INDICATOR FOR SURGICAL NAVIGATION CAMERA," Korean Patent # 1007715000000, issued Oct. 27, 2007.
9. **Hyun Jae Kang**, Cheol Young Kim, Se Ho Shin, "METHOD OF TREATING 3D MEDICAL VOLUME DATA SET," Korean Patent # 1009151230000, issued Aug. 26, 2009.
10. **Hyun Jae Kang**, Se Ho Shin, Cheol Young Kim, "METHOD OF OPERATING A MEDICAL NAVIGATION SYSTEM," Korean Patent # 1008804030000, issued Jan. 19, 2009.
11. **Hyun Jae Kang**, Cheol Young Kim, Jae-hoo Yoon, "BONE CUTTING GUIDE," Korean Patent # 1009060930000, issued June 29, 2009.
12. **Hyun Jae Kang**, Hyungmin Kim, Cheol Young Kim, "DEVICE FOR FIXING A REFERENCE FRAME AND CORRECTIONMETHOD IN MEDICAL NAVIGATION SYSTEM USING THE DEVICE," Korean Patent # 1007648150000, issued Oct 1, 2007.
13. **Hyun Jae Kang**, Hyunmin Kim, Won Chang Hu, Cheol Young Kim, "AUTOMATIC RE-REGISTERATION METHOD IN MEDICAL NAVIGATIONSYSTEM," Korean Patent # 1006769280000, issued Jan. 25, 2007.
14. **Hyun Jae Kang**, Hyungmin Kim, Won Chang Hu, Cheol Young Kim, "CORRECTION METHOD IN MEDICAL NAVIGATION SYSTEM," Korean Patent #1020050128832, issued Oct. 1, 2007.
15. **Hyun Jae Kang**, Chang Wan Oh, Cheol Young Kim, Won Chang Hu, Hyungmin Kim, "PROBE FOR MEDICAL NAVIGATION SYSTEM," Korean Patent # 1005994910000, issued July 4, 2006.
16. **Hyun Jae Kang**, Jeonhwan Sung, Hyungmin Kim, Won Chang Hu, Cheol Young Kim, "METHOD FOR CALIBRATING A MEDICAL MICROSCOPE IN MEDICALNAVIGATION SYSTEM," Korean Patent (Application date: 2005.03.26, Application No: 1020060103386)
17. **Hyun Jae Kang**, Jeonhwan Sung, Hyungmin Kim, Won Chang Hu, Cheol Young Kim, "APPARATUS FOR CALIBRATING A MEDICAL MICROSCOPE IN MEDICAL NAVIGATION SYSTEM," Korean Patent # 1005188370000, issued September, 2005.
18. Byungkyu Kim, **Hyun Jae Kang**, Youngwoo Namkung, "HYBRID TYPE MICRO MANIPULATION METHOD AND THE SYSTEM THEREOF", Korean Patent #0545962, January, 2006.
19. **Hyun Jae Kang**, Kyunghwan Kim, Eun Ha Song, Deok-Ho Kim, "METHOD AND DEVICE FOR ASSEMBLING MEMS COMPONENTS," Korean Patent #0473348, issued February, 2005.
20. **Hyun Jae Kang**, Jin Woo Lee, Beyong Kwon Ju, Sang Ju Kwon, Soo Kyung Chae, Jong-Oh Park., "A DUAL LIGHT-REFLECTION STRUCTURE AND A METHOD OF MANUFACTURING THE DUAL LIGHT-REFLECTION STRUCTURE," Korean Patent #0649124, issued November, 2006.

PUBLICATIONS (PEER-REVIEWED)

1. **Hyun Jae Kang**, Muyinatu A. Lediju Bell, Xiaoyu Guo, Emad M. Boctor, "Freehand spatial-angular compounding of photoacoustic images", in preparation.
2. Xiaoyu Guo, **Hyun Jae Kang**, Ralph Etienne-Cummings, Emad M. Boctor, "Active Ultrasound Pattern Injection System (AUSPIS) for Interventional Tool Guidance." PloS one 9.10 (2014): e104262.
3. Pezhman Foroughi, **Hyun Jae Kang**, Daniel A. Carnegie, Mark G. van Vledder, Michael Choti, Gregory D. Hager, Emad M. Boctor, "A Freehand Ultrasound Elastography System with Tracking for In Vivo Applications," Ultrasound in Medicine & Biology, Volume 39, Issue 2, February 2013, Pages 211~225
4. **Hyun Jae Kang**, Alexis Cheng, Emad M. Boctor, "MUSiiC ToolKit 2.0: Bidirectional Real-time Software Framework for Advanced Interventional Ultrasound Research." The MIDAS Journal - Computer Assisted Intervention and MICCAI 2012, International Workshop on Systems and Architectures for Computer Assisted Interventions.
5. Nathanael Kuo, **Hyun Jae Kang**, Danny Y Song, Jin U Kang, Emad M Boctor, "Real-time Photoacoustic Imaging of Prostate Brachytherapy Seeds Using a Clinical Ultrasound System", Journal of Biomedical Optics, 17(6), June 2012
6. **Hyun Jae Kang**, Philipp J. Stolka, Emad M. Boctor, "OpenITGLinkMUSiiC: A Standard Communications Protocol for Advanced Ultrasound Research". The MIDAS Journal - Computer Assisted Intervention and MICCAI 2011, International Workshop on Systems and Architectures for Computer Assisted Interventions 2011
7. Philipp J. Stolka, **Hyun Jae Kang**, Emad M. Boctor, "The MUSiiC Toolkit: Modular Real-Time Toolkit for Advanced Ultrasound Research". The MIDAS Journal - Computer Assisted Intervention and MICCAI 2010, International Workshop on Systems and Architectures for Computer Assisted Interventions 2010 (Beijing/China)
8. Byungkyu Kim, **Hyun Jae Kang**, Deok-Ho Kim, Jong-Oh Park, "A flexible microassembly system based on hybrid manipulation scheme for manufacturing photonics components," International Journal of Advanced Manufacturing Technology, vol.28, pp.379-386, 2006.
9. Deok-Ho Kim, Byungkyu Kim, **Hyun Jae Kang**, "Development of a piezoelectric polymer-based sensorized microgripper for micromanipulation and microassembly," Microsystem Technologies, vol.10, no.4, pp.275-280, 2004.
10. Deok-Ho Kim, Byungkyu Kim, Seok Youn, **Hyun Jae Kang**, "Cellular force sensing for force feedback-based biological cell injection," Transactions of the Korean Society of Mechanical Engineers (KSME), A, vol.27, no.12, pp.2079-2084, 2003.
11. Deok-Ho Kim, Byungkyu Kim, **Hyun Jae Kang**, Sang Min Kim, "Design, fabrication and performance evaluation of a sensorized superelastic alloy micro robot gripper," Transactions of the Korean Society of Mechanical Engineers (KSME), A, vol.27, no.10, pp.1772-1777, 2003.
12. Deok-Ho Kim, Byungkyu Kim, **Hyun Jae Kang**, "Force Feedback-based Microassembly: System Integration and Experiments," IEEE/ASME Transaction on Mechatronics, Oct., 2003.
13. Byungkyu Kim, **Hyun Jae Kang**, Deok-Ho Kim, Jong-Oh Park, "Flexible Microassembly System based on Hybrid Manipulation Scheme for Photonics Applications," IEEE Transaction on Industrial Electronics, Aug., 2003.

PUBLICATIONS (CONFERENCES)

1. Xiaoyu Guo, Alexis Cheng, Haichong Kai Zhang, **Hyun Jae Kang**, Ralph Etienne-Cummings, Emad M. Boctor, "Active Echo: A new Paradigm for Ultrasound Calibration", Medical Image Computing and Computer-Assisted Intervention– MICCAI 2014, 397-404.
2. Muyinatu A. Lediju Bell, Xiaoyu Guo, **Hyun Jae Kang**, Alexis Cheng, Behnoosh Tavakoli, Emad M. Boctor, "Improved contrast in laser-diode-based photoacoustic images with short-lag spatial coherence beamforming" IEEE Ultrasonics Symposium, 2014.
3. **Hyun Jae Kang**, Xiaoyu Guo, Reza Zahiri Azar, Alexis Cheng, Emad M. Boctor, "Software framework for spatially tracked pre-beamformed RF data with a freehand clinical ultrasound transducer." Proc. SPIE 9040, Medical Imaging 2014: Ultrasonic Imaging and Tomography, 90401W (March 20, 2014).
4. **Hyun Jae Kang**, Muyinatu A. Lediju Bell, Xiaoyu Guo, Russell H. Taylor, Emad M. Boctor, "Freehand spatial-angular compounding of photoacoustic images." Proc. SPIE 8943, Photons Plus Ultrasound: Imaging and Sensing 2014, 894361 (March 3, 2014).
5. Xiaoyu Guo, Behnoosh Tavakoli, **Hyun Jae Kang**, Jin U. Kang, Ralph Etienne-Cummings, Emad M. Boctor, "Photoacoustic active ultrasound element for catheter tracking." Proc. SPIE 8943, Photons Plus Ultrasound: Imaging and Sensing 2014, 89435M (March 3, 2014); doi:10.1117/12.2041625.

PUBLICATIONS (CONFERENCES) (Continued)

6. Alexis Cheng, Xiaoyu Guo, **Hyun Jae Kang**, Behnoosh Tavakoli, Jin U. Kang, Russell H. Taylor, Emad M. Boctor, "Concurrent photoacoustic markers for direct three-dimensional ultrasound to video registration." Proc. SPIE 8943, Photons Plus Ultrasound: Imaging and Sensing 2014, 89435J (March 3, 2014); doi:10.1117/12.2040321.
7. Haichong Kai Zhang, Alexis Cheng, Fereshteh Aalamifar, Hyun-Jae Kang, Gregg Trahey, Emad M. Boctor, "Robotic 2D Ultrasound Calibration Utilizing Trajectory of Moved Phantom", Ultrasound Imaging and Tissue Characterization, 2014.
8. Muyinatu A. Lediju Bell, **Hyun Jae Kang**, Xiaoyu Guo, Danny Y Song, Emad M. Boctor, "Real-time transurethral photoacoustic imaging of prostate brachytherapy seeds", Ultrasound Imaging and Tissue Characterization, 2014.
9. Xiaoyu Guo, Behnoosh Tavakoli, **Hyun Jae Kang**, Jin U. Kang, Ralph Etienne-Cummings, Emad M. Boctor,"All-optical Active Ultrasound Pattern Injection System", Ultrasound Imaging and Tissue Characterization, 2014.
10. Xiaoyu Guo, Behnoosh Tavakoli, **Hyun Jae Kang**, Jin U. Kang, Ralph Etienne-Cummings, Emad M. Boctor, "Localizing Surgical Tools with an Ultrasound-based Active Reflector Tracking System", Hopkins Imaging Conference, 2013
11. Xiaoyu Guo, **Hyun Jae Kang**, Muyinatu A. Lediju Bell, Ralph Etienne-Cummings, Emad M. Boctor,"Localizing Surgical Tools with an Ultrasound-based Active Reflector Tracking System", Ultrasound Imaging and Tissue Characterization, 2013.
12. **Hyun Jae Kang**, Muyinatu A. Lediju Bell, Xiaoyu Guo, Russell H. Taylor, Emad M. Boctor, "Freehand Spatial-Angular Compounding of Photoacoustic Images", Ultrasound Imaging and Tissue Characterization, 2013.
13. Daniel A. Carnegie, Emad M. Boctor, Xiaoyu Guo, **Hyun Jae Kang**, Nishikant Deshmukh, Pezhman Foroughi, E. Clif Burdette, R. Webster, J. Burgner, Michael Choti,"Accurate High-intensity Focused Ultrasound Ablation in a Porcine Liver Model Through Integration of real-time Image Guidance, Robotic Navigation, and Elastographic Monitoring", SAGES 2012, ET011, 2012.
14. **Hyun Jae Kang**, Nishikant Deshmukh, Philipp Stolka, Emad M. Boctora, E. Clif Burdette, "Ultrasound Imaging Software Framework for Real-Time Monitoring of Acoustic Ablation Therapy", SPIE Medical Imaging 2012, San Diego.
15. **Hyun Jae Kang**, Nathanael Kuo, Xiaoyu Guo, Danny Y Song, Jin U. Kang, Emad M. Boctor, "Software Framework of a Real-time Pre-beamformed RF Data Acquisition of an Ultrasound Research Scanner", SPIE Medical Imaging 2012, San Diego.
16. Seth Billings, Nishikant P. Deshmukh, **Hyun Jae Kang**, Russell H. Taylor, Emad M. Boctor, "System for Robot-Assisted Real-Time Laparoscopic Ultrasound Elastography", SPIE Medical Imaging 2012, San Diego.
17. Nathanael Kuo, **Hyun Jae Kang**, Travis DeJournett, James Spicer, Emad M. Boctor, "Photoacoustic Imaging of Prostate Brachytherapy Seeds in Ex Vivo Prostate," SPIE Medical Imaging 2011 (Lake Buena Vista, FL/USA)
18. Ziv Yaniv, Pezhman Foroughi, **Hyun Jae Kang**, Emad M. Boctor, "Ultrasound calibration framework for the image-guided surgery toolkit". SPIE Medical Imaging 2011 (Lake Buena Vista, FL/USA)
19. Nishikant P. Deshmukh, Hassan Rivaz, Philipp J. Stolka, **Hyun Jae Kang**, Gregory D. Hager, Mohamad E. Alaf, Emad M. Boctor, "Real-time GPU based ultrasound Elastography". International Conference on the Ultrasonic Measurement and Imaging of Tissue Elasticity, 2010, Utah
20. Nishikant P. Deshmukh, Hassan Rivaz, Philipp J. Stolka, **Hyun Jae Kang**, Gregory D. Hager, Mohamad E. Alaf, Emad M. Boctor: "Real-time GPU-based Analytic Minimization/Dynamic Programming Elastography". HP-MICCAI 2010 (Beijing, China)
21. Philipp J. Stolka, **Hyun Jae Kang**, Michael Choti, Emad M. Boctor, "Multi-DoF Probe Trajectory Reconstruction with Local Sensors for 2D-to-3D Ultrasound". 2010 IEEE International Symposium on Biomedical Imaging (ISBI) (Rotterdam, The Netherlands)
22. Hassan Rivaz, **Hyun Jae Kang**, Philipp J. Stolka, Emad M. Boctor, "Novel reconstruction and feature exploitation techniques for sensorless freehand 3D ultrasound". SPIE Medical Imaging 2010 (San Diego, CA/USA)
23. Deok-Ho Kim, Byungkyu Kim, **Hyun Jae Kang**, Jong-Oh Park, "Force Feedback-based Microassembly System Integrated with a Piezoelectrically Sensorized Microgripper," The 2nd International Conference on Mechatronics and Information Technology (ICMIT), Jecheon, Korea, December 4-6, 2003.
24. Byungkyu Kim, Deok-Ho Kim, Jong-Oh Park, Young Ho. Kim, Sang Ju Kwon, **Hyun Jae Kang**, Seng Hwan Jung, "Autonomous Biomanipulation Factory for Manipulating Individual Embryo Cells," (Invited Paper) IEEE/RSJ International Conference on Intelligent Robots and Systems, pp.74-88, Las Vegas, USA, October 27-November 1, 2003. (Workshop:Microrobotics for Biomanipulation)
25. Byungkyu Kim, **Hyun Jae Kang**, Deok-Ho Kim, Gwi Tae Park, Jong Oh Park, "Flexible Microassembly System based on Hybrid Manipulation Scheme," IEEE/RSJ International Conference on Intelligent Robots and Systems, pp.2061-2066, Las Vegas, USA, October 27-November 1, 2003.

PUBLICATIONS (CONFERENCES) (Continued)

26. **Hyun Jae Kang**, Sang Min Kim, Youngwoo NamGung, Byungkyu Kim, "Microassembly system for the assembly of photonic components," Proc. of KSPE Spring Annual Meeting, pp.241-245, June, 2003, Jeju, Korea.
27. Deok-Ho Kim, Byungkyu Kim, Seok Youn, **Hyun Jae Kang**, "Cellular Force Measurement for Force Feedback-Based Biomanipulation," KSPE Spring Annual Meeting, pp.237-240, June, 2003. (in Korean)
28. Deok-Ho Kim, Byungkyu Kim, **Hyun Jae Kang**, Sang Min Kim, "Fabrication and Sensorization of a Superelastic Alloy Microrobot Gripper using Piezoelectric Polymer Sensors," KSPE Spring Annual Meeting, pp. 251-255, June, 2003. (in Korean)
29. Geunho Lee, **Hyun Jae Kang**, Sang Joo Kwon, Gwi Tae Park, Byungkyu Kim, "Development of Biological Cell Manipulation System using Visual Tracking Method," Proceeding of IEEK Summer Conference, July 9~10, 2003, Kangwondo, Korea, pp.2911~2914.
30. Keunyoung Kim, **Hyun Jae Kang**, Kyunghwan Kim, "Direct Input device based on Augmented Reality," ICCAS2002, October 16 - 19, 2002, Muju Resort, Jeonbuk, Korea.
31. Byungkyu Kim, Kyunghwan Kim, **Hyun Jae Kang**, Deok-Ho Kim, Jae Hoon Lee, Sang Min Kim, "Hybrid Microassembly using Intelligent User Interface", The 10th G7-Advanced Manufacturing System Workshop, Seoul, Korea, Sept 6th, 2002 (in Korean).