# The MUSiiC Toolkit: Modular Real-Time Toolkit for Advanced Ultrasound Research

Philipp J Stolka*, Hyun-Jae Kang*, Emad M Boctor†
*Computing Science, Johns Hopkins University, Baltimore, MD/USA
†Department of Radiology, Medical Imaging Physics Division, Johns Hopkins Medical Institutions, Baltimore, MD/USA

***Abstract*—Ultrasound is a very useful and convenient modality in medical imaging, due to its relatively small footprint, high resolution, negligible adverse effects, and real-time imaging capability. However, it is fairly difficult to fully benefit from its resolution and speed in advanced imaging research because of the closed nature of most ultrasound systems. This becomes even more pressing when pursuing research in advanced ultrasound imaging modalities, requiring low-level data access.**
**We propose a modular and extensible open-source toolkit *MUSiiC S/W* for rapid development and system setup for ultrasound research. Aimed at algorithm and experimental system builders as well as at clinical research, it allows standardized access to various types of ultrasound-derived real-time data streams, incorporates various tracking facilities, volume reconstruction, and connections to external software tools. By providing well-defined open standard interfaces between those building blocks, we aim at easy extensibility for third-party researchers, giving access to a variety of ultrasound and tracking systems.**

## I. Introduction

### A. Background, Motivation and Objective

Where clinical ultrasound systems offer B-mode, Doppler, and sometimes elasticity imaging (*EI*) as standard imaging modalities, advanced ultrasound research – involving e.g. synthetic aperture imaging, adaptive beamforming, photoacoustic imaging, thermal imaging, high-performance elasticity imaging etc. – requires access to the underlying ultrasound radio-frequency (*RF*) data.

Although some manufacturers' systems allow the sampling and storage of RF data through sometimes proprietary, sometimes open interfaces (e.g. Ultrasonix [29], Siemens [2], Hitachi [21], ZONARE [14], or Philips/Agilent [7]), the resulting systems are relatively closed, and data is often saved into files in an asynchronous fashion, making access only offline and often requiring "monkey jobs" such as pressing buttons and scrolling during acquisition.

For research projects, once the basic RF data access hurdle is taken, the next step involves the computation of the mentioned derived imaging modalities, followed by displaying the results, tracking, image guidance, and a variety of other process steps depending on the particular application under consideration. The development of such systems will naturally draw on existing drivers, components and interfaces, but will probably introduce large complexity issues due to the amount of interacting parts. Real-time and image processing toolkits aim to reduce the time and development investment related to recurring modeling and interfacing tasks.

### B. Toolkit Landscape

A considerable amount of toolkits, frameworks, and component collections is populating the field of real-time and/or image guidance research and (ultrasound) system development. Apart from the low-level vendor APIs and related research toolkits (cf. above), several open-source projects have stepped up to this task.

One toolkit aiming at modular abstraction in image-guided therapy (*IGT*) in the widest sense deals with the problem at the networking level: OpenIGTLink [27] defines a protocol for data transmission between communication parties and provides a reference implementation for TCP/IP sockets. Considered data types include images, positions, status messages etc. It should be considered not so much an IGT toolkit but rather an important enabling technology, as many other approaches are building on the interoperability that OpenIGTLink provides.

SynchroGrab [1] for example encapsulates the low-level interfacing with US machines and tracking devices to provide tracked 3D US functionality. The resulting volume data is handed over via OpenIGTLink to compatible clients, e.g. 3D Slicer [22] for visualization and intervention guidance. As SynchroGrab calls on VTK to provide a video data source, it can fairly easily be extended to support other machines than the Ultrasonix ones it currently interfaces with, even using frame grabbing in case no open low-level access is available. Although the developers correctly realized the problems monolithic vendor-proprietary solutions pose, SynchroGrab itself again provides one functionality, although admittedly with the ability to extend and build on the source code.

One such extension is SynchroGrab4D [15], which allows tracked 4D scanning of cyclically moving organs. Although providing a good example of the use of extensible open-source ultrasound tools, it also illustrates the "single-functionality" property of an intermediate tool between acquisition and visualization.

The IGSTK (Image-Guided Surgery Toolkit) [9] recognizes the overhead involved in re-inventing classes and functionality related image guided surgery and provides extensive support for aspects like hardware-independent tracking, volume representations and manipulations, file I/O (e.g. DICOM), networking (by means of OpenIGTLink), registration (on ITK), calibration, GUI building (on VTK and Qt), and many others. It does not, however, include support for real-time ultrasound imaging.

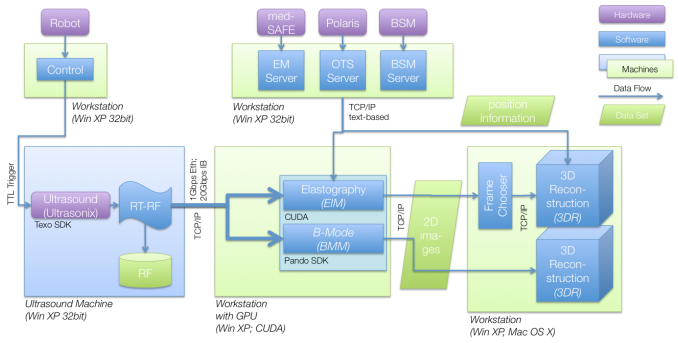


Fig. 1. Module connectivity graph e.g. for part of a navigated laparoscopic ultrasound system.

The ITK (Insight Toolkit) [10] is very widely used and concentrates on image analysis, filtering, registration, and segmentation tasks, but has only very limited focus on real-time aspects, and none on ultrasound interfacing at all.

One development that has a similar goal as IGSTK are the CISST libraries [4], incorporating support for robotics, algebra and operating system abstractions, vision, and intra-operative imaging. However, in terms of ultrasound support, they should be viewed more as a technology carrier for e.g. minimally-invasive surgery where US imaging is a useful modality for imaging, i.e. they rely on external support to receive US data.

Another interesting project is the Robot Operating System (ROS) [19], a real-time infrastructure toolkit providing architectural components for – in particular robotic – systems, the most important here being inter-process message passing in a publish/subscribe approach. Messages can be published in "topics" and are automatically distributed to subscribing clients, and parameter servers support e.g. central initialization handling.

Overall, it becomes clear that although many aspects of ultrasound-based image guidance and system design are dealt with by one or the other existing toolkit, none actually solves the problem of *providing a simple-to-use, hardware-independent, modular research environment that allows simple implementation of new, advanced ultrasound imaging algorithms and the application-agnostic, flexible setup of ultrasound research systems*.

### C. Overview

In Section II, the toolkit requirements and the single modules are presented. Section III shows particular architecture instances and experimental results together with an overview of the approach's limitations. Finally, Section IV concludes with a summary and an overview of future research directions.

## II. Approach

### A. Toolkit Design Requirements

There is no shortage of existing proprietary ultrasound system front-ends on the one hand – each machine should be shipped with one – and of research user interfaces and APIs on the other hand, where available from the system manufacturer. However, the wide area in between covers all sorts of clinical research settings, of medical imaging and biomedical engineering laboratories, and of signal processing groups, none of which is interested in inventing real-time imaging systems from scratch – so each project in each group sprouts off from another subproject, multiplying into unmanageable numbers of variations and branches.

Aiming at this problem and this audience, we propose a general modular ultrasound research toolkit (Fig. 1) with the following properties:

- *modular*: To allow for easy configurability, high reliability, and minimal system hardware footprints, modules following the "functional decomposition approach" provide only specialized functionalities, and thus are efficient and simple to develop (and debug). Also, they are to be provided both in source code and as separate executables.
- *networked*: Hardware constraints may render it necessary to deploy a research system on different machines/operating systems/locations, so inter-module communications have to be implemented using some form of network or inter-process communications stack.
- *extensible*: To be viable in research, the toolkit has to gracefully accommodate the integration of new algorithms and modules, which makes the next two properties important:
- *open interfaces*: Protocols, data types, and file formats throughout the system have to be well-defined, well-documented, and backwards-compatible (as much as possible) to allow interoperability between standard toolkit and third-party components.
- *open source*: For maximum impact, all network, system and hardware programming should rely on open-source toolkits, avoiding vendor SDKs where not absolutely necessary (such as for ultrasound hardware interfacing).
- *real-time*: In clinical application and for control purposes, high-bandwidth ultrasound data has to be processed with minimal latency.
- *parallel*: With current multi-core CPUs and widespread GPU adoption, the proposed high-bandwidth applications can be executed and balanced easily if implemented with parallel algorithms and independent execution threads.
- *portable and hardware-agnostic*: To ensure the toolkit's appeal to a wide audience, and to flexibly integrate into existing research software landscapes, all modules should be portable across the major operating systems where reasonably feasible (such as hardware-independent functions), and abstract away from the underlying hardware generating the data.
- *interaction with existing software*: As legacy and third-party applications will exist almost everywhere, the toolkit should adapt to them easily.

The proposed MUSiiC Toolkit entails two aspects – protocol/format definitions and module reference implementations. In the following, the separate executable modules provided by the toolkit so far are described roughly in processing order,

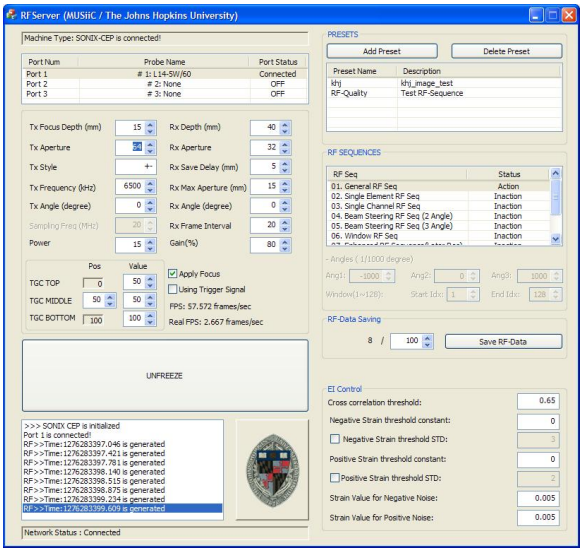


Fig. 2. Screenshot of the RF Server module (on Windows XP 32-bit) running on an Ultrasonix CEP machine.

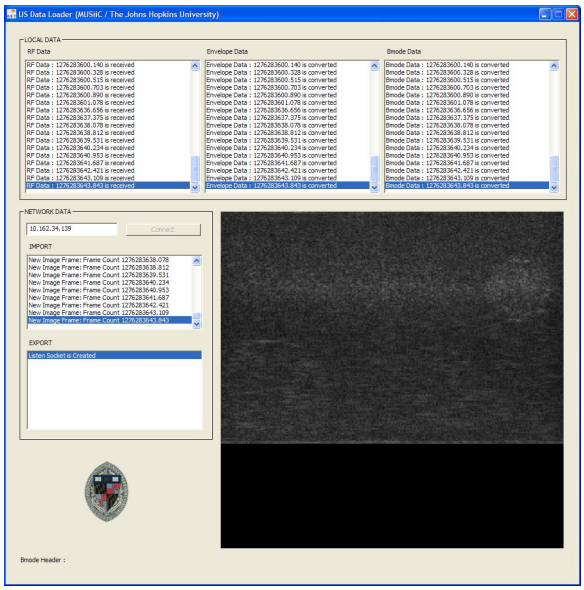


Fig. 3. Screenshot of the B-Mode Module (on Windows XP 64-bit) connected to an RF Server, displaying the real-time video stream.

with network and file protocols communicating between them.

### B. RF Data Acquisition

Obviously the first process step in any ultrasound-related activity is the physical acquisition of ultrasound data. Traditionally ultrasound systems performed the RF data acquisition and subsequent processing steps (e.g. B-mode computation) internally and provided a live view of the resulting video on screen. However, this poses a severe limitation on the further use of that data – not only is it necessary to "harvest" this data from the screen, often using some (potentially analog-to-digital) frame-grabbing mechanism, but also is this B-mode stream post-processed, i.e. log-compressed and perhaps noise filtered. Both problems make it essentially impossible to run advanced ultrasound imaging algorithms off that data.

Therefore, it is necessary to have low-level access to raw RF data as sampled by the analog-to-digital converters (*ADC*) of the ultrasound machine. The proposed toolkit includes an "RF Server" module (Fig. 2, showing a specific version for Ultrasonix machines) that interfaces with the hardware through the manufacturer's API, sets up parameters such as the probe firing sequence, depth, and apertures (among others), and initiates RF data sampling. The acquired frames can be either buffered and then written to disk, or be sent out to clients that perform further processing.

To allow synchronized data acquisition (e.g. for robot-based scans or photoacoustic imaging, cf. below) external RF-line- or frame-triggering is supported as well. Each RF frame receives an associated time stamp with $ms$ resolution which is preserved throughout all proposed modules. In a similar fashion, the frame is annotated in this step with a (currently binary) header containing both its pixel and physical dimensions as well as parameters like receive delay and beam steering angle necessary to reconstruct the frame later.

### C. B-Mode Computation

The next step in ultrasound imaging is conventionally the transformation of RF data into B-mode frames, showing the scanned object's acoustic impedance variations as a 2D grayscale image. The computation in the "B-Mode Server" module (Fig. 3) comprises envelope computation for the high-frequency (e.g. 20...40MHz) RF data through a Hilbert transform, log compression of sample data to adapt the large dynamic range of RF data for 8-bit screen display, and digital scan conversion to rectify and scale the envelope frame.

At this time, the image frame receives a text header, containing a "FRAME" tag, the original RF header time stamp, sizes etc.

### D. Elasticity Imaging

Up to this step it would be possible to use a closed ultrasound system and e.g. frame-grab a video stream of B-mode images. However, for advanced ultrasound imaging algorithms (such as synthetic aperture imaging, adaptive beamforming, photoacoustic imaging, thermal imaging, elasticity imaging etc.), direct access to the original RF data is essential.

One such modality is elasticity imaging (*EI*), where relative signal displacements between pairs of RF frames due to varying tissue compression are translated into stiffness estimates. Medically, these often correspond to histological findings such as lesions, tumors etc. While the actual compression can be effected in various ways – static through palpation motions, quasi-static by vibration, or dynamic through shear waves etc. – the underlying principle tracks signal patterns spatially with high resolution. Appropriate algorithms can be windowed pattern matching using normalized cross-correlation (*NCC*) [6], dynamic programming (*DP*) on sample intensities [16],

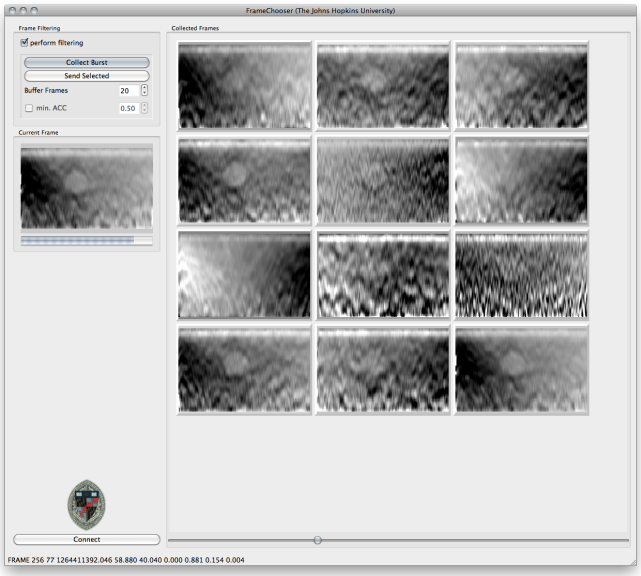


Fig. 4. Screenshot of the FrameChooser module (on Mac OS X) connected to an NCC-EIM, displaying the real-time video stream on the left, sampled and cached frames for manual selection on the right. A spherical lesion surrounded by homogeneous tissue within a synthetic phantom is shown.

analytic methods on DP results (*AM/DP*) to achieve sub-sample displacement estimation [17], or others.

All three have been implemented in various Elasticity Imaging Modules (*EIMs*) based on both serial algorithms and accelerated by using CUDA [5] for parallel GPGPU programming.

### *E. Quality-based Frame Selection*

For some video streams, it is possible to associate quality metrics with individual frames. For example with NCC-EIM output, each frame carries an average cross-correlation (*ACC*) value in its header, denoting the mean quality of pattern matches across the frame.

The "Frame Chooser" module (Fig. 4) allows the filtering of incoming video streams, either synchronously based on their associated quality value and a user-defined quality threshold before immediate re-sending, or asynchronously by sampling and buffering a preset number of frames, presenting them to the user for manual "point-and-click" selection, and then forwarding the selected frames on request.

### *F. 3D Ultrasound Reconstruction*

For intra-operative imaging, handheld ultrasound/US excels because of its simplicity, real-time nature, and no radiation. However, it is not easy to spatially interpret the images from 2D probes. In addition to the difficulty of highlighting relevant 3D features, handheld US imaging is an "immediate" modality with strong operator dependence and a lack of explicit volume reconstruction. This makes registration of new US data with pre-operative data sets difficult. Furthermore, it is desirable for many applications to allow visualization of the intervention volume along arbitrary planes, which is not generally possible with 2D ultrasound. In this place the 3D volume reconstruction (*3DR*) module comes in (Fig. 5, [24]), which connects to both a 6-DoF pose stream and an image frame stream, synchronizes them (by linear and *Slerp*-interpolation of poses to the nearest frame's timestamp), and performs frame insertion into the volume.

Once the current frame pose

$$Q := (P, R) = ((x, y, z), (\phi, \theta, \psi)) \in \mathbb{R}^3 \times SO(3) \qquad (1)$$

is known (e.g. from tracking, cf. below), a concurrently acquired 2D ultrasound frame (B-mode or EI) can be inserted into a 3D volume reconstruction. To do so, the 2D image pixels at position ${}^{img}p$ are projected into the 3D reconstruction space as voxels at ${}^{recon}p$ according to a pixel-nearest-neighbor (*PNN*) scheme [26]:

$$\begin{aligned} {}^{recon}p &= {}^{recon}M_{img} \cdot {}^{img}p & (2)\\ &= ({}^{recon}M_{track} \cdot {}^{track}M_{img}) \cdot {}^{img}p & (3) \end{aligned}$$

While ${}^{recon}M_{track} = Q_n$, i.e. is simply the current pose, the probe calibration matrix ${}^{track}M_{img} \in R^{4\times4}$ needs to be established first. Computed using well-known methods (cf. [12]), it can be read by the 3DR module from a file.

Entering new pixel values $v$ into the respective voxel at step $k+1$ is done by incremental averaging: $v_{k+1} = (v_k \cdot k + v)/(k+1)$, allowing constant-space iso-weighted averaging.

Because of the interactive nature of the handheld scanning process, the final extents of the reconstruction space cannot be determined in advance. Therefore, transgression of the current boundaries by the new slices' corners results in a (costly) dynamic expansion. However, its effects are mitigated by "preemptive" expansion, providing slightly more additional space than necessary just for the transgressing slice, based on the assumption that the scan motion can probably be extrapolated in the current direction. This "expansion factor" was set at 25% of the new overall size along that dimension, with satisfactory performance.

### *G. Tracking*

For any system requiring probe, tool, organ, or other positions in real-time, interfacing with tracking hardware is unavoidable. In spite of the multitude of different tracking approaches – optical/IR (e.g. NDI Polaris), electromagnetic/EM (e.g. Ascension medSAFE), visual camera-based (e.g. Claron microTracker), local-sensor-based as described in the following section, or even sensorless [18] – the basic result of tracking is always the same: a stream of up-to-6-DoF pose lines (position and quaternion orientation), one for each tracked object and each sampling instant:

```
<timestamp> <object-ID> A: <x> <y> <z>
<q0> ... <q3>
```

Thus, although each specialized tracking module is based on the particular device API, they abstract away this variability and are easily interfaced with multiple clients, e.g. 3D reconstruction or elasticity imaging [16].

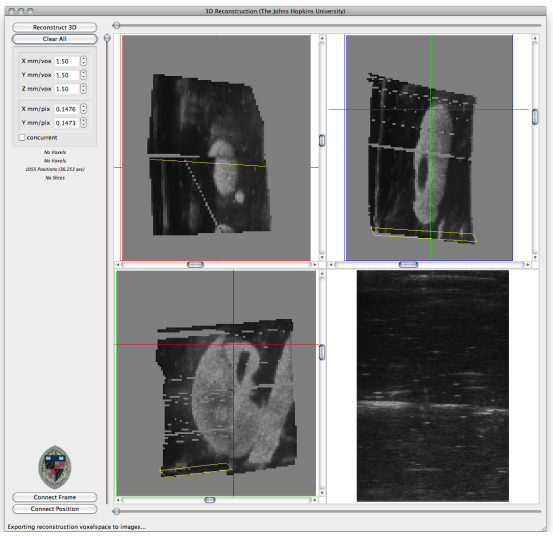

Fig. 5. Screenshot of the 3D Reconstruction module (on Mac OS X), showing a 3D B-mode reconstruction volume for a synthetic kidney phantom using electromagnetic tracking. The lower-right pane shows the current frame that is inserted; the yellow rectangle indicates the frame pose and size according to pose stream data and frame header information.

### H. Local-Sensor-Based Trajectory Reconstruction

One specialized tracking mechanism is the BeeSpaceMouse ( [23], [24], Fig. 6), using only local sensors (that are measuring local and/or relative data, unlike e.g. global optical or EM tracking). It is able to perform reconstruction of the probe trajectory $T_n = (Q_{i=1\ldots n})$ with up to 5 DoF relative to an arbitrary initial pose $Q_0$. Using optical mouse camera components and a three-axis accelerometer mounted on a probe bracket, it is able to track US probe trajectories along e.g. skin surfaces.

Motion $\Delta p$ (interpreted as homogenous translation vectors) and orientation $R$ (as homogenous rotation matrices) are accumulated into positions $P$ according to

$$P_n = P_0 + \sum_{i=1}^{n} R_i \Delta p_i \tag{4}$$

Using only the 2-DoF rotational information, the current absolute orientation $R_n$ is available directly, so the probe pose can be expressed as $Q_n = (P_n, R_n)$ and used as real-time pose input for the 3DR module.

### I. MITK-based Registration and Guidance

Currently, the 3D reconstruction module – apart from displaying orthogonal slices – is able to export its US volume either as 3D voxelspace data or as a batch of $n$ 2D slices. The 3D-B-mode or 3D-EI data can be imported e.g. into a 3D-US/CT registration [11] functionality of the MITK application. There it can serve to register pre-operative CT planning data for partial nephrectomy interventions [25] to the OR situation to enable target region overlays onto endoscopic video (Fig. 7).

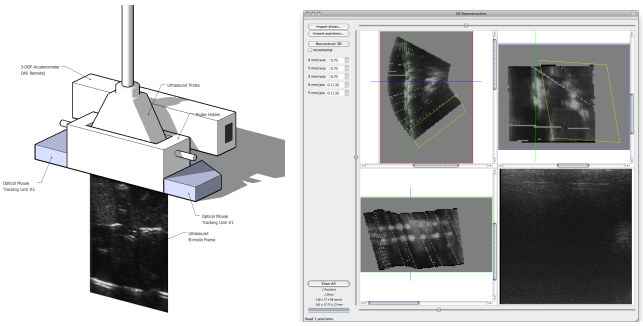

Fig. 6. Schematic view of the BeeSpaceMouse local sensor tracking bracket (left), 3D reconstruction result of a synthetic phantom (right).

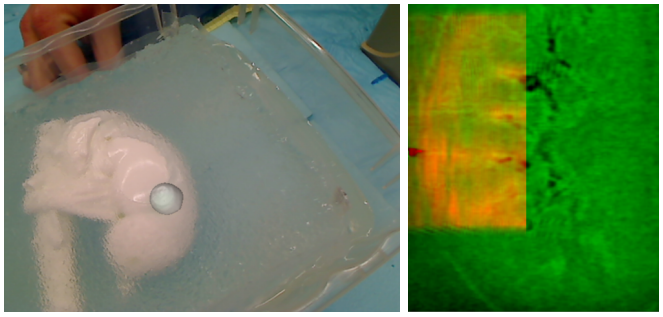

Fig. 7. Graphical model overlay of a user-segmented target region from CT data onto an endoscopic video stream of a phantom partial nephrectomy (left). The model can be registered using MITK with either 3D-B-mode or 3D-EI (right, overlaid in red onto green CT).

### J. Inter-Process Communications

As all modules are built, distributed, and deployed as separate executables – and therefore run as different processes without shared memory – inter-process communications need to be explicitly built in. Connections are established according to a (multi-)client–server paradigm over TCP/IP sockets. Data flow is purely unidirectional "push" to simplify networking protocol implementation. Congestion has to be handled by the client, silently dropping packets (frames or poses) whenever necessary.

This allows very flexible setups and the interception or splicing of connections with simple tools such as *netcat*, *tee*, and "$>, >>, |$" stream redirection operators.

### K. Various

One component not described in detail here (because it interfaces with, but is not part of the MUSiiC Toolkit) is e.g. a three-DoF robot (controller by Galil Motion Inc.; Fig. 8) for high-precision 3D-US volume acquisition. By following a predefined path, it is possible to perform repeatable palpation motions for EI with frame acquisition triggering into the RF Server module via a TTL line to the Ultrasonix machine [11].

Another hardware component is a laser for photoacoustic imaging, triggered by an external function generator, which itself again triggers the acquisition of a single RF line to assemble pseudo-pre-beamformed RF data over time.

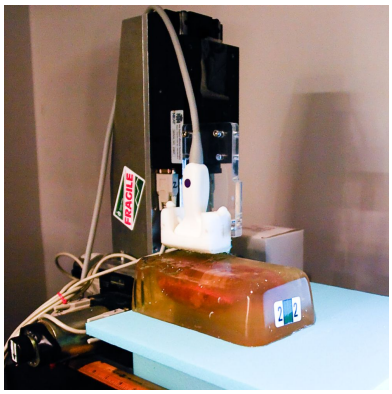


Fig. 8. Robot holding an ultrasound probe, triggering RF frame acquisition during palpation motions, resulting in 3D-EI volumes as in Fig. 7.

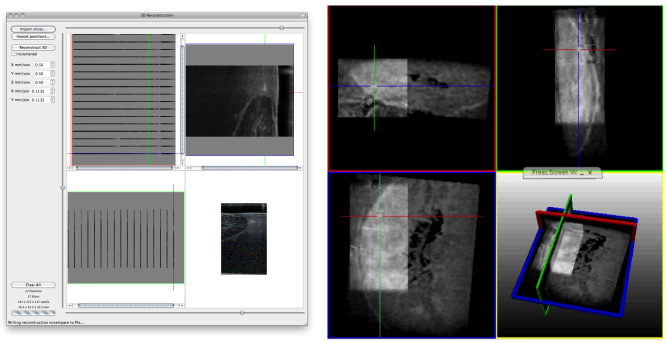

Fig. 9. Robotic 3D US scan of an *ex-vivo* kidney phantom (left); registration to CT using MITK (right; courtesy of M. Keil/Fraunhofer IGD).

Another legacy software component visualized a moving tracked 2D US frame in a 3D view, by interfacing directly with an Ascension EM tracker and frame-grabbing US or EI data from the screen of an Ultrasonix RP machine via a video connector. Using the well-defined interface protocols for poses and frames, adapting this visualization into the MUSiiC Toolkit to connect to digital US data and to communicate with other trackers was little work.

These examples serve to illustrate how simple interfacing external components with the MUSiiC Toolkit parts is for existing projects.

## III. Results

### A. Hardware Description

Ultrasound system integration mainly focused on two Sonix RP and Sonix CEP machines (Ultrasonix Medical Corp.), using handheld L12-5 and L14-5 linear array probes. Basic integration with a Siemens ACUSON Antares also exists.

Tracking system integration is mainly based on the Ascension medSAFE EM tracker (with mid-range and flatbed emitters), but also includes NDI Polaris optical tracking as well as the BeeSpaceMouse local sensor tracking.

The software modules have been tested and deployed across a variety of systems, among these Sonix CEP and RP machines (Windows XP 32-bit, Intel Core 2 Quad / 2.83 GHz, 2.75GB RAM), Pentium 4 / 3.0GHz, 3GB RAM)), workstations (Windows XP 64-bit, Intel i7 / 12GB RAM), Apple MacBook Pro (Mac OS X 10.6, Core 2 Duo / 2.66GHz, 6GB RAM) and iMac (Mac OS X 10.6, Core 2 Duo / 2.13GHz, 3GB RAM), and a Linux HPC cluster.

Several machines incorporated NVidia Tesla (D870 and C1060) and GeForce 9600, 9600M GT, or 9400M GPUs.

### B. Software Description

Apart from the caveat of hardware-dependent features in some software modules (e.g. ultrasound interfacing, external trigger ports, robot interfacing, availability of GPUs, tracking system drivers, ...), all software modules are aimed to be portable across all major operating systems (Mac OS X, Windows, and Linux), based on standard C++, the *boost* libraries, and the Qt 4 toolkit. GPU programming is based on CUDA.

The BeeSpaceMouse trajectory reconstruction is performed using an interactive module written in Java, tested on Mac OS X and Windows.

Using the ultrasound manufacturer Ultrasonix' API (*Texo* SDK), real-time RF data is acquired and transmitted via TCP/IP to the downstream modules, with transformation into B-Mode and elastography images (using NCC-EI and DP-EI) in real-time (at around 5...16fps, depending on algorithm and input size).

Finally, the 3DR module reconstructs volumes interactively using a pixel-nearest-neighbor approach (achieving around 30...40fps).

### C. Use Cases

The presented toolkit served as the demonstrator backbone of a joint Johns Hopkins/Fraunhofer-Gesellschaft project, where a navigation system for laparoscopic partial nephrectomies with elasticity-based registration and endoscopic video overlay (Fig. 7) was developed. To generate ground-truth EI data, the robotic setup of Fig. 8 collected US data which was then 3D-reconstructed in the 3DR module (Fig. 9) and registered to CT using an MITK functionality [11] based on the open-source MITK toolkit for interactive medical image processing [13].

Currently validation efforts of the BeeSpaceMouse integration are underway, with human volunteer studies in cooperation with the Johns Hopkins Hospital upcoming. Initial testing shows that handheld local-sensor-based tracking with 3D reconstruction in the proposed toolkit is feasible (Fig. 6, [24]).

It is also possible to set up remote US visualization or supervision systems; in one such setting US data was transmitted from the B-Mode Module running on the ultrasound machine to a remote in-house workstation via 1Gbps Ethernet, in another between Columbia University, NY and the Johns Hopkins labs.

In an ongoing cooperation with the NIH/NLM, the B-mode and EI data generated using the described toolkit is used as input to an adaptive volume rendering algorithm

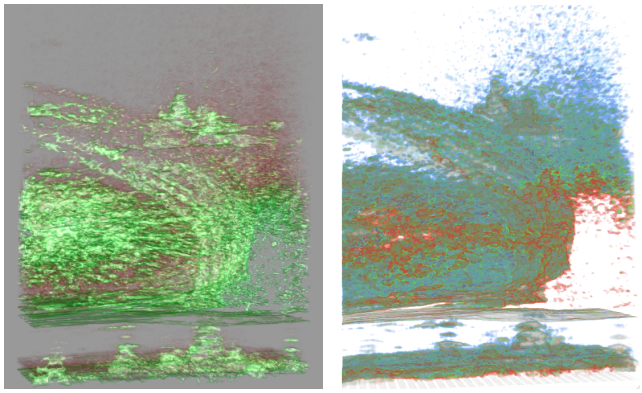

Fig. 10. Volume renderings of a robotic 3D US scan of an *ex-vivo* kidney phantom, focusing on internal structures or the renal boundaries; courtesy of J. Caban/NLM.

(Fig. 10 shows some of the preliminary results of the adaptive rendering technique).

### D. Limitations

The proposed approach requires the manual instantiation of the communications graph, i.e. the launching each module and connecting it with its respective data servers. Service disruptions (i.e. crashing modules and connection losses) can induce ripple effects, taking down larger sets of components or at least requiring manual intervention to reset the system to the desired connectivity state.

At irregular intervals (depending on the actual trajectory), 3D volume expansion is necessary whenever new slices fall outside of the current volume. To keep the voxelspace representation simple and access times approximately constant, this requires a re-copying of the complete 3D information into an expanded structure. While memory requirements are still reasonable (several hundreds of MB), actual memory allocation and access caused short data flow interruptions and loss of real-time behavior. This effect was minimized through both "preemptive expansion" (cf. above) and an application-dependent limit on the overall size of the reconstruction volume (on the order of $(200mm)^3$ at an isometric resolution of $0.5mm$).

## IV. Conclusion

### A. Discussion

We have presented the modular MUSiiC Toolkit for real-time ultrasound research, comprised of a variety of specialized executable modules, communicating image and tracking data via network sockets between each other. With well-defined network protocols, the fast and simple system setup allows quick reconfiguration according to application needs. Modules interfacing with hardware (e.g. ultrasound or tracking systems) allow the abstraction of implementation details on the executables level, where new hardware can be easily adapted by using the established network protocols.

Because of the multi-processing approach, load balancing – either on multi-core CPUs or as a distributed system – comes naturally and for free with the fine-grained functional modularization in this toolkit.

With the easy implementation of algorithmic modules using the provided real-time data streams, and with the concomitant research potential of this toolkit, we experienced (and further expect) very good adoption of toolkit modules and development "best practices" throughout the user group, already extending well beyond the authors' laboratory into academia, research agencies, and industry.

Several algorithms are already present in the toolkit's "reference implementation", such as various elasticity imaging algorithms, volume reconstruction, local-sensor-based trajectory reconstruction, and hardware interfaces. Unlike a fully-formed Venus springing into existence, this toolkit is under intense development – new modules will continually be added, and external algorithmic additions are very welcome. For further information – and protocol and format definitions in particular – refer to *https://musiic.lcsr.jhu.edu/Research*.

### B. Outlook

The roadmap for future development of the MUSiiC Toolkit contains both new features and new components.

First, it is important to note that the toolkit operates according to the *source-filter-sink* model, where the filter modules perform a single specialized function only. Currently, many of those modules also include graphical user interfaces (GUIs), diluting the approach and complicating automation and user interaction, making GUI centralization a near-term goal. At the same time, a standard XML file format for module initialization files will become desirable.

Integration with other hardware includes e.g. the Siemens Antares ultrasound machines, 3D wobbler probes (for acquisition and scan conversion), etc. We are also developing a module for integration of the Ultrasonix pre-beamformed data acquisition system (*DAQ*) [28].

Obvious extensions to the toolkit by the community include sensorless 3D-US tracking [18] [8] or tracking-based quality metrics for EI frames.

This raises the issue of data caching, e.g. for modules that require caching of past frames such as elasticity imaging. A data distribution system (*publish/subscribe DDS*) architecture would enable central buffering with little local overhead. At the same time, the existing OpenIGTLink protocol is an obvious choice to make the MUSiiC network protocol standards-compliant. Thus, the extension of the backwards-compatible OpenIGTLink with multi-client servers, automatic reconnects, new data types (e.g. RF data and freeform XML) (some of these aspects have already been described by [3]), and a DDS backbone reference implementation is a tantalizing mid-term aim. In fact, current work includes OpenIGTLink network-level compatibility of the toolkit modules and standardization of file formats, i.e. with persistent-OpenIGTLink-derived semantics. This will also make the MUSiiC Toolkit interoperable with existing software such as 3D Slicer [22]. Of course, DICOM compatibility is another desired feature.

We are also looking forward to integration of the MUSiiC Toolkit with the open-source CISST libraries [4] and in particular the SAW Surgical Assistant Workstation [20] and the daVinci surgical robot.

## ACKNOWLEDGMENT

The authors would like to thank Drs. Russell H. Taylor of the NSF ERC-CISST at the Johns Hopkins University, Alfred Yu of the Medical Engineering Program/Dept. of Electrical and Electronic Engineering at the Hong Kong University, and Jesus Caban at the NIH/National Library of Medicine for their valuable suggestions and support. This work was partially funded by the National Science Foundation under cooperative agreement NSF ERC-CISST EEC 9731748. Further support was provided by internal funds of the Johns Hopkins University. Equipment support was generously provided by Ultrasonix (Sonix RP).

## REFERENCES

[1] J. Boisvert, D. Gobbi, S. Vikal, R. Rohling, G. Fichtinger, P. Abolmaesumi: "An Open-Source Solution for Interactive Acquisition, Processing and Transfer of Interventional Ultrasound Images". Insight Journal, 2008

[2] S.S. Brunke, M.F. Insana, J.J. Dahl, C. Hansen, M. Ashfaq, H. Ermert: "An Ultrasound Research Interface for a Clinical System". IEEE Transactions on Ultrasonics, Ferroelectrics, and Frequency Control, vol. 54, no. 1, January 2007

[3] Chinzei K., Tokuda J.: "Extension to OpenIGTLink; Smart Socket Connection, XML as Message, Logging, and One-to-multi Relaying". MIDAS Journal - Systems and Architectures for Computer Assisted Interventions, 2009

[4] https://trac.lcsr.jhu.edu/cisst

[5] http://developer.nvidia.com/object/gpucomputing.html

[6] N. Deshmukh, H. Rivaz, E. Boctor: *GPU-Based Elasticity Imaging Algorithms*. MICCAI-Grid Workshop, London/UK, 2009

[7] C.M. Fabian, K.N. Ballu, J.A. Hossack, T.N. Blalock, W.F. Walker: "Development of a parallel acquisition system for ultrasound research". SPIE, Vol. 4325, 54 (2001)

[8] R.J. Housden, G.M. Treece, A.H. Gee, R.W. Prager: "Calibration of an orientation sensor for freehand 3D ultrasound and its use in a hybrid acquisition system". BioMedical Engineering OnLine 2008, 7:5

[9] K. Cleary, P. Cheng, A. Enquobahrie, Z. Yaniv (eds.): "IGSTK: The Book". http://www.igstk.org

[10] http://www.itk.org/

[11] M. Keil, Ph. J. Stolka, M. Wiebel, G. Sakas, E. McVeigh, R. H. Taylor, E. Boctor: *Ultrasound and CT Registration Quality: Elastography vs. Conventional B-Mode*. IEEE International Symposium on Biomedical Imaging ISBI 2009, Boston/MA

[12] L. Mercier, T. Lang, F. Lindseth, L.D. Collins: "A Review Of Calibration Techniques For Freehand 3-D Ultrasound Systems". Ultrasound In Medicine and Biology, Vol. 31, No. 2, Pp. 143165, 2005

[13] http://www.mitk.org/wiki

[14] L. Mo, D. DeBusschere, G. McLaughlin, et al.: "Compact ultrasound scanner with simultaneous parallel channel data acquisition capabilities". IEEE International Ultrasonics Symposium, pp. 1342-1345, 2008.

[15] D.F. Pace, D.G. Gobbi et al.: "An open-source real-time ultrasound reconstruction system for four-dimensional imaging of moving organs". Insight Journal, 2009

[16] H. Rivaz, E. Boctor, P. Foroughi, R. Zellars, G. Fichtinger, G. Hager: "Ultrasound elastography, a dynamic programmimg approach". IEEE Transactions on Medical Imaging, 27(10):1373-1377, 2008.

[17] H. Rivaz, P. Foroughi, I. Fleming, R. Zellars, E. Boctor, G.D. Hager: "Tracked regularized ultrasound elastography for targeting breast radiotherapy". Medical Image Computing and Computer Assisted Intervention (MICCAI), pages 507-515, September 2009

[18] H. Rivaz; H-J. Kang; Ph. J. Stolka; E.M. Boctor: "Novel reconstruction and feature exploitation techniques for sensorless freehand 3D ultrasound". SPIE Medical Imaging 2010 (San Diego, CA/USA)

[19] http://www.ros.org/wiki/

[20] https://www.cisst.org/saw/Main_Page

[21] V. Shamdasani, U. Bae, S. Sikdar, et al.: "Research interface on a programmable ultrasound scanner". Ultrasonics, vol. 48, pp. 159-168, 2008.

[22] http://www.slicer.org/

[23] Ph. J. Stolka, J. Choi, J. Wang, M. Choti, E. Boctor: *5-DoF Trajectory Reconstruction for Handheld Ultrasound with Local Sensors*. IEEE-International Ultrasonics Symposium IUS 2009, Rome/Italy

[24] Ph. J. Stolka, H. J. Kang, M. Choti, E. Boctor: *Multi-DoF Probe Trajectory Reconstruction with Local Sensors for 2D-to-3D Ultrasound*. IEEE International Symposium on Biomedical Imaging ISBI 2010, Rotterdam/Netherlands

[25] Ph. J. Stolka, E. Boctor, M. Keil, E. McVeigh, R. H. Taylor: *A 3D-Elastography-Guided System for Laparoscopic Partial Nephrectomies*. SPIE-Medical Imaging 2010, San Diego/USA

[26] O. V. Solberg, F. Lindseth, H. Torp, R. E. Blake, T. A. Nagelhus Hernes: *Freehand 3D Ultrasound Reconstruction Algorithms - A Review*. In: Ultrasound in Med. & Biol., Vol. 33, No. 7, 2007

[27] J. Tokuda, G.S. Fischer et al.: "OpenIGTLink: an open network protocol for image-guided therapy environment". Int J Med Robotics Comput Assist Surg 2009; 5: 423434.

[28] Tsang IKH, Yiu BYS, Cheung DKH, Chiu HCT, Cheung CCP, Yu ACH: *Design of a multi-channel pre-beamform data acquisition system for an ultrasound research scanner*. IEEE International Ultrasonics Symposium, 2009; 1840-1843.

[29] T. Wilson, J. Zagzebski, T. Varghese, Q. Chen, M. Rao: "The Ultrasonix 500RP: A Commercial Ultrasound Research Interface". IEEE Transactions on Ultrasonics, Ferroelectrics, and Frequency Control, vol. 53, no. 10, October 2006