CISUS: An Integrated 3D Ultrasound System for IGT Using a Modular Tracking API

Emad M. Boctor[*] [a], Anand Viswanathan [a], Steve Pieper [b], Michael A. Choti [c], Russell H. Taylor [a], Ron Kikinis [b], and Gabor Fichtinger [a]

[a] Computer-Integrated Surgical Systems and Technologies (CISST), Johns Hopkins University 3400 North Charles Street, New Engineering Building 315 Baltimore, MD 21218-2681;

[b] Surgical Planning Laboratory AMB II, L1-Room 0069, Radiology Brigham & Women's Hospital 75 Francis St. Boston, MA 02115;

[c] Johns Hopkins School of Medicine, Baltimore, MD 21287-4606.

ABSTRACT

Ultrasound has become popular in clinical/surgical applications, both as the primary image guidance modality and also in conjunction with other modalities like CT or MRI. Three dimensional ultrasound (3DUS) systems have also demonstrated usefulness in image-guided therapy (IGT). At the same time, however, current lack of open-source and open-architecture multi-modal medical visualization systems prevents 3DUS from fulfilling its potential. Several stand-alone 3DUS systems, like Stradx or In-Vivo exist today. Although these systems have been found to be useful in real clinical setting, it is difficult to augment their functionality and integrate them in versatile IGT systems. To address these limitations, a robotic/freehand 3DUS open environment (CISUS) is being integrated into the 3D Slicer, an open-source research tool developed for medical image analysis and surgical planning. In addition, the system capitalizes on generic application programming interfaces (APIs) for tracking devices and robotic control. The resulting platform-independent open-source system may serve as a valuable tool to the image guided surgery community. Other researchers could straightforwardly integrate the generic CISUS system along with other functionalities (i.e. dual view visualization, registration, real-time tracking, segmentation, etc…) to rapidly create their medical/surgical applications. Our current driving clinical application is robotically assisted and freehand 3DUS-guided liver ablation, which is fully being integrated under the CISUS-3D Slicer. Initial functionality and pre-clinical feasibility are demonstrated on phantom and ex-vivo animal models.

Keywords: 3DUS, Image-Guided Therapy IGT, Multi-Modality Workstation, Visualization, Planning, Image-Guided Surgery, Robotic Ultrasound, Monitoring, Registration and Segmentation

1. INTRODUCTION

Image-guided therapy (IGT), the use of medical imaging for guidance of therapy, is not a new concept. Since the discovery of x-rays, various imaging methods (CT, MRI and US) have been used to localize normal anatomical structures and pathologic lesions, as well as to locate surgical instruments. The core components of IGT usually involve image analysis, segmentation, registration, and visualization. IGT uses cutting edge *engineering* and *medical imaging* research to change the current practice of surgery. The implementation of IGT will allow the usage of medical image analysis to act as a surgical navigation tool to guide the surgeon through the patient's anatomy via visualization of the anatomical changes during surgery. IGT has already led to improvements in surgical procedures in terms accuracy, speed, as well as shorter hospitalization and an improved overall outcomes. These trends are expected to continue on an accelerated path in the foreseeable future.

Besides a medical imaging device, other hardware such as tracking devices and medical robots may also be useful for IGT. A tracking device is an apparatus that tells the whereabouts of objects in 3D space and returns "real-time"

[*] eboctor@ieee.org; phone 1 410 516-4779; www.cisst.org

information concerning the position and orientation of those objects relative to a given frame of reference. These devices can also play a substantial role in building 3DUS data from series of tracked 2DUS images. A robot or mechanical arm allows for steady positioning and precise manipulation of surgical instruments in and around the target anatomy and it can also be used to manipulate US scanners.

From the information provided by preoperative images, surgeons can choose the best course of action to treat the patient. Medical images can not only be used for the purposes of diagnosis and surgical planning before surgery, but also for visualization during surgery of the anatomical Region Of Interest (ROI), which may be fully embedded in the organ and cannot be directly visualized. During both acquisition of medical images and surgery, the target and surrounding anatomy are susceptible to motion and deformation, typically caused by respiration and mechanical contact with the surgical instruments. Therefore, it is a logical imperative to track the 3D location of the anatomy of interest in interventions concerning sensitive or vital structures. Through IGT systems one should be able to detect organ motion as well as organ deformation by using real-time imaging (in our case US) to track and monitor the dislocation and deformation of anatomy.

Another aspect of IGT is the synthesis of different imaging modalities, a.k.a. multimodality registration. Typically, by acquiring preoperative 3D volumes in a high resolution imaging modality such as CT or MRI, we can register another imaging modality that is real-time such as fluoroscopy and US, to provide accurate and updated visualization information to the surgeon during the procedure. This would provide a minimally invasive and precise approach to treatment of a wide array of organ systems and diseases. As minimally invasive surgical applications gain prominence, the role of IGT will become increasingly more significant for capturing intraoperative surgical changes and providing integrated visualization.

Ultrasound imaging has emerged as a widely popular guidance modality for medical interventions, since it is portable, interactive, real-time, safe, cost-effective and convenient to use in outpatient clinics and sometimes even in office setup, in contrast to other modalities such CT and MRI. Consequently, US machines have become predominant in the operating room and have started to replace the X-ray's niche in intraoperative guidance especially, when soft tissues are concerned. Significant research has been dedicated to using US, but significant technical improvements are needed before the full potential of US can be realized, particularly in applications involving IGT. The main problems are to assemble individual 2D US images into 3D volumes [11] and then to relate the position of surgical tools with respect to the reconstructed 3DUS volume. These tasks critically require a reliable calibration framework as well.

In the current field of US IGT applications, there are only very few commercial/research US systems, such as Stradx [1] and In-Vivo [2], but these systems are proprietary and closed. Augmenting the functionality of these closed systems and the integration of them into an IGT environment are formidable technological and logistical/legal challenges. There have been successful attempts at integrating US in IGT applications as can be seen in MISON [3] is an intraoperative imaging system, which integrates 3D ultrasound and navigation and prostate brachytherapy systems also routinely use US [4,6] .

Although the systems above illustrate the feasibility and importance of US in an IGT setting, a standard and open environment for the development of US-guided therapy is still amiss. Our work presented here was intended to fill in this gap, by providing the US IGT research community with an open source platform to develop their specialized US IGT therapy systems using real-time 2D and/or 3D US imaging. To address these limitations, a robotic/freehand 3DUS open environment [7] (Computer-Integrated Surgery guided by US imaging, a.k.a. CISUS) is being integrated into the *3D Slicer* (Surgical Planning Lab, Brigham Woman Hospital and Harvard medical school) [5], an open source research tool intended for diagnostic visualization and surgical planning.

2. ARCHITECTURE OF THE CISUS SYSTEM

In this chapter, we present the following sections: (1) an overview of the CISUS system, (2) functional requirements, and (3) system design. The latter includes description of the development environment, CISUS architecture, implementation details, system extensibility, and the rapid prototyping interface.

2.1 System overview

The home environment for the CISUS software package is the *3D Slicer* [5,12], which is a development environment designed for open source analysis and visualization of medical data. The *3D Slicer* utilizes an underlying open source software package *Visualization Toolkit* (*VTK*), to handle the rendering and visualization pipeline while using *Tcl/Tk* for the user interface. Essentially *3D Slicer* is a set of *VTK* based *C++* classes and a structured user interface in *Tcl/Tk* designed to be used in IGT applications. These libraries are both compiler and operating system independent. *3D Slicer*'s main benefit is the inclusion of a wide array of useful features, including:

- A multiplane reformatting plane algorithm to display an arbitrarily oriented slice of a 3D volume
- The ability to display multiple volumes on the same slice.
- Display windows that show the axial, sagittal, and coronal 2D slices as well as a 3D graphical view.
- The ability to create, edit, and render surface models
- A trajectory assistance module that can display virtual objects in the 3D view and assist in planning the trajectory path of the object

Another significant element of *3D Slicer*'s core functionality is the multi-modal registration to semi-manually align two or more 3D volumes and a volume editor that contains a variety of segmentation tools. *3D Slicer* provides functionality to create rendered surface models from the 3D volume and use a decimation algorithm to reduce the number of triangles needed to represent those models. With the use of the above listed features, it is relatively straightforward to extend *3D Slicer* into IGT applications for both pre-operative and intra-operative surgical planning, which makes *3D Slicer* an ideal host for the proposed open 2D/3D US system, which we call CISUS.

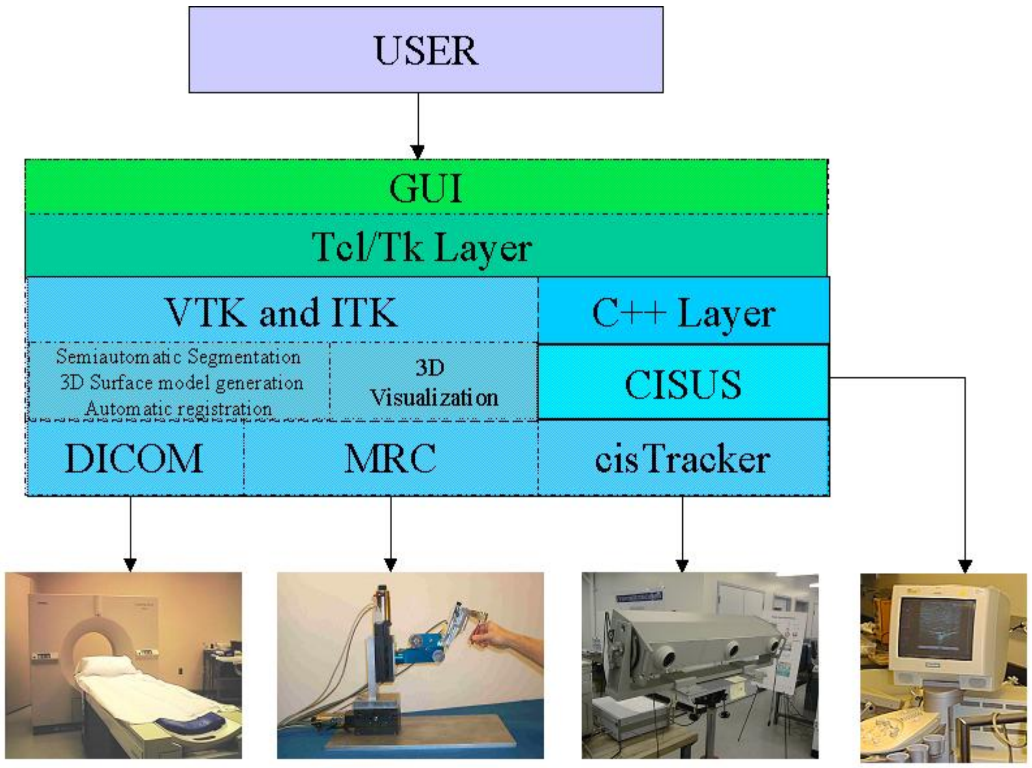
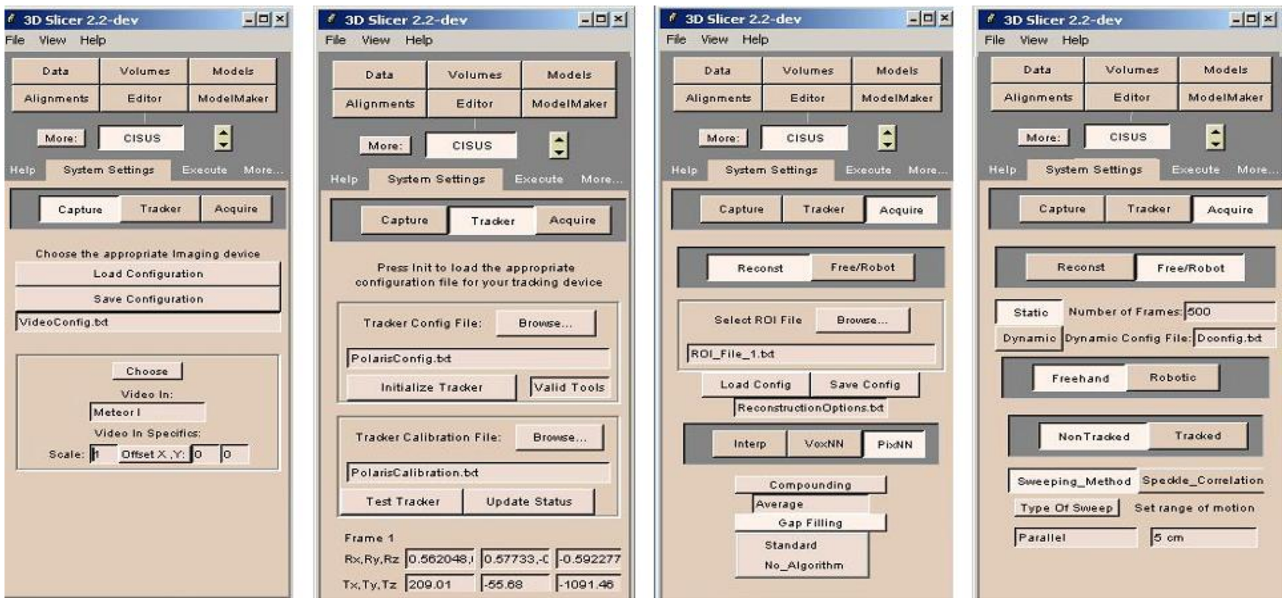


Fig. 1: The synergic integration of CISUS and 3D Slicer

The synergic integration of CISUS and *3D Slicer* is shown in Fig. 1. *3D Slicer* can be extended to include external libraries, such as the *cisTracker* and *mrc* modules for tracking devices and robotic control, respectively. CISUS can take advantage of all features of *3D Slicer* and use the DICOM format to read imaging modalities. CISUS was designed to integrate into the *3D Slicer* environment along with libraries for controlling medical robots and tracking devices (*mrc* and *cisTracker* respectively).


3D Slicer 2.2-dev
File View Help
Data Volumes Models
Alignments Editor ModelMaker
More: CISUS
Help System Settings Execute More...
Capture Tracker Acquire
Choose the appropriate Imaging device
Load Configuration
Save Configuration
Choose
Video In:
Video In Specifics:
Press Init to load the appropriate configuration file for your tracking device
Tracker Config File: Browse...
Initialize Tracker
Valid Tools
Tracker Calibration File: Browse...
Test Tracker
Update Status
Reconst Free/Robot
Select ROI File Browse...
Load Config Save Config
Interp VoxNN PixNN
Compounding
Gap Filling
Static
Dynamic
Freehand Robotic
NonTracked Tracked
Sweeping_Method Speckle_Correlation
Type Of Sweep Set range of motion

VideoConfig.txt
Video In: Meteor I
Scale: 1 Offset X ,Y: 0 0

PolarisConfig.txt
PolarisCalibration.txt
Frame 1
Rx,Ry,Rz 0.562048,(0.57733,-(-0.592277
Tx,Ty,Tz 209.01 -55.68 -1091.46

ROI_File_1.txt
ReconstructionOptions.txt
Average
Standard
No_Algorithm

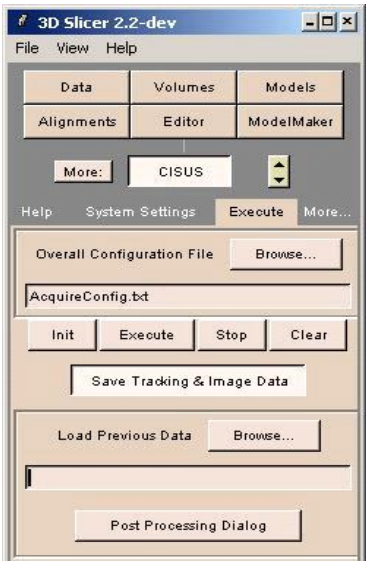
Number of Frames: 500
Dynamic Config File: Dconfig.txt
Parallel
5 cm

Fig. 2: Graphical user interfaces (GUIs) for the system settings.
From left to right: Capture Device, Tracker, Reconstruction, and Robotic/Freehand US

2.2 Functions implemented in CISUS

The functions of CISUS offer sufficient flexibility to instantiate an array of specific clinical/surgical applications. The functions provide flexible configuration of the specific application and means to controls the execution of interventions. There are separate graphical user interfaces (GUIs) to configure the image capture device, tracking device, image reconstruction parameters, and robotic/freehand US capture settings, as shown in Fig. 2.

- The "Capture GUI" allows for setting the parameters for the image acquisition device as configured by *vtkVideoSource*. These parameters can be saved or loaded from an input configuration file.
- The "Tracker GUI" configures the parameters for the tracking device as configured by *cisTracker*. There is a button selection to test if the tracking device is properly configured. There is also a selection to load the US probe calibration matrix for a specific calibrated probe.
- The "Reconstruction GUI" sets the options for building the volume and post-processing options to be applied on the volume. The user can choose from Voxel Nearest Neighbor (VNN), Pixel Nearest Neighbor (PNN), and Distance Weighted Interpolation methods.[13] The user can load a configuration file describing the ROI and another configuration file specifying the filling algorithm and its tuning parameters.
- The "Freehand/ Robotic GUI" allows the user to select the type of system based on the current hardware available. The user can then choose options such as if speckle correlation is used (in the case of an untracked freehand system) as well as the type of sweeping motion to be used. In the presence of a medical robot, the user can choose to configure the robot using a configuration file and specify force compliance mode. The user will have a button selection to test the configuration of the robot.


3D Slicer 2.2-dev
File View Help
Data Volumes Models
Alignments Editor ModelMaker
More: CISUS
Help System Settings Execute More...
Overall Configuration File Browse...
Init Execute Stop Clear
Save Tracking & Image Data
Load Previous Data Browse...
Post Processing Dialog

AcquireConfig.txt

Fig. 3: GUI for the execution phase

For the Execution phase, the user has a GUI to initialize the system and start data acquisition and 3DUS reconstruction (Fig. 3.) The user can decide whether to save the 3DUS volume and/or the 2D US images with the corresponding 3D position provided by the tracker. The user also has the option to load and save the above configuration parameters in files, for a rapid convenient configuration of the system.

2.3 System design

2.3.1 Development environment

The developing setup of the CISUS module requires the following software: *CMake* 1.8.3 [www.cmake.org], *Tcl/Tk* 8.3/8.4 [www.activestate.com], *VTK* 4.2 [www.vtk.org], *3D Slicer* 2.x [5]and the *cis-2* and *mrc-2* libraries [7]. *CMake* is a cross-platform makefile generator. *3D Slicer* uses *Tcl/Tk* for the GUI layer. The *SWIG* 1.3.19 [www.swig.org] software is also needed for an optional Tcl wrapped version of the *cis-2* library.

A number of steps are required for using the CISUS libraries. First, using *CMake*, we build the makefiles for *VTK* with the WRAP_TCL, USE_ANSI_STDLIB, and BUILD_SHARED_LIBS options all set to *ON*, the latter to generate dynamic libraries as the output of the makefile. After compiling the generated makefiles, we use *CMake* again to generate the makefiles for the *3D Slicer* code in the base directory. (It is advisable to check if *vtkSlicerBase.dll* and *vtkSlicerBaseTCL.dll* were indeed generated.) Next, we use *CMake* on the {SLICER_HOME}/Modules/vtkCISUS directory to generate the makefiles that are responsible for compiling and linking against precompiled *cis-2*, *mrc-2*, and *VTK* dynamic libraries. Finally, we build the CISUS module using the generated makefiles and set the proper path settings, and lastly add "vtkCISUS" to the SLICER_MODULES_TO_REQUIRE environment variable.

2.3.2 Architecture

Ultrasound-guided therapy may be used in a wide variety of clinical configurations, which CISUS addresses in two major steps: in a configuration and an execution phase. In the execution phase, the system performs the surgical intervention based on the settings in the configuration phase. As shown in Fig. 2, the "System Settings" tab has been divided into three categories: (1) "Capture Device" to configure a video input source for US equipment or to use *3D Slicer*'s DICOM reader. (2) "Tracking Device", if one is present, to configure a tracking device using the modular *cisTracker* library [7] that currently supports Polaris, Optotrak, Aurora, Flock Of Birds, and Flashpoint. (3) "Acquisition" to specify a multitude of operational parameters, such as scanning configuration, freehand vs. robotic; acquisition mode, static vs. dynamic. There is also a tab for serialization of the acquired data (images, tracking information, and force-sensor readings, and possibly more information). Robotic control is provided by the separate module named Modular Robot Control a.k.a. *mrc*.[7]

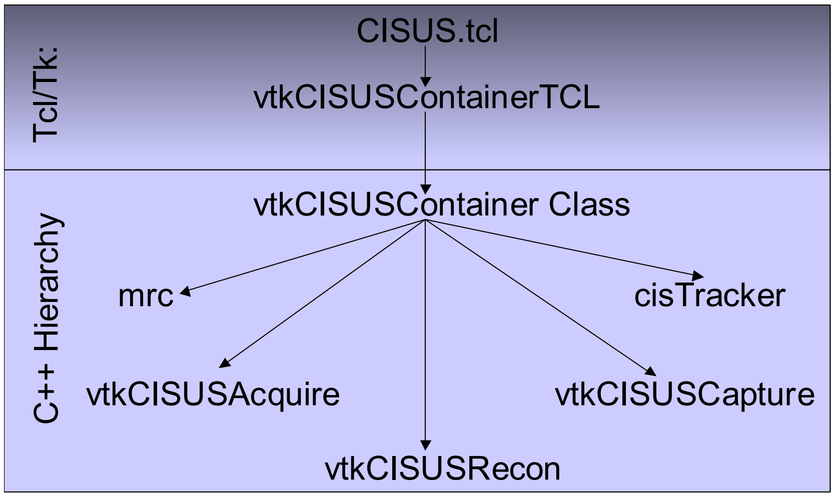


Fig. 4: An overview of the CISUS module architecture illustrating the class hierarchy as well as the relationship between the C++ and the Tcl layers.

CISUS consists of *VTK*-based C++ classes and a *Tcl/Tk* script for the GUI. The overview of the CISUS module architecture illustrating the class hierarchy is shown in Fig. 4. There is a class named *vtkCISUSContainer* that interfaces with the GUI and contains configuration variables. This class was designed to be the sole interface to the Tcl layer, by limiting the generation of Tcl wrappers to just the *vtkCISUSContainer* class. This scenario ensures that all GUI interactions take place through *vtkCISUSContainer* and all other classes rely on it to handle their own GUI traffic. After generating a *vtkCISUSContainerTCL* library, one can directly load the library from Tcl. This *vtkCISUSContainer* class also serves as the entry point for all other related classes. The *vtkCISUSContainer* class is also responsible for the execution phase of the system and to serialize (and de-serialize) the entire system setup as well as the acquired 2D US image data along with the computed 3D volume. There are three classes, each responsible for the configuration of the tracking device used, for the Acquisition phase, and for the Reconstruction phase, respectively. These classes are serialized such that data (both positional and ultrasound images) and configuration options can be saved into a file and loaded from a file.

The Tcl/Tk script loads the *vtkCISUSContainerTCL* library and displays a GUI within the *3D Slicer* for the user to interface with. Working in this window management environment, the user can configure and operate the system and also seamlessly switch to other *3D Slicer* modules and back without losing any of CISUS data.

2.3.3 Implementation

The *vtkCISUSCapture* class handles the video input device and allows for settings of the scale and offset of the US image. *VTK* is a powerful open source visualization package that handles the rendering pipeline of visual programming. Since our host application was *3D Slicer*, which relies on *VTK*, we chose to use *VTK's* VideoSource libraries to handle the video input from the US machine. This class uses a pointer to a *vtkVideoSource* object. By using *vtkVideoSource*, one can immediately use one of the preexisting derived classes for Matrox frame grabbers and generic windows acquisition cards. If support of another card is needed, one can simply add the appropriate derived class for *vtkVideoSource*. Since the base class of *vtkVideoSource* defines standard generic interface for video inputs, there are only superficial modifications to make a new class of *vtkVideoSource* be used in the CISUS module.

The *vtkCISUSAcquire* class handles the acquisition and configuration of the US image and tracking data. This class is responsible for the synchronization between the hardware components and for determining the type of operational mode the system is using. This class contains the US images along with related data (tracking data, encoder values, and force sensor readings).

The *vtkCISUSRecon* class handles the ROI parameter as well other options to configure the volumetric US reconstruction algorithm. This class uses the input of the configuration file or the input from the user interface to determine which types of algorithms are available.

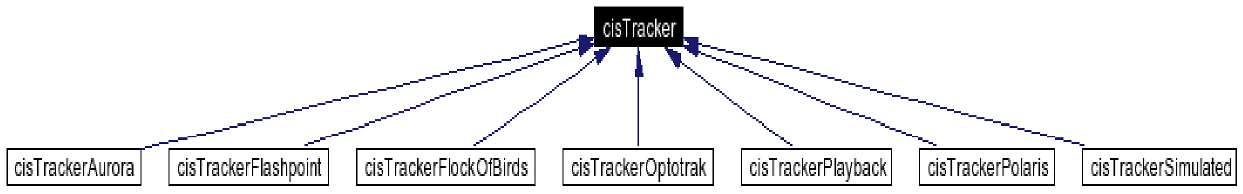


Fig. 5: The inheritance graph of the *cisTracker* library

The CISUS module uses the Engineering Research Center (ERC) 's libraries [7] to handle tracking devices and medical robotic hardware. For the tracking devices, *cisTracker* is used. This generic tracker class defines a basic outline of what actions tracking devices should be able to perform. The class is independent of the particular tracking device and operating system used. Classes derived from this general class must be tailored to the hardware specifications of a given tracker. This library is to be used for standalone systems, where the application using the tracker library is run on the computer to which the tracking device is physically attached. The *cisTracker* class also contains a simulated tracking device class to allow the usage of a virtual tracking device to aid developers to test their code without actually having a physical tracking device connected to the system. Fig. 5 illustrates the inheritance tree of the base class *cisTracker* and the specialized derived classes.

For Robot control, the CISUS module uses the modular robot control library, or *mrc*. The purpose of *mrc* is to provide a standard interface for distributed modular medical robots for computer integrated surgery (CIS) robot applications while providing a standard usage of C/C++ libraries provided by robot hardware vendors. The *mrc* library allows for seamless use of multiple robot platforms in CIS applications, without hard coding any of the robot's particulars in the application's source code. As shown in Fig. 6, the *mrc* library was designed specifically to control a variety of robots including the LARS [17], the Brigham open-MR robot [18], the Steady Hand [19], and the Neuromate [20], among several others.[21] For force-compliant control of surgical assistants, the *mrc* also supports various force sensors.

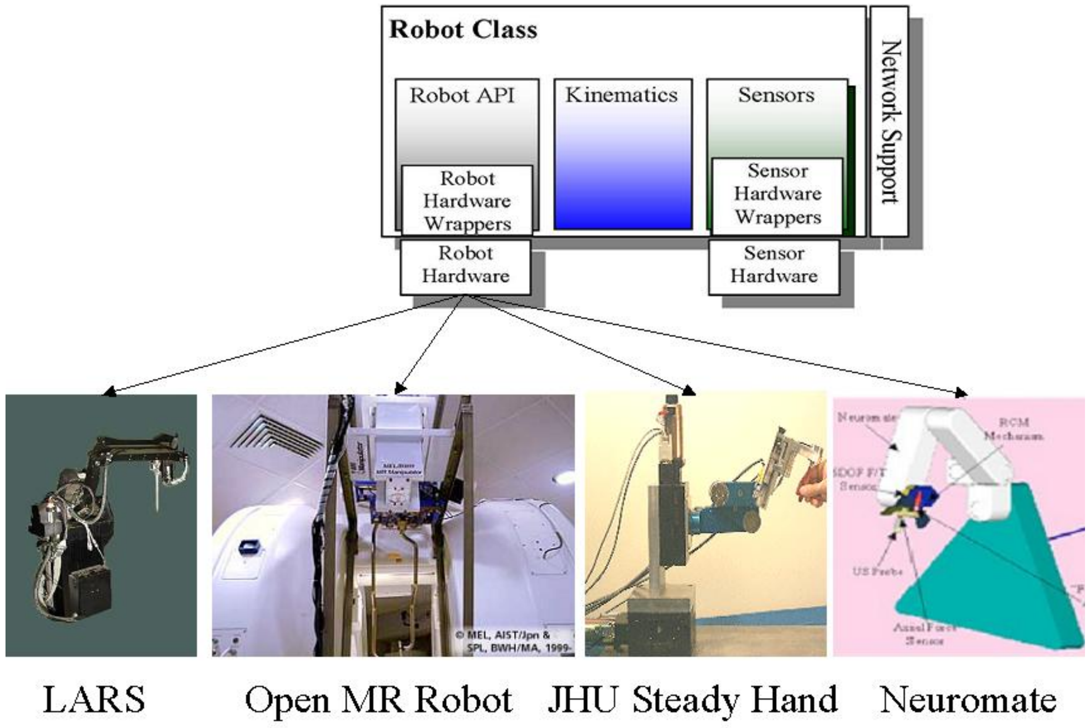


Fig. 6: A sample of medical robots with which the *mrc* software library can interface

2.3.4 Extension to CISUS

The architecture of the CISUS module was designed to allow for flexible extension with minimum amount of new code added. To add a new algorithm or method to the CISUS, the following steps are followed:

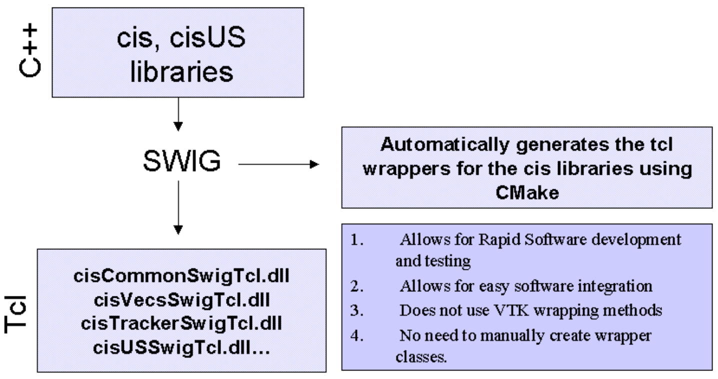
1. Add the code to the appropriate class {Acquire, Capture, Reconstruction}.
2. Add the methods to wrap the inserted function in the *vtkCISUSContainer* class.
3. Edit the GUI interface in the CISUS.tcl file to make the new function appear in the GUI.

To add new classes to the CISUS module, the following simple process is followed:

1. Place the classes in the {SLICER_HOME}\Modules\vtkCISUS\cxx directory and edit the CMakeListsLocal.txt to add the new .cxx files to the LOCAL_NO_WRAP_SRCS variable.
2. Use the *vtkCISUSContainer* class as an entry point for the new class and add the appropriate Tcl code to the GUI in the CISUS.tcl file.
3. Run *CMake* to regenerate the *vtkCISUS* makefiles and then recompile the library.

2.3.5 Rapid prototyping interface

In the process of developing the CISUS system, we explored the possibility of using wrapped C++ libraries into Tcl using the interface compiler, *SWIG*. By combining the powerful makefile generator *CMake* with the *SWIG* development tool, we were able to implement an automatic generation of the Tcl wrapped libraries. This Tcl layer was very helpful in providing a foundation for rapid software development and testing. Normally *VTK* uses the Tcl interface generator *Cable* to wrap classes into other languages. Initially our automatic wrapping strategy was based on using *CMake*



Automatically generates the tcl wrappers for the cis libraries using CMake

1. Allows for Rapid Software development and testing
2. Allows for easy software integration
3. Does not use VTK wrapping methods
4. No need to manually create wrapper classes.

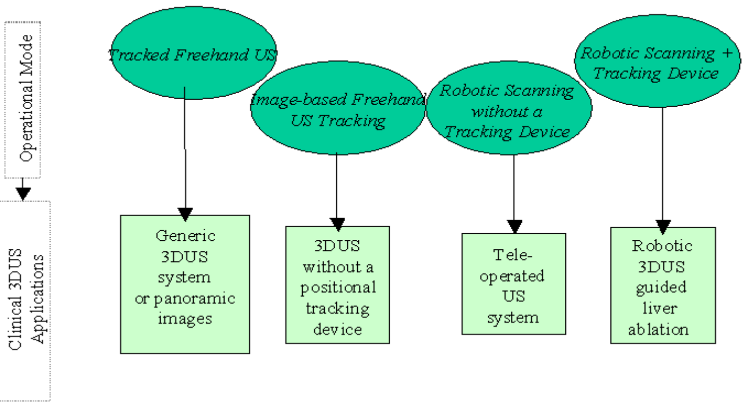
Fig. 7: The rapid software development pipeline

and *Cable*. Due to limited support of the *C++* STL libraries as well as advanced templated code in the *Cable* package, the *SWIG* package was utilized instead. However in the development of our algorithm, we realized that the synchronization between the tracking device, the video input and if present, a robotic device to be performed in the *C++* layer to reduce any temporal delays. After coming to this decision, we instead chose the path of using *VTK* inheritance to wrap only the *vtkCISUSContainer* class to interface to the Tcl layer. However, we left the implementation of using the *SWIG* route of the Tcl interface for the ERC libraries in order to facilitate faster software development for other developers of the CISUS system. The *SWIG* interface for the CISUS module also allows easier software integration. In addition to Tcl wrapped libraries, *SWIG* is also able to produce Perl, and Python wrapped libraries.

One important consideration is that although *3D Slicer* contains the functionality of multimodality imaging, semiautomatic segmentation, 3D surface model generation, automatic registration, and 3D visualization, the CISUS system code solely relies on *VTK* and can be transplanted into a GUI environment outside of *3D Slicer*.

3 CURRENT IGT APPLICATIONS

Based on the clinical application requirement specifications gathered during the setup phase, there might be four possible operational modes, as shown in Fig. 8. The first configuration mode, "Tracked Freehand US" is utilized for any generic 3DUS application. The second configuration mode, "Image-Based Freehand US Tracking" is a cost effective 3DUS system without the usage of a physical tracking device[8,9]. The third operational mode, "Robotic Scanning without a Tracking Device" utilizes the robot's encoder position instead of a tracking reading; this configuration can be demonstrated by the ultrasound positioning robot developed at the University of British Columbia.[10] Finally, the last operational mode, "Robotic Scanning + Tracking Device" illustrates a synergetic 3DUS system that incorporates both the tracking device and robotic control; The previous CISUS system developed in the ERC-CISST lab is an example of this operational mode by utilizing the LARS robot and a magnetic tracking device.[14,15]



Operational Mode
Clinical 3DUS Applications
Tracked Freehand US
Image-based Freehand US Tracking
Robotic Scanning without a Tracking Device
Robotic Scanning + Tracking Device
Generic 3DUS system or panoramic images
3DUS without a positional tracking device
Tele-operated US system
Robotic 3DUS guided liver ablation

Fig. 8: Possible operational modes for the CISUS and their corresponding clinical applications

The authors have utilized a preliminary version of CISUS system to develop different US-guided liver ablation prototypes. These prototypes vary from freehand to robotic 3DUS guidance, also from manual to robotically-guided needle insertion, and utilize a homogeneous user interface through the use of CISUS- *3D Slicer*. Fig. 9 shows a snapshot for the integrated CISUS-*3D Slicer* system in an ex-vivo liver experiment.

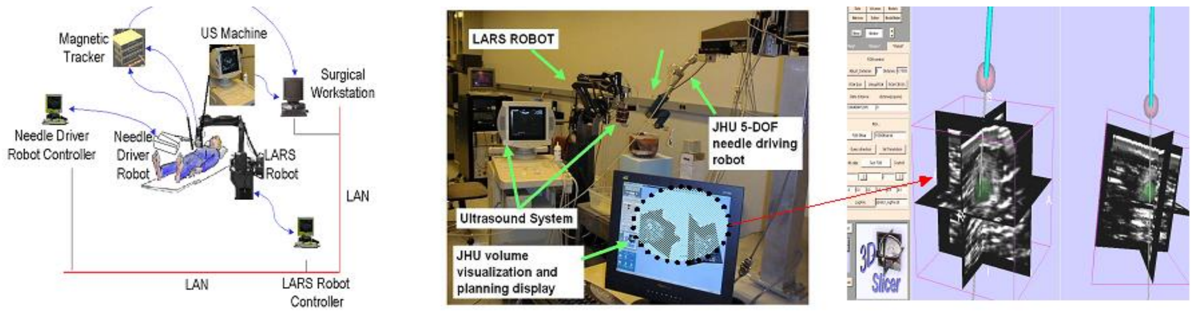


Fig. 9: Robotically assisted 3DUS liver ablation system; system layout (left), and experimental setup (middle), and a typical planning screen with CISUS-3D-Slicer interface (right).

At the CIMIT center in Boston, Ellsmere et al. are developing a novel system [16] that tracks a laparoscopic ultrasound transducer and displays the plane of the US image relative to a 3D model of the patient reconstructed from body CT (Fig. 10), thereby providing important spatial cues for the operating surgeon.

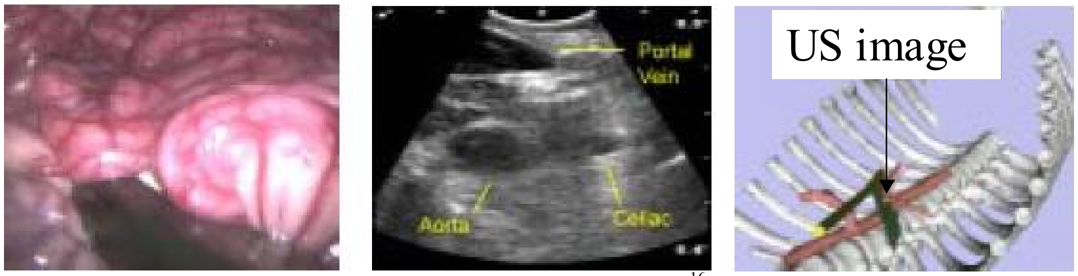


Fig. 10: An IGT application using laparoscopic ultrasound [16] (Courtesy of J. Ellsmere.)

4 CONCLUSIONS

The available 3DUS systems are both, difficult to augment functionality and to integrate into a clinical application. The robotic/freehand 3DUS open environment (CISUS) and *3D Slicer* as an open source research tool intended for diagnostic visualization and surgical planning has demonstrated the potential of solving these limitations. The integrated 3D US system, "CISUS for IGT", uses a modular tracking API and provides the following salient features:

1. A generic 3DUS visualization workstation that could be used in different clinical applications
2. A flexible system that supports or can be extended to support any type of US image acquisition (through frame grabber cards or other video input)
3. A flexible system that incorporates external hardware to allow for specialized configurations (i.e. a tracking device or a robot)
4. Rapid clinical/surgical US based applications developing tool for researchers
5. This new open, cross- platform compiler independent architecture will serve as a valuable tool to the medical imaging community
6. Most importantly, an unprecedented open synergetic environment for US-guided therapy

The CISUS-*3D Slicer* open system substantially assisted in rapidly developing different research tools and prototype systems in our laboratories. Two beneficiaries of CISUS are the previously mentioned, 3DUS-guided liver ablation (Fig. 9) and the laparoscopic ultrasound (Fig. 10) projects. In future work, we plan to fully integrate the CISUS system with medical robot control and test it under different operational modes (static vs. dynamic), integrate new hardware. A current challenge is improve on the available documentation and code maintenance, thereby encouraging the IGT research community to participate in the CISUS project, either as users or developers. One of the drawbacks of the system is that the US beam calibration module exists as a stand-alone MATLAB based toolkit and is currently not part of the CISUS system under *3D Slicer.*

ACKNOWLEDGEMENTS

We are grateful to Mr. John Walsh (ALOKA Inc.) for providing the SSD1400 US equipment. We also thank Anton Deguet for his input and assistance in various aspects of the project and Andy LaMora for final proofreading. We thank Dr. James Ellsmere for the images used in Fig. 10. The project was supported by the National Science Foundation under the Engineering Research Center grant #ERC 9731478.

REFERENCES

1. Richard W. Prager, A. H. Gee, L. Berman, "*Stradx: real-time acquisition and visualization of freehand 3DUS*", Medical Imaging Analysis 3(2), pp129-140, 1998.
2. Sakas G, "*InViVo Interactive Visualizer of Large Scalar Voxel Fields*", Computer Graphic topics, 4(2), pp. 12-13, 1992.
3. MISON (Centre for Medicine and Technology), NORWAY http://www.mison.no.
4. COMPUTERIZED MEDICAL SYSTEMS (CMS), St. Louis, Missouri http://www.cms-stl.com.
5. MIT AI Lab and the Surgical Planning Lab at Brigham & Women's Hospital http://www.slicer.org.
6. Varian Medical Systems, Palo Alto, CA http://www.varian.com.
7. ERC-CISST LAB Johns Hopkins University http://www.cisst.org/ resources/software.
8. R. W. Prager, A. H. Gee,G. M. Treece, C. J. C. Cash,and L. H. Berman, "*Using Image-based Regression to Acquire Freehand 3D Ultrasound*", CUED/F-INFENG/TR 436, June 2002.
9. Wendy L. Smith, Aaron Fenster, "*Analysis of an image-based transducer tracking system for 3D ultrasound*", Medical Imaging SPIE 2003, Proc. SPIE Vol. 5035, p. 154-165.
10. Abolmaesumi, P. Salcudean, S.E. Wen-Hong Zhu Sirouspour, M.R. DiMaio, S.P., "*Image Guided Control of a Robot for Medical Ultrasound*" IEEE Trans. Rob. & Auto., 18(1) Feb 2002.
11. A. Fenster, D. B. Downey, and N. H. Cardinal, "*Three Dimensional Ultrasound Imaging*" Physics in medicine and biology, vol 46, pp. 67-99, 2001.
12. D. Gering, A. Nabavi, R. Kikinis, W. Eric L. Grimson, N. Hata, P. Everett, F. Jolesz, W. Wells III, "*An Integrated Visualization System for Surgical Planning and Guidance using Image Fusion and Interventional Imaging*", Medical Image Computing and Computer-Assisted Intervention (MICCAI), pp. 809-819, 1999.
13. R. N. Rohling, A. H. Gee, and L. H. Berman, "*Radial Basis Function Interpolation For 3D Ultrasound*", CUED/F-INFENG/TR 327, July 1998.
14. Boctor EM, Fischer G, Choti M, Fichtinger G, Taylor R, "*Dual-Armed Robotic System for Intraoperative Ultrasound Guided Hepatic Ablative Therapy: A Prospective Study*", IEEE 2004 International Conference on Robotics and Automation (accepted).
15. Boctor EM, Taylor RH, Fichtinger G, Choti MA, "*Robotically assisted intraoperative ultrasound with application to ablative therapy of liver cancer*", Proc. SPIE Vol. 5029, p. 281-291, Medical Imaging 2003: Visualization, Image-Guided Procedures, and Display; Robert L. Galloway, Jr.; Eds.
16. James Ellsmere, Jeffrey Stoll, David Rattner, David Brooks, Robert Kane, William Wells, Ron Kikinis and Kirby Vosburgh, "*A Navigation System for Augmenting Laparoscopic Ultrasound*", Medical Image Computing and Computer-Assisted Intervention (MICCAI), pp. 184 - 191, 2003.
17. Taylor RH, Funda J, Eldridge B, Gruben K, LaRose D, Gomory S, Talamini M, Kavoussi LA, and Anderson JH, "*A Telerobotic Assistant for Laparoscopic Surgery*", IEEE EMBS Magazine Special Issue on Robotics in Surgery. 1995. pp. 279-291.
18. Chinzei K, Hata N, Jolesz A, Kikinis R, "*Surgical Assist Robot for the Active Navigation in the Intraoperative MRI: Hardware Design Issues*", Proc. 2000 IEEE/RSJ International Conf Intelligent Robots and Systems (IROS 2000), 1: 727-32, 2000.
19. Russell Taylor, Patrick Jensen, Louis L. Whitcomb, Aaron Barnes, Rajesh Kumar, Dan Stoianovici, Puneet Gupta, ZhengXian Wang, Eugene deJuan, and Louis Kavoussi, "*A Steady-Hand Robotic System for Microsurgical Augmentation*", International Journal of Robotics Research, 18(12):1201-1210 December 1999.
20. Integrated Surgical Systems, Davis, CA http://www.robodoc.com/eng/index.html.
21. Imaging Science and Information System, Medical Center, Washington DC http://www.imac.georgetown.edu/.