MedicalStudio: a medical component-based framework

Daniela G. Trevisan¹², Vincent Nicolas¹, Benoit Macq¹, Luciana P. Nedel²

¹Communications and Remote Sensing Laboratory, Université catholique de Louvain,
   (UCL) Place du Levant 2, Bat. Stevin, 1348 Louvain-le-Neuve, Belgium

²Instituto de Informática – Universidade Federal do Rio Grande do Sul (UFRGS)
   Caixa Postal 15.064 – 91.501-970 – Porto Alegre – RS – Brazil

{trevisan, vincent.nicolas, macq}@tele.ucl.ac.be, nedel@inf.ufrgs.br

Abstract. This paper introduces MedicalStudio: a composable, open-source easily evolvable cross-platform framework that supports surgical planning and intra-operative guidance with augmented interactions. It is designed to integrate the whole computer aided surgery process to which both researchers and clinicians participate. In this paper we provide a description of the framework architecture as well as some medical components and applications already developed into this framework. Collaboration with several research centers and medical clinics has shown the versatility and promising dissemination of this medical framework.

1. Introduction

Advances in scanning technology and other data collection systems provides a wide spectrum of useful and complementary information about a patient’s status to research and clinical domains. To combine these different types of information into a coherent presentation assuring usable and cognitively adequate interaction within the Operating Room becomes extremely difficult. More sophisticated methods of analysis, guidance and interaction need to be created to achieve these objectives.

Concerning some of the existents systems for surgical planning and intraoperative guidance, 3D Slicer [Gering, et al 1999] has been one of the first open-source application enabling data fusion and interventional imaging. Julius [Keeve, et al 2001] is another extensible, cross-platform software framework providing a complete medical processing pipeline, but more targeted to visualization and analysis, whereas IGSTK [Cleary, et al 2004] is the latest framework currently under development for open-source component-based rapid prototyping of image guided surgery applications.

Research efforts in Image Guided Surgery (IGS) systems and image processing are investigating how to leverage on such techniques to develop more effective, integrated solutions for the planning, simulation, and finally intra-operative guidance systems either based on a navigation concept or including human-computer interaction systems. As pointed out in [Cleary, Chung and Mun 2004], other problems regarding the use of new technologies are the lack of compatible and interchangeable equipments and limited communication among surgeons and others in the team especially during surgical procedures.
MedicalStudio proposes to address these issues by realizing a component-based framework targeted to medical interventions and equally attentive to research and therapeutic concerns. It consists of a unique framework for planning, simulating and performing interventional procedures assuring more compatibility within the surgical workflow. Its modular architecture can easily manage abstractions of hardware peripherals and directly make data available from them. Components developed in collaboration with several research centers and medical clinics have shown the promising dissemination and versatility of this medical framework in various disciplines.

The paper is structured as follows. In section 2 we first describe the component-based architecture. In the General Purposes Components and Applications sections we illustrate examples of high level software components and applications implemented within the medical framework. The final section is devoted to the conclusions and the presentation of future plans.

2. MedicalStudio Framework

This section gives an overview on MedicalStudio architecture. It then details the core functionalities and implementation choices of the framework.

2.1 Architecture

![Diagram](a) The ARCH software architecture (b) MedicalStudio architecture extending the ARCH model.

A framework allowing the centralization of all tasks of assisted surgery must have a consistent and evolvable architecture. For that reason MedicalStudio is based on a component architecture. Such software architectures have been widely discussed either for multimodal interactions [Bouchet and Nigay 2004] or visualization systems [Chi and Riedl 1998]. The software architecture implemented in MedicalStudio is a modified version of the ARCH model [Bouchet and Nigay 2004] (Figure 1(a)) to add a specialization of the input and output modalities. The principal components of the architecture illustrated in Figure 1(b) are detailed as follows:

1) **Functional Core** contains all data processing components such as registration algorithms, I/O filters, etc.
2) **Functional Core Adapter** is an abstraction layer that allows the event manager to communicate with the functional core.

3) **EventManager** is the dialog controller between interactions and functional core. When interactions occur, the event manager will propagate them to the functional core, and if needed interpret them before.

4) **Interaction** represents user input components such as mouse, keyboard, magnetic pen, vocal recognition, etc.

5) **Tools** are components that map interactions into data processing or visualization modification. For example, a mouse click will be translated into a rotation (view manager) by the navigation tool, but it will be translated into a data modification (functional core) by a segmentation tool.

6) **View Manager** manages all view and knows how data can be visualized on which type of view.

7) **Renderers** are components which render specific data type onto specific views. An example is the rendering of an image on a 3D view with the raycasting algorithm. In order to clarify the development process and to fix where components should be in the visualization pipeline systems [Chi and Riedl 1998], classified them into operators. MedicalStudio architecture groups the data operators into data components, the visualization operators into output components, and adds input components. There are plans to refine this subdivision to add more control on the visualization pipeline.

### 2.3 Framework Implementation

In order to assure cross-compatibility in terms of execution and development, MedicalStudio is written in C++. The language is commonly used by the signal and image processing community. Publicly available libraries are used to provide well known functionalities and avoid re-implementation of already validated methods. These libraries have been chosen in function of their specifications, their development language, their cross-compilation possibilities and the community working with them. VTK\(^1\) is used for all visualization tasks. GTK\(^2\) is used for all graphical user interfaces. ITK\(^3\) is used for signal and image processing. Dcmtk\(^4\) is used for Dicom 3.0 standard compliance. Figure 2 is showing how these libraries are inserted into the architecture pipeline. This list of libraries is not fixed. Thanks to the modular architecture any other library can be linked into a new component providing high extensibility to MedicalStudio. We define a plug-in as a group of components compiled together into one shared library. These plug-ins can group components by any criteria but by convention it is preferable to group them either by type or procedure context. An xml file is created for each plug-in describing its content. With this, the kernel is able to create a repository of all available components without loading them into memory, plug-ins will be loaded lately, only when required letting resources available for data or processing. This facilitates the distribution process as only two files will be needed to add this procedure to the basic MedicalStudio platform. This allows easy customization of final applications. For example all the components for a specific neurosurgery procedure will be grouped into one plug-in. For applications that do not need the

---

\(^1\) VTK: [http://www.vtk.org](http://www.vtk.org)  
\(^2\)GTK: [http://www.gtk.org](http://www.gtk.org)  
\(^3\)ITK: [http://www.itk.org](http://www.itk.org)  
\(^4\)Dcmtk: [http://www.dicom.offis.de/dcmtk.php.en](http://www.dicom.offis.de/dcmtk.php.en)
neurosurgery plug-in (e.g. maxillo-facial surgery), removing the two files is enough to eliminate the unwanted functionalities.

![Diagram of MedicalStudio component-based architecture pipeline.](image)

**Figure 2. MedicalStudio component-based architecture pipeline.**

### 3. General Purpose Components

In the following part we will present examples of generic software components that can be found in MedicalStudio to develop more complex scenarios for the comprehensive support of interventions (planning, simulation, intraoperative guidance) and clinical research. This includes components for registration, segmentation and 3D reconstruction and augmented visualization. Thanks to the component-based architecture, while developing a new plug-in in MedicalStudio the programmer does not need to care about implementing basic tools such as 2D and 3D rendering, view organization and coordination, colors settings, transparency settings or performing operations such as scaling, rotations and zooming. All these functionalities as well as other plug-ins integrated into MedicalStudio can easily share all the data required for their execution.

#### 3.1 Multimodal Registration

Currently MedicalStudio supports two kinds of multimodal registration: rigid and non-rigid.

1) **Rigid Registration**: the rigid registration algorithm relies on the Insight Registration and Segmentation Toolkit. It makes use of a Simultaneous Perturbation Stochastic Approximation of the cost function gradient to seek for the optimum. This method has first been introduced by Spall (1998). The estimate of the gradient is fast and more robust to the presence of local minima than classical gradient descent schemes. Such registration algorithm was validated in collaboration with the Radiation Oncology Dept., St-Luc University Hospital, Belgium. A group of fifteen patients with pharyngolaryngeal tumors were imaged by CT scan, MRI (T1-and T2 weighed) and PET (transmission and FDG emission) with constrain masks. All these images were automatically registered using Mutual Information as criterion. Four of the fifteen patients were registered manually. The results from both methods were compared in terms of accuracy, reproducibility, robustness and speed. Maximum Euclidean deviations to the reference transformations were smaller than 1.7mm for PET-CT registration and smaller than 4.7mm for MR-CT registration. Furthermore, the automated method converged to validated results for clinical cases where experts failed...
using a manual registration. Automated registration needs 2 to 8 min on standard platforms. Figure 3 illustrates the MedicalStudio registration interface.

Another kind of rigid registration uses the surface-based algorithm. It minimizes the mean square distance between the points representing the real object and the surface from segmented MRI images. For that we applied the implementation of Saito’s EDT (Euclidean Distance transform) found in Cuisenaire (1999). Results of this method are illustrated in Figure 4. This method is reused by other specialized medicalstudio plug-ins such as the Transcranial Magnetic Stimulation (TMS) application described in Noirhomme, et al (2004) and the ACROGuide application described in section 4.

2) Non-rigid Registration: The use of the SPSA (Simultaneous Perturbation Stochastic Approximation) method has been investigated for optimizing a large set of parameters characterizing a non-rigid deformation. We use volumetric tetrahedral meshes as non-rigid deformation models. Their main advantage is the capability to deal with non-uniform sampling of the image domain and to allow multi-grid representation of the deformation. The deformation is constrained by the linear elastic energy acting as a regularization term. This regularization term can allow more flexibility in some regions than in others by giving different mechanical properties to elements in different regions. The implementation allows to run this algorithm in parallel on Symmetric Multi Processor architectures (SMP) by distributing independent computations of the cost functions to different threads. This work has been developed in collaboration with
3.2 Segmentation and 3D Reconstruction

This plug-in contains various components allowing the segmentation and appropriate labeling of anatomical structures for 3D reconstruction. As well as methods for manual segmentation where borders are drawn directly onto the raw image dataset, one of the classic methods for performing the task is intensity based filtering from MRI or CT dataset using the Marching Cubes algorithm [Lorensen and Cline 1987] and acting in the same way as thresholding segmentation. Figure 5 illustrates the result of this method. The correct reconstruction also requires connectivity filtering to extract cells that share common points and satisfy a scalar threshold criterion. Another algorithm implements automatic, atlas-guided segmentation [Haese, et al 2003] which is suitable in presence of deformed anatomy caused by tumors and operates through a combination of rigid and non rigid registration components. The computed transformations map the atlas segmented structures onto the subject volume. The plug-in includes level set segmentation [Osher and Sethian 1988] with active contour modeling for boundary object detection letting an initial interface evolve towards the object boundary. The level set implementation automatically manages the topology changes of the evolving interface, allowing detection of several objects.

Figure 5. Reconstructed 3D model from MRI data with segmented structures.

4. Applications

This section illustrates examples of medical applications developed within the MedicalStudio framework by making use of all the available plug-ins for Image Guided Surgery generic components as well as the basics tools for view management, visualization and manipulation of images and volumes.

The flowchart that represents the system designed by us to support maxillofacial surgery including pre-operative CT scanning and planning to surgical guidance is shown in Figure 6. The process begins with the images acquisition using a CT scanner. Then, a threshold is used to filter the images, separating the bones and the other soft tissues. After this processing, the images are segmented and a 3D model of the skull is reconstructed from the segmented images. Using the reconstructed 3D model and the 3D Medical Assistant for Orthognathic Computer Surgery, a path-line representing the osteotomy path is then designed. The next step involves the hardware calibration and the registration of real and virtual objects in the ACROGuide component. This step
consists in picking points over patient mandible using an optical tracker system and then register them with the 3D surface reconstructed before from patient CT images.

![Figure 6. System flowchart supported by MedicalStudio to provide surgical guidance during maxillofacial surgery.](image)

### 4.1 3D Medical Assistant for Orthognathic Computer Surgery

This application contributes to the realization of a comprehensive orthognatic image guided surgery system aim to improve accuracy and performance of surgical acts throughout the different phases involved (diagnosis, planning, simulation and surgery). A wizard is being developed to guide the surgeon during the 3D cephalometric analysis [Olszewski, et al 2003] (Figure 7). The analysis is divided into 16 modules which determine and quantify facial asymmetry, dentomaxillofacial dysmorphosis and cranio maxillo facial deformations. As the whole analysis uses 3D concepts, the interface allows the visualization and navigation in 2D (using the three orthogonal axes) and in 3D (using a polygonal reconstruction of the bones).

![Figure 7. Maxillo-facial computer assisted planning. (left) Planes position (right) Wizard for assistance.](image)
Figure 7 shows one of the steps that the surgeon should do. It emphasizes the region of interest where the action is to take place. Effects of completed actions are highlighted and should be easy for the user to distinguish.

4.2 ACROGuide

In collaboration with the Service de Stomatologie et Chirurgie Maxillo-faciale at Saint Luc Hospital, in Brussels, we proposed the development of an application using a mixed reality interface to guide surgeons in this type of surgeries. The goal of this application is to increase the first surgery success, avoiding a second intervention. To achieve this objective, we are mixing real and virtual images by projecting a virtual guidance path-line on the patient mandible live video images. Then, the surgeon should cut the patient mandible paying attention to follow this path-line and avoiding touching the dental nerve [Trevisan, et al 2006]. When the markers are recognized by the system, two spheres are displayed on the screen: one attached to the tool; and the other, bigger, displayed in the middle of the tool (Figure 8). The smaller one indicates if the tooltip is placed (i.e. assuming the green color) or not (i.e. assuming the blue color) on the correct position indicated by the path-line (red line) projected into the image video. The big one indicates the distance between the tooltip and dental nerve using three colors: gray means go ahead; blue means pay attention (i.e., you are less than 3mm from the nerve); red means stop (i.e., you will probably touch the nerve).

5. Conclusions and Future Works

In this work we have presented MedicalStudio\( ^5 \) which is a medical framework supporting several surgical guidance aspects. The system is suited for the processing of multimodal image data registration as well as augmented visualization. The main advantages of this medical framework can be summarized as following: 1. Flexibility: the component-based architecture allows the integration of new components sharing resources offered by other plug-ins furnished by a large community of developers. In this case only the plug-ins used are loaded. 2. Inter-operability: basic resources such as reading and visualizing support to several medical images formats guarantee the

\( ^5 \) MedicalStudio is actually free available for download at http://www.medicalstudio.org.
exchange of data between pre and intraoperative phases. 3. Benchmarking: having different approaches integrated as plug-ins allows to researchers compare results. 4. Cross-platform: the same applications scenarios can be compiled currently to Windows, Linux and MacOSX platforms. 5. Independent Validation: the component-based architecture favors independent validation of single modules. It is an advantage to reuse already validated components when building a new medical application.

On going work is focused on the integration of a new component for rendering using hardware graphic in collaboration with the Computer Graphics Laboratory at University Federal of Rio Grande do Sul. The support of graphics hardware for texture-based visualization allows efficient implementation of rendering techniques that can be combined with interactive sculpting tools to enable interactive inspection of 3D datasets (Figure 9) [Huff, et al 2006]. This integration will test the flexibility of MedicalStudio to support such kind of application.

![Figure 9. Example of an interactive sculpting session where the volume of interest (VOI) is a torso dataset. Visualization is shown for each tool: (a) original VOI, (b) VOI carved by the eraser tool, (c) VOI carved by the digger tool, and (d) VOI carved by the clipper tools.](image)

Subsequent versions of MedicalStudio will include integration with PACS providing DICOM network compatibility and extend image support format such as electrocardiogram and ultrasound images which require a differentiated image processing treatment. Finally future works will address how advanced technologies such as further intra-operative devices, robotics and telecollaboration can be supported by this framework and integrated seamlessly into the operation room.

6. Acknowledgment
This work was partially supported by the HEROL (WALEO 21/5132) research project and CNPq (Brazilian Council for Research and Development)

7. References


