An Open Source Framework for Surgical Simulation

Release 1.00

M.A. Rodriguez-Florido[1,2], N. Sánchez Escobar[1], R. Santana[1] and J. Ruiz-Alzola[1,2]

October 31, 2006

[1]Center for Technology in Medicine, Dep. Signals & Communications, University of Las Palmas de Gran Canaria, Canary Islands, Spain
{marf,norberto,rsantana,jruiz}@ctm.ulpgc.es
[2]Canary Islands Institute of Technology (ITC, S.A.), Canary Islands, Spain
{marf,jruiz}@itccanarias.org

Abstract

In this work a software framework for the virtual surgical simulation is presented. Our environment ESQUI[1] is based on an open software and multi-platform architecture (Unix, Windows, etc.), resulting in a suitable framework for research and development. The modularity and connectivity between its components allow the environment to be extensible and adaptable to any type of virtual simulation. The virtual surgical scene is described by an XML file that contains all the necessary information for the simulation. This file description language has been named Surgical Reality Modeling Language (SRML). Although our framework is still under development and needs validation in education and clinical training, our environment ESQUI has proved to be a modular platform that runs across most important areas of virtual surgical simulation.

Contents

1 ESQUI Overview **2**
1.1 Implementation and Software Requirements . . . 2
1.2 Surgical Scenes . . . 3
1.3 Scenes Description File: SRML . . . 3
1.4 Simulation: Collisions and Deformations . . . 5
1.5 Haptics . . . 5

2 Conclusions and Future Lines **6**

A Install **7**

[1]The acronym comes from the Spanish "Entorno para Simulación QUIrúrgica", i.e. Framework for Surgical Simulation

B Quick start: An example 8
B.1 Keyboard commands 8

During the last years, surgical simulation has become a versatile means for learning and training surgery, mainly minimally invasive surgery (MIS)[15], which due to its peculiarities (new procedures of access in the intervention, specialized instruments, etc.) needs specific learning and training. For MIS learning and training, surgical simulators do not only contribute to the synthetic creation of interactive scenes that reproduce certain interventions with certain properties, perhaps difficult to reproduce in animals or in corpses, but also reduce the time needed by experts supervising the learning or training process. Therefore, one of the challenges of the technology applied to the medicine has been the surgical simulation. This challenge has a significant impact on groups and research centers, as well as on the surgical industry. Let's consider examples such as the Spanish Network SINERGIA (www.ctm.ulpgc.es/sinergia), with more than ten groups (among them we have universities, clinical centers and hospitals) in Spain, and the industrial studies of trends [13].

In this paper we propose a framework called ESQUI based on multi-platform open source software. Although our environment is intended for any surgical technique, at this moment it is applied to laparoscopic surgery. Our framework includes 3D modeling, simulation and haptic connectivity. In addition, our software uses a description file of the surgical scene, which we have named Surgical Reality Modeling Language (SRML).

The paper is organized as follows. First, we make a description of the software architecture of ESQUI, describing briefly the functionality of each one of its components. Then, we present the description SRML file, and the advantages of its use in a surgical simulation environment. Finally, we present some conclusions and future lines.

1 ESQUI Overview

The environment ESQUI is a software framework that has been developed in our R&D group as part of our efforts in surgical simulation. Mainly, ESQUI is the result of the second author Master Thesis [11] (supervised by the first author), our participation in the National Network Sinergia (www.ctm.ulpgc.es/sinergia), and the interest of the ITC,S.A. in this line of work.

At this moment, ESQUI is distributed under the Creative Commons' "by-attribution" license (http://creativecommons.org/licenses/by/2.5).

The framework deals with the three great areas of surgical simulation: design and generation of 3D scenes, software algorithms for simulation (collisions, deformations, etc.), and management of the information in the surgical scene. Although ESQUI has been developed for laparoscopic surgery, it can be applied to any other.

1.1 Implementation and Software Requirements

Surgical simulators are computer software projects including some type of hardware (i.e. haptic device) to interact with the user. This kind of software is usually too big and complicated to be written by one person acting alone and it almost always requires collaboration and joint development. Also, many platforms (Unix, windows, etc.) can be used for developing. Therefore, in our opinion, the success of a surgical simulator project depends of the development tools used for the generation of extensible and re-usable software, and the population of the community that support that tools.

ESQUI has been implemented using open source and multi-platform software: Blender [1], python [16], VTK [18][19], and Tcl/Tk [7]. The wide community of these toolkits or packages allows ESQUI benefits of the control version quality and improvement carried out by the international community. In our environment, except Blender, the software has been programmed in our group. We have defined the basic structure of the framework around VTK, recoding or wrapping algorithms under VTK/Tcl-Tk.
Platform or compiler dependency is avoided using Cmake open-source make system [12]. We control versions using subversion [6], and the code is documented using doxygen [3]. In the near future, we will include an interface with DART [2] for testing the software.

1.2 Surgical Scenes

Although many software packages exist for 3D modeling, in ESQUI we use an open source and multi-platform software: Blender [1]. This software package presents some points of interest for our project:

- Open source and multi-platform.
- Tools for 3D design and modelling.
- A command interface under python that allows to include scripts to modify the 3D scene.

Based on these points we find some advantages:

- To share 3D scenes from different platforms.
- Connect VTK to a 3D modeling software.

We have modified the work of [8] to exchange 3D models (polygons and textures) between Blender and VTK. This connectivity is very important for us, because 3D models generated from real medical images (MRI, CT, US, etc.), using own or third-party software (i.e. diSNei [10], Slicer [20], etc.), can be loaded in Blender to add the effects of texture and realism, and finally used in the virtual simulator. Then, the surgical scenes can be made synthetically or with real information.

In figure 1, we present an example of a 3D model scene under Blender.

1.3 Scenes Description File: SRML

The complexity of a virtual surgical scene (virtual models, textures, illumination, location of the models, biomechanical properties, instruments, etc.) needs a description file that should be interchangeable and extensible. XML offers a standard for the interchange of data, and presents some important properties:

- The integration of data between applications and also within the own application.
- It shows clearly the relationship between data and information.
- Extensible and adaptable to new or future specifications.

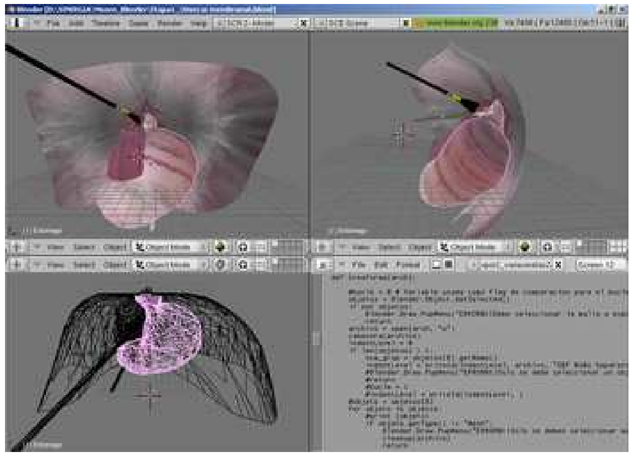

Figure 1: 3D modeling using Blender. The figure shows a surgical scene of the laparoscopic Nissen’s procedure.

Therefore, it seems suitable to implement a description file of a surgical scene using XML. With this objective the Surgical Reality Modeling Language, from now SRML, was born. SRML contains:

- Path to the virtual models files of tissues (.vtk for the models and .jpg for the textures) and the surgical instruments.
- Information about the tissues biomechanic properties and the biomechanical model to be used.
- Relative position of cameras and surgical instruments as well as positions and restrictions of movement.
- Environment properties in the surgical scene (direction of the gravity force, luminosity, etc.).

In the text below, we show an example of a SRML file.

```
<!DOCTYPE SRML SYSTEM ’SRML.dtd’>
<SRML>
<Undeformable>
 <UnDefModel VTKName=’Escenarios/Cuerpo_Inclusion.vtk’
TextureFile= ’Escenarios/black.jpg’></UnDefModel>
 <UnDefModel VTKName= ’ Escenarios/FORMA1/Marco.vtk ’
TextureFile= ’ Escenarios/GRIS.JPG ’></UnDefModel>
</Undeformable>
<Deformable>
<DefModel VTKName= ’Escenarios/FORMA1.vtk ’ TextureFile= ’ Escenarios/FORMA1/FORMA1.JPG ’
DEFModel= ’ MASS-SPRING ’ StrongFactor= ’0.000000’ CanBeImpudent= ’1’ CanGetHookedUp= ’1’
ContainsFluids= ’1’ BEMYoungModulus= ’15.000000’ BEMPoissonRatio= ’0.400000’ BEMDensity= ’1.000000’
BEMAlpha= ’10.000000’ BEMOmmega= ’0.300000’ BEMGamma= ’50.000000’
BoundaryConditionsX= ’ {153 154 155 156 157 158 159 160 161 162 163 152 141 129 110 87 64 45 32 20
21 19 17 15 13 11 9 7 5 3 1 0 22 33 46 65 88 111 130 142} ’
BoundaryConditionsY= ’ {153 154 155 156 157 158 159 160 161 162 163 152 141 129 110 87 64 45 32 20
21 19 17 15 13 11 9 7 5 3 1 0 22 33 46 65 88 111 130 142} ’
BoundaryConditionsZ= ’ {153 154 155 156 157 158 159 160 161 162 163 152 141 129 110 87 64 45 32 20
21 19 17 15 13 11 9 7 5 3 1 0 22 33 46 65 88 111 130 142} ’></DefModel>
</Deformable>
```

```
<Entorno>
<Name name= ’ FORMA1 ’></Name>
<Gravity Place= ’0.000000 1.000000 0.000000’ Direction = ’1’></Gravity>
<Ligth  EnviromentLigth= ’0.000000 0.000000 0.000000’
DifuseLigth= ’1.000000 1.000000 1.000000’
EspecularLigth= ’1.000000 1.000000 1.000000’
LigthIntensity=’0.750000’>
</Ligth>
<Tools VTKName= ’../Tools/tools1.vtk’ Ratio= ’0.300000’
PointOfEntry= ’-1.474732 -4.723502 -7.500000’
InitPoint= ’-2.616987 -4.723502 -2.632223’
Orientation= ’0.000000 0.114988 0.000000 0.993367’>
</Tools>
<Tools VTKName= ’../Tools/tools2.vtk’ Ratio= ’0.300000’
PointOfEntry= ’-8.671877 -4.845428 -7.500000’
InitPoint= ’-7.632318 -4.845428 -2.609262’
Orientation= ’0.000000 -0.104528 -0.000000 0.994522’>
</Tools>
<Camera OpenAngle=’0.698131’ PointOfEntry= ’-5.034461 -4.793622 -7.500000’
InitPoint= ’-5.034461 -4.793622 -7.500000’ Orientation= ’0.000000 0.000000 0.000000 1.000000’>
</Camera>
</Entorno>
</SRML>
```

Using this type of description avoids to use multiple files that duplicate information, without hierarchy and structure, and it allows our simulator to be extensible to new properties of the surgical scene (a new tag could be defined in the SRML).

1.4 Simulation: Collisions and Deformations

ESQUI includes a software module for simulation (collisions and deformations). It has been implemented under VTK and Tcl/Tk, and our code design allows to include new algorithms (own or other people’s) for collisions and deformations. At the present version, we have wrapped under VTK the algorithms of collisions proposed in [17] (distributed under the terms of the GNU Lesser General Public License [4]), and we have coded under VTK the biomechanical model proposed in [14].

In Figure 2 we show an example of ESQUI applied to laparoscopic surgery.

1.5 Haptics

ESQUI is intended to work with haptic systems (man-machine interfaces). Our environment presents a VTK class to inherit the haptic’s API wrapped under VTK.

Presently, we are using ESQUI for laparoscopic surgery, and it connects to a commercial laparoscopic surgical system. If the user doesn’t have an haptic system, he is able to use the keyboard and mouse to drive the simulation.

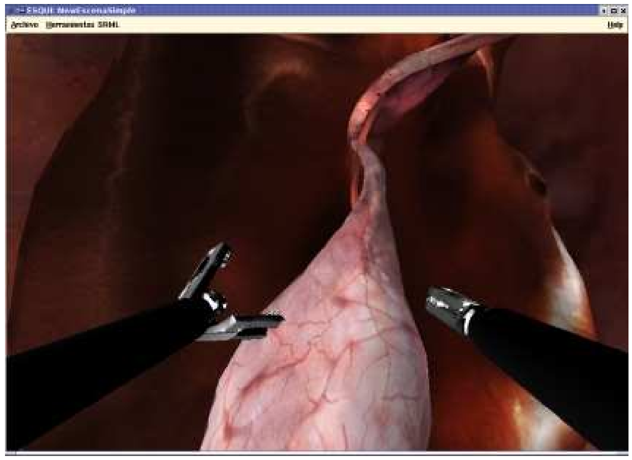

Figure 2: An example of ESQUI applied to laparoscopic surgery

2 Conclusions and Future Lines

In this paper we propose a framework for the virtual surgical simulation based on open source and multi-platform (Unix, windows, etc.) software. Our environment, ESQUI, includes a 3D modeling software (Blender) joined to a simulation software under VTK. Although it needs much work, we think that our software is an example of a open source project, and the community could use or adapt it to specific applications, and moreover, contribute to its development.

Our contribution to the community with this work could be summarized as:

- Description of the surgical scene. At the moment, there is no standard to describe surgical scenes which makes difficult the exchange of information between different simulators, users and developers. Therefore, with the SRML we propose an adaptable and extensible solution.
- A multi-platform and open source based software. Our environment has been developed under libraries/languages with a wide community of development. The code is adaptable and extensible.

Although ESQUI is being applied to laparoscopic surgery, our idea is to be able to extend it to any type of MIS technique. The type of MIS procedure and the haptic related information will be included in the SRML file. In addition, we are working to include more algorithms (own or other people's) for collisions [9] and deformations, and studying the viability to recode under VTK some third-party algorithms (i.e. OpenTissue [21], Spring [5], etc).

Future lines are focused to develop a database software module for ESQUI to include scores in the didactic exercises for surgeons training. Nevertheless, our framework still needs more development and clinical validation to be able to use it for surgical training.

Acknowledgments

This work was supported by the Spanish Ministry of Education and Science (TIC-2001-38008-C02-01 and TEC2004-06647-C03-02), the European Commission (SIMILAR NoE FP6-507609), and the Carlos III Institute of Health (SINERGIA FIS-G03/135). The first author is partially funded under the program *Torres Quevedo* of the Spanish Ministry of Education and Science and the European Social Funds, PTQ2004-1443. Third author is partially funded by the University of Las Palmas Foundation (FULP) and NogalMetal under the program *innova*. Thanks to the Spanish Network Sinergia, specially to Diana Suárez-Martel and Manuel Bernal from CTM-ULPGC, J. Blas Pagador from CCMI, and Oscar López from MedicLab.

In this section we present some indications to build the software and run a simple example.

A Install

At this moment, the ESQUI project distribution includes the folders:

- Tcl: Tcl/Tk scripts to run the software GUI (main.tcl) and the modules.
- Doc: Doxygen vtk classes documentation.
- misc: Python script to exchange virtual models with Blender.
- src: The vtk-C++ source code. This folder is organized as: - ColDetect: Source code for collision detection.
  - Colisions: Source code for managing the collisions.
  - Comun: Common code used in ESQUI.
  - Haptic: Source code for controlling/including haptics devices.
  - BEM: Mathematical functions.
  - Tools: Code for surgical tools management.
  - T2Mesh: Biomechanic model source code.
  - Wrapping: Wrapper for Tcl.

ESQUI has been developed using vtk 4.4.2 and Tcl/Tk 8.4.13. This distribution has been compiled/tested under windows XP (VS7) and linux(gcc), using the cross-platform development tool CMake (www.cmake.org).

The installation process follows the vtk project building steps. In the next lines we summarize this process:

1. Download the source code.
2. Extract the source code using winzip (windows) or tar command (tar zxvf file.tgz)under unix.
3. If you want to use the software with haptics (LWS-Immersion Corp),uncomment the line associated in the CMakelist.txt file before building. Also, remember to follow the indications in the ”Tcl/main.tcl” file to use the LSW haptic device, and uncomment the line ”Simulacion” at *haptic_good_idle_proc* {} in *LoadAndRunExerciseFromSRML.tcl*.
4. Create a ESQUiBin folder and run ccmake in:

   ccmake *path of ESQUI Source code*

   i.e.(under linux): /*path*/ESQUibin/ccmake ../ESQUI if ESQUI is in *path*.
5. Follow the ccmake configuration. Remember to set the VTK_DIR path (the folder where you have compiled vtk and you have your VTKConfig.cmake file), and the Tcl/Tk libraries path. The wrapping and other configuration are done by default (they are included in the CMakelist.txt file).
6. Make the vtk-project

   i.e. (under linux) /*path*/ESQUIbin/make
7. You shouldn’t have any error. If you have any problem, please send an email to {marf,rsantana}@ctm.ulpgc.es.

8. Set the environment variables: LD_LIBRARY_PATH and TCLLIBPATH to the path where are the (.so/dlls) vtk/wrapping libraries. If you have other Tcl/Tk versions, you should declare the TK_LIBRARY and TCL_LIBRARY variables.

   i.e. (under linux)

   export LD_LIBRARY_PATH=/*path vtk libraries*:/*path*/ESQUI/EsquiBin/bin/

   export TCLLIBPATH=/home/marf/software/ESQUI/ESQUI/Wrapping/Tcl

9. Execute *path ESQUI source Code*/Tcl/main.tcl using your vtk version.

Note: We have found a bug while we were compiling the STL libraries under Linux using the GCC 4.0.4. To avoid this bug, you should use a gcc lower version (i.e. GCC 3) or fix it (you should remove "stl::" in some lines).

B Quick start: An example

In this section we describe how to run the software using a simple example. To start the demo you have to execute the *main.tcl* file in the Tcl folder. Select *File-Open* and choose the file */ESQUI/Tcl/Scenes/SceneFormVTK/FormVTK.srml*. In figure 3, we show the software's snapshots while user are running this quickstart.

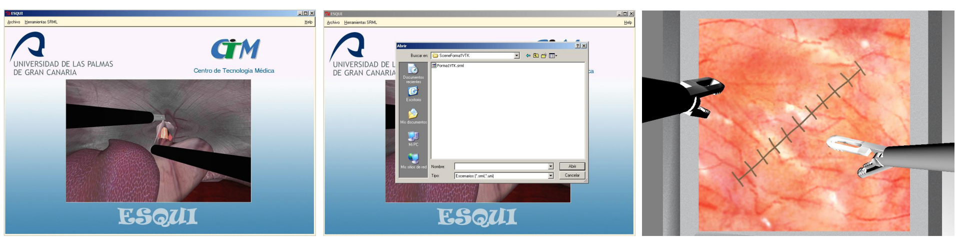

Figure 3: Snapshots's sequence while user is opening a SRML file.

B.1 Keyboard commands

If the haptic device is not used, the surgical simulation can be driven with the computer keyboard and mouse. Press "C" before "D"(right tool) or "I" (left tool) to set one of the surgical instruments. The mouse's right button controls the deep of the selected tool like the traditional VTK interactor. The mouse's left button controls the lateral movement. "Z" and "X" letters allow rotate the selected tool, and "A" and "S" opens and closes it respectively.

References

[1] *Blender*. http://www.blender.org. 1.1, 1.2

[2] *Dart*. http://public.kitware.com/dart. 1.1

[3] *Doxygen*. http://www.stack.nl/ dimitri/doxygen. 1.1

[4] *GNU Lesser General Public License*. http://www.opensource.org/licenses/lgpl-license.php (last visit 2006-07-05). 1.4

[5] *Spring*. http://spring.stanford.edu/. 2

[6] *Subversion*. http://www.subversion.org. 1.1

[7] *Tcl/Tk*. http://tcl.tk. 1.1

[8] *VTK and Blender*. http://www.ualberta.ca/AICT/RESEARCH/Vis/VTKBlender/index.html (last visit 2006-07-05). 1.2

[9] *vtkBioeng*. http://sourceforge.net/projects/vtkbioeng. 2

[10] Alberola C. Cárdenes R. Martín M. Martín M. A. Rodriguez-Florido M.A. and Ruiz-Alzola J. diSNei: A collaborative environment for medical images analysis and visualization. *Lecture Notes in Computer Science*, 1935:814–823, 2000. 1.2

[11] N. Sánchez Escobar. *Diseño e Implementación de un Simulador Quirúrgico en Cirugía Mínimamente Invasiva*. Master Thesis - Universidad de Las Palmas de Gran Canaria, Las Palmas, Spain, 2006. 1

[12] Martin K. and Hoffman B. *Mastering Cmake*. Kitware Inc. Clifton Park NY, ISBN 1-930934-09-2. Schroeder and K. Martin and B. Lorensen, 2003. 1.1

[13] Nicola Di Lorenzo and Jenny Dankelman. Surgical training and simulation. In *Business Briefing: Global Surgery - Future Directions: www.touchbriefings.com/pdf/1438/ACF535.pdf*, 2005. (document)

[14] C. Monserrat. *Modelos Deformables de Tejidos Elsticos en Tiempo Real*. PhD thesis, Universidad Politécnica de Valencia, Valencia, Spain, 1999. 1.4

[15] R. Aggarwal K. Moorthy and A. Darzi. Laparoscopic skills training and assessment. *Br J Surg, 91(12)*, 91(12):1549–1558, 2004. (document)

[16] Mark Pilgrim. *Dive Into Python - http://diveintopythong.org*. 2004. 1.1

[17] C. Laugier R. Rodriguez and K. Sundaraj. *Multimodal and incremental modelling of space and motion:ColDetect Library*. http://www.inria.fr/rapportsactivite/RA2003/e-motion2003/module7.html (last visit 2006-07-05). 1.4

[18] W. Schroeder, K. Martin, and B. Lorensen. *The Visualization Toolkit, An Object Oriented Approach to 3D Graphics*. Kitware Inc, 1998. 1.1

[19] W. Schroeder, K. Martin, and B. Lorensen. *The VTK User's Guide, Version 4.4*. Kitware Inc - ISBN 1-930934-08-4, 2003. 1.1

[20] Slicer. *http://www.slicer.org*. 1.2

[21] K. Erleben J. Sporring and H. Dohlmann. Opentissue-an open source toolkit for physics-based animation. *InsightJournal - ISC/NA-MIC/MICCAI Workshop on Open-Source Software*, 2005. 2