A Visualisation Platform for Shoulder Replacement Surgery

Charl P. Botha and Frits H. Post

Computer Graphics and CAD/CAM group, Faculty of Information Technology and Systems
Delft University of Technology, The Netherlands
{c.p.botha,f.h.post}@its.tudelft.nl

Abstract. This note presents a new software platform that has been designed to enable the integration and investigation of visualisation in multiple facets of the shoulder replacement process. What distinguishes this platform from available CAS (Computer Assisted Surgery) software solutions is a combination of its focus on visualisation (and specifically of the shoulder), its flexible and scalable architecture (without loss of domain-specific benefits) and the planned integration of predictive modelling.

1 Introduction

The DIPEX (Development of Improved endoProstheses for the upper EXtremities) program[1] is a clinically driven, multidisciplinary research effort that aims to improve radically joint replacement surgery of the upper limb. Cooperating parties include several research groups of the Delft Technical University, the Orthopaedics Department of the Leiden University Medical Centre, the Free University of Amsterdam and the Erasmus University of Rotterdam.

As an important component of DIPEX, the use of visualisation for joint replacement surgery is being studied. In order to facilitate the prototyping of new visualisation paradigms as well as the integration of existing methods with the orthopaedic surgical process, a new software platform was required. Existing solutions were either general purpose visualisation packages (e.g. IBM OpenDX, Application Visualization System[1]) or purpose-designed CAS (Computer Assisted Surgery) and bio-mechanical modelling solutions (e.g. Integrated Surgical Systems' Orthodoc[2], the orthopaedic modules of Brainlab VectorVision[3] and SIMM[4]).

With general purpose visualisation packages, the lack of domain-specific refinements limits use to visualisation experts: this makes clinical deployment for testing purposes very difficult. On the other hand, more specialised CAS solutions focus on per-operative assistance with very little pre-operative planning facilities and almost no predictive modelling capabilities. At the time of writing, there are no commercially available CAS packages and very few bio-mechanical modelling packages that deal with the shoulder joint. In addition, purpose-designed solutions (CAS and bio-mechanical) in general do not offer state-of-the-art visualisation capabilities nor are they accessible for easy extension. These factors render them unsuitable as experimental platforms for fundamental research.

The DSCAS1 (Delft Shoulder Computer Assisted Surgery) platform, attempts to fill this vacuum. It is a cross-platform, extensible and scalable software platform for the prototyping and integration of visualisation methods in the shoulder replacement process. In order to capitalise maximally on this visualisation research, the platform is also being designed to integrate various kinds of predictive modelling, such as the Delft Shoulder Model.[5]

2 Functionality

2.1 Basic visualisation

Fundamentally DSCAS1 assists with the direct visualisation of regular grid medical volume data, e.g. CT or MRI modalities. It is able to read both DICOM[6] and HDF[7] (Hierarchical Data Format) datasets. It allows the user to explore volumes by making use of fast orthogonal slice viewers. With these slicer viewers, the user can "move" along any of the orthogonal directions and have the displayed slices continuously and instantaneously updated.

[1] See http://www.wbmt.tudelft.nl/mms/dipex/index.htm.

To further help the user orient herself with regards to the dataset, these orthogonal slices are also shown with their correct positions and orientations in a wire-frame cube representing the physical extents of the dataset. Figure 1 depicts three orthogonal slice viewers and the cube viewer.

As is expected of most medical visualisation software, the platform has hardware-accelerated surface rendering support for viewing arbitrary polygonal surface representations. Figure 2 has an example of two model viewers in action.

DSCAS1 also has direct volume rendering capabilities: it is able to perform real-time ray-casting directly on the volume data in order to yield interactive 3D visualisations without traditional surface reconstruction. This functionality is implemented as a plug-in. Plug-ins are discussed in section 3.2 and an example of a volume rendering can be seen in figure 3.

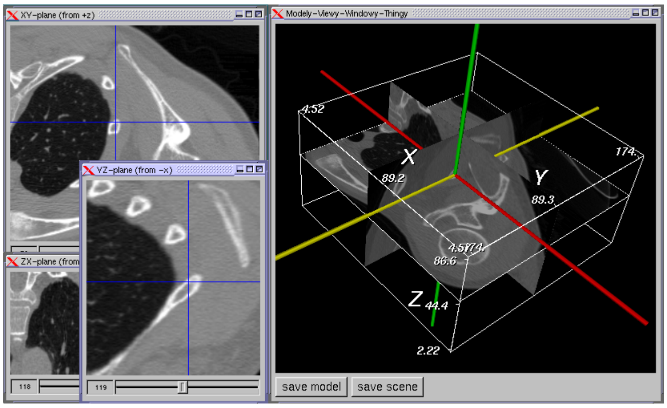


Fig. 1. Three slice viewers and the cube viewer.

2.2 Surface reconstruction

Polygonal surface representations of bony structures in the CT data can be reconstructed easily by making use of the built-in 3D region growing and surface extraction routines. After the user has picked a seed-point and thresholds, the platform automatically segments the connected bone structure. This segmentation can then be used to reconstruct a 3D surface. The user has control over several extraction parameters. Figure 2 shows an interactive rendering of a scapular surface that has been extracted from in-vivo CT data. The segmentation results for a single slice as well as a part of the reconstruction configuration dialogue can also be seen.

2.3 Assemblies

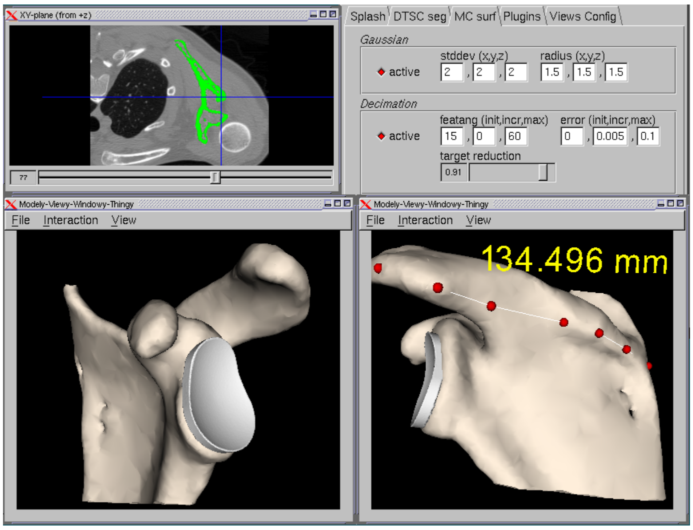
All reconstructed surfaces are collected in a central data entity pool (described in section 3.1) and can be used to create arbitrary assemblies of geometric structures (e.g. a shoulder structure consisting of a separately reconstructed humerus, scapula, thorax and clavicula). Externally generated geometric surface descriptions can also be added to the data entity pool, which makes for an elegant method of simulating prosthesis placement.

Traditionally, orthopaedic surgeons plan shoulder prosthesis placement by making use of an X-ray of the patient's shoulder as well as several transparencies of possible prosthesis CAD designs. These transparencies are then each overlaid (by the surgeon) on the X-ray until a satisfactory fit is found. This assists the surgeon with her decision on a type and configuration of prosthesis as well as its placement.

At the moment, DSCAS1 can provide this functionality extended to three dimensions. Because of the data entity pool architecture, a selected prosthesis CAD model can be placed in a reconstructed virtual shoulder. One important advantage is that the surgeon is not limited to basing her judgement on a 2D-projection of a 3D problem, but rather on a more information-dense manipulable 3D model. Figure 2 shows a glenoid component implanted in a reconstructed scapula.

2.4 Measurement

Arbitrary piecewise linear contours can be measured by clicking on consecutive points in either the slice viewer or a 3D model viewer. This assists the researcher or clinician in forming a quantitative interpretation of the patient shoulder under investigation. Figure 2 demonstrates this functionality where the scapular spine has been measured. The red spheres represent line segment nodes and the white lines the segments themselves. To facilitate measurement, the nodes automatically jump to the nearest bone (or object) surface when initially placed. However, a node can be freely moved after having been placed.



77

Gaussian

◆ active stddev (x,y,z) 2 , 2 , 2 radius (x,y,z) 1.5 , 1.5 , 1.5

Decimation

◆ active featang (init,incr,max) 15 , 0 , 60 error (init,incr,max) 0 , 0.005 , 0.1

target reduction 0.91

Fig. 2. Scapular surface reconstructed from in vivo CT data shown with "implanted" glenoid component. On the right, this same assembly has been used to measure the length of the scapular spine.

3 Architecture

3.1 Conceptual

The design is centred around a repository of data objects called the *data entity pool*. The platform can operate on any combination of available data entities to modify them in-place, to create new data entities or to present them to the user in some fashion. Data entities can obviously be loaded into the pool directly from secondary storage. All presentation and user-interaction takes place via *views*. A single data-entity can be presented using multiple differently configured views at the same time.

This very closely resembles the Document-View architecture that originated from the SmallTalk Model-View-Controller architecture[8] and was popularised by the Microsoft Foundation Classes. In the Document-View architecture the View and Controller objects are unified.

3.2 Plug-in System

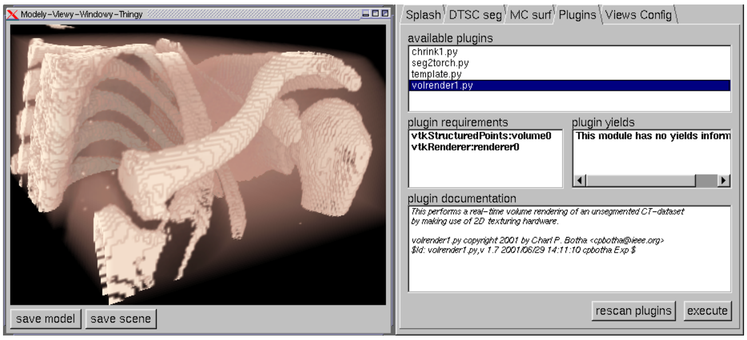
This architecture very clearly separates the data from the methods that operate on it. This means that by exposing the platform's data structures to the user, we can allow her to modify and create, during run-time, methods that operate on these data structures. In short, this is DSCAS1's plug-in system.

This plug-in system was implemented by embedding the Python[9] scripting language. Algorithms can be programmed in Python and executed within the platform infrastructure, i.e. they have access to all internal data entities and are able to create new entities in the data entity pool. In addition, there is no compile-link cycle involved: scripts can be modified and instantaneously re-executed in a single execution session of the platform. This is a very powerful method of rapidly developing, prototyping and even deploying new ideas and algorithms.

Figure 3 shows a real-time volume rendering of a human shoulder as performed by a DSCAS1 Python plug-in.

3.3 Low-level

DSCAS1 makes use of several freely available open-source libraries. Probably most prominent of these, VTK[10] is used as the visualisation back-end. Development takes place primarily on Linux, but the



available plugins

chrink1.py
seg2torch.py
template.py
volrender1.py

plugin requirements

vtkStructuredPoints:volume0
vtkRenderer:renderer0

plugin yields

This module has no yields inform

plugin documentation

This performs a real–time volume rendering of an unsegmented CT–dataset by making use of 2D texturing hardware.

volrender1.py copyright 2001 by Charl P. Botha <cpbotha@ieee.org>
$Id: volrender1.py,v 1.7 2001/06/29 14:11:10 cpbotha Exp $

Fig. 3. The plug-in system UI and output of the volume rendering plug-in. This volume rendering is done with the relatively standard 2D texture-mapping method.

software has been designed to be platform-independent. Any available graphics hardware is well-utilised even for functionality other than traditional accelerated polygon rendering. For instance, the slice viewers make use of the interpolation functionality in available texturing hardware to afford very fast image zooming.

4 Conclusions and Future work

A new platform for the study of visualisation in the shoulder replacement process was developed. This paper briefly discussed the current functionality of this software and its architecture. Important properties of the platform are: It is uniquely dedicated to the development and application of visualisation in the shoulder replacement surgery process; Its particular form of extensibility alleviates the often-encountered conflict between the requirements of a specialised application and those of a flexible and open architecture; It is being designed to integrate with advanced predictive modelling for surgery planning.

In the near future, an investigation will be made into the possibilities of optimising or approximating the Delft Shoulder Model so that it can be used interactively in DSCAS1. This would mean that a researcher or surgeon would be able to make changes to a specific shoulder configuration graphically (e.g. relocating a muscle force line) and interactively see the the results as predicted by the shoulder model. Other types of predictive modelling (e.g. finite element analysis of prosthesis-bone interfaces) are also to be integrated with the platform.

In general, the platform will be used to investigate many issues w.r.t. novel roles of visualisation and modelling in the shoulder replacement process. In the course of these investigations, the software will evolve to allow a broader range of visualisation methods and other algorithms.

References

1. C. Upson, T. Faulhaber, D. Kamins, D. Laidlaw, D. Schleigel, J. Vroom, R. Gurwitz, and A. van Dam, “The Application Visualization System: A Computational Environment for Scientific Visualization,” *IEEE Computer Graphics and Applications*, pp. 30–42, July 1989.
2. P. Kazanzides, B. Mittelstadt, B. Musits, W. Bargar, J. Zuhars, B. Williamson, P. Cain, and E. Carbone, “An integrated system for cementless hip replacement,” *IEEE Engineering in Medicine and Biology Magazine*, vol. 14, pp. 307–313, May-June 1995.
3. H. Gumprecht, D. Widenka, C. Lumenta, R. Maciunas, G. Barnett, P. Kelly, and D. Roberts, “BrainLab VectorVision neuronavigation system: Technology and clinical experiences in 131 cases,” *Neurosurgery*, vol. 44, no. 1, pp. 97–105, 1999.
4. S. L. Delp and J. P. Loan, “A Computational Framework for Simulating and Analyzing Human and Animal Movement,” *IEEE Computing in Science and Engineering*, vol. 2, pp. 46–55, September/October 2000.
5. F. C. van der Helm, “A finite element musculo-skeletal model of the shoulder mechanism,” *Journal of Biomechanics*, vol. 27, no. 5, pp. 551–569, 1994.
6. National Electrical Manufacturers’ Association, “NEMA Standard PS3: Digital Imaging and Communications in Medicine (DICOM),” 2000.
7. NCSA University of Illinois at Urbana-Champaign, *HDF Reference Manual*, December 2000. `ftp://ftp.ncsa.uiuc.edu/HDF/HDF/Documentation/HDF4.1r4/Ref_Manual/`.
8. G. Krasner and S. Pope, “A Description of the Model-View-Controller User Interface Paradigm in the Smalltalk-80 system.,” *Journal of Object Oriented Programming*, vol. 1, no. 3, pp. 26–49, 1988.
9. G. van Rossum, *Python Reference Manual.* Python Software Foundation, April, 2001.
10. W. Schroeder, K. Martin, and B. Lorensen, *The Visualization Toolkit.* Prentice Hall PTR, 2nd ed., 1999.