

The multimod application framework: A rapid application development tool for computer aided medicine

Marco Viceconti^a, Cinzia Zannoni^b, Debora Testi^a, Marco Petrone^b,
Stefano Perticoni^a, Paolo Quadrani^b, Fulvia Taddei^{a,*},
Silvano Imboden^b, Gordon Clapworthy^c

^a Laboratorio di Tecnologia Medica, Istituti Ortopedici Rizzoli, Via di Barbiano, 1/10, 40136 Bologna, Italy

^b C.I.N.E.C.A. Supercomputing Centre, Italy

^c University of Luton, United Kingdom

ARTICLE INFO

Article history:

Received 21 June 2006

Received in revised form

19 September 2006

Accepted 20 September 2006

Keywords:

Computer-aided medicine

Data fusion

Application framework

ABSTRACT

This paper describes a new application framework (OpenMAF) for rapid development of multimodal applications in computer-aided medicine. MAF applications are multimodal in data, in representation, and in interaction. The framework supports almost any type of biomedical data, including DICOM datasets, motion-capture recordings, or data from computer simulations (e.g. finite element modeling). The interactive visualization approach (multimodal display) helps the user interpret complex datasets, providing multiple representations of the same data. In addition, the framework allows multimodal interaction by supporting the simultaneous use of different input-output devices like 3D trackers, stereoscopic displays, haptics hardware and speech recognition/synthesis systems. The Framework has been designed to run smoothly even on limited power computers, but it can take advantage of all hardware capabilities. The Framework is based on a collection of portable libraries and it can be compiled on any platform that supports OpenGL, including Windows, MacOS X and any flavor of Unix/Linux.

© 2006 Elsevier Ireland Ltd. All rights reserved.

1. Introduction

Information and communication technology (ICT) is widely used in healthcare, but mostly to manage the flow of administrative information that is required to operate today. In contrast, the penetration of information technology as a support to prevention, diagnosis, planning, treatment, monitoring and rehabilitation is much less significant. One reason may be that *computer aided medicine* (CAM) suffers from high software costs. A generalized approach to software development does not succeed, as CAM applications are effective only when they

are tightly focused [1]. Thus, CAM requires the availability of a wide variety of highly specialized applications, which obviously reduces the potential market size for each individual application. This combined with the fact that the applications usually require very sophisticated user interfaces, contributes significantly to the spiraling software development costs.

Other ICT domains have faced similar problems and, in all cases, one solution was found to be the development of rapid application development (RAD) environments. A typical example is that of virtual reality, where toolkits and

* Corresponding author. Tel.: +39 051 6366864; fax: +39 051 6366863.

E-mail address: taddei@tecnio.ior.it (F. Taddei).

0169-2607/\$ – see front matter © 2006 Elsevier Ireland Ltd. All rights reserved.

doi:10.1016/j.cmpb.2006.09.010

application frameworks are becoming quite popular: DIVERSE¹ and CAVELib² are cross-platform APIs for developing virtual reality applications; Vega Prime³ is a VR toolkit; VR Juggler⁴ is an integration platform.

In contrast, very little is available for CAM applications. A few medical image-processing frameworks have been proposed as tools also to develop CAM solutions [2,3]. Recently, a RAD for medical image processing based on choreographed web services was described [4]. The Image-Guided Surgery Toolkit (IGstk),⁵ a high-level component-based framework providing common functionality for image-guided surgery applications [5] is explicitly focused on computer-aided surgery.

The Julius framework⁶ provides a general software development environment for rapid application production [6]. The Julius object-oriented architecture can be extended with specific plug-ins, and the framework comes with a set of standard plug-ins such as a DICOM 3.0 loader, 2D slice views and 3D volume and surface visualization. Components for basic manual segmentation algorithms, measurement tools, and drivers to hook up optical instrument tracking or haptic devices are reported to be under development. While elegant in its architectural design, the Julius system appears rather incomplete and strongly focused on computer-aided surgery.

Thus, at the current state of the art, there is no widely available tool that allows the rapid development of CAM applications that possess both the degree of specialization necessary for their effectiveness and all of the features demanded by the end users.

Faced with this situation, a group of researchers working on the Multimod project, which was supported by the European Commission, designed and developed the *multimod application framework* (MAF) [7], an open source application framework for computer aided medicine. The aim of this paper is to provide an in-depth description of the MAF, illustrated by example applications in which it has been successfully applied.

2. The multimod application framework

The MAF provides a three-layered set of software libraries that different users can exploit to realize a disparate range of software systems.

A first possible use case is that of a researcher who wants to make a specialized algorithm available to the users. The researcher will be highly motivated to ensure that his/her algorithm is put into use but will not generally want to waste time in developing those parts of the application that are irrelevant to his/her own research but, nonetheless, must be present for the user to utilize the software. These items could include import filters, GUIs, efficient visualization pipelines, etc.

The second use case is that of an application expert who wants to develop a specialized computer aided medicine application targeting a specific medical diagnostic, treatment or rehabilitation service. This user will want to focus on the application and will want to develop it as quickly, doing as little as possible in terms of implementation.

The third use case is that of the final user. Here, it is necessary to distinguish between the medical technician, who will normally be involved in data preparation, and the medical professional, who will want to focus on the specific medical service to be performed. The technician usually has some understanding of the technical issues involved with the elaboration of digital data, and wants a powerful and productive user interface to help with preparing and organizing the data. The medical professional, on the other hand, requires a user interface that is as familiar as possible, so he/she can focus on the task, rather than on the application.

From these definitions of the scope and uses of the MAF, a four-layer architecture was derived (Fig. 1).

2.1. Multimod foundation layer

The lowest level, called the *multimod foundation layer* (MFL), can also be considered as a software distribution—a collection of some of the best open source libraries available today, assembled into a coherent set. All visualization services are based on the Visualization Tool Kit (VTK) [8].⁷

Registration and segmentation services are derived from the Insight Tool Kit (ITK).⁸ Other specialized libraries provide collision detection, encryption, etc. All the other core services provided by the MAF are contained in a library that we developed, which (for historical reasons) is based on the architecture of VTK, and is coherently named SVTK. The MFL is usually accessed directly only by those programmers who are developing and maintaining the MAF itself.

2.2. Low abstraction layer

The *low abstraction layer* (LAL) provides the fundamental elements of each MAF application: user interface, views, operations, interactors, and *virtual medical entities* (VMEs).

The VMEs are the data objects. Each VME may contain a block of metadata, a binary object dataset, and a pose matrix that defines the position and orientation of the dataset with respect to its parent VME in the hierarchy. Each VME type is associated with a list of visualization pipelines that can process it, a serialization method and, in some cases, with a data processing pipeline that can manipulate the dataset in real time. The *VME tree* is a hierarchical tree of VMEs that defines the data structure.

In the default application, the VME tree is portrayed on the right of the screen (Fig. 2). Because of the relative posing mechanism, each VME can be translated, rotated, and scaled independently from its type. Those few Views whose performance would be drastically impacted by the fact that the dataset is arbitrarily oriented with respect to the View camera,

¹ <http://diverse-vr.org>.

² <http://www.vrco.com/CAVELib/OverviewCAVELib.html>.

³ http://www.multigen-paradigm.com/products/runtime/vega_prime.

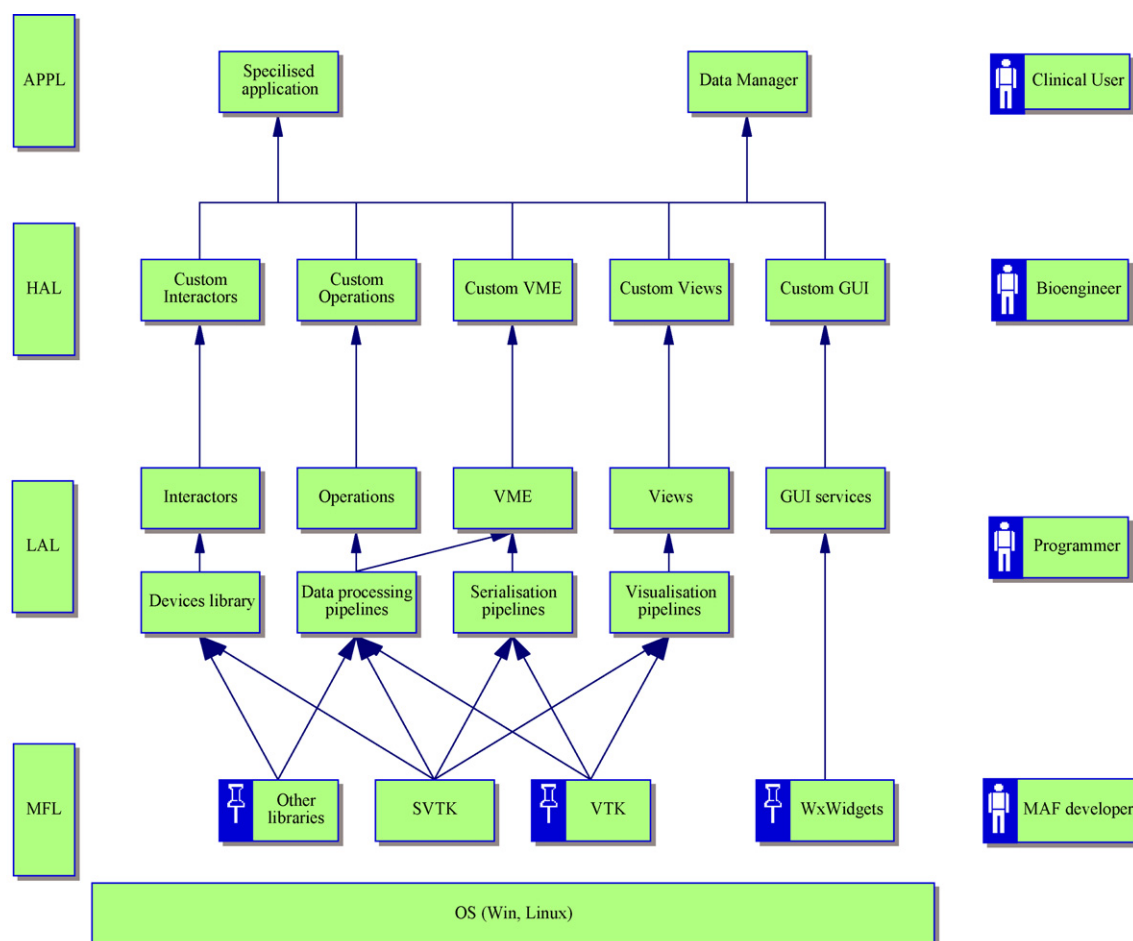
⁴ <http://www.vrjuggler.org>.

⁵ <http://www.igstk.org>.

⁶ <http://www.julius.caesar.de>.

⁷ <http://public.kitware.com/VTK/>.

⁸ <http://www.itk.org>.



90

Fig. 1 – General architecture of the multimod application framework. For each layer of the architecture indicated on the left, the type of user most likely to exploit it is indicated on the right.

are defined as Local, e.g. they can render only one VME of that type at a time; the camera is then re-oriented always to be properly aligned with the VME reference system.

The *Interactors* are used to integrate non-conventional input devices into MAF applications. The default application uses the mouse as interactor; however, it is possible to control the application with 6-DOF trackers (3D mouse, trackball, tracking gloves, or any rigid object tracked in stereophotogrammetry), haptics, gesture recognition engines, etc. For example, the HipOpMS VR environment – a MAF application for the simulation of total hip replacement surgery, developed as part of the Multisense project [9] – can employ either a dual-haptic control with force feedback, or a dual-hand control with stereophotogrammetry-tracked gloves to provide elementary gesture recognition [10]. In general, an Interactor is connected to a View.

A View is a particular visual representation of the data space. Depending upon its type, each VME is visualized in each view with a certain visualization pipeline. In this sense, a View can be seen as a visualization container, within which we associate with each data type, the visual representation that is to be used for it.

In any MAF application, it is possible to create as many Views as desired, of the same or different type, and in each

View, the user can choose to visualize any VME in the VME tree.

In general, a View should combine visualization pipelines that provide a coherent representation of the data space. So, for example, in the DRR (digitally reconstructed radiograph) View, the volume objects are transformed into 2D pseudo-radiographs, while surface objects are rendered with a flat renderer that creates the typical visual aspect of a strongly radio-opaque object on a radiograph. In this way, if a CT volume and a surface model of a surgical tool are visualized in this View, the result will be what the clinician would expect to see (Fig. 3).

This mechanism allows the user to define Views as visual representations of multiple data types in a way that is coherent for the final use. This metaphor is totally different from other scientific visualization programs, which are mostly based on the concept of the visualization pipeline. The user finds, in the Views, a pre-defined set of visualization pipelines, providing a coherent representation one to the other, which dramatically simplifies the user's workflow, an essential requirement for applications targeting non-technical users.

An *Operation* is a procedure that manipulates the VME. An Operation can: create a new VME; edit its pose matrix, its

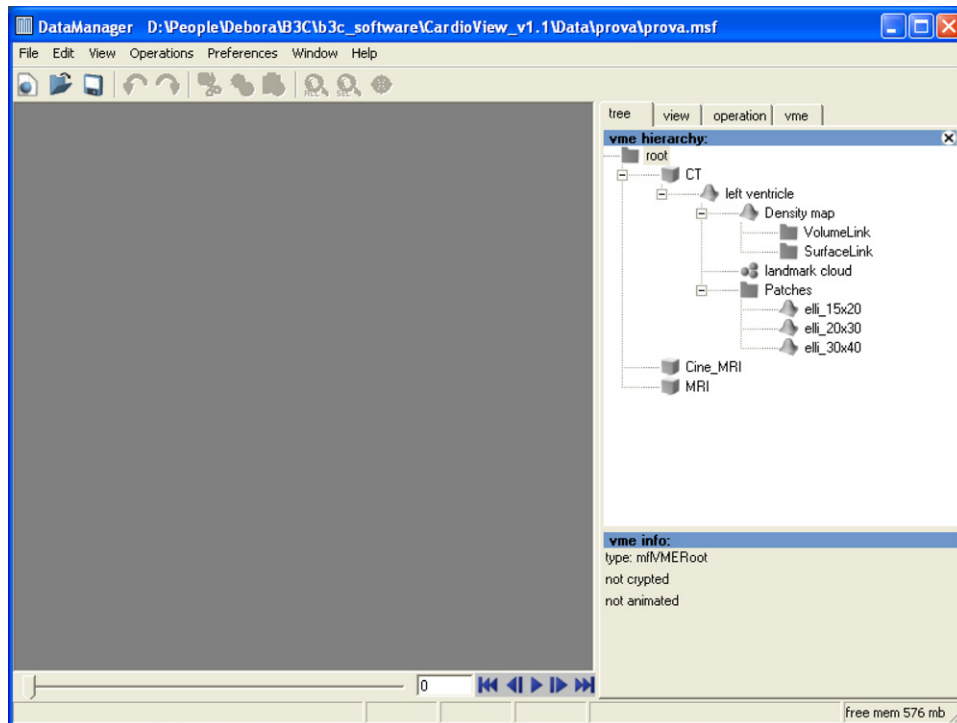


Fig. 2 – Default interface layout of a MAF application. The VME hierarchy box on the right shows the VME tree; other tabbed windows contain the controls for the selected view, the current operation, and the selected VME. The time bar is visible at the bottom.

metadata, or its binary object; change its position in the VME tree or its attributes (name, colors, etc.). The Operation can refer to a manipulation as simple as “translate VME”, to something as complex as a full-blown simulation engine.

An Operation typically relies on one or more data processing pipelines. However, certain operations can create a special VME type, called a *derived VME* (dVME), which contains the

data processing pipeline. In these VMEs, the dataset is the result of processing another dataset (usually that contained in another VME). A derived VME can be seen as a real-time Operation, and the building block for a dataflow pipeline in the VME tree. If the user creates an iso-surface dVME and connects it to a cardioCT volume, advancing the time bar the CT volume changes to the next frame, and the iso-surface dVME



Fig. 3 – Digitally reconstructed radiograph view of a CT scan volume and the polygonal surface model of a surgical implant. The visualization pipelines for the two different VME types are chosen to provide the user with a coherent representation.

Table 1 – List of the most important operations available in the current release of the MAF

Operation	Description
FEM surface	Creates a derived polygonal surface VME of the mesh external surface
Add FEM materials	Operation that import the material properties of the finite element model
Add single time fem results	Operation that import, and add to the FEM VME, the single-time result of a FEM simulation
Add multitime fem results	Operation that import, and add to the FEM VME. the multi-time result of a FEM simulation
Extract labels	Operation that extracts the 3D surface of the selected label
Blended slicer	Create an arbitrary plane that visualizes, as blended image, the FEM results on the CT dataset
Color volume	Operation that maps over a CT dataset the results of a FEM simulation
Labelize	Operation to labelize a medical imaging dataset on threshold base
VOI density	Operation that computes the average and maximum HU values of the voxels that are comprised in a closed surface
Create alias	An alias VME is simply a link to another VME present in the same tree
Filter surface	Clean-up a polygonal surface of double faces, dangling faces, wrong normals, etc.
Create group	Combine multiple Vme into a single object that can be moved, rotated and scale as a whole
Autofit	Computes the best zoom for a View that fits all or selected VME in the scene
ROM	Computes the range of motion between multiple articulating surfaces
Maps	Computes the scalar value distribution of a 3D image over a 3D surface placed inside it
Volume slicer	Creates a 2D image that is the real time calculation of the intersection between a plane and a volume
Create geometry	Creates a derived VME that generates a polygonal surface representation of parametric elementary solids
Create RefSys	Creates a reference system, with respect to which spatial transformations can be defined
Freeze	Derived VME update dynamically: this command create a new VME that free the VME data at a given instant
Open attributes window	Displays a special View that shows all meta information associated to a VME
Add note to VME	Each VME can be annotated in text or HTML
Open with external program	External data imported in the VME tree, such as an excel file, can be opened with this command
Extract isosurface	Generate an isodensity surface using an algorithm much faster than Marching Cube, which makes it interactive
Surface mirror	Mirror any surface VME
Volume resample	Operation that change volumes resolution
Clip surface	Operation that clip a surface with another surface
Crop volume	Operation that crop any volume with an arbitrary rectangle
Reparent	Operation that let you change the position of a VME in the VME tree, while retraining its position in space
Distance meter	Operation that places a linear element which dinamically computes the distance between two points
Svncro time	Operation to register time histories of two VME
Create joint centre	Place a landmark at the centre of a sphere of best fit
Register landmark clouds	Point based registration that uses single value decomposition
MAF transform	Powerful operation that let you translate, rotate and scale interactively any VME
2D measure	Small measurment toolbox that let you take 2D measurments on any View
Explode/collapse landmark cloud	Special grouping operation for landmarks
Register surface (icp standard)	Surface-based registration that uses the interactive closest point algorithm
Create landmark	Virtual palpation operation

is automatically recomputed on this new volume. In principle, by concatenating chains of dVMEs it would be possible to build very complex processing pipelines, although this approach has not been fully exploited so far. Table 1 summarizes the list of Operations that are currently implemented in the MAF.

All of these elements are wrapped into a user interface that is built using the wxWidget cross-platform GUI API.⁹ The Default application exposes, under Windows, an MDI user interface with a Menu Bar, a Tool Bar, and a tabbed column on the right. The remainder of the layout is free for the Views that the user can open and organize at will. The tabbed column provides one tab for each family of components: VME, Oper-

ation, View. The VME Tab shows the VME tree and provides a simple means to select and de-select the VME that is seen in a View. The Operation and the View tabs host the specific controls that each running Operation, or each selected View, may require (Fig. 4).

2.3. High abstraction layer

At its higher level, the framework provides a fully developed application logic, which is wrapped in the Logic class. Any MAF application is an instance of this class. The simplest way to build a MAF application is entirely to subscribe to this application Logic, and, when loading the VME types, the Operations and the Views that the application requires, use only those immediately available in the MAF. A new MAF application, fully based on the default MAF logic,

⁹ <http://www.wxwindows.org/>.

can thus be created with just the following few lines of code:

```
// — MyApp.h —
#include wxWindows.h
#include mafServices.h
class MyApp: public wxApp
{
void OnInit();
void OnExit();
mafLogic *m.Logic;
}
```

together with a few lines of code to define, in the initialization method, which Views and Operations the application should include:

```
// — MyApp.cpp —
#include MyApp.h
void MyApp::OnInit()
{
    m.Logic = new mafLogic();
    m.Logic->AddView (new mafV1());
    m.Logic->AddView (new mafV2());
    m.Logic->AddOperation (new mafOP1());
    m.Logic->AddOperation (new mafOP2());
    m.Logic->Show();
}
void MyApp::OnExit()
{
    delete m.Logic;
}
```

If a specialized application requires an application logic different from the default, the expert programmer can override one of more methods, or even entirely replace the Logic class with a brand new one. Of course, as in any framework, the efficiency of this development model is inversely proportional to the amount of modification from the default.

If the default Logic class is employed, and all VMEs, Views and Operations are loaded, what is obtained, after compilation, is an application called the Data Manager (DM). DM can act as a useful test bed for developers to perform early tests on a specific element of a MAF application they are creating. Further, because of its extensive capabilities, the application is already being used in some research labs for managing complex projects involving the fusion and the manipulation of large amounts of disparate biomedical data [11–14].

3. Examples of MAF applications

In this section, we shall briefly describe some current projects that use the MAF to develop their software tools. This section will also provide the opportunity to demonstrate the MAF in action in some relevant cases such as subject-specific analysis of the risk of skeletal fracture, a VR simulator for hip surgery, and a visualization console for a microCT device.

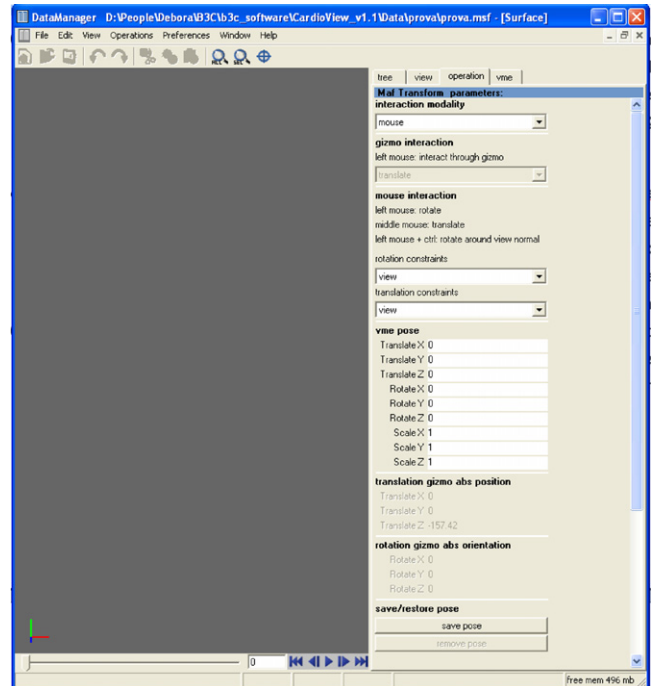


Fig. 4 – Default MAF application interface. On the right is visible the Operation tab, that is currently displaying the interface of the Transform, an operation that allows the user to change the position, orientation and scaling of each VME.

3.1. Subject-specific analysis of the risk of skeletal fracture

Orthopedic surgeons are sometimes faced with complex skeletal reconstructions because of trauma or because of bone tumors. The difficulty increases substantially when the patient is still growing—a child or an adolescent; in these cases, the surgeon cannot use conventional implantable devices, because they would not grow with the patient's skeleton.

A possible solution is a complex reconstruction that combines homologous transplants (bone grafts from cadavers) or autologous transplants (bone grafts taken from other regions of the patients' skeleton) with plate(s) and screws to obtain an initial mechanical stability. But the clinicians are then faced with another problem: to find the most intense rehabilitation program that will not produce a fracture of the reconstructed site. Unfortunately, due to the complexity of the reconstruction, no *a priori* knowledge is available on the strength of these reconstructed bones. Using the MAF, we have implemented a solution, called BuildOp, that allows the evaluation of the risk of fracture on reconstructed bone segments.

The patient undergoes CT and MR imaging of the reconstructed region and, immediately afterwards, a full gait analysis session. The imaging data are imported in BuildOp in DICOM format, while the movement data are imported in C3D format. The imaging data are collected while the patient is wearing radio-opaque reflective markers, which

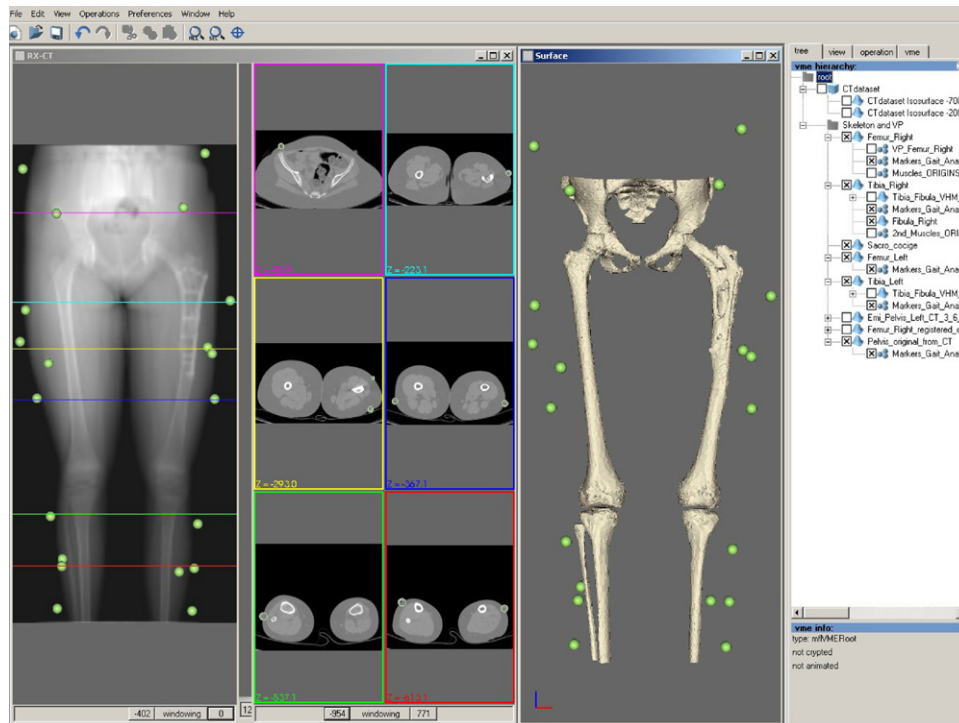


Fig. 5 – The radio-opaque markers are identified on the RX-CT view (left) and their relative position is shown with respect to the 3D model of the lower-limb skeleton (right).

are then tracked in the movement analysis session. This provides a framework of fiducial markers that are used to align the respective reference systems of the imaging devices and the motion-capture devices. The exact position of each marker is identified on each imaging dataset using the RX-CT view (Fig. 5). The same landmarks are located in the motion dataset, and a rigid registration operation based on single value decomposition is used to register the various data.

Bones are segmented from CT data, and muscles from MRI data. When (as is frequently the case) MRI data are not available, a virtual palpation tool (Fig. 6) allows the user to define the shape and position of each muscle from the CT images. A user-defined dictionary is visible on the right. This helps to assign the proper name to each anatomical landmark; the use of conventional names also simplifies the point-based registration with other data, such as movement analysis capture.

The most relevant skeletal landmarks are identified on the patient's skeleton (Fig. 7) and are then used to register a standardized atlas of muscles' origins and insertions on to the patient-specific skeleton morphology, using an affine registration algorithm.

Muscle volumes are used to estimate the lines of action and the physiological cross-sectional area of each muscle bundle. Bone surfaces, muscle lines of action, and landmark trajectories are exported to a multi-body dynamics application (ADAMS, MSC Software, USA) in which special algorithms perform the dynamic optimization required to register the bones to the motion (to allow for the skin artifacts) and to compute the muscle and joint forces. All results are imported back

in BuildOp using the ASCII importers for scalar and vector datasets.

Another possibility, currently under investigation, is to extract the muscle volumes directly from the CT images. A special tool, which was specially developed for this, is currently being validated (Fig. 8) [15].

Bone surfaces are also exported to a mesh generation program (Hyperworks, Altair Engineering Inc., USA) that is used to create finite element models of the reconstructed region and of the contralateral region (Fig. 9). The FE mesh is imported into BuildOp using the ASCII Mesh import filter; a special operation called *Bonemat* computes the local mechanical properties of the bone tissue for each finite element by performing a local integration over the CT data [16]. The FE model, subjected to the forces computed previously, is solved using a general-purpose solver (Ansys, Ansys Inc., USA); the stress and strain results are then imported back to BuildOp using the FE results import filter.

Special visualization tools superimpose the stress results on the imaging data, providing an ideal environment for engineer–clinician communication (Fig. 10).

The BuildOp application was developed over a long period of time, as the result of several internal research projects. The method still uses various specialized programs, since it would not make sense to try to replicate them inside the MAF. It is important to stress how, in this fashion, the MAF can provide an effective repository for all algorithms and methods developed in research projects, allowing them to be reused within a single coherent application, with obvious advantages in terms of usefulness and productivity.

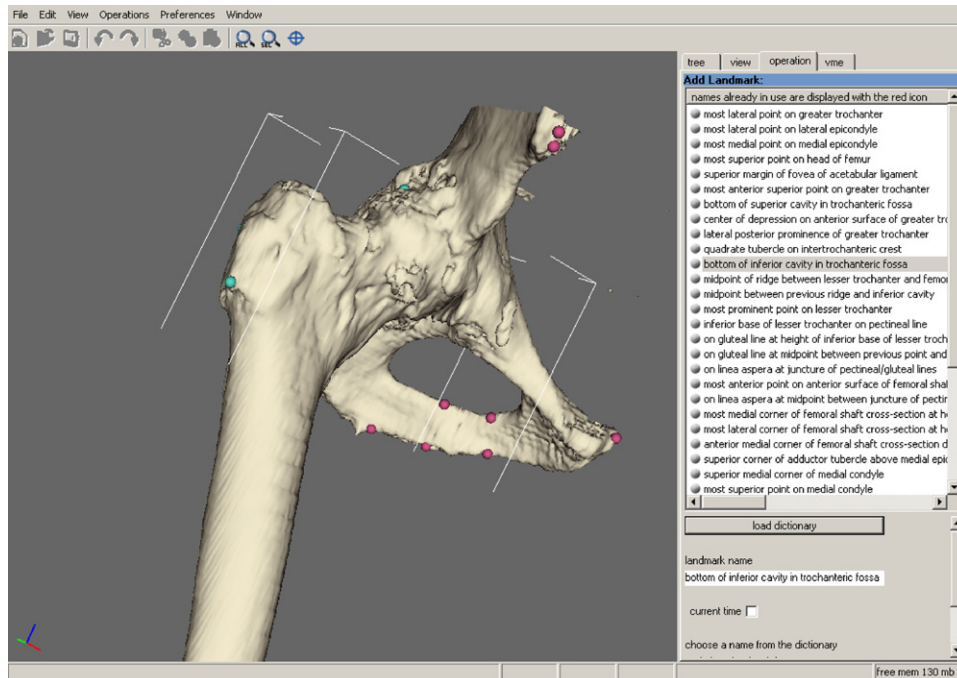


Fig. 6 – Virtual palpation operation—interactively placing point landmarks in correspondence to relevant anatomical features.

3.2. VR simulator for hip surgery

Specialized software applications are available to help surgeons in planning total hip replacement (THR) surgery

[17]. However, even when based on a CT scan, these do not normally take advantage of the intrinsic 3D nature of the data and of the interaction with the prosthetic components.

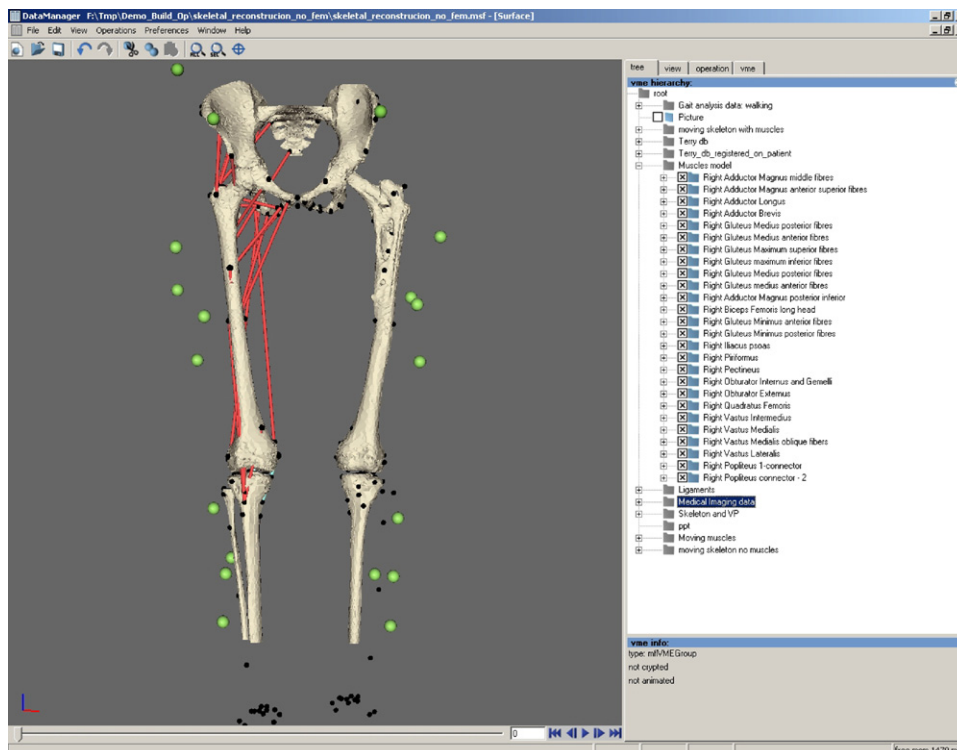


Fig. 7 – The muscle origin and insertion points were registered to the patient's skeleton (black dots) and then used to identify the muscle lines of action (right leg) from the skeletal landmarks that were virtually palpated on the bone segments.

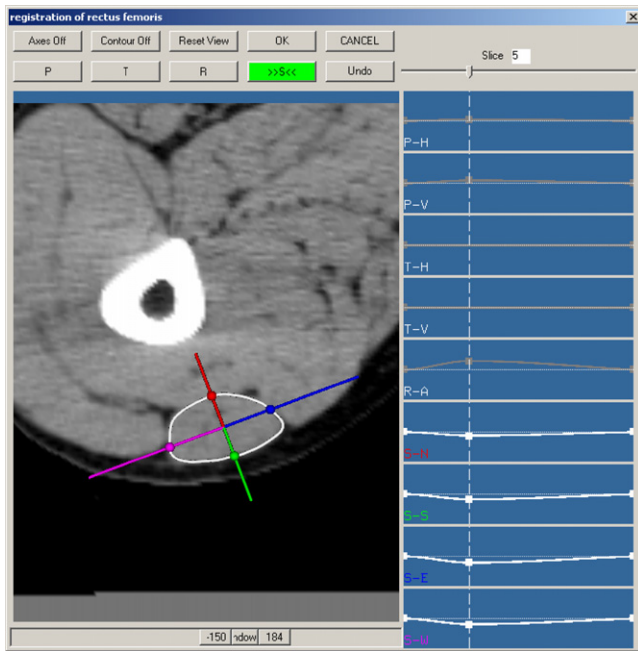


Fig. 8 – Interactive template-based elastic registration tool used to segment low-contrast 3D images; here it is being used for the segmentation of muscle volumes from CT images.

An innovative system for the pre-operative planning of THR, called HipOpMS, has been developed, based on the MAF (Fig. 11). HipOpMS is a multimodal/multisensorial system for THR planning which integrates technologies relevant to an immersive environment—stereoscopic display, different 6DOF tracking devices, speech recognition, haptic feedback, etc. The patient's data can be visualized using different forms of stereoscopic display, and the objects can be manipulated with a two-handed interaction, with one hand being used to control the camera and one hand to perform the current operation.

The software includes advanced functions for the planning of the surgical access and muscle retraction [18]. The surgeon marks the line of incision on the external surface by placing two landmarks and then moves the markers indicating the positions of the retractors to the desired positions, while feeling the skin surface with the haptic device. The system then illustrates the aperture thus created in the skin, allowing the user to inspect the muscle beneath (Fig. 12). It is possible to adjust the curvature of the free parts of the incision boundary to reflect different levels of tension being applied by the skin retractors. In Fig. 12, the tension values are high, so the edges of the incision are nearly straight.

When the planned incision is finalized, the surgeon must decide whether to cut or to retract each muscle on the route to the operation site. A retractor is provided (the green tool in Fig. 13) with which the user can grab each muscle in turn and retract it to try to expose the surgical region. During the muscle retraction, the user is given force feedback in order to understand if it is possible to retract the muscle sufficiently to achieve the acetabular capsule without causing undue tissue damage.

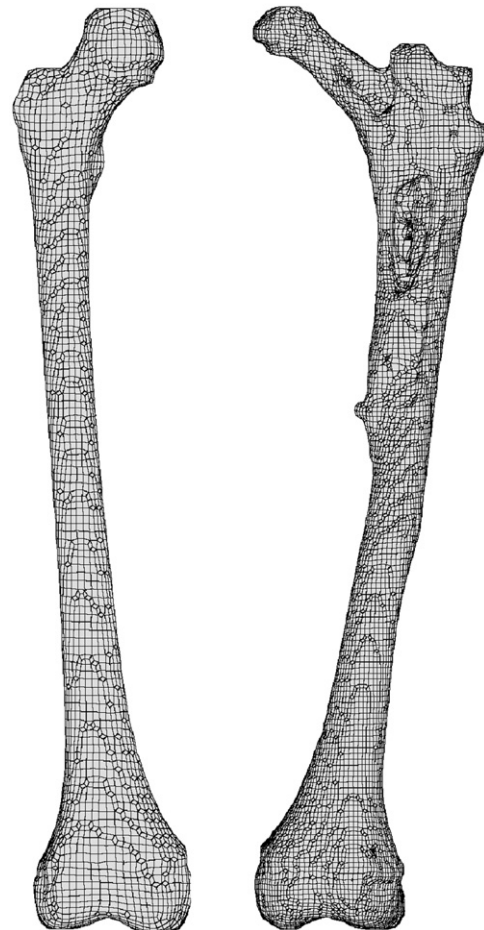


Fig. 9 – The finite element meshes of the operated and intact contralateral femurs.

In the subsequent steps, specialized functions are made available to position the prosthetic components in the patient's anatomy (changing, if necessary, the size and/or type of the component) with the help of an arbitrary slice of the CT volume. At any time during the positioning, the surgeon can check a series of functional indicators—extension of bone-implant contact (Fig. 14), range of motion of the operated joint (Fig. 15), limb lengthening/shortening on joint reduction, balancing of soft tissues, and alteration of articular muscle moment arms—to judge the likely outcomes.

The HipOpMS application was developed over a long period, mostly as a result of funding received from a European research project. It again illustrates how the MAF can provide an effective repository for all algorithms and methods developed in research projects, and also how it allows integration with non-conventional input/output devices.

3.3. Visualization console for a microCT device

The microCT application is an example on how the MAF can be used to develop an application in a very short time, without reducing the quality of the end product.

There was a necessity to build a bridge between a microCT scanner and a machine that realizes physical prototypes using

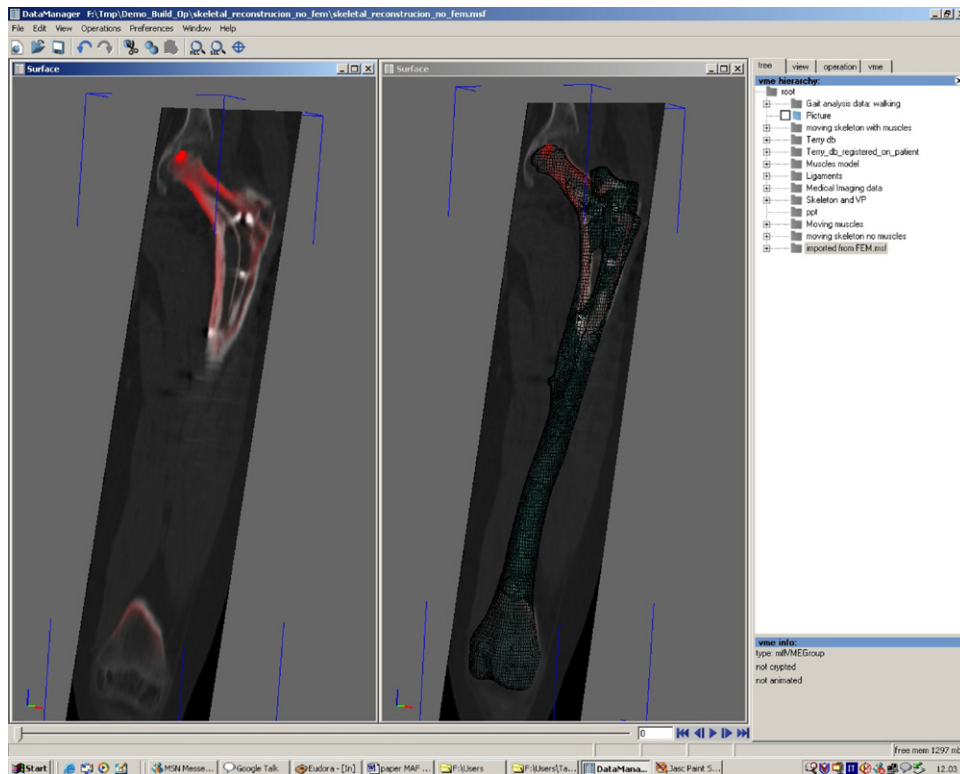


Fig. 10 – The risk of fracture, as predicted by the FE model, is mapped on to the CT dataset of the patient as a superimposed red field and can be viewed by cutting the CT dataset with an arbitrary slice. On the right, the FE mesh is shown in the same view. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of the article.)

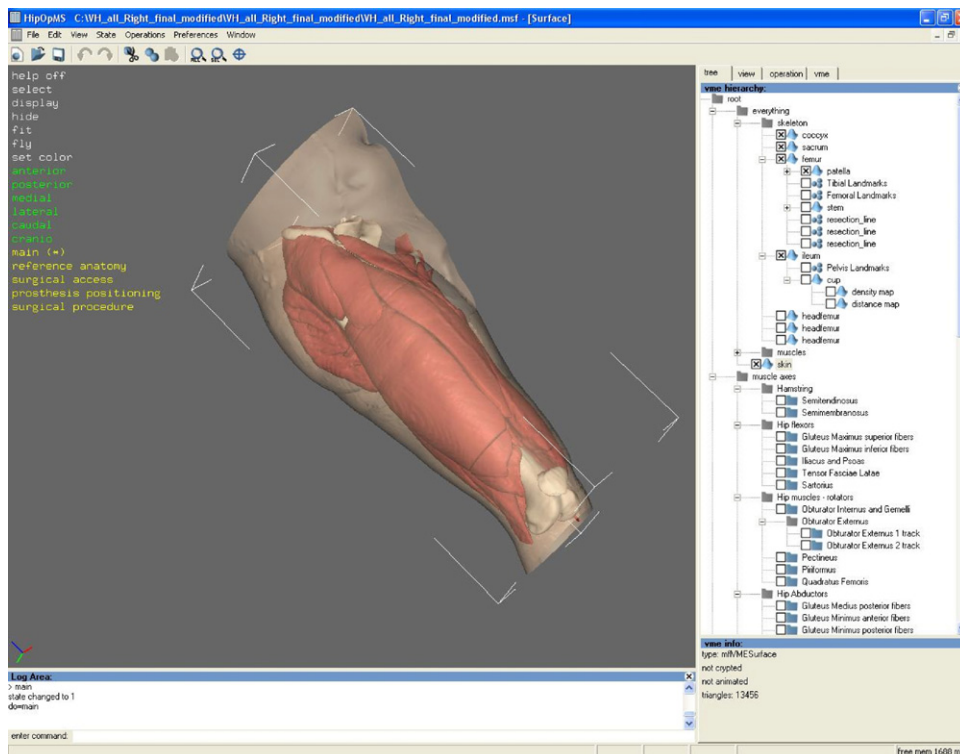


Fig. 11 – The HipOpMS software interface. The colored text commands on the left are designed to support the speech recognition control of the application.

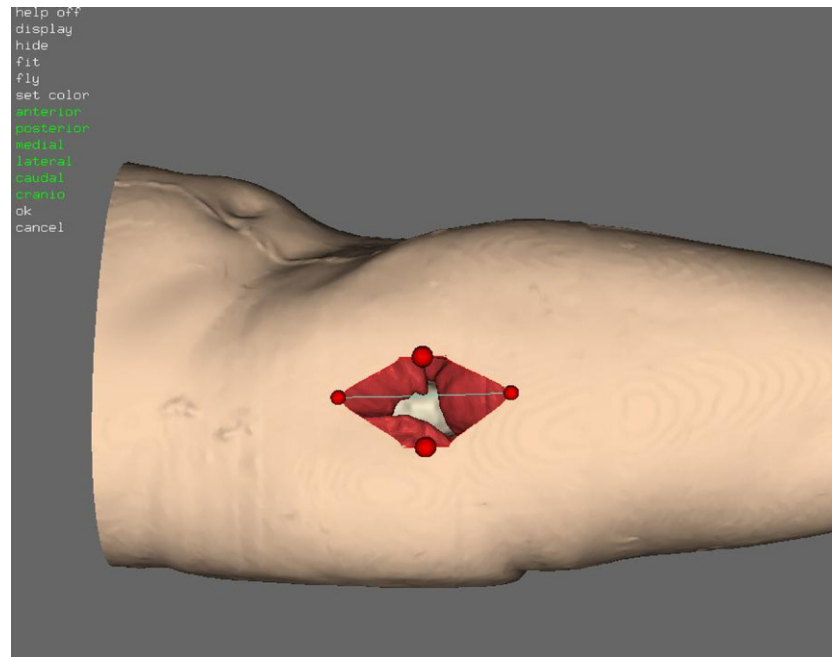


Fig. 12 – Definition of the skin incision. By moving the four landmarks, the user can define the size and position of the skin incision.

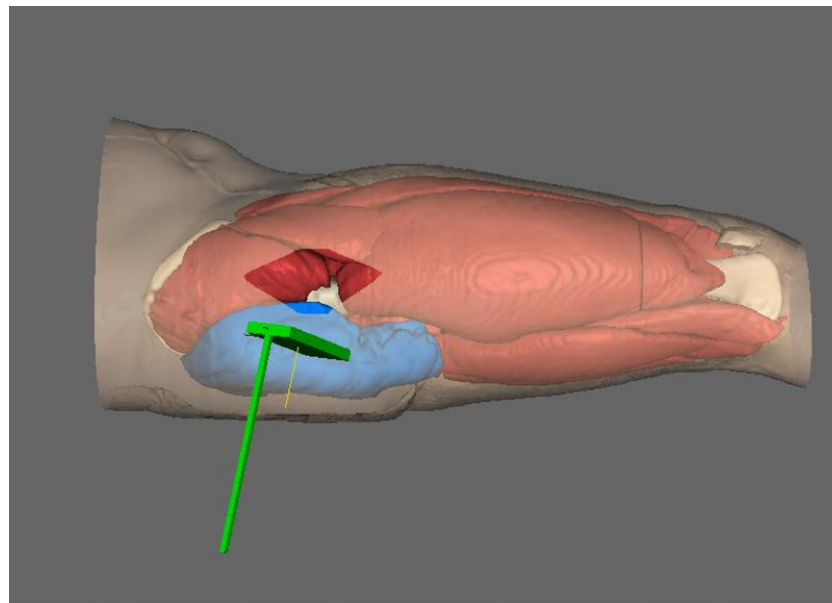


Fig. 13 – Muscle retraction. The skin is rendered semi-transparent to make visible the principal muscle groups.

stereolithography, including a visual preview of the volume imported from the CT scanner, based on the rendering of isodensity surfaces. All of these elements were already present in the MAF framework: a DICOM importer, an STL exporter and the operation for isosurface generation. In considerably less than one day's work, the application was realized (Fig. 16).

This was achieved by plugging the three operations needed into a vertical application, plugging a view for rendering the isosurface and unplugging interface elements not needed for the application, such as the time bar and log bar.

4. The road ahead

The MAF is now at the center of many major initiatives. The AneurIST consortium (EU project IST-2005-027703) will use the MAF to develop the next-generation technology for assessing the risk of rupture in cerebral aneurysms. The LHDL consortium will use the MAF to develop the data fusion and data visualization environment of the Living Human Project,¹⁰ a

¹⁰ <http://www.tecno.ior.it/VRLAB/LHP/>.

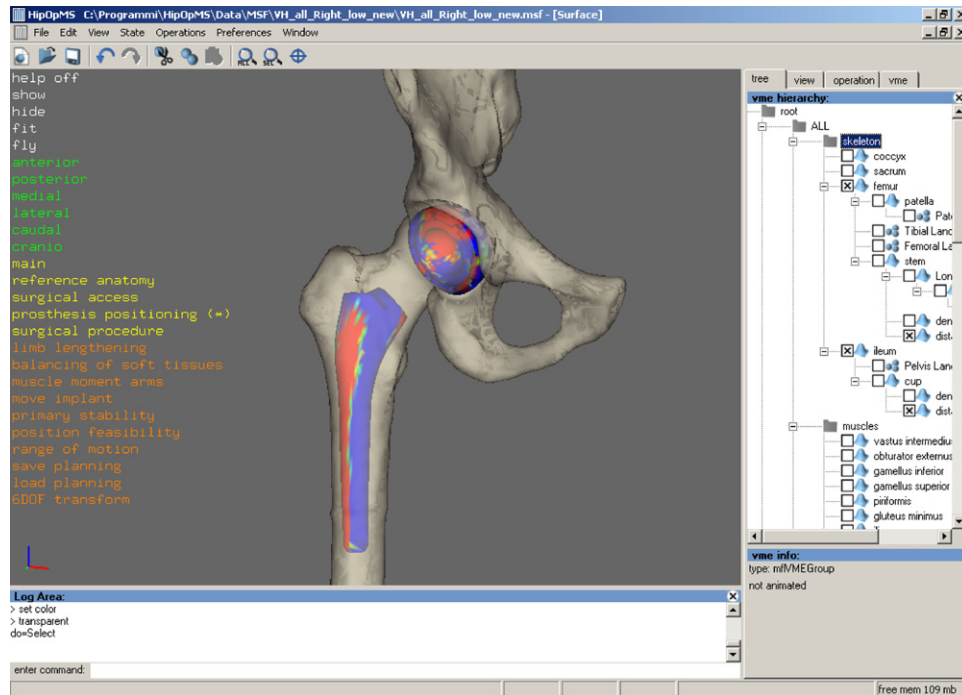


Fig. 14 – Density and distance maps. Depending upon the planned position for the prosthetic components the user can visualize, as a color map, the density of the bone tissue that is touching each point the implant surface, or the distance between the implant surface and the endosteal cortical wall. Both forms of information have great clinical relevance to deciding whether the chosen position is biomechanically adequate.

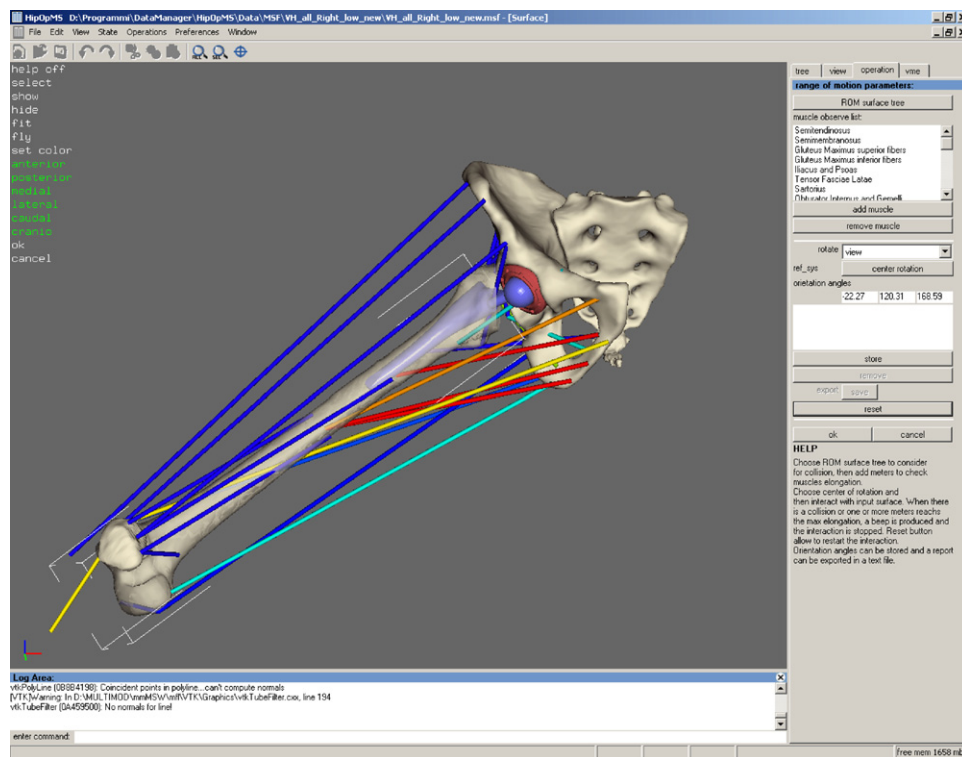


Fig. 15 – Range of motion evaluation including of muscle elongation. Once the planning is performed, the system simulates the physiological movements of the hip joint, while computing excessive elongations of the muscle fibers and eventual impingements between the prosthesis and the bones.

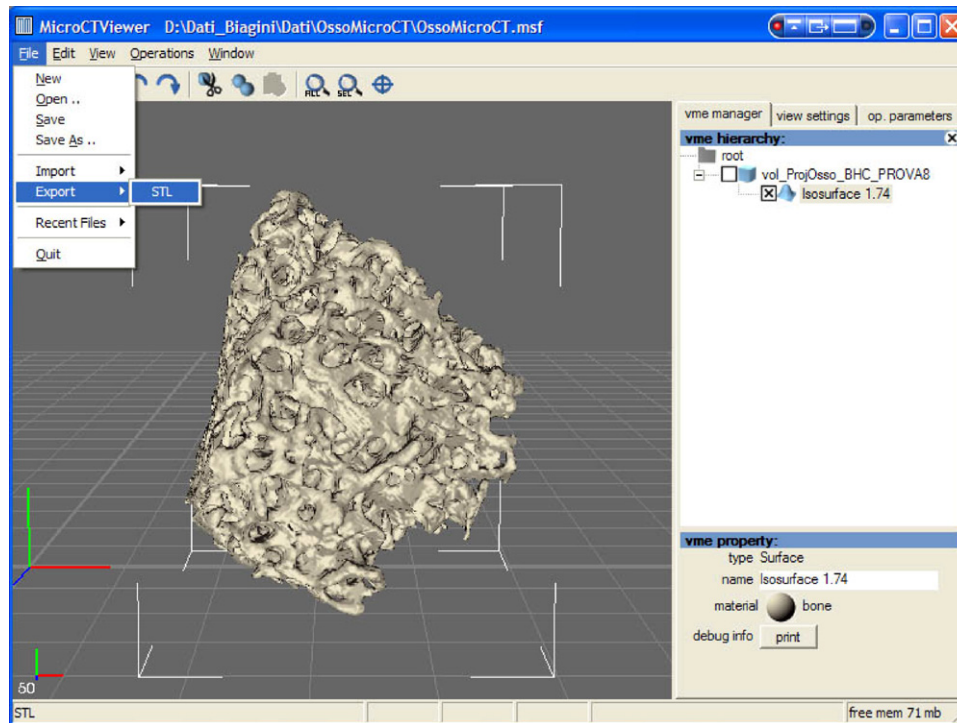


Fig. 16 – MicroCT application. The main view represents the isodensity surface of the microCT dataset generated from a fragment of human cancellous bone.

European initiative aimed at creating a worldwide repository for the creation of the musculo-skeletal physiome. B3C, a spin-off company of CINECA, is developing commercial applications based on the MAF for cardiac surgery planning, total hip replacement planning, and a solution for the visualization of high-resolution diagnostic images on portable devices. A number of research groups spread around the world have started to use the MAF and the Data Manager for their internal research projects.

Further, now that the framework has proved its capabilities and has become mature, we plan to position it in a more strategic perspective. The MAF is distributed under a BSD-like license agreement very similar to that under which VTK is distributed. This license allows every user to develop commercial applications using the MAF without any explicit authorization and without paying any royalty to the MAF developers. In addition, MAF is currently available as Open Source, and since various institutions own different portions of the code, it is unlikely these access conditions can be changed in the future. Last, but not least, MAF is based on lower-level libraries with a proven record for open source collaborative development, such as VTK or WxWidget.

For all of these reasons, MAF has the credibility to candidate itself as the repository of choice for all developers willing to contribute to a shared code base in computer aided medicine. Furthermore, the ability of MAF to load and merge almost any biomedical data, makes it an ideal means to share and exchange biomedical data.

The need for the creation of large repositories of algorithms and datasets is central to different grand vision of future

biomedical research, including the US BIRN,¹¹ the International Physiome Project,¹² the European Virtual Physiological Human initiative,¹³ the Living Human Project¹⁴ and the UK Integrative Biology project.¹⁵ All of these initiatives need large numbers of researchers in computational bioengineering and biomedical informatics to accumulate their algorithmic and data resources, so that these materials may begin to achieve the critical volume required by these grand challenges. In our opinion, the MAF can be a candidate tool for this accumulation process.

Thus, we hope that the MAF will continue to grow and improve over the next five years and that this paper can motivate other skilled developers to contribute to the code base of the Framework, and to use it in many new projects.

5. Mode of availability

The MAF is freely available at <http://www.openmaf.org>.

Acknowledgements

Here we shall thank many people who contributed to the realization of the MAF, among the others Alessandro Chiarini,

¹¹ <http://www.nbirn.net/>.

¹² <http://www.physiome.org.nz/>.

¹³ <http://www.europhysiome.org/>.

¹⁴ <http://www.livinghuman.org/>.

¹⁵ <http://www.integrativebiology.ox.ac.uk/>.

Luca Astolfi, Alexander Savenko, Mel Krokos, Hai Lin, Feng Dong, Nenad Filipovic and Yan Yang.

REFERENCES

- [1] M. Viceconti, R. Lattanzi, C. Zannoni, A. Cappello, Effect of display modality on spatial accuracy of orthopaedic surgery pre-operative planning applications, *Med. Inform. Internet Med.* 27 (2002) 21–32.
- [2] W. Birkfellner, K. Huber, A. Larson, D. Hanson, M. Diemling, P. Homolka, H. Bergmann, A modular software system for computer-aided surgery and its first application in oral implantology, *IEEE Trans. Med. Imaging* 19 (2000) 616–620.
- [3] R.A. Robb, D.P. Hanson, R.A. Karwoski, A.G. Larson, E.L. Workman, M.C. Stacy, Analyze: a comprehensive, operator-interactive software package for multidimensional medical image display and analysis, *Comput. Med. Imaging Graph* 13 (1989) 433–454.
- [4] T. Mitsa, P. Joshi, An evolvable, composable framework for rapid application development and dynamic integration of medical image processing web services, *Web Technol. Appl. Serv.* (2005) 7–12.
- [5] K. Cleary, L. Ibanez, S. Ranjan, B. Blake, Igstk: A Software Toolkit for Image-guided Surgery Applications, *CARS 2004*, Elsevier, 2004.
- [6] E. Keeve, T. Jansen, B. von Rymon-Lipinski, Z. Burgielski, N. Hanssen, L. Ritter, M. Lievin, Julius—a software framework for visualization, simulation and navigation in computer-aided-surgery, *Lect. Notes Comp. Sci.* 2673 (2003) 302–310.
- [7] M. Viceconti, L. Astolfi, A. Leardini, S. Imboden, M. Petrone, P. Quadrani, F. Taddei, D. Testi, C. Zannoni, The multimod application framework, in: *Eighth International Conference on Information Visualisation (IV'04)*, IEEE Computer Society Press, 2004, pp. 15–20.
- [8] W. Schroeder, K. Martin, B. Lorensen, *The visualization toolkit an object-oriented approach to 3d graphics*, Prentice Hall, Upper Saddle River, 1998.
- [9] D. Testi, C. Zannoni, G.J. Clapworthy, M. Viceconti, N.G. Tsagarakis, A multimodal and multisensorial pre-operative planning environment for total hip replacement, in: *Third International Conference on Medical Information Visualisation–BioMedical Visualisation*, IEEE Computer Society, 2005, pp. 25–29.
- [10] S. Imboden, M. Petrone, P. Quadrani, C. Zannoni, R. Mayoral, G.J. Clapworthy, D. Testi, M. Viceconti, D. Neiberg, N.G. Tsagarakis, D.G. Caldwell, A haptic enabled multimodal pre-operative planner for hip arthroplasty, in: *First Joint Eurohaptics Conference and Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems (WHC'05)*, IEEE Computer Society, 2005, pp. 503–504.
- [11] V. Sholukha, A. Leardini, P. Salvia, M. Rooze, S. Van Sint Jan, Double-step registration of in vivo stereophotogrammetry with both in vitro 6-dofs electrogoniometry and ct medical imaging, *J. Biomech.* (2005).
- [12] A. Leardini, L. Astolfi, S. Fantozzi, M. Viceconti, M.G. Benedetti, F. Catani, Advanced multimodal visualisation of clinical gait and fluoroscopy analyses in the assessment of total knee replacement, *Comput. Meth. Prog. Biomed.* 79 (2005) 227–240.
- [13] F. Taddei, A. Stagni, M. Cappello, R. Manfrini, U. Albisinni, M. Viceconti, Kinematic study of a reconstructed hip in paediatric oncology, *Med. Biol. Eng. Comput.* 43 (2005) 102–106.
- [14] F. Taddei, M. Manfrini, S. Giacomini, M. Viceconti, A computer assisted approach for monitoring and planning biological reconstructions in orthopaedic oncology: preliminary results and future perspectives, *Chir. Organ. Mov.* 88 (2003) 211–215.
- [15] M. Krokos, D. Podgorelec, G.J. Clapworthy, R.H. Liang, D. Testi, M. Viceconti, Patient-specific muscle models for surgical planning, in: *Third International Conference on Medical Information Visualisation–BioMedical Visualisation*, IEEE Computer Society, 2005, pp. 3–8.
- [16] F. Taddei, A. Pancanti, M. Viceconti, An improved method for the automatic mapping of computed tomography numbers onto finite element models, *Med. Eng. Phys.* 26 (2004) 61–69.
- [17] R. Lattanzi, M. Viceconti, C. Zannoni, P. Quadrani, A. Toni, Hip-op: an innovative software to plan total hip replacement surgery, *Med. Inform. Internet Med.* 27 (2002) 71–83.
- [18] R.H. Liang, G.J. Clapworthy, M. Krokos, R. Mayoral, Real-time predefined shape cutaway with parametric boundaries, in: *Third International Conference on Medical Information Visualisation–BioMedical Visualisation*, IEEE Computer Society, 2005, pp. 227–231.