Real-Time Visualisation within the Multimod Application Framework

M Krokos[1], A Savenko[1], G J Clapworthy[1], H Lin[1], R Mayoral[1], M Viceconti[2], S Van Sint Jan[3]

[1] Department of Computing & Information Systems, University of Luton, Park Square, Luton LU1 3JU, UK
[2] Laboratorio di Tecnologia Medica, Istituti Ortopedici Rizzoli, via di Barbiano 1/10, Bologna 40136, Italy
[3] Department of Anatomy (CP 619), Faculty of Medicine, University of Brussels (ULB), Belgium

{mel.krokos, gordon.clapworthy} @luton.ac.uk, viceconti@tecno.ior.it, sintjans@ulb.ac.be

Abstract

This paper gives an overview of real-time visualisation algorithms developed under the EC-funded project Multimod to support a novel paradigm for the virtual representation of musculo-skeletal structures. These algorithms are fully integrated into the Multimod Application Framework (MAF), an open-source freely-available software framework for the rapid development of medical visualisation applications. MAF is based on the Visualisation Toolkit (VTK) and other specialised toolkits, e.g. for image registration and segmentation, collision detection or numerical computation. MAF provides a range of high-level components that can be easily combined for rapid construction of visualisation applications that support synchronised views. The majority of algorithms available within the standard underlying MAF toolkits were frequently either too slow or too general for our purposes. We have thus implemented computationally efficient versions of existing algorithms, e.g. for surface and volume rendering, and more importantly, developed new techniques, e.g. for X-ray rendering and designing volume rendering transfer functions. To achieve interactive rendering we have employed a scheme for space partitioning. The emphasis is on exploiting the characteristics of medical datasets (e.g. density value homogeneity) but further utilising the hardware-accelerated capabilities of modern graphics cards. In this context, calculations are moved into hardware as appropriate while avoiding dependency on specialised features of particular manufacturers so as to ensure real code portability.

1. Introduction

The EC-funded project Multimod, currently in its closing stages, has developed and validated a new virtual representation paradigm for anatomical objects using multimedia, multi-modality views of biomedical information.

Many systems that have pursued only visual realism have met with resistance from medical practitioners who, in their training and professional life, have dealt with medical data in a particular way and learnt to interpret it accurately. In contrast to those systems, within Multimod the medical data is fully available in all the conventional ways, but it is augmented within the same context by a multiplicity of additional forms of representation. The display focuses on clinical relevance and employs multiple synchronised views, each view simulating a different imaging modality. Manipulations in one window are immediately and automatically reflected in all of the other windows.

Some views portray medical imaging modalities (e.g. RX, CT, MRI, endoscopy), possibly animated through motion data. Others involve less conventional anatomical representations, e.g. visualisation of anatomical objects in a rather global physiological context, such as inside a moving body. Another example is the visualisation of functional data related to a specific biomedical application and generated by specialised diagnostic modalities, image processing algorithms or even complex multi-physics simulations.

The core code developed within the Multimod project is the Multimod Application Framework (MAF) currently in its first public release [1]. MAF is an open-source, freely-available software framework allowing rapid development of medical visualisation applications and it is wrapped around the Visualisation Toolkit (VTK) [2] and other freely-available biomedical toolkits, e.g. for image registration and segmentation, collision detection or numerical computation. The next release of MAF will support multimodal interaction with tracking devices, e.g. two-handed interaction, and integration of

Proceedings of the Eighth International Conference on Information Visualisation (IV'04)
1093-9547/04 $ 20.00 IEEE

multisensory control via haptics and speech. See Viceconti et al. [3] for details.

Over the last few years the sizes of medical datasets have continued to increase steadily as a result of improved resolution of the imaging devices and they may nowadays even be measured in gigabytes. Meanwhile, the memory of an average-specification workstation has only recently become sufficiently large to store a "standard" size dataset, so providing appropriate visualisation tools continues to produce many challenges.

Several off-the-shelf surface and volume rendering techniques are available within the standard libraries incorporated into MAF. However, they are often either slow or too general for the purposes of the planned Multimod demonstrators [4].

The main visualisation task was to optimise these techniques as much as possible based on "standard" underlying hardware. Further, to support specialised demonstrator features, new rendering modules were developed, e.g. dynamically-reconstructed radiographs for registration purposes.

A very common scenario in visualising a medical dataset is that only a limited number of voxels are of interest. Typically a volume-rendering algorithm (with the notable exception of shear-warp by Lacroute & Levoy [5]) deals only with a small region of interest within a dataset at any given time. As a result, a large number of optimisation techniques found in the literature rely on space subdivisions, e.g. Fujimoto & Tanaka [6].

One of the most widely used space-partitioning schemes is the octree – see Meagher [7] – which organises a volumetric dataset as a hierarchy of bounding boxes. Octrees originally proved very effective in rendering complex polygonal scenes, and have been widely adopted in rendering volumes. Space partitioning allows almost every data processing task to be performed more efficiently. The bounding volumes can be classified for relevance, so that irrelevant areas are quickly discarded, e.g. in ray casting, empty regions can be leapt over or sampled only sparsely. For very large volumes, only relevant sub-volumes would be loaded into RAM for processing.

To optimise performance for surface and volume rendering within MAF, we employed a space-partitioning scheme. The idea was to define a structure less general than an octree but offering a considerable increase in processing speeds due to increased ease of object manipulation. The hierarchical organisation of our structure offers another significant benefit – rendering different levels of detail (LODs), i.e. a multiresolution representation is built in. This allows operations to be performed according to circumstances and provides adaptability particularly when dealing with complex or time-consuming operations. The transition between different LODs in rendering is rapid and practically seamless.

As noted previously, dataset sizes in biomedical applications can be prohibitively large. Even top-of-the-range, state-of-the-art graphics cards do not allow interaction with such datasets in real time, while standard graphics boards may require rendering times within the order of tens of seconds. Using multiresolution, a complex dataset can be rendered at lower resolutions during interactive tasks; the system then reverts instantaneously to higher quality resolutions once user manipulation ceases. This allows us to adhere to pre-determined frame rates according to an application's individual requirements.

The multiresolution philosophy was also used in optimising all volume rendering algorithms integrated into MAF. These optimisations rely on quantifying homogeneity of data to enable the algorithms to work with blocks of voxels (subvolumes) rather than single voxels.

An important issue that influenced our development was the different groups of potential MAF users. These groups are categorised by the available hardware. Low-end users will possess a standard PC with no special graphics acceleration. Standard users will have a modern PC with a good graphics card. At the high end of the spectrum, specialised users will be equipped with a graphics workstation with top-of-the range graphics acceleration, including specialised volume rendering cards such as VolumePro [8]. We have moved as much as possible of the algorithmic execution into the hardware for fast rendering for specialised users. However if graphics support is not available, our implementation reverts to a software-based execution. To achieve interactive rates under all possible circumstances, some image quality may have to be traded for speed; this is achieved through extensive use of lower LODs and subsampling.

This paper gives a brief description of visualisation components we have developed specifically within MAF. Section 2 presents a fast iso-surface generator while hybrid rendering (that is mixing volumes and geometries) and X-ray rendering are discussed in Sections 3 and 4 respectively. Section 5 deals with methods for specifying transfer functions suitable for informative volume rendering. Finally, Section 6 contains a summary and pointers to future developments.

2. Iso-Surface Rendering

The surface renderer employs a significantly enhanced version of the marching-cubes iso-surfacing algorithm, Lorensen & Cline [9], in conjunction with multiresolution to produce surface models within a region of interest in a volume dataset.

The standard marching cubes approach suffers from two significant drawbacks: surface extraction is very slow (a typical time for extracting a surface from a standard size CT dataset is in the region of tens of seconds) and resulting surface models generally contain large numbers of very small triangles. When trying to manipulate such large polygonal models interactively, computational performance can be considerably degraded even on high-end, graphics-supported workstations.

The iso-surfacing algorithm tackles these problems by employing a number of optimisation techniques such as multiresolution, min-max blocks, point caching and multi-threading.

Multiresolution allows us to generate lightweight preview models simply by skipping pixels at regular intervals. Min-max blocks essentially pre-compute minimum and maximum values for data blocks, thus allowing the rendering algorithms to skip irrelevant blocks of data (that is, with values below or above the iso-surface threshold) very quickly. Point caching ensures that vertices on a polygonal model are computed only once.

The speed improvement of iso-surfacing in MAF compared to standard VTK iso-surfacing is normally in the range of 30-100 times faster (even without multiresolution). This performance gives users the ability to select an optimal isosurface threshold interactively, an impossible task with the standard implementation. See Van Sint Jan et al. [10] for discussion of the use of this algorithm in practice.

3. Volume Rendering

Although a volume renderer is the centrepiece of modern medical visualisation software, the large amounts of data that volume renderers have to deal with in typical situations have previously made them very slow and thus impractical.

Numerous researchers, such as Guthe et al. [11], have tackled this problem by developing hardware-based algorithms to render volumes interactively. However, such algorithms have many limitations (e.g. reduced colour resolution, high distortion) and, moreover, lack some important functionalities (e.g. support for complex transfer functions).

To address these issues, the volume renderer developed within MAF is a hybrid hardware/software renderer that employs the standard capabilities of modern graphics cards to accelerate operations while retaining a software-based kernel to support advanced volume rendering functions, e.g. lighting, 2D transfer functions and trilinear data interpolation.

Additionally, this renderer employs various software optimisations that allow us to achieve interactive frame rates. Among these are: multi-resolution data sampling, adaptive pixel sampling, render caching and efficient Bresenham-like volume traversals. A detailed discussion of these concepts is outside the scope of this article and will be presented in a future paper. This volume renderer replaces the shear-warp method by Dong et al. [12] used initially, and offers improved rendering quality with only minor speed compromises.

Another aspect of the volume renderer is hybrid rendering, that is, visualisation of volumetric data using both geometric (surface-based) and volumetric methods within the same image (Fig. 1). Hybrid rendering plays an important role within the Multimod demonstrators. For example, showing the position and orientation of cut planes can significantly improve the viewer's understanding of a volumetric dataset. For orthopaedic operation planning, this functionality is an indispensable tool for depicting surgical instruments, or prostheses, together with the volumetric data for a particular limb position, thus allowing surgeons to see their instruments in relation to important structures within a volume of interest. For example, a surgeon has the ability to define the anatomically correct position and orientation of the femoral and pelvic components of an implant and to determine its optimal size.

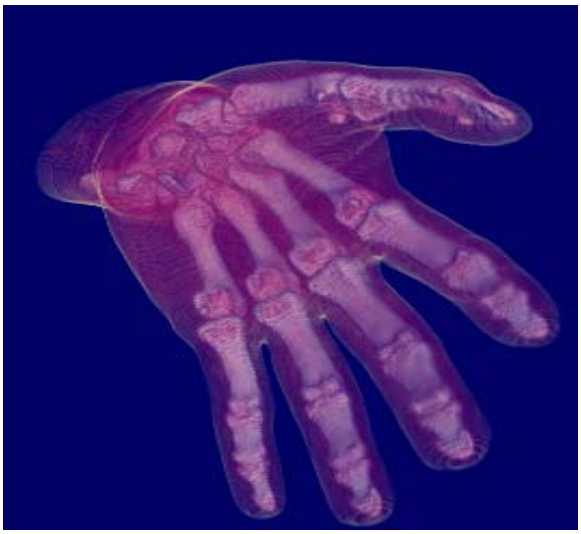

Figure 1. Hybrid rendering

Geometric rendering may not necessarily be confined to medical instruments or implants, it can also include tissue, e.g. a skin iso-surface or a soft tissue model.

4. X-Ray Rendering

The X-ray renderer reconstructs radiographs from volume datasets, an important feature in orthopaedics visualisation applications. This renderer essentially computes a 2D projection (e.g. an X-ray image) from an arbitrarily-oriented CT volume with respect to the projection source. This opens up the possibility for computing the roto-translation to be applied to the CT volume (or to the source) in order to find the synthetic image that best matches a real radiograph (see Fig. 2). Based on recently-developed methods, real radiographic images captured during a patient's treatment can be accurately matched to preoperative CT data – see LaRose [13].

Figure 2. X-ray rendering

Because the matching process (registration) may require hundreds of renderings in order to achieve a desired accuracy, the main development focus was on achieving interactive rendering rates (15-30 frames per second). To this extent, a range of computational capabilities of modern graphics cards were exploited, e.g. texture mapping, accumulation buffers and hardware-accelerated geometric transformations. Software-based techniques were also used, e.g. object culling. Multiresolution was used to select optimal accumulation buffer resolution, slice sampling distance and texture sizes.

An issue faced when reconstructing radiographs was the very high dynamic range of the resulting images. An automatic brightness mapping (auto-exposure) tool then becomes a must-have feature. In MAF, we implemented a histogram-based auto-exposure algorithm that allows users to interact with a model without having to adjust image settings manually.

5. Transfer Functions

Perhaps one of the most common problems that prohibits the effective use of volume rendering techniques is the difficulty in specifying a transfer function (TF). A transfer function assigns optical properties, e.g. opacity/colour, to data values in a dataset. A "good" transfer function can make a vast difference in rendered image quality, but automatic derivation of such a function is difficult as it is heavily dependent upon the semantics of the dataset to be visualised.

The most common scheme for TF specification is by trial and error. This involves manually editing a typically linear function by manipulating "control points" and periodically checking the resulting volume rendering. Even if specialised hardware support is available, e.g. a VolumePro board [8], this method can be very laborious and time-consuming. The problem lies in the lack of a precise correspondence between control point manipulation and its effects on the rendered images.

Another scheme based on the design galleries paradigm of Marks et al. [14] generates a large number of volume renderings simultaneously, each resulting from a different transfer function. The user then selects renderings satisfying individual requirements, thus implicitly optimising the transfer function. The challenges are automatically to generate different transfer functions that produce a sufficiently wide spread of dissimilar output renderings and to present these renderings in an effective way. A typical session may involve hundreds of renderings. To ensure interactivity, real-time hardware volume rendering functionality is essential.

The contour spectrum algorithm computes metrics over the data values of a dataset and the resulting spectrum is displayed within the user interface as a collection of signature curves, each representing a different attribute – Bajaj et al. [15]. Such curves offer an alternative concise way of revealing global characteristics of datasets and can be very helpful in creating transfer functions.

Edge detection concepts for locating boundaries are employed by a method based on distance maps originated by Kindlmann & Durkin [16]. A computationally inexpensive pre-processing step requiring minimal user intervention is performed to construct a 3D volume histogram of data value against first/second derivatives. A distance map is afterwards produced to record the relationship between data value and boundary proximity. Using the distance map, users can interactively experiment with a variety of settings but the transfer functions are always usefully constrained

by the boundary information measured in a given dataset.

Based on the previous method, multi-dimensional transfer functions introduced recently by Kniss et al. [17] provide an effective way to extract materials and their boundaries that is applicable not only to scalar but also to multi-variate datasets. Multi-dimensional transfer functions allow voxel classification based on a combination of values, thus increasing the probability that a feature can be uniquely isolated in the transfer function domain. An unavoidable drawback is, of course, the increased memory consumption that is necessary to store all relevant transfer function variables at a voxel sample point.

The transfer function design interface currently employed in MAF is founded on the aforementioned method and employs 2D functions. The end-user interacts with a set of direct manipulation widgets (triangles and rectangles). Each widget precisely corresponds to a different material and widgets are blended automatically to compute an overall transfer function.

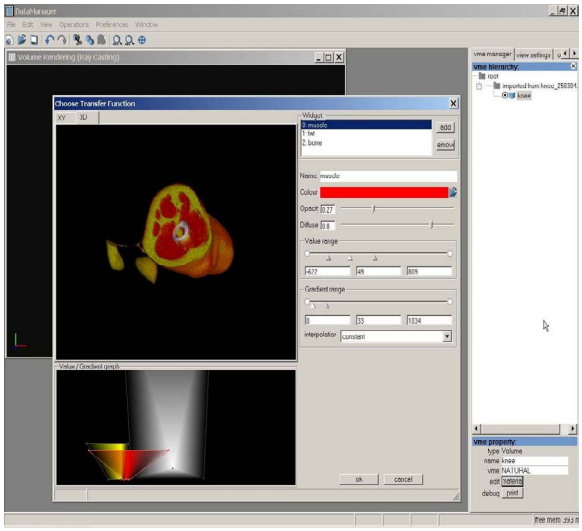

Figure 3. Transfer function design using 2D widgets

This interaction process allows users without specialised graphics knowledge to specify, quickly and intuitively, appropriate transfer functions for informative volume rendering. Nevertheless, as pointed out in Kniss et al. [17], novice users will typically need to pass through a training period to learn to appreciate the information provided by the widgets. As this may not be an easy or desirable process for medical professionals who have traditionally employed only 2D views, MAF allows meaningful and fast visual feedback through a 2D slice preview of the actual volume rendering appropriately superimposed on a slice-based view of the data being rendered, as in Botha & Post [18]. This greatly facilitates the correlation of structures in the volume rendering with their recognisable counterparts in a traditional 2D view. Also, it is easier to quantify the impact of small changes being made to the transfer function widgets.

Initial feedback based on early MAF demonstrators suggests that transfer function specification using 2D functions constitutes a far superior tool for correctly assigning colour and opacity. Our tests indicate that, compared to traditional techniques, e.g. 1D linear ramps, the overall specification process takes a fraction of the time. Figure 3 shows a transfer function for a 100MB CT dataset. Three widgets are shown (one rectangular and two triangular) for bones muscles and skin. The overall process lasted less than 10 minutes for a user previously unfamiliar with the system.

6. Conclusions

This paper has given a brief description of the visualisation modules within the Multimod application Framework, the core code developed under the EC-funded Multimod project.

All the visualisation modules described have been integrated into the Lower Abstraction Layer (LAL) of MAF. For every module, a wrapper class was created encapsulating the pipeline creation routines (e.g. data preparation and connection of visualisation framework elements to each other), the visualisation module and a specialised user-interface (e.g. as a volume rendering view or transfer function interface). This allows rapid creation of MAF-based applications, as different visualisation modalities can be plugged in an appropriate LAL class.

We have implemented computationally efficient versions of existing algorithms for surface and volume rendering and have also developed new techniques for X-ray rendering and designing volume rendering transfer functions. We have employed a space subdivision scheme to achieve interactive rendering. The emphasis has consistently been on exploiting the hardware-accelerated capabilities of modern graphics cards appropriately. Although algorithmic execution is moved into hardware for high-end graphics workstations, our implementation reverts to software execution if graphics support is unavailable.

The Data Manager [19] is the core MAF demonstrator and supports all of the aforementioned visualisation modules, including 2D transfer functions and hybrid rendering.

The current transfer function (TF) implementation constitutes the basis for further developments into higher

dimensions. A range of questions need to be addressed, e.g. what is an "appropriate" set of metrics upon which TFs are defined, what is a "good" way of defining TFs given these metrics, what is an "easy" way of manipulating widget shapes in spaces higher than 2D, what is an efficient way of coping with increased memory requirements in dimensions higher than 2D and what will be the acceptance of medical professionals when tools such as higher dimension TFs are employed within standard clinical environments.

Future work will also be extended to include point rendering and clustering.

Point rendering uses points, rather than polygons, as primitives for rendering surfaces implying that no connectivity information is required. Point rendering is thus particularly suitable for data organised as point clouds. Although suitability within a MAF context is unproven we believe that a hybrid renderer that combines point and volume rendering appropriately may be useful.

Clustering employs a network of computers for a single process and might be particularly beneficial within an educational context, e.g. in a group of low-spec PCs in an academic laboratory. As, by nature, clustering implies algorithm parallelisation, the main challenges are dynamic volume subdivision into (not necessarily equal) subvolumes to strike a balance between network traffic and the computing capabilities of individual PCs. Fast algorithms are necessary for compositing / blending the final image, together with new techniques for parallel volume rendering.

Acknowledgement

This work was supported by the Information Society Technologies Programme of the European Commission under the project Multimod (IST-2000-28377).

References

[1] http://www.tecno.ior.it/multimod

[2] http://public.kitware.com/VTK/index.php

[3] M. Viceconti, A. Leardini, C. Zannoni, D. Testi, F. Taddei, L. Astolfi, M. Petrone, S. Imboden and P. Quadrani, "The Multimod Application Framework", *Proc. Information Visualisation 04*, IEEE Computer Society Press, 2004

[4] http://www.tecno.ior.it/multimod/mm_demonstrators.html

[5] P. Lacroute and M. Levoy, "Fast Volume Rendering Using a Shear-Warp Factorization of the Viewing Transformation", *ACM Comp. Graphics (SIGGRAPH '94)*, pp. 451-458, 1994

[6] A. Fujimoto and T. Tanaka, "Accelerated Ray Tracing", *Proc. Computer Graphics, Visual Technology & Art*, Springer-Verlag, pp. 40-65, 1985

[7] D. Meagher, "Geometric Modelling Using Octree Encoding", *Computer Graphics & Image Processing*, 19(2), pp. 129-147, 1982

[8] http://www.terarecon.com

[9] W. Lorensen and H. Cline, "Marching Cubes: a High Resolution 3D Surface Construction Algorithm", *Computer Graphics*, 21(4), pp. 163-169, 1987

[10] S. Van Sint Jan, M. Viceconti and G.J. Clapworthy, "Modern Visualisation Tools for Research and Education in Biomechanics", *Proc Information Visualisation 04*, IEEE Computer Society Press, 2004

[11] S. Guthe, M. Wand, J. Gosner, W. Straßer, "Interactive Rendering of Large Volume Data Sets", *Proc. IEEE Visualization 2002*, pp. 53-60, 2002.

[12] F. Dong, M. Krokos and G.J. Clapworthy, "Fast Volume Rendering and Data Classification Using Multiresolution Min-Max Octrees", *Computer Graphics Forum,* 19(3), pp. 359-368, 2000

[13] D. LaRose, "Iterative X-Ray/CT Registration Using Accelerated. Volume Rendering", PhD Thesis, Tech. Report CMU-RI-TR-01-10, Robotics Institute, Carnegie Mellon University, May 2001

[14] J. Marks, B. Andalman, P.A. Beardsley, W. Freeman, S. Gibson, J. Hodgins and T. Kang, "Design Galleries: a General Approach to Setting Parameters for Computer Graphics and Animation", *ACM Comp. Graphics (SIGGRAPH '94)*, pp. 389-400, 1997

[15] C.L. Bajaj, V. Pascucci and D.R. Schikore, "The Contour Spectrum", *Proc. IEEE Visualization 97*, pp. 167-173, 1997

[16] G. Kindlmann and J.W. Durkin, "Semi-Automatic Generation of Transfer Functions for Direct Volume Rendering", *Proc. IEEE Symp. Volume Visualization*, pp. 79-86, 1998

[17] J. Kniss, G. Kindlmann and C. Hansen, "Multidimensional Transfer Functions for Interactive Volume Rendering", *IEEE Trans. on Visualization. & Comp. Graphics*, 8(3), pp. 270-285, 2002

[18] C.P. Botha and F.H. Post, "New Technique for Transfer Function Specification in Direct Volume Rendering using Real-time Visual Feedback", *Proc. SPIE Symposium on Medical Imaging*, Vol. 4681, 2002

[19] http://www.cineca.it/B3C/MAF/datamanager.html