Drishti, A Volume Exploration and Presentation Tool

Ajay Limaye[1]

National Computational Infrastructure National Facility, The Australian National University, Canberra ACT 0200, Australia

ABSTRACT

Among several rendering techniques for volumetric data, direct volume rendering is a powerful visualization tool for a wide variety of applications. This paper describes the major features of hardware based volume exploration and presentation tool - *Drishti*. The word, *Drishti*, stands for vision or insight in Sanskrit, an ancient Indian language. *Drishti* is a cross-platform open-source volume rendering system that delivers high quality, state of the art renderings. The features in *Drishti* include, though not limited to, production quality rendering, volume sculpting, multi-resolution zooming, transfer function blending, profile generation, measurement tools, mesh generation, stereo/anaglyph/crosseye renderings. Ultimately, *Drishti* provides an intuitive and powerful interface for choreographing animations.

Keywords

Drishti, volume rendering, 3D printing, surface rendering

1. Introduction

In the past decade with the easy availability of powerful graphics hardware via GPUs, volume rendering has positioned itself as a potent tool for interactive exploration of volumetric data sets. Over the years many volume rendering libraries and frameworks such as Voreen [1], Tuvok [2], OpenGL Volumizer [3], VolPack [4], java based library RTVR [5] and VTK [6] have been made available to the users to build applications in. Complete volume rendering systems such as VolView[7], OsiriX[8], VisageRT[9], 3DSlicer[10] are more geared towards medical data. Simian [11], a program developed mainly for research in volume rendering, was the first one to employ two dimensional transfer functions – where voxel values along with their neighbourhoods are considered in deciding colour and opacity for that volume element. Visualization packages such as Amira[12], ParaView[13] and ImageJ[14] also provide volume rendering as an option for viewing volumetric data. VGStudioMax[15] provides visualization and analysis capabilities. All these packages are used extensively on CT data.

Most of these softwares provide only rudimentary animation capabilities, where camera movements can be choreographed. Many packages allow clipping along standard axes, but more general volume sculpting facilities such as cropping or dissection are generally unavailable. *Drishti* supports a unique mesh generation and colouring options that are currently not available in any of the commercial or free software.

[1] Ajay.Limaye@anu.edu.au

Developments in X-Ray Tomography VIII, edited by Stuart R. Stock,
Proc. of SPIE Vol. 8506, 85060X · © 2012 SPIE · CCC code: 0277-786/12/$18
doi: 10.1117/12.935640

The remainder of this paper is organized as follows. The *Drishti* interface is described in Section 2. Various volume sculpting options are introduced in Section 3. The animation facility is discussed in Section 4. The mesh utility is described in Section 5 and other features are discussed in Section 6. The paper concludes with a summary and future directions.

2. Interface

Drishti, from Sanskrit, means representation, a framework for vision/perception/cognition. The software handles scalar data over rectilinear grid and is written using OpenGL [16] with Qt [17] for the user interface. This cross-platform open source software is available for download from http://code.google.com/p/*Drishti*-2. The software runs on OpenGL 2.0 capable graphics hardware. The package has two modules – a data importer and the renderer.

2.1. Drishti Import

Drishti Import converts the volume data from a variety of open or user defined formats into the format that *Drishti* Render reads (8-bits per voxel). The import facility has plugin interface to read in different formatted inputs. At the time of writing this document, the importer can read and convert raw, netCDF 3.0[18], Analyze[19], HDF 4.0[20], image stacks, DICOM [21] and MetaImage[22] volume file formats.

The importer provides a capability to resample, filter, enhance contrast and trim the volumes. Mesh generation facility is also available from within the importer. A histogram and colour gradient panel is provided to apply colour to the grayscale data. These colour image slices can be saved.

The importer also allows loading of coloured image slices for viewing in the *Drishti* Render as three or four channel (RGB/RGBA) volumes.

2.2. Drishti Render

The renderer module handles 3D volumetric data of scalar field defined over a regular grid. The volumetric data is rendered using hardware texture based volume rendering method [23]. The volume is loaded into texture memory of the graphics card. View aligned polygons are textured and composited to generate the final image. *Drishti* renderer can visualize, simultaneously, up to four 3D datasets or up to four 4D datasets or single coloured RGB/RGBA volume.

The interface for the *Drishti* renderer consists of two components – the display screen where the volume rendered images will appear, and the user interface dialogs via which transfer function definition, shadow controls etc can be modified (Figure 1). When a volume is loaded, the data is sub-sampled if necessary so as to fit the entire volume in the texture memory of the graphics card. This version serves to give a global view of the data. A box widget is provided for selection of the sub-volume for detailed exploration. The selected region might also be sub-sampled in order to fit in the texture memory. *Drishti* thus provides multi-resolution zoom facility that allows users to view arbitrarily large datasets. User can switch the display window to show either the entire volume or the selected region in higher resolution.

Various panels and switches that allow manipulation of transfer functions, lighting and shadow controls, keyframe animation, volume information, stereo viewing, mouse grabbing etc. are accessible via top menu bar.

Transfer functions map voxel information to optical properties. In scalar volumes, gradient magnitude characterizes how quickly values are changing in a given neighbourhood. Its inclusion in the domain of the transfer function allows the distinction between homogeneous regions (low gradient magnitudes) and transition regions (high gradient magnitudes). *Drishti* employs such a two dimensional transfer function making use of both value and gradient magnitude to assign colour and opacity to the voxels.

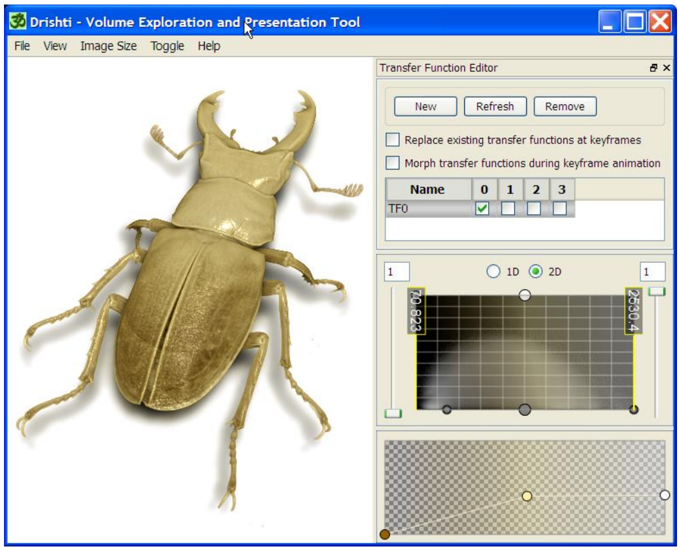
Drishti - Volume Exploration and Presentation Tool
File View Image Size Toggle Help
Transfer Function Editor
New
Refresh
Remove
Replace existing transfer functions at keyframes
Morph transfer functions during keyframe animation
Name 0 1 2 3
TF0
1D
2D
70.823
2530.4

Figure 1. *Drishti* Renderer interface. On the left side is display window for volume rendered images. On the right side is the graphical user interface. Display window can be switched to show entire volume or selected sub volume. Shown is a volume rendered micro-CT scan of a stag beetle[24].

The user is presented with a histogram of the volume data over which the transfer function is drawn. Choice can be made between value-only (1D) histogram and value-gradient magnitude (2D) histogram (Figure 2). Widgets are provided for specifying transfer function shape and colour and opacity definitions. Standard and user defined colour/opacity maps can be loaded.

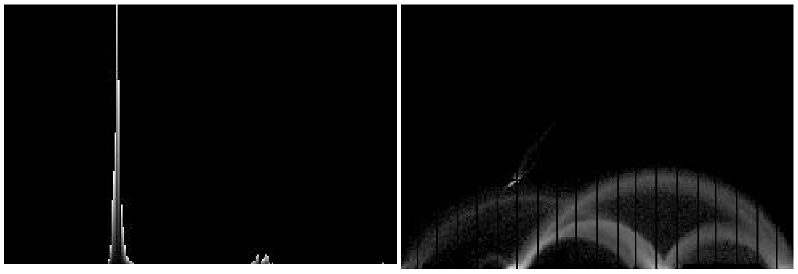

Figure 2. On the left is an value (1D) histogram and on the right is the value-gradient magnitude (2D) histogram of tooth dataset [25]. In both the graphs, value is plotted along the horizontal axis. The gradient magnitude is along the vertical axis in the 2D histogram.

Each transfer function is given a name. The user selects the transfer functions to composite for the final image makeup. Such a facility allows for creation of large number of transfer functions which bring out specific features in the data, and to switch the visibility of these features as and when required. This is invaluable during exploration and presentation. It also helps in reducing the visual clutter and emphasizing the essential features (Figure 3).

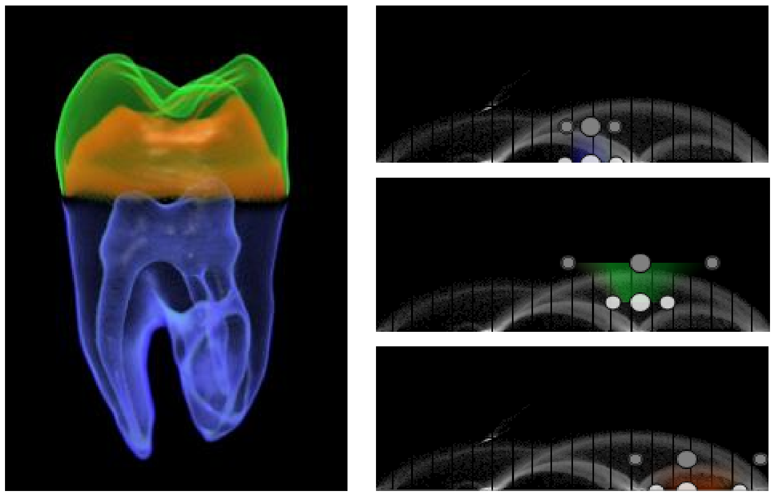

Figure 3. This figure shows the effects of different transfer functions on a tooth dataset [25]. Three transfer functions are defined for 3 different features in the dataset.

Transfer functions can be grouped together in various transfer function sets. A transfer function can belong to multiple sets. When multiple volumes are visualized together, each volume gets assigned a different transfer function set, so that user can control what transfer functions get applied to what volume. Transfer function sets are also used when blending transfer functions, where rendering using a particular set of transfer functions is restricted to certain regions.

3. Volume Sculpting

Sculpting has been shown to be useful in volumetric applications. For example, researchers often need to explore the inner structures of their simulated or sampled datasets by gradually removing or slicing the material. *Drishti* provides sculpting facility via clipping planes, cropping, blending, dissection, reveal, path tools and bricks as shown in Figure 4.

A clipping plane is used to cull the data against the plane. Each clip plane is depicted by its normal and a point on the cutting plane (Fig. 4a). There is no restriction on the number of clipping planes. The data culling via individual clipping planes can be toggled. Images and text annotations can also be embedded on to the clip planes.

Cropping facility provides more a flexible sculpting facility compared to clip planes. Users can make cuts using geometric shapes such as box, ellipsoid, cylinder and its various permutations shown via an example in Fig. 4b.

Blending of transfer functions allow users to display different information in a specific region by using different transfer functions to the one that is applied to the whole data. For example in Fig. 4c, the central region is showing different information (pores – air phase) to the one shown on the outside (grains – solid phase). Like cropping, blending also uses various geometric shapes to define the required region of interest.

Dissection allows tearing to peek inside the data as shown in Fig. 4d. Here a dissection widget is placed on the top layer of a rabbit liver lobe to create an incision and see the vessels inside. Dissection widget allows for different type of cuts such as eye shape, wedge shape or a hole.

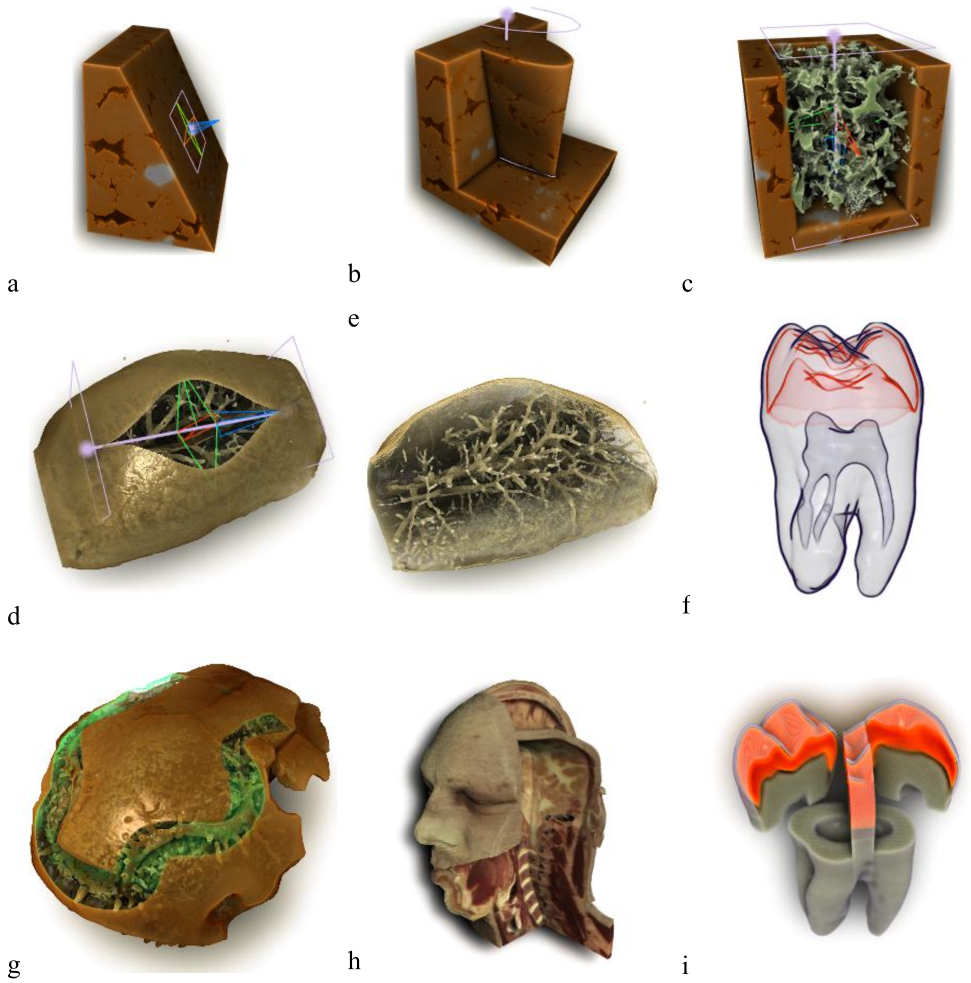

Figure 4. This figure shows various volume sculpting options in *Drishti*. First three image (a),(b) and (c) show a rock data set. (d) and (e) show a rabbit liver lobe. (f) and (i) show tooth. (g) shows a fossilized lung fish (Gogonasus) [26] snout. (h) shows RGB volume from visible human dataset [27]. Sculpting methods shown are (a) clip plane, (b) crop, (c) blend transfer function showing pore phase, (d) dissection showing vessels, (e) reveal making top layer transparent (f) line drawing (g) sculpting using path to show inner structure (h) curved slab using path and (i) bricking

The reveal functionality (Fig. 4e) allows users to make certain regions transparent based on the surface normal. Inner and outer surfaces are decided based on direction of normal with respect to the user. If normal at a point is directed towards user, then the point is said to lie on the outer surface. Reveal facility can be used to make outer surface more transparent in order to reveal the inner regions of the data. User can explore data from all the sides just by moving around the data. As the user moves around the data, surface facing the user will always be made transparent thus revealing the interior regions from all sides. Users are also provided with controls for transparency, cone of angle for normal to be considered as facing user and mixing of original surface with modified one. By manipulating cone of angle, users can also create line drawing like images (Fig. 4f), by only displaying points that have normal pointing at right angles to the viewer.

Paths are splines that can be used for multiple purposes. One of the uses is to sculpt the volumes either by cropping or by blending a different transfer function in the region specified by the path. Users can excavate, extract tubular structures, generate curved slabs or fill excavated region with different transfer function as shown in Fig. 4g and 4h. Paths provide a more flexible way for performing cropping and blending operations.

The volume data can be subdivided into smaller chunks called bricks. Each brick can have its own set of transfer functions and can be rotated, translated and scaled as shown in Fig. 4i. Bricking allows users to expose the interior of a solid region by slicing along major axes.

4. Animation

Animations in *Drishti* are generated using what is called as keyframe based animation. Users choreograph camera moves and various parameters and saves important frames in an animation sequence. These saved frames are called keyframes. A keyframe saves all the information necessary to generate the required image. A small snapshot of the image generated from the keyframe acts as a place holder in the animation panel (Fig. 5).   Information that needs to be stored include, though not limited to, transfer functions, lighting, bricks, clip planes, paths, volume bounds, background colour, stereo settings, annotations, camera parameters, crop, dissect, blend and reveal. These settings are then interpolated by the program for intermediate frames. Various interpolation options such as linear, smoothstep, easy in and easy out are supported.

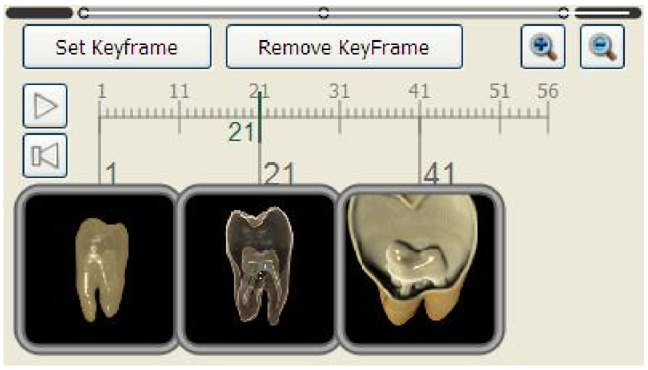

Figure 5. Animation panel showing the stored keyframes.

Such an interface provides an intuitive and powerful interface for choreographing animations. The animations can be rendered as a set of images or can be saved in a movie format. The keyframe and volume information is stored in a project file.   Users can import transfer functions as well as keyframes and other settings from another project. Users have an option of storing mono/left and right stereo images and movies.

5. Mesh Generation

Almost all available software packages allow users to generate isosurface meshes for a given voxel value. The isosurface will usually have a single colour all over the mesh. Moreover, data cannot be cropped, clipped or sub-sampled, before the mesh generation.

In *Drishti*, users can generate water tight meshes [28] either by using voxel values or voxel opacities. *Drishti* offers more than just an isosurface mesh. It allows users to apply transfer functions and generate surface mesh enclosing the opaque region. In order to achieve this, an opacity volume is first created using the transfer functions. This volume is converted to binary form based on the opacity – 0 for transparent region and 255 for non transparent region. The volume is then smoothed and an isosurface mesh is generated from this volume.   Colours are baked into the mesh based on transfer functions used to define the enclosing region. This is achieved by applying transfer function to each vertex in the mesh. In order to bring out the subsurface details, software shoots rays inside the mesh to collect and composite colour information (Fig 6a) onto the mesh surface. Local shadowing effects can also be applied to the meshes by casting rays from every direction of the surface and collecting visibility information. This visibility information is then used to decide on the local darkening of the mesh (Fig 6b).

All the cropping and blending operations applied in the volume rendered image are also applied during mesh generation (Fig. 6c). As with volume rendered image, these operations can be used to reduce the clutter in mesh or expose hidden regions in the volume.

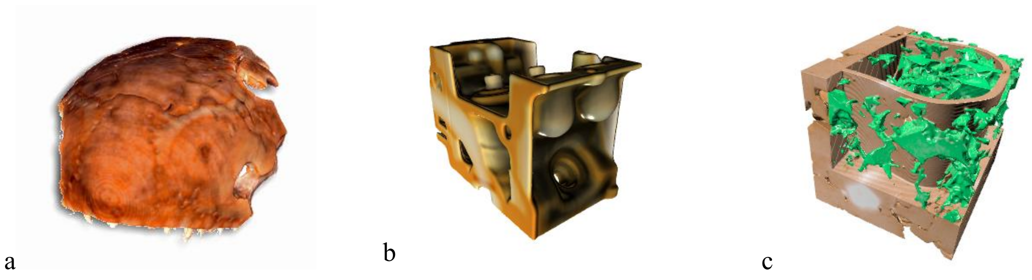

Figure 6. Meshes generated from *Drishti* – (a) fossilized lung fish (Gogonasus) snout showing subsurface details. (b) local darkening applied to engine block [25]. (c) rock (brown) with pores (green) using paths for blending transfer function.

The mesh algorithm runs in the CPU and so it is advantageous to work with sufficient RAM. In the case of large datasets, a sub-sampled version of data can be supplied for mesh generation. If the dataset cannot be accommodated in main memory, then slabs of the dataset are loaded and mesh generation process is run on each slab. The resulting mesh slabs are then combined together to form a complete mesh for the given structure.

The meshes generated are water tight and have colour information at each vertex, thus making them highly suitable for 3D printing. This way of generating physical reproductions with internal details painted onto the surface is an innovation in 3D printing.

6. Miscellaneous Features

The software is meant for interactive use, therefore in order to maintain interactive frame rate, *Drishti* employs two volume sampling levels – coarse level for mouse dragging and a refined level when there is no mouse interaction. These two sampling levels can be controlled by the user to suit the graphics card. Viewers can also operate *Drishti* in active as well as passive (anaglyph/crosseye) stereo modes. The software allows both perspective as well as orthographic projections. Users can save image sequences as well as movies in mono-, stereo- and cubic- formats. Cubic formatted images are required for generation of fisheye images suitable for dome projections in planetariums. *Drishti* implements state of the art lighting and shading for volume rendering (Fig 7a). Users have control over depth cueing, light position, soft shadowing, emissive lighting, light diffusion and shadow casting on backplane. The lighting and shadowing parameters can be animated. All animation and other settings are stored to a project file. The user only needs to load the project file to restore these settings back. *Drishti* also supports batch rendering.

Paths are very versatile objects in *Drishti*. Users can embed text annotations onto paths, this allows for displaying information next to or on a feature in 3D (Fig. 7b). Interrogation of raw data values and coordinates at various points is supported. Path lengths and angle displays are possible. The data profile and thickness profile can also be obtained along a path. Most of the softwares allow for profile generation along a ray. *Drishti* allows profile generation along a curved path. This profile information can be saved to a text file.

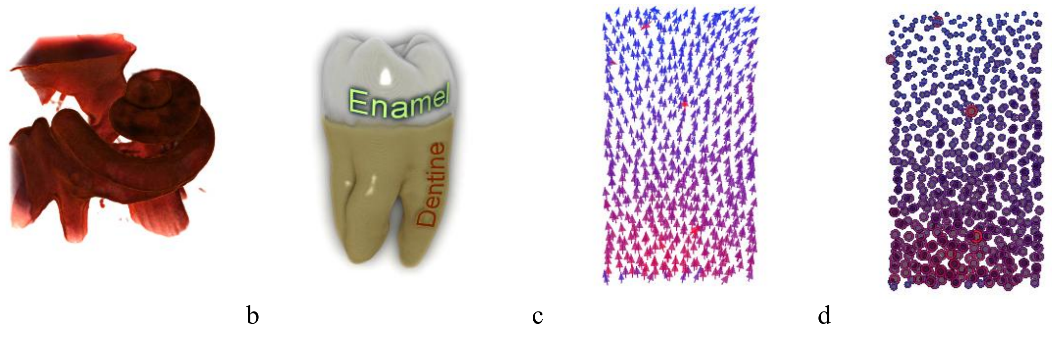

Figure 7. (a) Fossilized whale cochlear – light is placed behind the object. (b) Tooth showing 3D text on dentine and enamel. (c) vector data showing direction vectors (d) same data as (c) displaying magnitude as circles. Colours for (c) and (d) are assigned based on vector magnitude – high to low – red to blue.

Vector data can be displayed as direction vectors using arrow glyphs (Fig 7c). Scatter plots can be displayed as circle glyphs with radius based on value at that point. Users can choose to display vector magnitudes as circle glyphs with radius defined by vector magnitude (Fig 7d). Vector datasets can be filtered and colours assigned based on vector magnitude (Fig 7c, d).

Surface area and total volume calculations are supported for a given transfer function. All non-zero opacity voxel are considered for these calculations. In the case of total volume computation, all non-zero opacity voxels are added up, where as for surface area calculations, all interior voxels are eliminated before the computations are done. The non-zero opacity voxels are decided based on currently applied transfer functions.

Mesh generated using *Drishti* can also be embedded with the volume rendering. Meshes can be cropped and clipped to view interior. Front surfaces can also be made transparent for the same purpose.

7. Conclusions and Future Work

In this paper various features in *Drishti* have been presented. Many options for volume sculpting to tease out features and explore volumetric datasets are discussed. *Drishti* provides production quality presentation and animation facilities. Mesh generation and colouring capability to bake subsurface details onto surface mesh along with cropping and transfer function blending are the standout and unique features in *Drishti*.

Drishti will be extended to include more analysis tools. Work is underway to extend the present rendering engine to include progressive rendering in order to handle multi-gigabyte datasets.

The source code and executables for the software can be downloaded from http://code.google.com/p/*Drishti*-2/

8. Acknowledgements

Drishti software development is made possible because of continuing support from NCI National Facility, ANU. The author would like to thank Drew Whitehouse and Dr. Tim Senden, ANU for discussions on implementation of various features in *Drishti* and making available liverlobe, whale cochlear and rock datasets. Furthermore the author would like to acknowledge Dr. Gavin Young, ANU for lung fish Gogonasus fossil dataset and Dr. Mohammed Saadatfar for vector data. The author would also like to acknowledge gratefully the contributions from the *Drishti* user group https://groups.google.com/group/drishti-user-group

8. References

[1] Voreen, http://www.voreen.org
[2] T. Fogal, J. Kruger, "Tuvok, an Architecture for Large Scale Volume Rendering", Vision, Modelingand Visualization, 2010.
[3] P. Bhaniramka, Y. Demange, "OpenGL volumizer: a toolkit for high quality volume rendering of large data sets", Proceedings of the 2002 IEEE symposium on Volume visualization and graphics: 45-54, 2002.
[4] VolPack, graphics.stanford.edu/software/volpack
[5] L. Morz, H Hauser, "RTVR – a flexible Java library for interactive volume rendering", IEEE Visualization, 279-286, 2001.
[6] The Visualization Toolkit, http://www.vtk.org
[7] VolView, http://www.kitware.com/products/volview.html
[8] Osirix, http://www.osirix-viewer.com
[9] VisageRT, Mercury Computer Systems, Inc, lifesciences.mc.com
[10] 3DSlicer, http://www.slicer.org
[11] Simian, http://www.cs.utah.edu/~jmk/simian
[12] Amira, http://www.amira.com
[13] ParaView, http://www.paraview.org
[14] ImageJ, http://rsbweb.nih.gov/ij
[15] VGStudioMax, http://www.volumegraphics.com
[16] OpenGL, http://www.opengl.org
[17] Qt, http://qt.nokia.com
[18] NetCDF, http://www.unidata.ucar.edu/software/netcdf/
[19] Analyze, http://wideman-one.com/gw/brain/analyze/formatdoc.htm
[20] HDF4, http://www.hdfgroup.org/products/hdf4/
[21] DICOM, http://medical.nema.org/
[22] MetaImage, http://www.itk.org/Wiki/ITK/MetaIO/Documentation
[23] B. Cabral, N. Cam, J. Foran, "Accelerated volume rendering and tomographic reconstruction using texture mapping hardware", 1994 Symposium on Volume Visualization: 91-98, 1994.
[24] http://www.cg.tuwien.ac.at/research/publications/2005/dataset-stagbeetle/
[25] http://www9.informatik.uni-erlangen.de/External/vollib/
[26] J.A. Long, G.C. Young, T. Holland, T.J. Senden, E.M.G. Fitzgerald, "An exceptional Devonian fish from Australia sheds light on tetrapod origins", Nature 444, 199-202, 2006.
[27] http://www.nlm.nih.gov/research/visible/visible_human.html
[28] T. Lewiner, H. Lopes, A.W. Vieira, G. Tavares, "Efficient implementation of Marching Cubes' cases with topological guarantees", Journal of Graphics Tools, 8 (2) (2003), pp. 1–16