Contents lists available at ScienceDirect

Computerized Medical Imaging and Graphics

journal homepage: www.elsevier.com/locate/compmedimag

Dynamic real-time 4D cardiac MDCT image display using GPU-accelerated volume rendering

Qi Zhang*, Roy Eagleson, Terry M. Peters

Imaging Research Laboratories, Robarts Research Institute, University of Western Ontario, London, Ontario, N6A 5K8, Canada

ARTICLE INFO

Article history:
Received 18 February 2008
Received in revised form 16 March 2009
Accepted 8 April 2009

Keywords:
Real-time
4D volume rendering
Retrospective electrocardiographic (ECG) gating
Graphics processing units (GPUs)
Stereoscopic rendering
Optical density radiation model
Ray-casting
Volume of interest (VOI)
Dynamic multiresolution
Image-guided therapy
Cardiac intervention

ABSTRACT

Intraoperative cardiac monitoring, accurate preoperative diagnosis, and surgical planning are important components of minimally-invasive cardiac therapy. Retrospective, electrocardiographically (ECG) gated, multidetector computed tomographical (MDCT), four-dimensional (3D + time), real-time, cardiac image visualization is an important tool for the surgeon in such procedure, particularly if the dynamic volumetric image can be registered to, and fused with the actual patient anatomy. The addition of stereoscopic imaging provides a more intuitive environment by adding binocular vision and depth cues to structures within the beating heart. In this paper, we describe the design and implementation of a comprehensive stereoscopic 4D cardiac image visualization and manipulation platform, based on the opacity density radiation model, which exploits the power of modern graphics processing units (GPUs) in the rendering pipeline. In addition, we present a new algorithm to synchronize the phases of the dynamic heart to clinical ECG signals, and to calculate and compensate for latencies in the visualization pipeline. A dynamic multiresolution display is implemented to enable the interactive selection and emphasis of volume of interest (VOI) within the entire contextual cardiac volume and to enhance performance, and a novel color and opacity adjustment algorithm is designed to increase the uniformity of the rendered multiresolution image of heart. Our system provides a visualization environment superior to noninteractive software-based implementations, but with a rendering speed that is comparable to traditional, but inferior quality, volume rendering approaches based on texture mapping. This retrospective ECG-gated dynamic cardiac display system can provide real-time feedback regarding the suspected pathology, function, and structural defects, as well as anatomical information such as chamber volume and morphology.

1. Introduction

Most cardiac disorders result from anatomical structure defects, and cardiac surgery is a common strategy to treat such problems. Knowledge of cardiac anatomy, regional myocardial contraction patterns and deformations over the entire cardiac cycle is necessary for accurate diagnosis of cardiac disease and therapeutic planning. Traditionally, cardiac surgery involves a sternotomy and the use of a heart–lung machine. However, the trauma associated with opening the chest, along with potential generation of embolic and systemic immune-system reactions linked to the use of a heart–lung machine, suggest the need to explore procedures that are less invasive than those currently in use [1]. These limitations have motivated our laboratory to explore means of performing surgery inside the beating heart in a minimally invasive fashion [2].

In recent years, new multidimensional medical imaging modalities have enabled real-time 4D cardiac data acquisition using multidetector computed tomography (MDCT), or magnetic resonance imaging (MRI) scanners in a standard clinical environment. These data are acquired at evenly spaced time points throughout the cardiac cycle, and dynamic information relating to the cardiac motion may be extracted by examining and tracking the 4D data throughout a complete cardiac cycle [3]. The extracted motion parameters can then be employed to assess anatomical morphology, to diagnose cardiac dysfunction, to track disease, and to guide surgical interventions. If the dynamic data are synchronized with the patient ECG signal and registered to the patient, a virtual representation of the patient's dynamic anatomy can be provided to provide a context during minimally invasive cardiac procedures [4], enabling the surgeon to visualize the full volumetric presentation of the beating heart in real time.

New techniques for efficiently modeling and rendering these 4D cardiac datasets, along with accurate surgical instrument tracking, offer new approaches for minimally invasive cardiac therapy. Such techniques reduce the side effects of open chest cardiac surgery, and are often facilitated with robotics [5]. Further progress in this direction has been made by introducing surgical tools into the chambers through the heart wall to perform surgery inside the beating heart

* Corresponding author. Tel.: +1 519 663 5777x24136; fax: +1 519 663 3900.
E-mail address: qzhang@imaging.robarts.ca (Q. Zhang).

0895-6111/$ – see front matter © 2009 Elsevier Ltd. All rights reserved.
doi:10.1016/j.compmedimag.2009.04.002

[6]. In these cases, high quality real-time cardiac dynamic volume visualization is key to the continued development of the intracardiac intervention planning and surgery. Stereoscopic imaging can be employed to improve both the realism and precision of these image-guided procedures.

Volume rendering [7,8], which preserves all the anatomical information of the original volumetric data, is the preferred means for visualizing 3D and 4D data in the treatment of heart and coronary artery disease during minimally-invasive cardiac surgery. However, the rendering process is computationally expensive, and even with graphics hardware support and the use of fast CPUs available in today's computers, high quality real-time dynamic cardiac visualization is still a challenge for volumetric display algorithms. In particular, stereoscopic 4D cardiac visualization incurs a double computational cost, since it must render the volume to the left and right graphic depth buffers separately for each frame. Therefore, visualization methods such as multiplanar reformation (MPR) [9,10], surface rendering (SR) [11], and maximum intensity projection (MPI) [12,13] are often employed instead to decrease the computational cost and increase the display speed. However, in this process, either 2D methods are used to display 3D objects, or only a small fraction of the volumetric data is used to construct and display the image. Under these conditions, 3D anatomical context and relationships must be mentally reconstructed by the observer. In addition, it is difficult to extract and render the surface of some imaging modalities, such as ultrasound images, which must be fused with the rendered MRI or CT volume. Thus, these three substitution strategies can only be employed as adjunct display methods, rather than optimal visual solutions for rendering dynamic medical data. In an attempt to address these challenges, special purpose hardware VolumePro (TeraRecon, San Mateo, Calif) [14] and commercial workstation with InSpace (Siemens Medical Solutions) [15] have been employed for 4D cardiac volume rendering without ECG gating. However, the Cube-4 volume rendering architecture implemented on the VolumePro board with only orthogonal projection or the texture-mapping based hardware acceleration used in the InSpace system make such solutions infeasible, since they cannot provide optimal realism or high image quality for routine clinical applications.

In this paper, we present our solutions to some of these challenges. Modern graphics processing units (GPUs) have recently demonstrated a dramatic increase in programmability, parallelism and computational precision [16,17]. Such flexibility has enabled us to implement the light intensity emission radiation model for dynamic volume rendering and manipulation directly on a modern GPU [18]. This rendering scheme performs real-time raycasting computations directly on the GPU vertex and fragment processors in parallel, providing image quality that is arguably better than that of slower, non-interactive software-based raycasting implementations, but with a rendering speed that is competitive with the classical, but inferior quality, volume rendering approaches based on texture mapping. In addition, we propose a new dynamic volume texture binding technique, which allows us to dynamically bind the 3D texture of the cardiac volumes to the GPU-accelerated rendering engine, while visualizing the beating heart without serious performance degradation. In our application, we use ECG signals to retrospectively synchronize the beating volume rendered heart to the actual patient. We also present a novel algorithm, which operates in conjunction with ECG gating, to compensate for rendering latency. In addition, a dynamic multiresolution approach is described that allows a region of interest to be emphasized without compromising display speed, and a new color and opacity adjustment strategy is presented to improve the uniformity of the rendered multiresolution cardiac images. Finally, we describe a stereoscopic visualization technique that has been added to the software system to increase the reality and precision of the 4D cardiac volume rendering and manipulation environment.

2. Volumetric display and stereoscopic rendering

The introduction of modern X-ray multidetector computed tomography (MDCT), magnetic resonance imaging (MRI), and positron emission tomography (PET), has enabled the acquisition of highly detailed medical images. However, the large amount of data that comprises these images creates a challenge for their effective display and interpretation. Real-time stereoscopic volume rendering can address many of these limitations, and provide support for a variety of medical applications, such as diagnosis, radiation therapy planning and guidance, surgical planning, reconstructive surgery, medical education and research [19]. In this section, we briefly introduce some medical applications of monoscopic and stereoscopic volumetric image display techniques.

2.1. Rendering methods

Direct volume rendering (DVR) is a major technique for 3D medical data display, creating images of an entire dataset without explicitly extracting surfaces corresponding to features of interest. DVR is therefore a promising technique for visualizing complex anatomical structures within a 3D dataset, eliminating many of the disadvantages of MPR, SR and MIP image rendering methods. Commonly used volumetric display methods include software-based raycasting [7,8], shell rendering [20], splatting [21], shear-warp [22], hardware-based texture mapping [23,24], and GPU-accelerated rendering [25].

There are many medical applications of volume rendering algorithms. Audigier et al. [26] integrated splatting and raycasting algorithms with interactive 3D medical image segmentation, providing users with visual feedback at the end of each segmentation iteration. Lei et al. [27] described shell rendering to provide a high-quality volume visualization of vascular information in a single 3D contrast-enhanced magnetic resonance angiography (CE-MRA) image data. Texture mapping techniques were employed by Holmes et al. [28] to visualize trans-urethral ultrasound (TUUS) for prostate treatment. Levin et al. [29] used this technique to interactively visualize multimodality dynamic cardiac datasets with 4D motion segmentation on standard consumer graphics cards, and a similar algorithm was employed by Lehmann et al. [30] to render the beating heart in real-time, in which a hierarchical memory structure was employed for high cache and bandwidth efficiency. A viewing-direction based dynamic texture slice resampling algorithm, running on graphics hardware, was proposed in our previous work [31]. This approach improved the image quality and rendering speed of the texture mapping based volume rendering algorithms for real-time display. Usually, patients with cardiac arrhythmia can be treated by catheter ablation of the affected anatomic region, and Wang et al. [32] have demonstrated the use of volume visualization of 3D ultrasound to guide cardiac ablation procedures. Using such visual guidance, the catheter can be precisely positioned to point toward the target tissue region, and the ablation can be accomplished accurately, avoiding the time-consuming "trial-and-error" iterative sequence in this procedure [33].

2.2. Stereoscopic rendering

Stereoscopic visualization adds depth to the perceived image by creating a binocular view of an object [34], which enables observers to more easily understand complex vessel structures, providing them with more efficient assessment of diseases such as vascular aneurysms, stenotic narrowing of vessels, and vascular malformations [35]. Stereoscopic display is a compelling adjunct

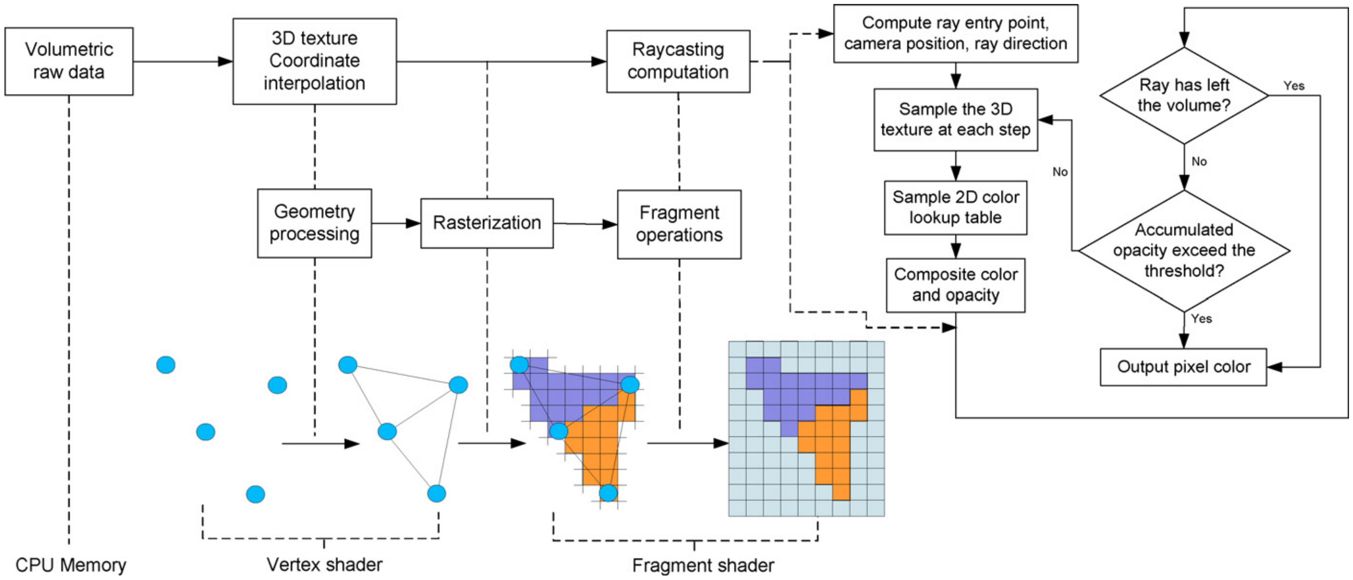

Volumetric raw data
3D texture Coordinate interpolation
Raycasting computation
Compute ray entry point, camera position, ray direction
Sample the 3D texture at each step
Sample 2D color lookup table
Composite color and opacity
Ray has left the volume?
Yes
No
Accumulated opacity exceed the threshold?
No
Yes
Output pixel color
Geometry processing
Rasterization
Fragment operations
CPU Memory
Vertex shader
Fragment shader

Fig. 1. Schematic description of the programmable hardware graphics pipeline and the GPU-based volume raycasting algorithm. The ray casting computation is performed on the GPU fragment processor, in which an early ray termination technique is embedded in the rendering process.

to surgical navigation, and is used in the operating room (OR) for the integration of digital subtraction angiograms with MRI and CT for neurosurgical planning purposes [36–38]. Stereoscopic rendering has proved to be a major advantage when planning the shape and position of a 3D lesion that must be created precisely with respect to a functional atlas that has been merged with the data, and has also been employed to recover and monitor surgical workflow [39]. To facilitate ultrasound guided needle biopsies, a unified augmented reality (AR) ultrasound platform was developed by Khamene's group [40], who overlaid composited 2D and 3D ultrasound images into a stereoscopic video view. Recently, Wang et al. [41] employed a GPU-based rendering algorithm to stereoscopically display volumetric CT images at interactive rates for lung cancer screening.

3. Materials and methods

The challenge in implementing a 4D volume rendering system is to efficiently handle a vast amount of volumetric data. A real-time interactive system must employ high performance hardware, based on multiprocessing and parallel memory organization. Current graphics processing units (GPUs) are programmable with their performance increasing at a much higher rate than that of typical CPUs. In addition, the inherent parallelism of GPU's can be employed to enhance performance. Therefore, GPU-based algorithms have begun to provide the most efficient means for rendering large volumetric data sets. In this section, we first introduce the programmable graphics GPU pipeline, and then provide a description of the optical density radiation model, which is implemented on GPUs for raycasting computation. We then demonstrate our dynamic multiresolution techniques with opacity and color adjustment for image uniformity improvement, along with the stereoscopic rendering algorithm. Finally, we describe the procedure for retrospectively synchronizing the dynamically rendered cardiac volume with ECG signals, along with the entire real-time 4D cardiac image rendering and manipulation system.

3.1. Programmable graphics pipeline

Modern graphics hardware is designed to operate on large continuous streams of vertex and fragment data. A typical graphics pipeline is described in Fig. 1: vertices (points in 3D space) are first transformed and assembled into primitives, such as triangles, polygons, rectangles, etc., by the vertex units. These primitives are then rasterized into geometric fragments, composed of a large stream of elements. Texture mapping is performed as a part of the rasterization procedure, during which the texture image is interrogated for the correct color to be added to the fragments. These fragments are further processed by the fragment units, and then written to the frame buffer, which is copied into the graphics card video memory to produce images on the screen.

In older generation graphics pipelines, the geometry sent to the GPUs was static, and the operations that could be performed on fragments were limited. GPU functions were therefore restricted to certain fixed pipelines. The new generation of graphics cards has evolved to a level where the vertex and fragment units are now user programmable by developing small programs called vertex shaders and fragment shaders, which permit direct control of the vertex and fragment processors during the graphics pipeline. Modern GPUs support the Shader Model 4.0 features [17], such as dynamic branching, multiple rendering targets, and computation of texture coordinates per pixel, etc. A further improvement in the new GPUs is the increase in pixel depth from 32 bits per pixel to 128 bits per pixel, which means that each red, green, blue, and alpha component can now have 32-bit floating point precision throughout the graphics pipeline. The increased data precision combined with the enhanced programmability means that complicated scientific visualization and computation algorithms can now be implemented directly on GPUs with much greater efficiency, accuracy, and realistic visual effects than ever before.

3.2. Optical density radiation model

The volume rendering algorithm discussed in this paper is based on the particle light absorption and emission model, which is derived from the volumetric density scattering and diffuse reflection model introduced by Blinn [42]. It describes light passing through, and being reflected by clouds of randomly distributed small particles with similar single scattering properties. In this section, we derive a optical model for the raycasting computation on a GPU, which we refer to as the optical density radiation model, and illustrate the volume rendering pipeline from both physical and mathematical perspectives.

The raycasting procedure involves extending a ray from a specified camera position through each pixel on the screen viewport, and into the volume. Consider a volume with the intensity $\rho(x(t), y(t), z(t))$. For a ray transmitted from t_a to t_b, the associated optical density $\tau(t_a, t_b)$ can be expressed by Eq. (1)

$$\tau(t_a, t_b) = \int_{t_a}^{t_b} \rho(x(t), y(t), z(t))\mathrm{d}t \tag{1}$$

We assume that the above casting ray is sampled at N intervals that can be uniform or variable, and this sampling rate satisfies the Nyquist criterion, so the original image can be reconstructed from these sampling points without aliasing. Trilinear interpolation is employed to acquire scalar values at these $N+1$ points. For the above casting ray, the light emitted by a voxel at the sampling position $t_i = (x(t_i), y(t_i), z(t_i)), (0 \le i \le N)$ can be set as $I(t_i)$. When it is transmitted to the camera position t, the light intensity $I(t_i)$ will be attenuated because of the absorbtion and scattering by the voxels it encounters on the ray. The attenuation is proportional to the Poisson distribution of the volume optical density $\mathrm{e}^{-\tau(t_i,t)}$, so the light arrived at the camera is $I(t_i)\mathrm{e}^{-\tau(t_i,t)}$. If we set the position where the casting ray enters the volume to t_a, and its exit point to t_b, we have the relationship: $t_0 = t_a$ and $t_N = t_b$ $(t_a < t_i < t_b, 0 < i < N)$. Thus, the total accumulated light arrived at the camera along this casting ray is:

$$I(t_a, t_b) = \sum_{i=0}^{N} I(t_i)\mathrm{e}^{-\tau(t_i,t)} \tag{2}$$

Since every point between t_a and t_b on the casting ray emits light, this illumination will be transmitted to the camera position with an approximate attenuation. The distance between the points can be assumed to be infinitesimally small, so the total light arriving at the camera $I(t_a, t_b)$ can be expressed by the following integral

$$I(t_a, t_b) = \int_{t_a}^{t_b} I(s)\mathrm{e}^{-\int_{t_a}^{s}\rho(x)\mathrm{d}x}\mathrm{d}s \tag{3}$$

We assume the sampling interval $\Delta t = (t_b - t_a)/N$ to be sufficiently small that the light emitted by every point in the Δt is constant. The Riemann sum can be used to numerically approximate the ray integral expressed by Eq. (3).

$$\begin{aligned} I(t_a, t_b) &\approx \sum_{i=0}^{N}\left[I(t_a + i\Delta t)\Delta t \mathrm{e}^{-\sum_{j=0}^{i-1}\rho(t_a + j\Delta t)\Delta t} \right] \\ &= \sum_{i=0}^{N}\left[I(t_a + i\Delta t)\Delta t \prod_{j=0}^{i-1} \mathrm{e}^{-\rho(t_a + j\Delta t)\Delta t} \right] \end{aligned} \tag{4}$$

If we use the first-order Taylor series expansion at the point Δt= 0 to approximate the light attenuation term $\mathrm{e}^{-\rho(t_a+j\Delta t)\Delta t}$, then the intensity $I(t_a, t_b)$ of the light arrives at the camera in Eq. (4) can be expressed as

$$I(t_a, t_b) \approx \sum_{i=0}^{N}\left\{ I(t_a + i\Delta t)\Delta t \prod_{j=0}^{i-1}[1 - \rho(t_a + j\Delta t)\Delta t] \right\} \tag{5}$$

Now we use I_i to express the light intensity sum from every voxel on the ray segment from the point $t_a + i\Delta t$ to $t_a + (i+1)\Delta t$ $(0 \le i \le N-1, I_i = I(t_a + i\Delta t)\Delta t)$, and we also use α_j to express the opacity sum of all the voxels on the ray segment from the point $t_a + j\Delta t$ to $t_a + (j+1)\Delta t$ $(0 \le j \le i-1, \alpha_j = \rho(t_a + j\Delta t)\Delta t)$. Then Eq. (5) can be reduced to

$$I(t_a, t_b) \approx \sum_{i=0}^{N}\left\{ I_i \prod_{j=0}^{i-1}\left[1 - \alpha_j\right] \right\} \tag{6}$$

Eq. (6) is the front-to-back ray casting computation formula. On the casting ray, at every sampling point p_i $(0 \le i \le N)$, there is a scalar value s_i derived from trilinear interpolation. A four-channel transfer function $\Gamma(s_i)$ based on the Phong optical model [42] is used to map the scalar value s_i to colors $C_{i\gamma}$ $(\gamma = r, g \text{ and } b)$ and opacity α_i. The ray casting computation (6) is used to calculate each channel of the final color for the screen pixel of this casting ray. This process is repeated for each screen pixel and results in an image of the volume.

3.3. GPU-based implementation

Raycasting is a popular technique for rendering scalar volumes, and is well suited for parallel implementations since the rays are processed independently. It is based on the above optical density radiation model, which describes the real world physical phenomena, i.e., the behavior of light interacting with particles in a volume. Ray casting has the advantage of flexibility and high rendered image quality. While software based implementations are too slow for a real-time medical applications, their implementation in modern GPUs can overcome this limitation. Our raycasting algorithm runs directly on the graphics vertex and fragment processors, and its pipeline is described by Fig. 1 and the following steps:

(i) Load the volumetric raw data and color table to the graphics memory as 3D and 2D textures. Transform their texture coordinate from object space to local texture space. These transformed texture coordinates are first loaded to the vertex shader, interpolated, then output to the fragment shader.

(ii) In the fragment shader, the interpolated 3D texture coordinates are first scaled by factors derived from the dimension of the volumetric raw data, and are then used as the entry point for the casting ray. The algorithm then obtains the camera position from the model view matrix. The casting ray entry point and the camera position are used to compute the ray direction.

(iii) The 3D texture volume is sampled at each sampling point along the casting ray, acquiring voxel values from the α channel, in which the scalar value is stored. The values derived from every two consecutive sampling steps are used to sample the 2D texture based color lookup table to get the r, g, b color and opacity α. Alpha blending is used to assign the derived color value to the destination color.

(iv) Check the accumulated opacity, if it has exceeded a preset opacity threshold, exit the loop, go to (v); otherwise, update the sampling position, go to (iii). If the casting ray has left the 3D texture volume, exit the loop, go to (v).

(v) Alpha blend the derived destination color with the background color, and set the result as the output fragment color.

3.4. Dynamic multiresolution

Physicians are often interested in only part of the medical volume, and may need to change the volume of interest (VOI) interactively to find the lesion, tumor and other tissue abnormalities, but the entire volumetric context is still needed in the visualization procedure. However, most published algorithms are based on a hierarchical data structure [43,44], which is determined by the VOI and built before the rendering process, so it is difficult to interactively change the high resolution VOI region, which makes this technique unsuitable for clinical applications. Therefore, we

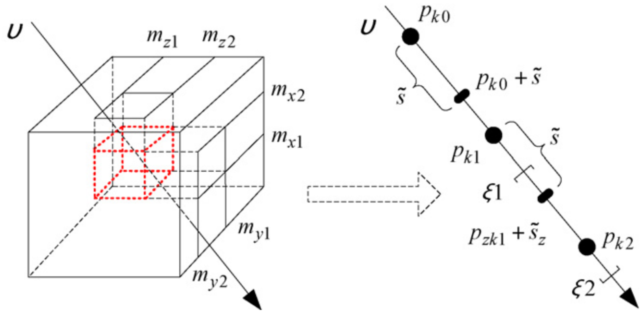


Fig. 2. Dynamic multiresolution pipeline. Left: a volumetric dataset with a VOI $V_x \cap V_y \cap V_z$; right: dynamic sampling approach with multiple sampling step selection criterion.

propose a new GPU-based dynamic multi-resolution technique, in which the VOI can be changed in real-time. We also apply a color and opacity adjustment technique to improve the visual uniformity of the final multiresolution volume rendered images.

As shown in Fig. 2, we assume $V_\sigma = [m_{\sigma 1}, m_{\sigma 2}]$, $\sigma \in \{x, y, z\}$, so the VOI can be calculated by: VOI = $V_x \cap V_y \cap V_z$. During the volume rendering process, the ray casting computation is performed on the GPU, and the $m_{\sigma 1}$ and $m_{\sigma 2}$ are dynamically transferred from CPU main memory to GPU fragment shader as a data stream in real-time. For a specific viewing direction v and the corresponding resampling step $\hat{s}$, we employ two dynamic resampling steps: large $\tilde{s}$ (in our implementation, $\tilde{s} = n\hat{s}$, $n = 2, 3, \cdots$) and small $\bar{s}$ (in our implementation, $\bar{s} = \hat{s}$). Then, for every sampling point on the casting ray, we check which sampling step size is to be used for the following step. For a position p, the small sampling step $\bar{s}$ is used if either p or $p + \tilde{s}$ is located in the range of interest VOI. Otherwise, the large sampling step $\tilde{s}$ is employed. For example, for points p_{k1} and p_{k2}, the following sampling step will use $\bar{s}$; while for point p_{k0}, $\tilde{s}$ will be employed. Therefore, the VOI will be rendered with uniform high resolution in all viewing directions, while the other regions will be displayed with low resolution.

From Eq. (6), we can see that the total intensity accumulated on the casting ray at the current sample point t_i is the intensity at this point I_i multiplied by all the transparencies $(1 - \alpha_j)$ encountered so far on this ray. Fig. 3 illustrates a front-to-back compositing procedure, in which the sample points are composited starting at point t_a on the ray and composite toward the exiting point t_b. For point t_i with intensity I_{t_i} and opacity α_{t_i} on this ray, the intensity I_{in} and opacity α_{in} are the composited values up to this point. The new values I_{out} and α_{out} are the results of front-to-back compositing I_{t_i} and α_{t_i} with I_{in} and α_{in}, which are calculated by:

$$I_{out} = I_{in} + I_{t_i} \times (1 - \alpha_{in}) \tag{7}$$

$$\alpha_{out} = \alpha_{t_i} + \alpha_{in} \times (1 - \alpha_{t_i}) \tag{8}$$

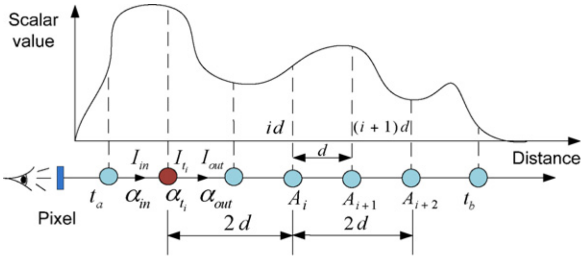


Fig. 3. Front-to-back compositing procedure with multiresolution sampling strategy. Sample points are composited together in a front-to-back order, starting from the point t_a and compositing toward the existing point t_b. d is the original sampling step, and $2d$ is the new sampling step used in the no VOI volumetric regions.

When we sample the casting ray with low resolution, we lose some points, and since the intensity and opacity of these missing points are unknown, the final results using the compositing of Eqs. (7) and (8) will be changed. Therefore, there will be color artifacts in the low resolution region, as shown in Fig. 12(a) in subsection 4.1.4. However, we can assume that the color and opacity of the missing points can be estimated by their neighbors. For example, in Fig. 3, when we use the double sampling step, point A_{i+1} will be missed. When the composited color and opacity of the point A_{i+2} are computed, we introduce a scaling factor λ to make this adjustment, changing Eq. (7) to

$$I_{out} = I_{in} + I_{A_{i+2}} \times (1 - \alpha_{in}) \times \lambda \tag{9}$$

Here, the scaling factor λ can be changed during the dynamic multiresolution rendering process. The setting of λ depends on the relationship of the low-resolution contextual volume and the high-resolution VOI, the intensity to color and opacity mapping rule, as well as the rendered dataset. The initial value of λ is set to 1.0, and the adjustment step is 0.1. Generally, the range of λ is between 1.5 and 2.5. The final value of λ is based on visual inspection.

This user-controlled color and opacity adjustment strategy makes the color and opacity in the low resolution region almost the same as that in the high resolution VOI, increasing the visual uniformity of the resulting multiresolution volume rendered cardiac images, as illustrated in Fig. 12.

3.5. Stereoscopic rendering

3D stereo display has proved to be valuable in neurosurgical navigation [36–38], and is also commonly used in augmented reality (AR) enhanced surgical applications [39,40]. A common approach is the so called “toe-in” method, which the camera has a fixed and symmetric aperture (Fig. 4 (a)). Each camera is pointed at a single focal point. This approach introduces vertical parallax, most noticeable for objects in the outer field of view, causing discomfort for the viewer.

To overcome this problem, we employ a parallel axis asymmetric frustum perspective projection algorithm. The view vectors for each camera remain parallel during the stereoscopic imaging process, therefore no vertical parallax is introduced and there is no distortion in the created 3D stereo views. We create a perspective view for each eye, and the rendering engine renders the volume to the left- and right-eye frame buffer, respectively. Objects that lie in front of the projection plane have negative parallax, appearing in front of the computer screen, while objects behind the projection plane have positive parallax, placing their images behind the screen.

In our rendering system, the OpenGL API *glFrustum*$(left, right, bottom, top, l_n, l_f)$ is used to create the perspective projection, setting the field of view of the camera. Through experiment, we find the optimal parameter settings for the stereoscopic 4D cardiac rendering, delivering a balanced stereo image that makes use of both positive and negative horizontal parallax without introducing vertical parallax.

As shown in Fig. 4, the parameters l_n and l_f are the distance from the camera to the near and far clipping plane, respectively, $d = l_n + 2 \times (l_f - l_n)/3$ is the focal distance, i.e., the distance from the camera to the projection plane. $\theta = 45^\circ$ is the camera aperture, and c is the separation of the cameras. We then compute the following parameters, which are used to create the perspective frustums: $\tilde{x} = d \times \tan(\theta/2), \tilde{y} = d \times \tan(\theta/2) \times \zeta, c_x = c/2 = \tilde{x} \times l_n/d, c_y = c_x \times \zeta$, here $\zeta = h/w$ is the aspect ratio of the height (h) and the width (w) of the display window.

We then build the two perspective view frustums for the left and right eyes, respectively. First, for the left eye, the scene is translated to the right by c_x using *glTranslated*$(c_x, 0, 0)$, while for the right

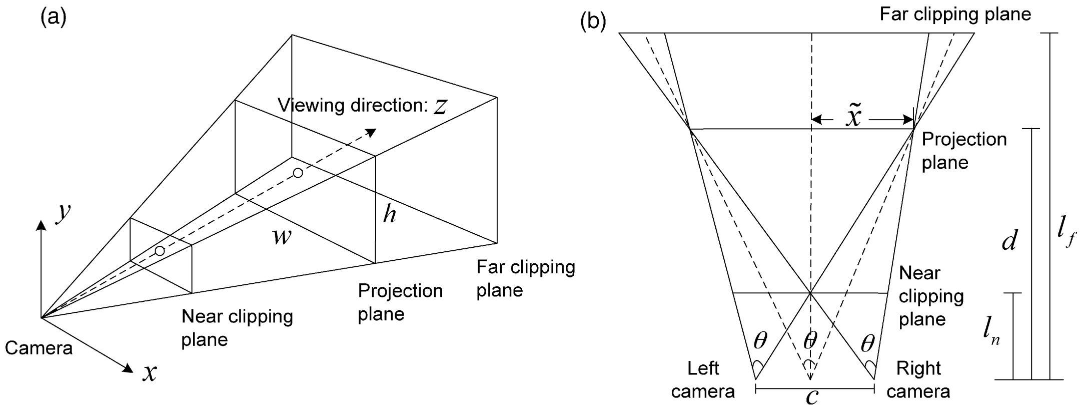


Fig. 4. Diagram showing the 3D perspective view frustum (a); and the asymmetric perspective view frustum 2D (*xz* plane) projection (b). θ is the camera aperture, d is the focal distance, and c is the distance of the camera separation.

eye, it is moved to the left by c_x using *glTranslated*$(-c_x, 0, 0)$. Then, we introduce a adjustment factor $\gamma = 0.2 \times c_x$ to optimize the horizontal parallax. The computed frustum parameters are $top = c_y$, $bottom = -c_y$, $right = c_x + \varepsilon$, and $left = -c_x + \varepsilon$. For the right eye, the frustum is shifted to the left, so $\epsilon = -\gamma$, while for the left eye, it is moved to the right, so $\varepsilon = \gamma$.

During 3D stereo visualization, we use either a ViewSonic CRT monitor, Nvidia Quadro 4 graphics hardware (NVIDIA Corporate, Santa Clara, USA) for display, where Silicon Space Shutter Glasses (3DTV Corp., USA) are employed to synchronize the stereo pairs for the two eyes, or a Dimension Technologies Inc. auto stereoscopic display that does not require the use of special eyewear.

3.6. 4D image acquisition

The latest implementations of MDCT can acquire data from the beating heart and generate 4D data sets with a retrospective ECG-gated reconstruction algorithm. Here we report the results from a dynamic cardiac phantom (Chamberlain Group, MA, USA), a normal canine heart, and clinical patient data, acquired using a GE Healthcare (GEHC) Lightspeed volume CT (VCT) scanner. Each dataset consists of ten volumes over a single cardiac cycle acquired with retrospective ECG gating.

3.6.1. Retrospective ECG-gating

ECG-gating methods may be achieved in two ways: retrospective and prospective [45,46]. Retrospective ECG gating is needed for helical scanning, where a continuous spiral scan is acquired with the ECG-signals recorded simultaneously. To clearly visualize the defined time points in the cardiac cycle, the acquired scan data are selected for image reconstruction with respect to a pre-defined cardiac phase registered with the recorded ECG signals. A specified R-wave delay time defines the start point of data that is used for image reconstruction. On the other hand, in prospective ECG gating, the ECG signal is used to trigger data acquisition in a user-defined phase of the cardiac cycle (usually diastole), where the least cardiac motion occurs.

The advantage of retrospective over prospective ECG-gating, is that the data acquisition is independent of the patient's heart rate. Therefore, this technique is less sensitive to arrhythmia, and it is possible to visualize the imaged region during different time points over the cardiac cycle. The disadvantage of this ECG-gating technique is the prolonged data post processing procedure and the associated higher radiation dose acquired by the patient, as the X-ray beam is on continuously during the scan. Since we use the ECG to gate the volume rendered heart beating speed during the 4D cardiac data real-time visualization process, we employed retrospective ECG-gating during 4D cardiac data acquisition.

3.6.2. Phantom data

The beating heart phantom we used in some of our examples provides its own ECG trigger. These signals were recorded during the acquisition of 3D spiral MDCT images, using a GEHC Lightspeed VCT scanner. The X-ray tube current was 200 mA during acquisition, the beam energy was held at 120 kVp, the table translation speed was 3.75 mm/rotation, and the imaging time was 5 s. The field of view and image resolution were 16.2 cm and 512 × 512, respectively. The reconstructed voxel size was $0.488 \times 0.488 \times 0.625\,\text{mm}^3$, and the ECG was used to retrospectively reconstruct 10 volumes throughout the cardiac cycle.

3.6.3. Canine data

MDCT images of a normal canine heart were acquired on the same GEHC Lightspeed VCT scanner. A standard 3 lead ECG configuration and an ECG Cardiac Trigger Monitor (Model 3150, Ivy Biomedical Systems, Inc., Branford, Connecticut, USA) were used to acquire the actual ECG signals. The R-wave peaks in the ECG signals were detected by the monitor and produced a 100 ms tirgger pulse (Fig. 5), beginning at the peak of each R-wave. The animal was anesthetized with isoflurane, paralyzed with vecuronium, and

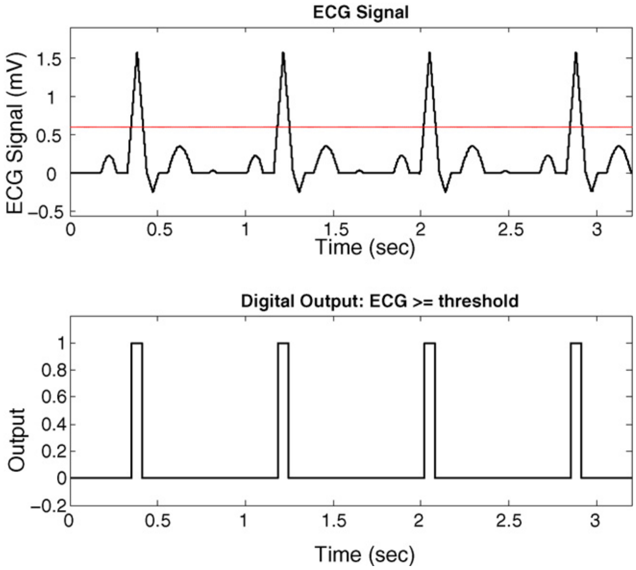


Fig. 5. Retrospectively recorded ECG waveforms with corresponding monitoring digital trigger pulse output. Upper figure shows an ECG waveform with the trigger threshold (red line), and lower figure illustrates the triggered digital outputs. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of the article.)

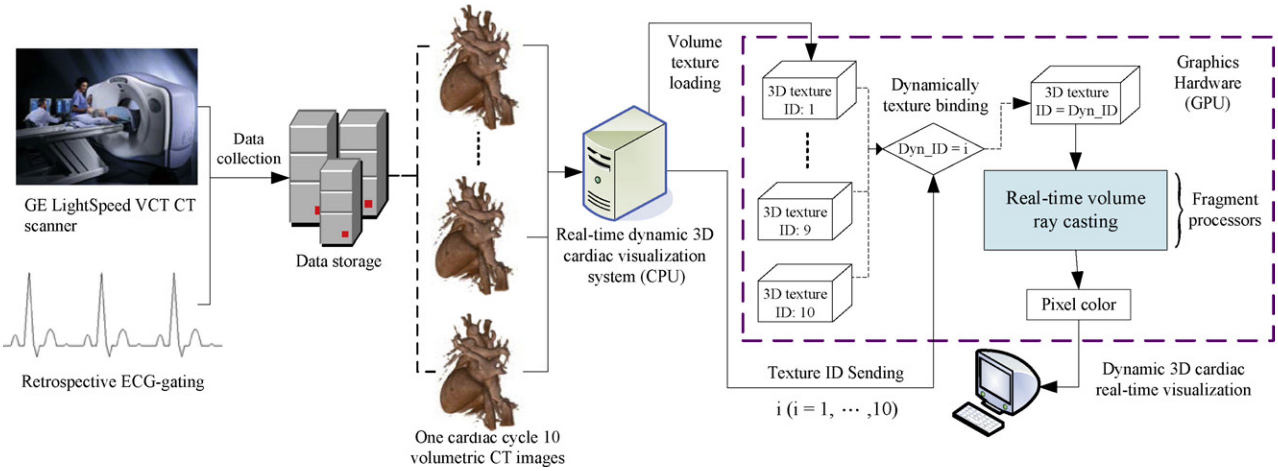


Fig. 6. The process of retrospective ECG-gated 4D cardiac image real-time visualization. Ten CT images were acquired from a GEHC Lightspeed VCT scanner with time information during the cardiac cycle derived from the corresponding recorded ECG signal. The required 4D data were loaded into the graphics cards as a set of volume textures, and a dynamic texture binding technique is employed to real-timely bind different volume texture ID with the GPU-accelerated raycasting volume rendering engine, which directly runs on the GPU vertex and fragment processors. This 4D CT cardiac visualization process renders each of the cardiac volumes as a new frame in real time.

mechanically ventilated to minimize artifacts due to breathing during the entire procedure. This animal study was carried out under a protocol approved by the Research Ethics Board of the University of Western Ontario, Canada. After placing the dog in the CT gantry, intracardiac contrast was enhanced by an intravenous injection of a contrast agent bolus, followed 12 seconds later by a dynamic 3D MDCT scan.

The ECG recorded during imaging was used to retrospectively bin projections into their relevant phases, and dynamic 3D MDCT images were reconstructed with $0.35 \times 0.35 \times 1.25\ \text{mm}^3$ voxels. Each acquisition was performed in helical mode using technique factors of: 120 kVp, 400 mA, slice thickness 1.25 mm, and the table translation speed and field of view of 3.75 mm/rotation and 18 cm, respectively. Each 4D image comprised ten 3D frames equally spaced within the R-R interval. The length of this interval was set by the subjects' heart rates, which ranged from 70 to 85 bpm.

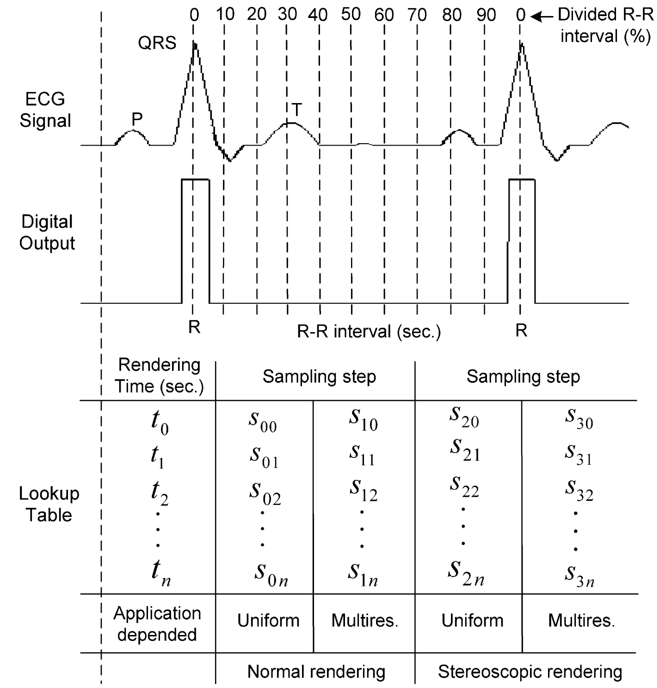
3.6.4. *Human data*

Human cardiac MDCT images were also acquired using the same GEHC VCT scanner, as part of standard clinical imaging studies. The X-ray tube voltage, current and field of view were 120 kV, 600 mA and 20 cm, respectively, and the time of breath hold was about 75 rotations through 12 cm during 7 s. ECG leads were placed on both wrists and the left forearm of the patients to avoid artifacts. The R-wave peaks in the ECG signals were detected by the monitor, which generated a digital trigger pulse (Fig. 5). Fifty seconds after intravascular injection of 100 ml nonionic Omnipaque iodinated contrast agent (300 mgI/ml) at 5 ml/s, the scan was started with ECG-gating technique. The average heart rate of the subject was 72 bpm, resulting in an R-R interval length of approximately 0.83 s. The images were reconstructed with a retrospective ECG-gated reconstruction algorithm to create 3D data sets from raw data at each of ten equally-spaced phases throughout the cardiac cycle. Finally, the acquired data were binned according to the ECG signal and reconstructed with $0.39 \times 0.39 \times 0.75\ \text{mm}^3$voxel dimensions.

3.7. *Cardiac dynamic rendering*

Fig. 6 describes the cardiac dynamic rendering process. First, ten cardiac volume data sets (phantom, canine or human) were reconstructed at ten equally spaced time points over one cardiac cycle using the raw data and their corresponding retrospective ECG signals acquired from the MDCT scanner, and the procedure is described above. These ten reconstructed volumes were saved in the data storage workstation. The data were then loaded into graphics card memory as a set of 3D textures, during which time a unique texture ID was assigned to each of them as each 3D texture was generated. Next, during the real-time dynamic rendering process, our real-time 3D raycasting rendering engine ran directly on the GPU's vertex and fragment processors, with rays being cast from each screen pixel through the texture volume to perform the pixel color computation. The actual texture volume through which the rays traversed was dynamically determined by the texture ID, during which the dynamic texture binding technique was used, enabling our GPU-based 3D rendering engine to display different cardiac volume at different time points, without requiring that they be reloaded during the rendering process. Finally, we sent the texture ID of the ten MDCT volumetric images to our rendering engine in a loop sequence gated with digitized ECG signals, so that the beating heart can be dynamically visualized with high image quality. The ECG gating the dynamic beating heart rendering procedure is formulated as follows:

- *Initialization*: the ECG leads are placed on both wrists and the left forearm of the patients to acquire several ECG signals. A programmable ECG digitizer is used to convert the clinical ECG signals to digital signals: set the threshold $\xi = (\max(ecg) - mean(ecg))/2$ to estimate the ECG signal ecg R-peak point (Fig. 5), when $ecg \geq \xi$, then the digital trigger outputs 1, otherwise outputs 0. Based on the digital trigger outputs, calculate the R-R interval ΔT of a cardiac cycle.
- *Rendering speed estimation*: as shown in top of Fig. 7, the R-R interval is temporally divided into ten equal intervals, with increments of 10%, corresponding to the ten cardiac volume reconstruction time points. The volume rendering time Δt for each volume is $\Delta T/10$.
- *Lookup table construction*: as illustrated in the bottom of Fig. 7, we use the calculated Δt to build a sampling step lookup table. First, determine the rendering model (normal or stereoscopic), and determine whether or not to use the dynamic multiresolution technique. Next, build the lookup table with the time step 0.02 second: $t_0, t_1, \cdots, t_n$ ($10 \leq n \leq 20$, Δt is the median), and the corresponding sampling steps are $s_{i0}, s_{i1}, \cdots, s_{in}$, $i \in \{0, 1, 2, 3\}$.
- *ECG-gating*: during the rendering process, the actual volume display is delayed by one cardiac cycle with respect to the ECG signal. When the system receives the calculated R-R interval ΔT,



	Rendering Time (sec.)	Sampling step		Sampling step	
Lookup Table	t_0	s_{00}	s_{10}	s_{20}	s_{30}
	t_1	s_{01}	s_{11}	s_{21}	s_{31}
	t_2	s_{02}	s_{12}	s_{22}	s_{32}
	⋮	⋮	⋮	⋮	⋮
	t_n	s_{0n}	s_{1n}	s_{2n}	s_{3n}
	Application depended	Uniform	Multires.	Uniform	Multires.
		Normal rendering		Stereoscopic rendering	

Fig. 7. Diagram illustrating the ECG signal with P, QRS, and T point, and the corresponding digital trigger outputs. There are 10 divided R-R interval (%) with 10% increase at each step. The bottom shows the sampling step lookup table, there are four columns corresponding to the four rendering methods.

computes Δt, then checks the lookup table to find $t_i \le t \le t_{i+1}$ $(i = 0, 1, \cdots, n-1)$. The sampling step $s = s_i + \alpha(s_{i+1} - s_i)$, where the interpolation factor $\alpha = (t - t_i)/(t_{i+1} - t_i)$.

- *Adjustment*: for the calculated sampling step s, the actual rendering time $\tilde{t}$ may not equal to the t, so we compute the changing ratio $\kappa = (\tilde{t} - t)/t$, then a new sampling step $\tilde{s} = s \times (1 + \kappa)$ will be used for the next cardiac volume rendering.

3.8. System description

In this project, we developed a comprehensive dynamic real-time visualization and manipulation software system to visualize and manipulate dynamic cardiac images. If the ECG signal is acquired during the image acquisition, it can be used to synchronize the dynamic data set with the patient, by reference to the patient's clinical ECG signals in the operating room (OR).

Our system is designed with an object oriented approach, and is implemented using C++, OpenGL, GL Shading Language (GLSL) and extended Visualization Toolkit (VTK, Kitware, Inc., USA) classes. The KWWidgets toolkit is used to build the graphics user interface (GUI). Our new algorithms and cardiac function diagnosis techniques are designed and implemented according to the VTK working pipeline (Fig. 8): the mapper class provides the geometric representation for an actor, and when the renderwindow asks the graphics pipeline to render the image, the mapper converts the image into the OpenGL graphics representation and renders it.

As shown in Fig. 8, this software platform supplies functions for GUI and IO management, and multiple rendering methods, i.e., direct volume rendering (DVR), maximum intensity projection (MIP), surface rendering (SR) and multiplanar reformation (MPR, axial, coronal, and sagittal) display. To help users to simplify the process of navigation and interacting with the medical datasets, it supplies interactive transfer function, color lookup table, clipping plane and measurement tools for rendered cardiac image adjustment, and it also supplies the opacity, position, and windows-level control for the surface and cross-plane display. In particular, it acquires the real-time digitized ECG signals from the patient, and supplies functions to synchronize the dynamic volume visualization with the clinical ECG signals. In this process, the dynamic multiresolution feature can be used to emphasize the volume of interest (VOI) while maintaining a real-time rendering speed. In addition, stereoscopic rendering is added to enhance the realism of the displayed image. This software platform is extensible, and can include modules such as tracking, endoscopes, image segmentation and registration, etc., to supply more functions for comprehensive clinical diagnosis, planning and image-guided minimally-invasive cardiac interventions.

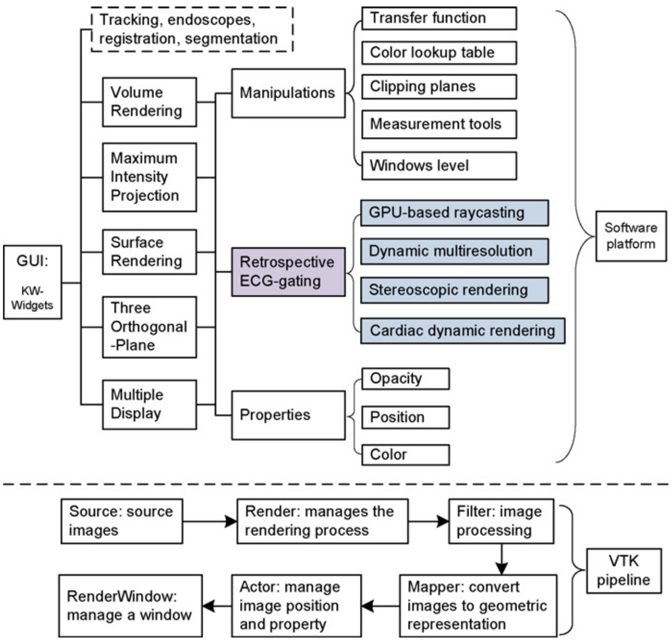


Fig. 8. The software system workflow (top) and the corresponding VTK image processing and visualization pipeline (bottom).

4. Results

We have implemented our real-time 4D cardiac visualization on a hardware system equipped with an Nvidia GeForce GPUs, Pentium D 2.8 GHz dual core processor and 2GB DDR2 main memory, and an ECG Cardiac Trigger Monitor (Model 3150, Ivy Biomedical Systems, Inc., Branford, Connecticut, USA). Our test environment also includes an ECG signal generator, and a programmable ECG signal digitizer, which is used to digitize the received ECG signals, and is directly connected to the computer parallel port for the system real-time binary ECG signal acquisition. The system runs on Windows XP and White Box Enterprise Linux x86 4.0, gcc version 3.4.3. The Nvidia driver version is 169.09.

4.1. Image analysis

We have implemented three volume visualization algorithms: 3D texture mapping (3DTM), software-based raycasting (SOFTRC) and hardware-accelerated raycasting (HWRC), and compared the relative performance of each. The implementation of 3DTM and SOFTRC are based on the algorithms described in the references [23] and [7], respectively, while HWRC is based on the algorithm presented in this paper. The three dynamic cardiac MDCT datasets described above were employed in our experiment for image quality comparisons and rendering performance analysis. The volume rendered images were interactively manipulated and evaluated by three radiologists and medical imaging specialists, assessing the

Table 1
The volume rendered image quality analysis and evaluations of the three volume rendering techniques. The ranking categories include the overall image quality impression (OIQI) and the tissue structure and relationship description (TSRD). The three data sets are utilized in this image quality analysis experiment, which corresponds to the images in Figs. 9–11, respectively.

Dataset	Evaluation	3DTM	SOFTRC	HWRC
Phantom heart	OIQI	1.5	1.5	1.0
	TSRD	2.0	2.0	1.0
Animal heart	OIQI	2.5	1.0	1.0
	TSRD	2.0	2.0	1.0
Human heart	OIQI	2.0	1.0	1.0
	TSRD	2.5	1.5	1.0
Mean	OIQI	2.0	1.2	1.0
	TSRD	2.2	1.8	1.0

rendered image quality of each algorithm. All the rankings in this section are the average scores, based on a 4-point rating scale.

(i) Overall image quality impression (OIQI): 1, excellent, smooth color and opacity changes, without visible artifacts, the small tissues, such as blood vessels, are clearly visible for detailed and reliable evaluation; 2, good with minor visible artifacts, the color and opacity changes are not perfectly smooth, the small structures are clear enough for adequate evaluation; 3, average with visible artifacts, the color and opacity of the rendered image are not smooth, the small structure's visibility compromises evaluation; 4, obvious artifacts, unsatisfactory visibility with inadequate evaluation.

(ii) Tissue structure and relationship description (TSRD): 1, exact structure description with excellent illustration of the tissue relationships without visible artifacts; 2, well delineated tissue structures with minor artifacts, sufficient relationship description, supplying good visual information for correct analysis of the relationship between tissues; 3, vaguely delineated tissue structures with obvious artifacts, but structures can be definitely visualized; the tissue relationships are scarcely illustrated; 4, the small tissue structures were not shown, and the relationships between the tissues could not be determined.

Table 1 describes the rendered image quality comparison result of the three volume rendering algorithms, using a phantom beating heart, as well as animal and human hearts as test data. We note that HWRC has the highest score in both OIQI and TSRD image quality estimation tests, being around 20% and 80% higher than the SOFTRC in the OIQI and TSRD metrics, respectively, while around 100% and 120% higher than 3DTM in these two categories. The SOFTRC rendered images are ranked higher than those rendered with 3DTM by 67% and 22% in the OIQI and TSRC, respectively. We also note that the ranking difference in the beating heart phantom is the smallest, probably because this dataset is simpler when compared with the animal and human dataset, as its tissue structures are easier to delineate and visualize. The ranking results of the canine and human datasets are similar, reflecting the similar structure and complication of these two datasets. However, it is more challenging to render the small vessels of the human data, especially for the 3DTM algorithm. The "striping" artifacts are obvious in both of these two datasets in the 3DTM rendered images, while in the human dataset they are more pronounced, since parts of the small vessels are not visualized in the artifact regions. As for the SOFTRC and HWRC rendered images, all these three datasets have similar visual results, while the HWRC rendered images are a little sharper and clearer with fewer artifacts, and the small structures are easily distinguished. In the following, we further illustrate the image quality comparisons based on the experts' opinions, which further explain the evaluation results of Table 1. In addition, we describe the effects of color and opacity adjustment techniques when applied in the dynamic multiresolution volume rendering, as illustrated in Fig. 12.

4.1.1. *Beating heart phantom*

Fig. 9 describes the rendered images of the beating heart phantom using the three rendering algorithms implemented in this paper. For the OIQI, the images rendered with 3DTM and SOFTRC are very good with only minor artifacts in the vessels and the surrounding tissues, the color and opacity changes are smooth, and adequate for reliable evaluation. The images rendered with the HWRC are excellent with smoothly changing color and opacity, with no noticeable artifacts. As for the TSRD, all the three images clearly show the tissue structures and their relationships, supplying enough visual information for clinical applications. However, the vessels of the 3DTM and SOFTRC rendered images have "striping" artifacts, while the image rendered with HWRC is free of these artifacts.

4.1.2. *Animal data*

The image quality comparison of the canine dataset is illustrated in Fig. 10, and while the overview image quality in the first row is visually good for all the three images, the HWRC rendered image is sharper than the other two. For the second row, there are obvious "striping" artifacts in the 3DTM rendered image, while the SOFTRC and HWRC are better, and free of visible artifacts. As for the TSRD comparison, all these three images exhibit the small tissue structures for clinical applications, however, the vessels of the 3DTM and SOFTRC rendered images somewhat blurred when compared to the image rendered with HWRC. The HWRC rendered image delivered the best illustration of the vessel relationships.

4.1.3. *Human data*

Fig. 11 describes the image results of the human heart. In the first row, we can clearly see the artifacts in the 3DTM rendered image, and its cardiac vessels are unclear, while the images created with SOFTRC and HWRC are very similar, both are visually satisfactory without visual artifacts. However, in the second row, the color and opacity of the HWRC rendered image are more uniform than those of the SOFTRC rendering image, and some small vessels are not visualized in the 3DTM rendered image.

4.1.4. *Dynamic multiresolution*

Fig. 12 describes the image quality improvement for the dynamic multiresolution rendered image when using the color and opacity adjustment technique. The volume of interest (VOI, enclosed by box) is rendered with high resolution. The low resolution regions are rendered with a sampling step 3 times that which was used in the high resolution section, therefore, increasing the rendering speed by a factor of about three. Fig. 12(a) is the multiresolution image without color and opacity adjustment, and it can be seen that the color difference between high resolution and low regions is obvious, the low regions are unclear and blurred, and the visual result is nonuniform and distracting. Fig. 12(b) shows the multiresolution image improved with color and opacity adjustment, the adjustment factor δ in the Eq. (8) being set 1.9. Note the image quality improvement in (b) compared with (a). Even though image (b) is still inferior to the original uniform image (c), its visual result is much improved when compared with the unadjusted image (a).

4.2. *Performance analysis*

The three datasets, e.g., beating heart phantom, canine heart and clinical human heart, were used in the performance analysis experiment. To demonstrate the impact of graphics hardware progress

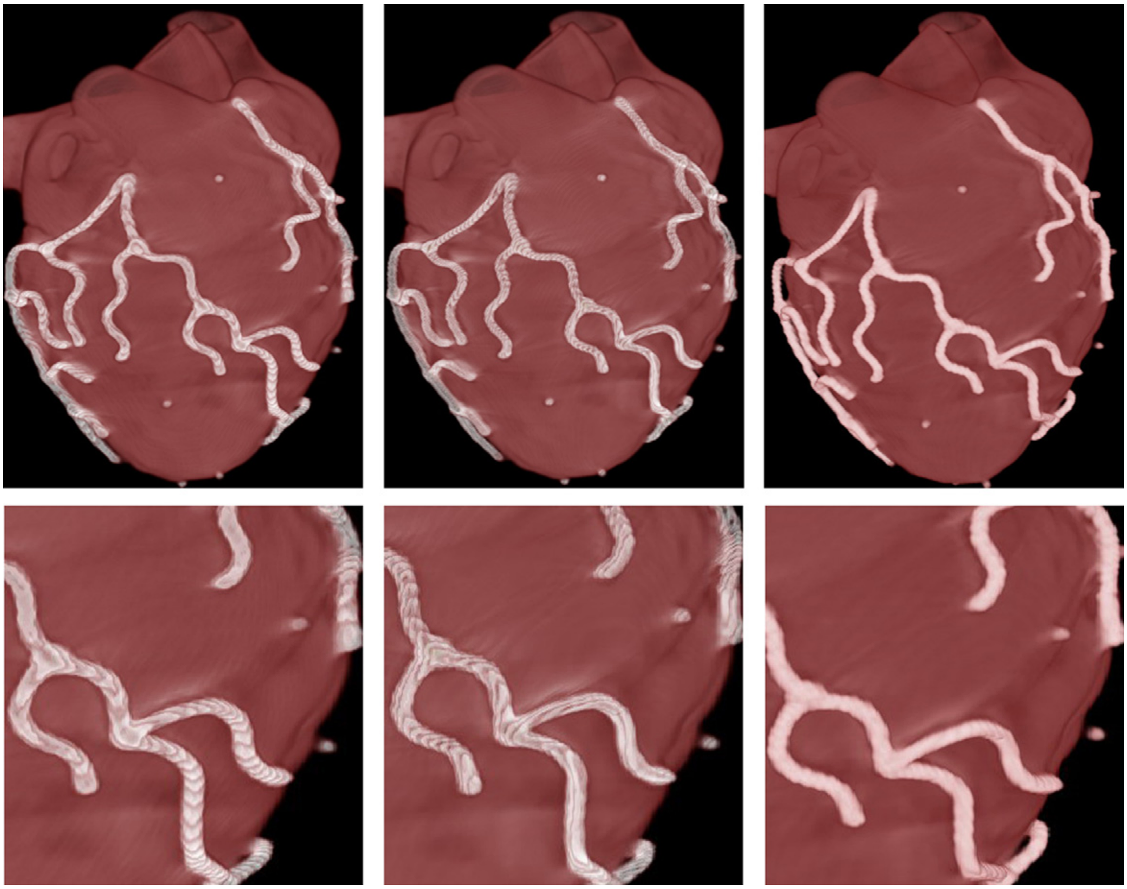


Fig. 9. The quality comparisons of the volume rendered beating heart phantom images using the three rendering algorithms. Left: 3D texture mapping (3DTM); middle: software-based raycasting (SOFTRC); right: HW-accelerated raycasting (HWRC). The second row is the zoomed part of the corresponding images in the first row.

on the system performance improvement, we used three generations of graphics card on the Intel Pentium D 820, 2.8 GHz, Dual Core processor and 2GB DDR system: Nvidia 6800 GS with 256 MB memory (GPU1), Nvidia 7900 GTO with 512 MB memory (GPU2), and Nvidia 8800 GTS with 640 MB memory (GPU3). To simulate the clinical application environment, the rendering windows were set to 800 × 800 with the camera positioned at 2.0 OpenGL units from the rendering object.

Table 2 describes the performance comparison of the static volume visualization algorithms, using the three datasets and evaluated on an Nvidia 8800 GTS GPU. The graphics hardware accelerated volume rendering algorithms (3DTM and HWRC) have much higher performance than the software-based rendering algorithm (SOFTRC), by up to 179 and 127 times respectively. We also note that when comparing the raycasting-based rendering algorithms (SOFTRC and HWRC) with the slice-based rendering algorithm (3DTM) on the three datasets, there are different performance trends. Both the raycasting-based rendering algorithms have higher performance on animal and human hearts when compared with their performance on the phantom heart, while the slice-based algorithm does not. This may be explained by the early ray termination acceleration technique employed in the raycasting algorithms, and the rendering speed is enhanced by 146% and 97% respectively when the dataset has normally distributed intensity. However, this acceleration scheme cannot be applied to the slice-based algorithm. When comparing the two hardware-accelerated algorithms (3DTM and HWRC), we find that when the dataset has normally distributed intensity and the early-ray termination technique can be applied, the HWRC has higher performance than the 3DTM by up to 100%, while for the dataset with low intensity, the 3DTM has around 38.7% higher performance than the HWRC. Both of these two hardware-accelerated algorithms can perform in real-time on these three experimental datasets.

Table 2
Performance comparison (average frames per second, i.e., fps) of three static volume visualization algorithms, using the Nvidia 8800 GTS GPU.

Dataset	SOFTRC	3DTM	HWRC
Phantom heart	0.24	43.0	30.6
Animal heart	0.53	50.0	61.2
Human heart	0.59	30.0	61.0
Mean ±STD	0.45	41.0	50.9

For the graphics hardware accelerated raycasting method (HWRC), the performance of the static (3D) and dynamic (4D) volume rendering on the three experimental datasets is illustrated in Fig. 13. The dynamic dataset volume rendering degrades the rendering performance from 19% to 54.1% in this experiment. We note that for the slow GPUs, such as GPU1, the performance degradation is small, only 19% for the beating phantom, while it is around 38% for the fastest GPU on the same dataset. The trend is true for the other datasets in this experiment.

There are several elements that affect the GPU performance, such as the number of GPU processors, core clock, texture fill rate, graphics memory speed, as well as memory bandwidth. As shown in Table 3, when the slow GPU (GPU1) is compared with the fast GPU (GPU3), the increase of the first three items are 5.6 times, 3.2 times, and 4.7 times respectively, and all these items will affect the static rendering. However, the memory bandwidth increases only by a factor of 2, which will work with the first

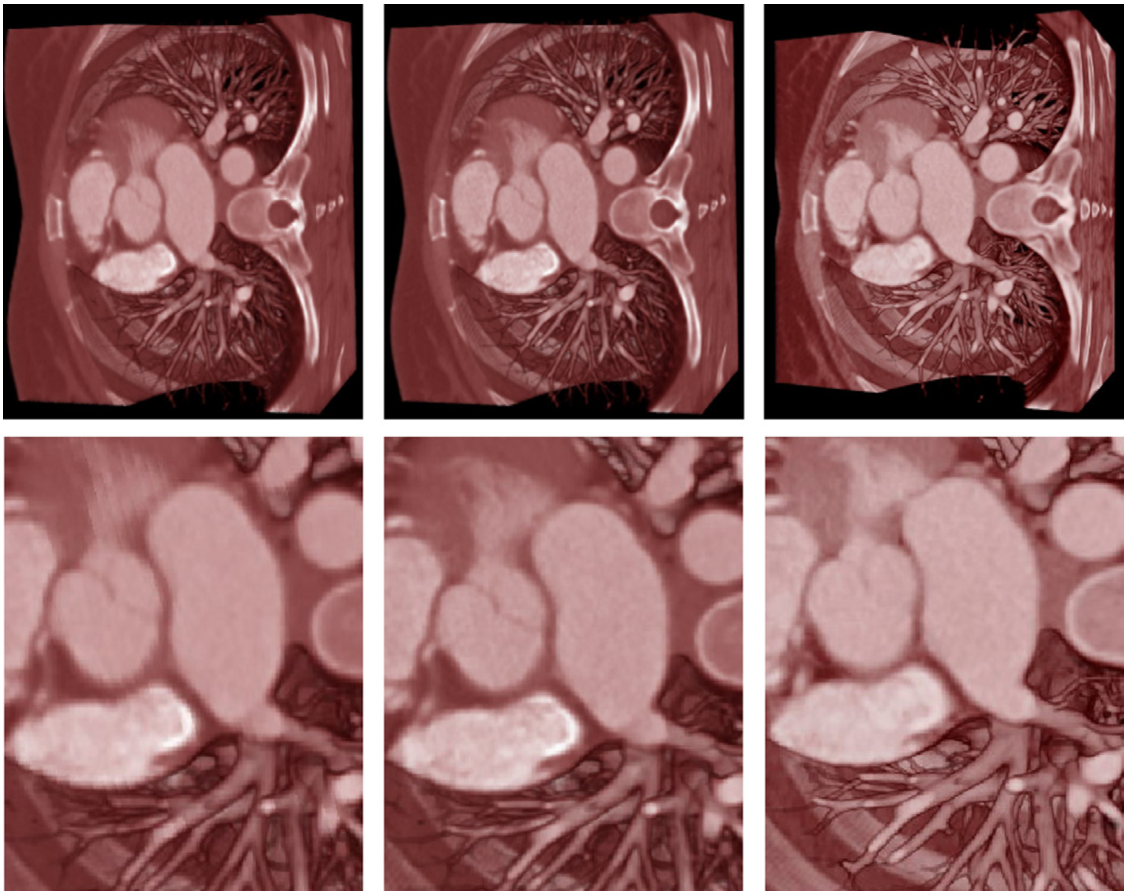


Fig. 10. The quality comparisons of the volume rendered canine dataset using the three rendering algorithms. Left: 3DTM; middle: SOFTRC; right: HWRC. The second row is the zoomed part of the corresponding images in the first row.

three items to affect the dynamic rendering. Compared with static rendering, for the dynamic rendering, when updating from GPU1 to GPU3, the bandwidth increase limitation lowers the level of performance enhancement (Fig. 13(a)–(c)), and increases the performance degradation (Fig. 13(d)).

Therefore, we attribute the performance degradation trend (Fig. 13) to the graphics card memory bus width limitation, and anticipate that when the bus width is increased in subsequent generation graphics hardware, the dynamic rendering speed will be greatly accelerated. In addition, current Nvidia 8800 series GPU can support real-time rendering of the ECG gated dynamic volumetric data.

Next, we tested the performance enhancement of the three rendering algorithms when using the newer generation graphics hardware. As described in Fig. 14, the newer generation GPUs (from GPU1 to GPU2, and to GPU3) can greatly increase the rendering speed of the two hardware-accelerated algorithms (3DTM and HWRC) on all these three datasets by up to 4.2 and 4.7 times respectively, while the performance for the software-based rendering algorithm (SOFTRC) does not change, or even degrades on the newer generation GPUs. For the beating heart phantom, where the intensity is low in most of the volume, the performance increase of the 3DTM is higher than the HWRC, while for the other two clinical datasets (canine and human), the performance enhancement of HWRC is higher than the 3DTM algorithm. We conclude therefore that when applied to rendering the general clinical datasets, the HWRC algorithm can benefit more from the new generation graphics hardware than the 3DTM approach.

In addition, we also tested rendering performance of the dynamic multiresolution approach. The performance improvement depends on the sampling step and the percentage of the high resolution VOI with respect to the entire volume. Typically, the speed improvement is up to 160% when the VOI is around 30% of the total. This method is used to improve the system performance while maintaining high image quality in the VOI. As for the stereoscopic rendering, there is, as expected, a performance degradation of around 50%. Nevertheless, the stereoscopic display increases the reality and precision during the quantitative interaction procedures, and when it is combined with dynamic multiresolution, the visual effect is greatly enhanced while maintaining high rendering efficiency.

Finally, we tested the retrospective ECG gating for the 4D cardiac real-time rendering, during which the HWRC rendering method and the GPU3 (Nvidia 8800 GTS) were employed. According to the procedure described in 3.7, we first set a threshold for the programmable ECG digitizer and initialized the system, as shown in Fig. 5. Then, based on the rendering model, we built a sampling step lookup table using the human cardiac dataset, as shown in Fig. 7. Next, the standard ECG leads were placed on a volunteer,

Table 3
The technical parameters of three Nvidia GPUs.

Parameters	GPU1	GPU2	GPU3
Number of processors	17	32	96
Core clock (MHz)	425	650	500
Memory speed (MHz)	500	1320	1600
Memory amount	256	512	640
Memory bandwidth (GB/s)	32	42.2	64
Texture fill rate (billion/s)	5.1	15.6	24

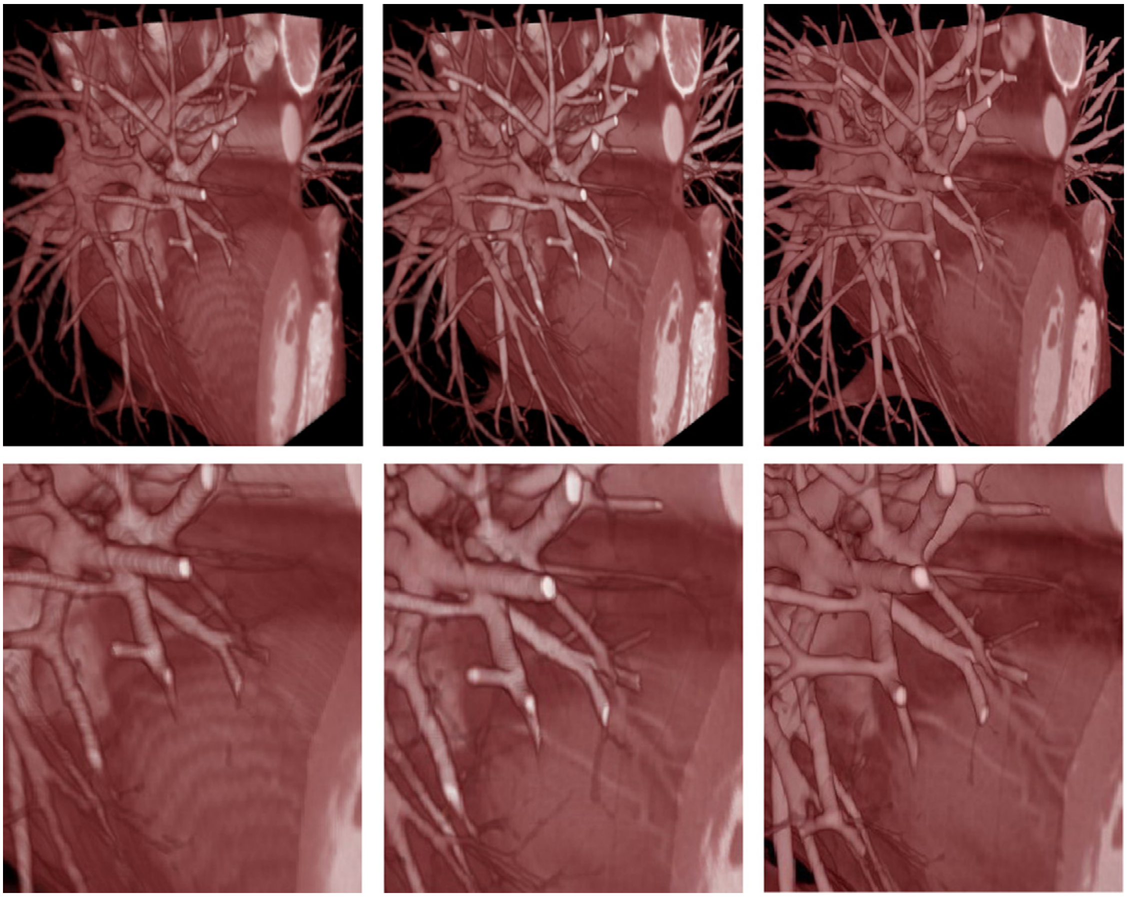


Fig. 11. Image quality comparison of 4D clinical human heart. Left: 3DTM; middle: SOFTRC; right: HWRC. The second row is the zoomed part of the corresponding images in the first row.

acquiring real-time ECG signals, which were then converted to binary digital signals and input to our system to synchronize the dynamic volume rendering speed of the heart. We recorded the time points at the end of every ECG R-R wave and the time points at the end of every volume rendering frame, and found that the average synchronization errors were within one cardiac sampling interval.

4.3. System analysis

Fig. 15 shows the ECG-gated beating heart real-time display and manipulation platform, which supports the following rendering methods: (1) multiplanar reformation (MPR), (2) surface rendering (SR), (3) maximum intensity projection (MIP), and (4) direct volume rendering (DVR). For DVR, the supported rendering

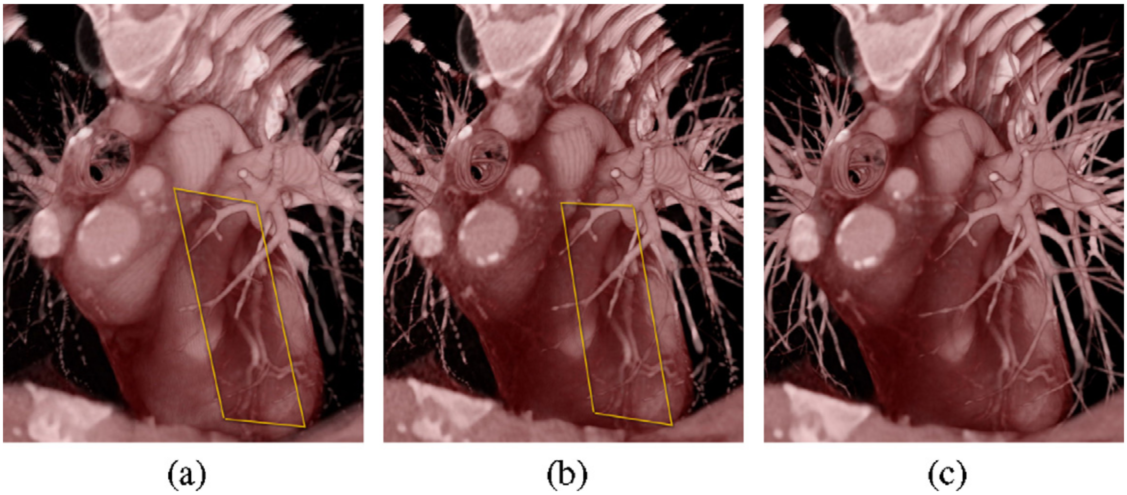


Fig. 12. Volume rendered images with dynamic multiresolution techniques. The high resolution region is outlined by yellow line. (a) Image without color and opacity adjustment; (b) image rendered with adjustment; (c) uniform image without multiresolution. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of the article.)

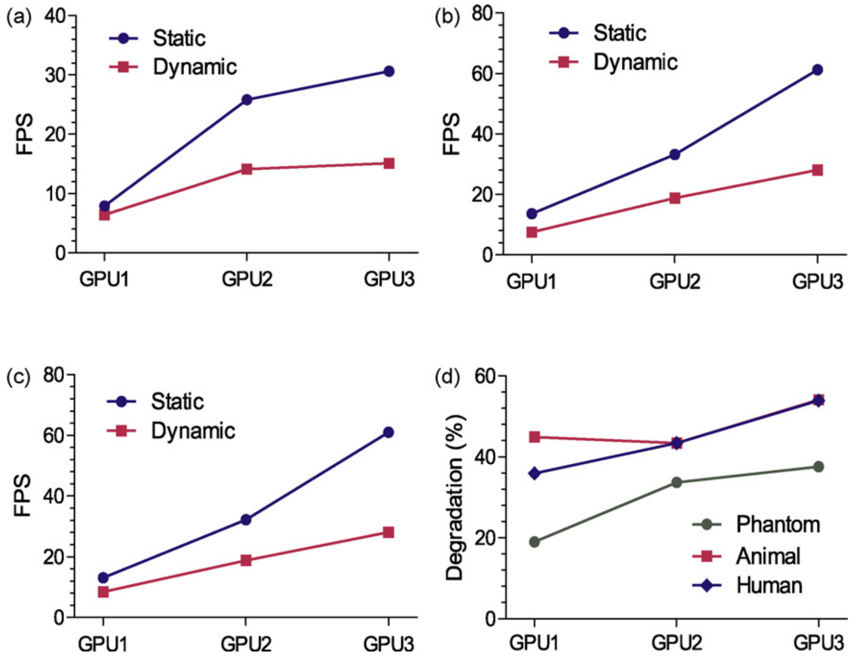


Fig. 13. Performance comparison (frames per second, i.e., FPS) of the static (3D) and dynamic (4D) volume visualization algorithms, using the three GPUs on the three datasets respectively. (a) Beating heart phantom, (b) canine, (c) human dataset, and (d) performance degradation for dynamic rendering.

algorithms include software-based raycasting (SOFTRC), hardware-accelerated texture mapping (3DTM) and hardware-accelerated raycasting (HWRC). This image shows a combination of DVR, SR and MPR, which provides multiple cardiac views to the observer. This system includes color and opacity editors for adjusting the appearances of the rendered volume, the surfaces describing the segmented cardiac structures, and three cross-planes through the rendered cardiac volume. There are also three separate windows for the display of 2D images corresponding to these three cross-planes, with a facility to interactively adjust the window and level of the grey-scale images. The pixel values of the 2D images can be interrogated, the rendered 2D and 3D cardiac images may be annotated, and quantitative morphological analysis can be performed within this environment.

For 4D cardiac volume rendering, the ECG signals are first digitized, and then transferred to the rendering system to synchronize the frequency of the volume rendered heart beat, using the algorithms described above. The cardiac structure changes can be recorded during this dynamic rendering procedure. This comprehensive ECG-gated dynamic cardiac display supplies a platform for cardiac function analysis and dynamic structure evaluation.

5. Discussion

In this paper, we have demonstrated a versatile environment for visualization and manipulation of dynamic cardiac image volumes based on commodity high-level graphics hardware. Specifically, this system incorporates the ability to synchronize the image display with a patient heart using ECG gating, implement a resolution-enhanced VOI facility, as well as provide a stereoscopic display. Through validation experiments, we have demonstrated that our GPU-accelerated algorithms are robust over a range of commonly available clinical MDCT data types and sizes.

We have also highlighted the advantages of the GPU-accelerated ray-casting (HWRC) algorithm over standard approaches employing 3D texture mapping (3DTM). The principal advantage of HWRC lies in the differences in the rendering pipelines between the two approaches. For 3DTM, since the sampling steps are determined by the intersections of the viewing rays with the view-aligned slices, they cannot be maintained constant for different viewing directions, resulting in “striping” artifacts being introduced into the rendered images (Fig. 9–11). In addition, the rendering pipeline is not flexible, and since only fixed back-to-front alpha-blending can

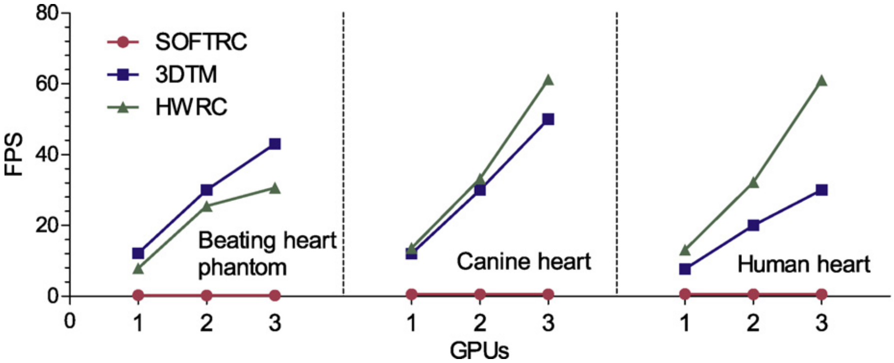
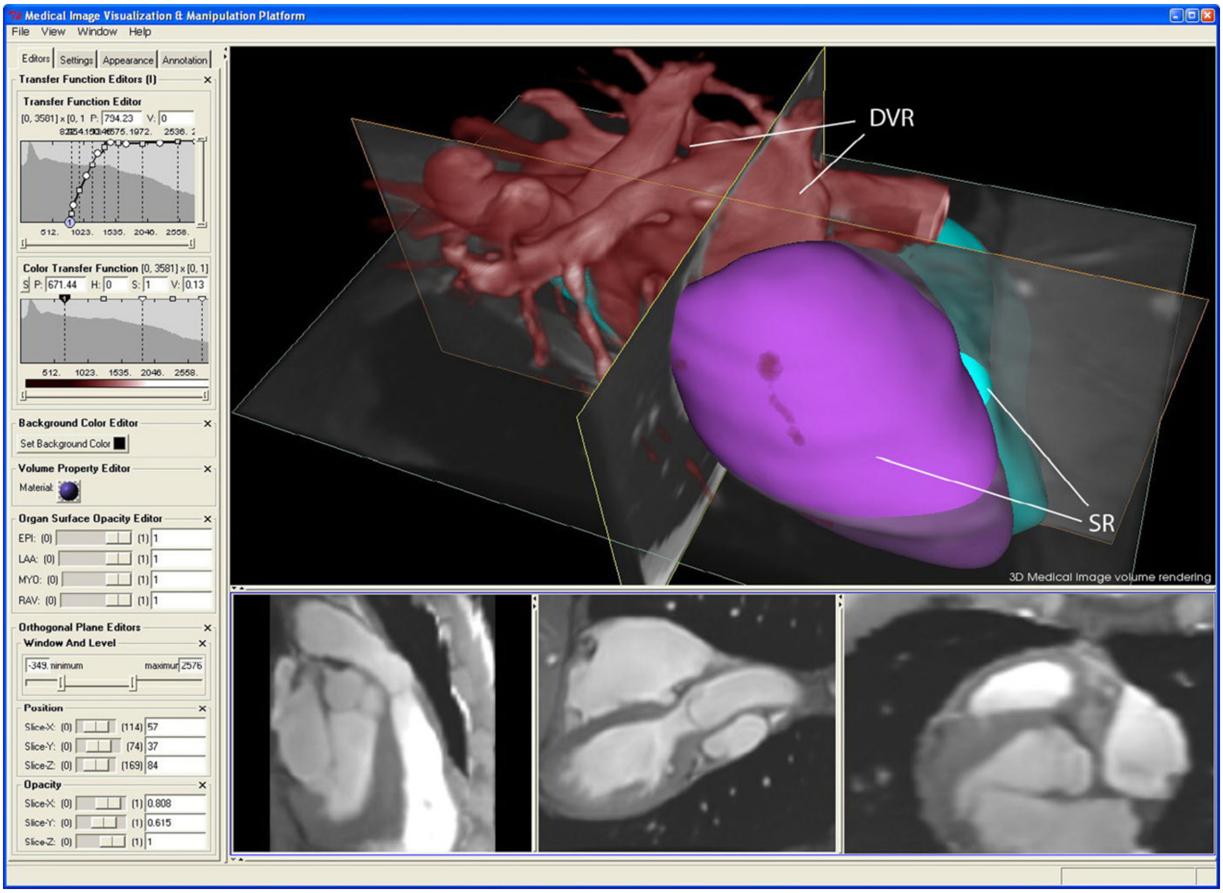


Fig. 14. Performance comparisons (frames per second, i.e., FPS) of the three volume rendering algorithms using three generation Nvidia GPUs: 6800GS (GPU1), 7800GTO (GPU2) and 8800GTS (GPU3). Left: results of rendering beating heart phantom; middle: results of rendering 3D canine heart image; right: results of rendering 3D human heart image. Note: unlike the GPU-accelerated 3DTM and HWRC algorithms, there are no noticeable frame rate changes for the SOFTRC algorithm when using different generation GPUs. The average frame rate of SOFTRC for the three datasets from top to bottom are 0.25, 0.56 and 0.60 fps, respectively.


Medical Image Visualization & Manipulation Platform
File
View
Window
Help
Editors
Settings
Appearance
Annotation
512.
1023.
1535.
2046.
2558.
DVR
SR
3D Medical Image volume rendering

Transfer Function Editors (I)

Transfer Function Editor
[0, 3581] x [0, 1 P: 794.23 V: 0

Color Transfer Function [0, 3581] x [0, 1]
P: 671.44 H: 0 S: 1 V: 0.13

Background Color Editor
Set Background Color

Volume Property Editor
Material:

Organ Surface Opacity Editor
EPI: (0) (1) 1
LAA: (0) (1) 1
MYO: (0) (1) 1
RAV: (0) (1) 1

Orthogonal Plane Editors
Window And Level
-349. ninimum maximur 2576

Position
Slice-X: (0) (114) 57
Slice-Y: (0) (74) 37
Slice-Z: (0) (169) 84

Opacity
Slice-X: (0) (1) 0.808
Slice-Y: (0) (1) 0.615
Slice-Z: (0) (1) 1

Fig. 15. 3D CT cardiac dataset are volume rendered with colored surfaces for FOI enhancement. Transfer functions, clipping plane, orthogonal plane with corresponding 2D slice views, and windows-level editor are designed for volume manipulation.

be used in the compositing process, early-ray termination acceleration techniques cannot be used. On the other hand, the HWRC algorithms cast rays through the volume for each arbitrary viewing direction, and the sampling points can be uniformly selected, resulting in higher quality images than those created with 3DTM. In addition, the HWRC rendering pipeline is more flexible, and can use the front-to-back compositing strategy, enabling the early-ray termination technique to be employed. Therefore, when rendering medical images with normal histograms typical of those of animal and human datasets, the HWRC algorithm has higher performance than 3DTM (Fig. 14). This trend is a function of the algorithm, and is independent of the graphics hardware used. We note that the rendering speed of HWRC is around 20–30 times faster than SOFTRC, and conclude that HWRC is an optimal solution for displaying dynamic volume rendered images, and that the efficiency of this algorithm can be further improved by a factor of two with the most recent generation of graphics hardware (Fig. 14).

Due to rapid advances in CT technologies, 4D cardiac images can be acquired with MDCT scanners with high temporal and spatial resolution [47], maintaining a high signal-to-noise ratio (SNR) and small voxel size, although at the expense of increased patient radiation dose. In addition, even though the stereoscopic display offers a wide field of view and a high degree of spatial perception, it doubles the computational cost of computing the images. The speed of hardware-accelerated volume rendering is limited by the voxel fill-rate, the size of texture memory, the I/O bandwidth, and the texture interpolation speed of the graphics hardware. For additional performance improvements, the scalable-link interface (SLI) technique (i.e., a multi-GPU solution developed by NVIDIA for parallel processing where multiple GPUs share the workload when rendering a 3D scene), can be considered. This technique can also be used with a graphics cluster structure [48] to further improve rendering speed. In addition, the spatial and temporal correlation could be exploited to accelerate the processing speed for 4D rendering [49]. When the data size exceeds the available texture memory, volume subdivision techniques could be employed for data compression [50]. However, as discussed above, this method is not suitable for dynamic multi-resolution approaches. In our evaluation studies, we first loaded one cardiac volume into the graphics memory, and then updated the volume in real-time for each frame. We found the time delay to be negligible when using the PCI Express × 16 graphics card, which has 4000 MB/s maximum data transfer rate.

To address the potential patient dose problem, Jakobs et al. [51] proposed an ECG-driven modulation method for cardiac image acquisition using maximum tube current at one time-point in the cardiac cycle and a reduced dose fraction for the remaining frames. In our lab, we have applied this method for dose reduction in the cardiac MDCT acquisition procedure [52], during which the mid-diastole (MD) phase was imaged with full dose, and the MD image then being independently registered to the lower dose images at other frames, using a non-rigid registration algorithm [3]. The MD frame was then warped through the cardiac cycle to generate a full 4D image but with ∼10% of the normal dose. This is therefore a potential approach for generating high SNR data for display throughout the cardiac cycle.

6. Conclusion

In this paper, we have presented a comprehensive system for real-time 4D image visualization and manipulation, specifically optimized for the display of cardiac datasets. With this system, all cardiac tissue types can be represented in a single image, and the spatial and temporal relationships between the heart volumes, organ surface and the underlying parenchyma can be displayed dynamically. The rendering pipeline was built inside the GPU fragment shader, resulting in image quality superior to that obtained with standard hardware-accelerated rendering algorithms, such as 3D texture mapping. We synchronized the volume rendered beating heart with the patient, using a real-time clinical ECG gating, and incorporated stereoscopic rendering to enable the display to be employed in a virtual-reality environment. Dynamic multi-resolution was also employed to balance rendering speed, manipulation convenience, and VOI selection and enhancement. Color and opacity is automatically adjusted in this mode to ensure visual uniformity in the rendered 4D images. Clinical ECG signals were digitized in real time and input to our system to synchronize the 4D volume rendering with the heart of a subject. To the best of our knowledge, this is the first attempt to use a retrospective ECG signal to synchronize high-quality 4D cardiac MDCT image rendering in a clinical environment, incorporating time-delay computation and compensation, visual reality enhancement, interactive VOI selection, and ECG-gating. We believe that this approach will form an important component for comprehensive cardiac intervention planning and guidance.

Acknowledgments

The authors would like to thank Drs. A. So, J. White, A. Islam, G. Guiraudon, T.-Y. Tee, M. Wierzbicki, X. Huang, and U. Aladl for valuable discussions and assistance in clinical data acquisition, J. Moore and C. Wedlake for technical support and assistance, and Ms. J. Williams for editorial assistance. The authors also thank the reviewers for their constructive comments. This project was supported by the Canadian Institutes for Health Research (Grant MOP 74626), the National Science and Engineering Research Council of Canada (Grant R314GA01), the Ontario Research and Development Challenge Fund, the Canadian Foundation for Innovation and Ontario Innovation Trust, and the Graduate Scholarships from the Ontario Ministry of Education and by the University of Western Ontario.

References

[1] Guiraudon GM, Jones DL, Baindridge D, Peters TM. Mitral valve implantation using off-pump closed beating intracardiac surgery: a feasibility study. Interactive CardioVascular and Thoracic Surgery 2007;6:603–7.

[2] Linte CA, Wierzbicki M, Moore J, Little SH, Guiraudon G, Peters TM. Towards subject-specific models of the dynamic heart for image-guided mitral valve surgery. Proceedings of MICCAI 2007;4792:94–101.

[3] Wierzbicki M, Drangova M, Guiraudon G, Peters TM. Validation of dynamic heart models obtained using non-linear registration for virtual reality training, planning, and guidance of minimally invasive cardiac surgeries. Medical Image Analysis 2004;8(3):387–401.

[4] Linte CA, Wiles AD, Hill N, Moore J, Wedlake C, Guiraudon G, et al. An augmented reality environment for image-guidance of off-pump mitral valve implantation. Proceedings of SPIE 2007;6509:65090N.

[5] Lehmann G, Habets D, Holdsworth DW, Peters TM, Drangova M. Simulation of intra-operative 3D coronary angiography for enhanced minimally-invasive robotic cardiac intervention. Proceedings of MICCAI 2002;2489:269–75.

[6] Guiraudon GM, Jones DL, Skanes AC, Bainbridge D, Guiraudon CM, Jensen SM. En bloc exclusion of the pulmonary vein region in the pig using off pump, beating, intra-cardiac surgery: a pilot study of minimally invasive surgery for atrial fibrillation. The Annals of Thoracic Surgery 2005;80:1417–23.

[7] Levoy M. Display of surfaces from volume data. IEEE Computer Graphics and Applications 1988;8(3):29–37.

[8] Höhne KH, Bomans M, Pommert A, Riemer M, Schiers C, Tiede U, et al. 3D visualization of tomographic volume data using the generalized voxel model. The Visual Computer 1990;6:28–36.

[9] Baek SY, Sheafor DH, Keogan MT, DeLong DM, Nelson RC. Two-dimensional multiplanar and three-dimensional volume-rendered vascular CT in pancreatic carcinoma: interobserver agreement and comparison with standard helical techniques. American Journal of Roentgenology 2001;176(6):1467–73.

[10] Shekhar R, Zagrodsky V. Cine MPR: interactive multiplanar reformatting of four-dimensional cardiac data using hardware-accelerated texture mapping. IEEE Transactions on Information Technology in Biomedicine 2003;7(4):384–93.

[11] Lorensen WE, Harvey EC. Marching cubes: a high resolution 3D surface construction algorithm. SIGGRAPH Computer Graphics 1987;21(4):163–9.

[12] Mroz L, König A, Gröller E. Maximum intensity projection at warp speed. Computers & Graphics 2000;24(3):343–52.

[13] René WMU, Frank JL, Ben JS, Johan PC, Suresh S. Use of maximum intensity projections (MIP) for target volume generation in 4DCT scans for lung cancer. International Journal of Radiation Oncology Biology Physics 2005;63(1):253–60.

[14] Saito K, Saito M, Komatu S, Ohtomo K. Real-time four-dimensional imaging of the heart with multidetector row CT. RadioGraphics 2003;23:e8.

[15] Lawler LP, Ney D, Pannu HK, Fishman EK. Four-dimensional imaging of the heart based on near-isotropic MDCT data sets. AJR American Journal of Roentgenology 2005;184(3):774–6.

[16] Pharr M, Fernando R. GPU gems 2: programming techniques for high-performance graphics and general-purpose cmputation (GPU gems). Addison-Wesley Professional; 2005.

[17] Owens JD, Luebke D, Govindaraju N, Harris M, Krüger J, Lefohn AE, et al. A survey of general-purpose computation on graphics hardware. Computer Graphics Forum 2007;26(1):80–113.

[18] Zhang Q, Eagleson R, Peters TM. Real-rime visualization of 4D cardiac MR images using graphics processing units. Proceedings of IEEE International Symposium on Biomedical Imaging (ISBI) 2006:343–6.

[19] Robb RA, Hanson DP. Biomedical image visualization research using the visible human datasets. Clinical Anatomy 2006;19(3):240–53.

[20] Udupa JK, Odhner D. Shell rendering. IEEE Computer Graphics and Applications 1993;13(6):58–67.

[21] Zwicker M, Pfister H, Baar JV, Gross M. EWA splatting. IEEE Transactions on Visualization and Computer Graphics 2002;8(3):223–38.

[22] Lacroute P. Analysis of a parallel volume rendering system based on the shear-warp factorization. IEEE Transactions on Visualization and Computer Graphics 1996;2(3):218–31.

[23] Cullip TJ, Neumann U. Accelerating volume reconstruction with 3D texture hardware. Technical report, University of North Carolina at Chapel Hill; 1994.

[24] Kniss J. Interactive texture-based volume rendering for large data sets. IEEE Computer Graphics and Applications 2001;21(4):52–61.

[25] Kruger J, Westermann R. Acceleration techniques for GPU-based volume rendering. Proceedings of the 14th IEEE Visualization 2003 2003:287–92.

[26] Audigier R, Lotufo R, Falc ao A. 3D visualization to assist iterative object definition from medical images. Computerized Medical Imaging and Graphics 2006;4(30):217–30.

[27] Lei T, Udupa JK, Saha PK, Odhner D. Artery-vein separation via MRA—an image processing approach. IEEE Transactions on Medical Imaging 2001;20(8):689–703.

[28] Holmes DR, Davisb BJ, Brucec CJ, Robba III RA. 3D visualization, analysis, and treatment of the prostate using trans-urethral ultrasound. Computerized Medical Imaging and Graphics 2003;27(5):339–49.

[29] Levin D, Aladl U, Germano G, Slomka P. Techniques for efficient, real-time, 3D visualization of multi-modality cardiac data using consumer graphics hardware. Computerized Medical Imaging and Graphics 2005;29(6):463–75.

[30] Lehmann H, Ecabert O, Geller D, Kiefer G, Weese J. Visualizing the beating heart: interactive direct volume rendering of high-resolution CT time series using standard PC hardware. Proceedings of SPIE 2006;6141:614109.

[31] Zhang Q, Eagleson R, Peters TM. Graphics hardware based volumetric medical dataset visualization and classification. Proceedings of SPIE 2006;6141:61412T.

[32] Wang AS, Narayan G, Kao D, Liang D. An evaluation of using real-time volumetric display of 3D ultrasound data for intracardiac catheter manipulation tasks. Volume Graphics 2005:41–5.

[33] Hastenteufel M, Yang S, Christoph C, Vetter C, Meinzer HP, Wolf I. Image-based guidance for minimally invasive surgical atrial fibrillation ablation. International Journal of Medical Robotics 2006;2(1):60–9.

[34] Sielhorst T, Bichlmeier C, Heining SM, Navab N. Depth perception—a major issue in medical AR: evaluation study by twenty surgeons. Proceedings of MICCAI 2006;4190:364–72.

[35] Maupu D, Van Horn MH, Weeks S, Bullitt E. 3D stereo interactive medical visualization. IEEE Computer Graphics and Applications 2005;25(5):67–71.

[36] Peters TM, Henri CJ, Munger P, Takahashi AM, Evans AC, Davey B, et al. Integration of stereoscopic DSA and 3D MRI for image-guided neurosurgery. Computers in Medical Imaging Graph 1994;18:289–99.

[37] Charland P, Peters TM. Optimal display conditions for quantitative analysis of stereoscopic cerebral angiograms. IEEE Transactions on Medical Imaging 2003;15(5):648–56.

[38] Dey D, Gobbi DG, Surry KJM, Slomka PJ, Peters TM. Automatic fusion of freehand endoscopic brain images to three-dimensional surfaces: creating stereoscopic panoramas. IEEE Transactions on Medical Imaging 2002;21:23–30.

[39] Navab N, Traub N, Sielhorst J, Feuerstein T, Bichlmeier MC. Action- and workflow-driven augmented reality for computer-aided medical procedures. IEEE Computer Graphics and Applications 2007;27(5):10–4.

[40] Khamene A, Vogt S, Azar F, Sielhorst T, Sauer F, Niemann HS. Local 3D reconstruction and augmented reality visualization of free-hand ultrasound for needle biopsy procedures. Proceedings of MICCAI 2003;2879:344–55.

[41] Wang XH, Good WF. Real-time stereographic rendering and display of medical images with programmable GPUs. Computerized Medical Imaging and Graphics 2008;32(2):118–23.

[42] Blinn J. Light reflection functions for simulation of clouds and dusty surfaces. Computer Gruphics 1982;16(3):21–9.

[43] Laur D, Hanrahan P. Hierarchical splatting: a progressive refinement algorithm for volume rendering. Computer Graphics 1991;25(4):285–8.

[44] Boada I, Navazo I, Scopigno R. Multiresolution volume visualization with a texture-based octree. The Visual Computer 2001;3(17):185–97.

[45] Flohr T, Schoepf UJ, Kuettner A, Halliburton S, Bruder H, Suess C. Advances in cardiac imaging with 16-section CT-systems. Academic Radiology 2003;10(4):386–401.

[46] Flohr T, Ohnesorge B. Developments in CT. Imaging 2006;18(2):45–61.

[47] Mahesh M, Cody DD. Physics of cardiac imaging with multiple-row detector CT. RadioGraphics 2007;27:1495–509.

[48] Kniss J, McCormick P, McPherson A, Ahrens J, Painter J, Keahey A, et al. Interactive texture-based volume rendering for large data sets. IEEE Computer Graphics Application 2001;21:52–61.

[49] Woodring J, Shen HW. Chronovolumes: a direct rendering technique for visualization time-varying data. In: Proceedings of International Workshop on Volume Graphics. 2003. p. 27–34.

[50] Wang C, Garcia A, Shen HW. Interactive level-of-detail selection using image-based quality metric for large volume visualization. IEEE Transactions on Visualization and Computer Graphics 2007;13(1):122–34.

[51] Jakobs TF, Becker CR, Ohnesorge B, Flohr TG, Suess C, Schoepf UJ, et al. Multislice helical CT of the heart with retrospective ECG gating: Reduction of radiation exposure by ECG-controlled tube current modulation. European Radiology 2002;12:1081–6.

[52] Wierzbicki M, Guiraudon GM, Jones DL, Peters TM. Dose reduction for cardiac CT using a registration-based approach. Medical Physics 2007;34(6): 1884–95.

Qi Zhang received his MSc in computer science from the University of Waterloo, and PhD in medical imaging with computer engineering discipline from the University of Western Ontario, Canada. He is currently working as a postdoctoral researcher at the Imaging Research Laboratories, Robarts Research Institute, London, Canada. His research focuses on real-time visualization, computer graphics, medical image processing and analysis, high-performance computation, GPU programming, and their applications in image-guided minimally invasive therapy and surgery.

Roy Eagleson is an associate professor in the Faculty of Engineering at the University of Western Ontario, an associate scientist at the Robarts Research Institute, and a scientist and Principal Investigator at CSTAR—The Canadian Surgical Technologies and Advanced Robotics research centre at LHSC, the London Health Sciences Centre, LHRI. His research includes: 3D Biomedical Visualization and Surgical Simulation, Human-Computer Interface Design for Tele-Robotics and Tele-Surgery, Haptic Interfaces and Programmable Graphical Interfaces (GPU programming).

Terry M. Peters is a scientist in the Imaging Research Laboratories at the Robarts Research Institute (RRI), London, ON, Canada, and professor in the Departments of Medical Imaging, Medical Biophysics and Biomedical Engineering at the University of Western Ontario. For the past 25 years, his research has focused on the application of computational hardware and software to medical imaging modalities in surgery and therapy. He has authored over 160 peer reviewed papers and book chapters, and has delivered over 130 invited presentations. Dr. Peters is a fellow of the Institute of Electrical and Electronics Engineers, the American Association of Physicists in Medicine and the Canadian College of Physicists in Medicine. He is an associate editor of IEEE Transactions on Medical Imaging and an executive member of the Board of the MICCAI Society, as well as its Treasurer.