Simulating 3-D Lung Dynamics Using a Programmable Graphics Processing Unit

Anand P. Santhanam, Felix G. Hamza-Lup, and Jannick P. Rolland

***Abstract*—Medical simulations of lung dynamics promise to be effective tools for teaching and training clinical and surgical procedures related to lungs. Their effectiveness may be greatly enhanced when visualized in an augmented reality (AR) environment. However, the computational requirements of AR environments limit the availability of the central processing unit (CPU) for the lung dynamics simulation for different breathing conditions. In this paper, we present a method for computing lung deformations in real time by taking advantage of the programmable graphics processing unit (GPU). This will save the CPU time for other AR-associated tasks such as tracking, communication, and interaction management. An approach for the simulations of the three-dimensional (3-D) lung dynamics using Green's formulation in the case of upright position is taken into consideration. We extend this approach to other orientations as well as the subsequent changes in breathing. Specifically, the proposed extension presents a computational optimization and its implementation in a GPU. Results show that the computational requirements for simulating the deformation of a 3-D lung model are significantly reduced for point-based rendering.**

***Index Terms*—Augmented reality, Green's function, lung physiology, spherical harmonics.**

I. Introduction

MEDICAL simulation of respiratory physiology allows the development of tools for applications ranging from teaching and training to surgical guidance. Of particular importance is the simulation and visualization of physically based lung dynamics related to lung morphology. A clinical evaluation of a human subject's regional lung morphology and behavior during breathing aids in deciding the future clinical intervention required for the patient [1]. In the case of lung disease, the physician would better understand the patient's response to various clinical intervention possibilities. This paper focuses on the simulation of physically based lung deformations presented in an augmented reality (AR) stereoscopic visualization environment. AR environments "augment" the users view of the real world with virtual information [2]. These AR environments have been used for applications from manufacturing and scientific guidance to entertainment. Visualizing three-dimensional (3-D) lung morphology in an AR environment involves graphically superimposing the 3-D lung models on either the patient himself or a human patient simulator (HPS) and visualizing their dynamics in a physiologically based manner. Fig. 1 shows a snapshot of a 3-D lung model superimposed over a HPS [3]. Such visualization may significantly improve the clinical teaching and surgical guidance of respiratory medical simulation. The key aims involved in achieving this visualization include: 1) tracking the real-world position of the patient or HPS; 2) computing the subsequent global positions of the 3-D lung models; 3) registering the 3-D models onto the patient or HPS; and 4) deforming and graphically rendering the 3-D lung models.

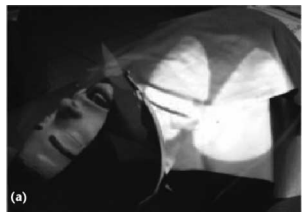


Fig. 1. View of lungs superimposed over a HPS in the Augmented Reality Center [3].

The technical challenges of designing and visualizing a medical simulation for lung morphology arise from its computational complexity. Specifically, in an AR environment, the position and orientation of the patient are updated by the tracking sensor every 16 ms. Such update requires deforming and rendering 3-D lung models at 30–66 times per second [3]. This subsequently limits the usage of high-resolution 3-D lung models for real-time deformation and visualization. A method to overcome this limitation was proposed in [4], in which the 3-D lung dynamics caused by the air-flow into the lungs were modeled using Green's formulation (GF). The lung deformation was precomputed for an upright orientation and simulated in real-time for the upright position. However, the changes in the 3-D lung deformations for changes in the patient position and orientation in an AR environment were not accounted for.

The method presented in this paper extends the 3-D lung dynamics previously modeled in [4] by off-loading the run-time deformation computations from the central processing unit (CPU) to the graphics processing unit (GPU). Specifically, the parameters of the deformation computation are transformed into frequency-space coefficients using spherical harmonic (SH) transformations before being off-loaded to the GPU. During run-time, the deformation is computed in the GPU as a dot product of the frequency-space coefficients. The end-result of such an approach allows the CPU to cater to the computational requirements of the AR environment, and the GPU to cater to

Manuscript received November 21, 2005; revised March 28, 2006. This work was supported in part by the Florida Photonics Center of Excellence (FPCE) and in part by the Link Foundation.

A. P. Santhanam is with the MD Anderson Cancer Center Orlando, Orlando, FL 32806 USA, and also with the Optical Diagnostics and Applications Laboratory, University of Central Florida, Orlando, FL 32816 USA (e-mail: anand@odalab.ucf.edu).

F. G. Hamza-Lup is with the Department of Computer Science, Armstrong Atlantic State University, Savannah, GA 31419 USA (e-mail: felix@cs.armstrong.edu).

J. P. Rolland is with the School of Computer Science, College of Optics and Photonics, and Institute for Simulation and Training, University of Central Florida, Orlando, FL 32816 USA (e-mail: jannick@odalab.ucf.edu).

Digital Object Identifier 10.1109/TITB.2006.889679

the computational requirements of deforming and rendering the 3-D model in this AR environment. Additionally, a method to approximate the airflow inside the lungs at any arbitrary orientation is discussed.

The paper is organized as follows. Section II discusses the related work in 3-D lung morphology modeling and deformation methods implemented using programmable graphics hardware. Section III briefly discusses the framework developed for 3-D lung morphology simulation. Section IV discusses the proposed method for simulating 3-D lung morphology and the representation of the associated computations in the GPU. Experimental results are presented in Section V. Section VI concludes the paper.

II. Related Work

Lung dynamics have been investigated for the verification of thoracic medical imaging equipment and for medical training purposes [5]. Initial methods to model the respiratory physiology were designed as regional pressure–volume equilibrium equations that are based on physiology and clinical measurements. Mechanical and electrical analogues of these balancing equations were simulated for understanding the lung's behavior [6]. These analogues substantially aided in providing controlled mechanical ventilations [7]. In order to obtain a computer-based simulation of the 3-D lung shape, deformation methods were further analyzed.

Deformation methods compute the displacement of surface positions of a 3-D virtual model. These methods have been previously employed in applications ranging from surgery simulations to animations. The deformation methods can be either geometrically based or physically based.

The geometrical methods model the lung deformation during respiration by changing the positions of the 3-D mesh surface points [8]. Of particular importance to medical simulations is the usage of SH for modeling deformations. There has been a significant amount of work performed using SH transformations for shape approximations of face, vertebra, and tumor 3-D models. In the case of face modeling, the shape of a 3-D model before and after the face deformation is represented using a set of SH coefficients. The SH coefficients of the intermediate 3-D face models are then represented as an interpolation [9], [10]. In the case of the vertebra, the SH-based shape approximation before and after the deformation is used to estimate the scaling and torsion movements undergone during the deformation [11]. In the case of tumor 3-D models, an SH-based shape approximation is used for the diagnostic purposes [12]. It is also worth mentioning that the SH-based shape approximations have also been applied for training artificial learning algorithms for 3-D model recognitions [13] and predicting soft tissue deformations [14]. Such methods can accurately model the change in shape of the lungs in real time but lack a physiological basis.

Physically based deformation methods attempt to model the lungs based on the physiology of breathing. The physically based deformation methods vary from simple functional representations such as mass-spring models [15] to representations using polynomial basis functions, such as finite-element methods (FEM) [16] and GF [17]. A FEM-based deformation was proposed by DeCarlo in order to visualize 3-D lung dynamics [18] and was further improved by Kaye in order to model pneumothorax related conditions [1]. In both these methods, a 3-D lung surface model was considered as a single-compartment model [1]. An alternative method for modeling lung deformations using a FEM-based functional representation of bronchioles and parenchyma was investigated by Tawhai and Burrowes [19]. In this method, the air-flow inside the bronchial airways was modeled using computational fluid dynamic methods. The volumetric lung space was divided into multiple compartments. The bronchial airway and the surrounding parenchyma were modeled using FEM [19]. The number of nodes or elements used for the 3-D virtual model limit the computational complexity of the FEM.

GF represents the physics of nonvibrating deformation and is computationally inexpensive as compared to the FEM. This formulation has been extensively used in medical simulation of anatomical organs and their haptic interactions [17]. Such formulation provides scope for real-time deformation of high-density 3-D models obtained from medical imaging methods that will subsequently lead to the development of real-time medical simulation and visualization. The computational complexity of GF is based on the number of elements in the 3-D model used and the behavior of the elastic tissue. Two key mathematical components of this method are: 1) the applied force on each node of the 3-D model and 2) the transfer function matrix, which is a square matrix representing the tissue property. The deformation is obtained as a matrix–vector multiplication of the transfer function matrix and the applied force vector [20].

For real-time requirements, the computational complexity can be reduced by either a decrease in the 3-D model complexity, or an increase in the computational speed of the processor. A decrease in the 3-D model complexity was discussed in [17] using wavelet transformations. In this approach, the GF for a given 3-D model at a high level-of-detail representation was first computed [21]. For any given application, a suitable level of detail was chosen and the 3-D model at that level of detail was computed. The GF was then modified from its original level of detail to the chosen level using wavelet transformations.

In order to increase the computational speed, we discuss the usage of GPUs, which are high-speed data-stream pipeline processors used for implementing fast graphics algorithms. Modern GPUs perform floating-point calculations much faster than most CPUs. For instance, a 3-GHz Pentium 4 can theoretically issue approximately six billion FLOPS. A vertex program in NVIDIA GeForce 5600 Ultra achieves approximately 20 billion FLOPS. Similarly, the memory bandwidth of a CPU is approximately 6 Gb/s while that of a GPU is approximately 25 Gb/s [22].

Programmable GPUs have been previously used for implementing both geometrically and physically based deformation methods. The geometrically based methods of particular importance were Key-framing and Vertex-skinning [23]. Key-framing allowed a cartoon artist to animate a 3-D character (model) by interpolating among a set of intermediate 3-D frame sequences. However, by implementing this method in a GPU the intermediate 3-D frames had to be stored [23]. Vertex-skinning

allowed a cartoon artist to create animations by associating the movement of a node of a 3-D model to the movement of a set of key nodes in the 3-D model. These key nodes as a group were represented as a matrix (referred to as a bone matrix). The animation was reduced to a matrix–vector multiplication in a GPU [24]. Physically based methods were implemented in a GPU using the "render-to-texture" feature. This feature allowed intermediate values of a computation to be stored into a texture memory in GPU, referred to as P-buffer [25]. This approach was used in obtaining physically based wave motions for fluid and cloud simulations [26] and FEM computations [27] in a GPU.

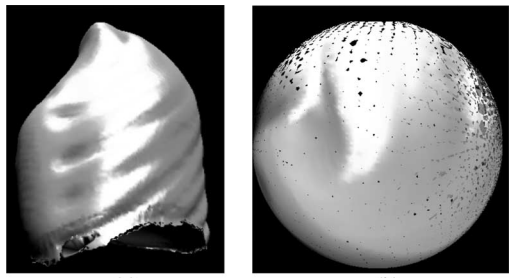


Fig. 2. Graphic representation. (a) 3-D polygonal lung model. (b) Polar coordinate representation.

III. Overview of the Method for 3-D Lung Deformation

In this section, we outline the methodology adopted for the dynamic simulation of 3-D lungs deformation. This method is subdivided into three stages. The first two stages have been addressed in [4], [28], and [29]. In the first stage, we parameterized the change in lung volume for a change in pressure (referred to as transpulmonary pressure) [28], [29]. This change in pressure, which causes the air-flow inside the lungs, accounts for both the local muscle resistance caused by anatomical components, such as lung tissue, diaphragm, and rib cages, as well as the motor drive of breathing controlled by a network of neurons in the medulla. This relation between the lung volume and the transpulmonary pressure is referred to as a pressure–volume (PV) relation. Both normal and abnormal PV relations may be simulated.

In the second stage, we estimated the change in the global lung shape for an increase in lung volume [4]. This was obtained using a physically based deformation method. Within the context of computer animation, a GF-based deformation was chosen since it has been observed that lung deformations do not undergo vibrations [30]. The total number of nodes on each of the 3-D high-resolution lung models is approximately 400 000. Such a large number of nodes facilitates effective modeling of both normal and pathophysical lung deformations. Also, a Young's modulus was first associated to every node of the 3-D lung model based on the lung's regional alveolar expansion. A unit force was then applied on each node and the transfer function matrix was computed using an iterative approach. In each step of the iteration, the force applied on a node was shared with its neighboring nodes based on a local average of Young's modulus as described in [4]. The iteration stopped when this sharing of applied force reached an equilibrium. At this point of equilibrium, the force shared by a node with its neighbors formed a row of the transfer function matrix. An upright orientation was considered in order to deform the 3-D lung model. The applied force caused by the air-flow inside the lungs was given by the vertical pressure gradient of lungs [31]–[33]. The computed force was then normalized so that the sum of the applied force magnitude on all the nodes was equal to a unit increase in volume. A unit increase in volume was set as the ratio between the tidal volume of human lungs (i.e., 500 mL) and the product of the deformation steps per second (i.e., 66.66 steps/s) multiplied by the ventilation rate of inhalation or exhalation (normally 3 s/breathing). Thus, for each increase in lung volume the subsequent change in lung shape was computed [4].

The third stage of the proposed methodology forms the focus of this paper. We consider a per-vertex approach for deforming and rendering 3-D lung models. We present a method to optimally compute the matrix–vector multiplication in a GPU during run-time. Specifically, the matrix–vector multiplication is represented in steps, which can be partially precomputed offline. The columns of the transfer function matrix are precomputed and represented using SH coefficients. These coefficients are obtained from orthonormal decomposition of the transfer function matrix using SH transformations [34]. The SH transformations approximate the transfer matrix with a negative-exponential convergence instead of a uniform convergence, which can be seen in wavelet transformations [35]. This property of SH transformations allows us to represent the transfer matrix using a minimal number of SH coefficients. While the 3-D models are rendered using a point-based rendering approach, a comparison of the frame rate per second (FPS) in using point-based rendering and polygon-based rendering is discussed in Section V. Additionally, the number of SH coefficients used for representing the transfer function row is much less as compared to the number of coefficients used for shape approximation (discussed in Section II). This advantage of the proposed method coupled with the per-vertex nature of an SH coefficient allow us to use GPU for improving the computation speed.

In the proposed approach, we make use of a unique property of normal lung deformations. In our approach, the transfer function remains a constant until the tissue properties undergo irreversible damage [36]. Since lungs do not undergo irreversible damage under normal breathing conditions, the transfer matrix can be considered a constant. The transfer function matrix is precomputed using the method proposed in [4]. The SH coefficients are transferred into the local memory of the GPU along with the 3-D model before starting the simulation. The SH coefficients of the force applied on the 3-D model are also precomputed. During the simulation, 3-D deformation is computed in GPU as a dot product of precomputed SH coefficients of the applied force and the transfer function matrix. The method is computationally inexpensive when compared to the dot product of the transfer function matrix and the applied force for a high-resolution 3-D model. The usage of this representation in deforming the 3-D lung model is further detailed in Section IV.

IV. Mathematical Model

In this section, we discuss the steps in the third stage of the methodology adopted for simulating 3-D lung dynamics. The nodes of the 3-D lung model [as shown in Fig. 2(a)] are represented using polar coordinates [as shown in Fig. 2(b)]. Such polar coordinates were obtained by projecting the nodes of the 3-D lung model [Fig. 2(a)] onto a spherical hull of large radius. It was observed that such an approach provided unique spherical coordinates for all the nodes of the 3-D lung model. Any shape degenerations in the 3-D lung model representing patho-physical conditions can be accounted for using conformal mapping methods [37] and is discussed in the future work. Let θ_I, ϑ_I represent the polar coordinate of node I of the 3-D lung model. At any instant during deformation, a force is applied on each node. From the force we can determine the function $F(\theta_J, \vartheta_J)$ that best approximates the force [38]. Similarly, we can also determine the function $T(\theta_I, \vartheta_I, \theta_J, \vartheta_J)$ that best approximates the transfer function matrix of the 3-D lung model, which represents the elastic interaction between node I and J associated with the deformation. The fundamental form of the transfer matrix is given in [20] and [34]. Let $D(\theta_I, \vartheta_I)$ be the displacement of the node I. For simplicity in notation, we denote $D(\theta_I, \vartheta_I)$ by $D(I)$, $F(\theta_J, \vartheta_J)$ by $F(J)$, and $T(\theta_I, \vartheta_I, \theta_J, \vartheta_J)$ by $T(I, J) \cdot D(I)$ using GF can be written as

$$D(I) = \sum_{J=0}^{N} F(J) T(I, J) \tag{1}$$

where N is the total number of nodes in the 3-D lung model. To further reduce the complexity of the computation, we approximate this function using a set of SH basis functions. The displacement and the transfer matrix have been previously shown to be elements of a Hilbert space, which can thus be represented again using SH polynomials [39]. We use this theoretical fact to modify the computations involved in the dot product on the right hand side of (1). A finite set of SH coefficients are used for representing an array of numbers. Let $Y_{lm}(\theta_J, \vartheta_J)$ represent the SH polynomials and $Y^*_{lm}(\theta_J, \vartheta_J)$ be their conjugates polynomial at node J [40]. For simplicity in notation, we denote $Y_{lm}(\theta_J, \vartheta_J)$ by $Y_{lm}(J)$ and $Y^*_{lm}(\theta_J, \vartheta_J)\sin(\theta_j)$ by $Y^*_{lm}(J)$. The applied force can be represented as

$$F(J) = \sum_{l=0}^{n} \sum_{m=-l}^{l} F_{lm}\, Y_{lm}(J) \tag{2}$$

where n specifies the total number of SH bands and is the square root of the total number of SH coefficients. The values of F_{lm} are the SH coefficients of the force where l and m are indexes for the SH coefficients and the SH polynomials. The SH coefficients can be computed from the applied force by the following relation:

$$F_{lm} = \sum_{J=0}^{N} F(J) Y^*_{lm}(J). \tag{3}$$

The I^{th} row of the transfer function can now be given as

$$T(J, I) = \sum_{l=0}^{n} \sum_{m=-l}^{l} T^I_{im} Y_{lm}(J) \tag{4}$$

where T^I_{lm} represents the SH coefficients of the Ith row of the transfer matrix. We now write (1) by substituting the expansions of the applied force and the transfer function row given in (2) and (4), respectively. Thus, (1) can be written as

$$D(I) = \sum_{J=0}^{N} \left[\sum_{l=0}^{n} \sum_{m=-l}^{l} F_{lm} \cdot Y_{lm}(J) \right] \cdot \left[\sum_{l'=0}^{n} \sum_{m'=-l'}^{l'} T^I_{l'm'} Y_{l'm'}(J) \right]. \tag{5}$$

In the orthonormal property of SH functions, the summation (for values of J from 0 to N) of the product of Y_{lm} and $Y_{l'm'}$ is 1 when l equals l' and m equals m' and 0 otherwise [39]. Equation given in (6) may now be expressed as

$$D(I) = \sum_{l=0}^{n} \sum_{m=-l}^{l} F_{lm}\, T^I_{im}\, A \tag{6}$$

where A is a constant (of value 1) representing the summation of the square of Y_{lm} (whose value is 1 because of its orthonormal property) for all values of J [39]. We now introduce the following modification to (6):

$$D(I) = \alpha \sum_{l=0}^{n} \sum_{m=-l}^{l} (F_{lm}\, T^I_{im}\, A) \tag{7}$$

where α is a coefficient, which allows accounting for the finite value of n. Further discussion on the choice of n and the corresponding value of α are discussed in Section V. The SH coefficients of both the applied force and the transfer function matrix can be precomputed and thus, we need to compute only (7) during run-time in order to obtain deformation of the 3-D model.

Having discussed the method to optimize the dot product, we now propose to compute SH coefficients of the applied force for different patient's orientation in run-time by interpolating among a set of specified orientations. The origin of the coordinate system is set to be the centroid of the lung model. The orientation of the lung model is now represented in terms of the rotation angles along the X, Y, and Z axes. Let p^i, q^i, and r^i be the arrays of precomputed SH coefficients for the applied force at rotation angle i of $\pi/2, \pi, 3\pi/2$, and 2 π along the X, Y, and Z axes, respectively. Let a, b, and c be arbitrary values for current rotation angles for the 3-D lung model. For such orientation, the SH coefficients for the applied force are computed by smoothly interpolating among p^i, q^i, and r^i. Let a_0, a_1, b_0, b_1, c_0, and c_1 be the angles of i that form the closest lower and upper limits *for* a, b, and c, respectively. The SH coefficients of the applied force for the orientation a, b, and c are now given by

$$f^{abc}_{lm} = f^a_{lm} + f^b_{lm} + f^c_{lm} \tag{8}$$

where

$$f^a_{lm} = \frac{[p^{a_0}_{lm} \cos(a - a_0) + p^{a_1}_{lm} \cos(a - a_1)]}{3}$$

$$f^b_{lm} = \frac{\left[q^{b_0}_{lm} \cos(b - b_0) + q^{b_1}_{lm} \cos(b - b_1)\right]}{3}$$

and

$$f_{lm}^{c} = \frac{[r_{lm}^{c_0} \cos(c - c_0) + r_{lm}^{c_1} \cos(c - c_1)]}{3}. \quad (9)$$

The advantage of using SH coefficients is that it reduces the dot product of two arrays of variable length of nonzero values to a dot product of two arrays of fixed length of nonzero values. Since the SH coefficients of the transfer function matrix and the applied force are precomputed, a significant amount of computation time is saved. Additionally, the SH coefficients can be directly modified for any change in the 3-D model's level of detail, which allows the method to comply with various application requirements.

We now summarize the third stage in three steps.

Step 1) In the first step, the position, color, and normal of a node in the 3-D lung model and the transfer function matrix are transferred into the GPU's vertex array. Since the GPU is a per-vertex processor, every row of the transfer matrix is associated with a node of the 3-D model and is transferred into the vertex array as multitexture coordinates.

Step 2) The second step involves computing the SH coefficients of the applied force array using (8) and (9) for the current orientation of lungs. The AR steps in tracking the orientation of the patient or HPS is beyond the scope of this paper and, thus, the orientation angles are obtained from AR components.

Step 3) The third step of the third phase is the implementation of (7) in the GPU during every frame. We use a GeForce4 FX 5600 for rendering and deforming the 3-D lung model. Specifically, we use a CG 1.1-based programmable vertex-shader, for the shading of the 3-D model (cg_simple [23]) and implementing the dot product of the per-vertex transfer matrix's row and the applied force in the vertex shader.

We now investigate the mathematical computation requirements of (1). For high-resolution 3-D lung models, it was observed that each row of the transfer function matrix has an average of 192 nonzero values. Thus, the computation of deformation of a node, using (1), will undergo an average of 192 multiplications and 191 additions. We now continue by analyzing the computational requirements of (7). We first compute the SH coefficients of the transfer function, which can be performed offline since the transfer function for lungs is considered a constant. The SH coefficients of the applied force at predetermined orientations can be computed from (3) offline. For any orientation of the 3-D lung model, the applied force is interpolated using (8) and (9). The deformation computation of (7) would be four vector (array of four floating-point numbers) dot products and three additions per node in the Nvidia's GPU, which reduces the number of computations by approximately one-third.

V. Lung Deformation Results

In this section, we first quantify the increase in speed obtained by using GPU-based deformations. Specifically, we show that when GPU-based deformations are used, an increase in the 3-D model's complexity, and the AR system's computational complexity of the tracking and the registration components do not create a significant lag in simulation. We then discuss the graphic outputs obtained from the deformation approach.

TABLE I
System Benchmark Using Learning_VAR

Learning_VAR approach	FPS
GPU-based computations	109.8
CPU-based computations	55.6

The frame-rates per second (FPS) for the implementation system were computed using Nvidia's benchmark program, learning_VAR [23] and are reported in Table I. It can be seen that the system provides a higher frame rate with the GPU-based simulation (vertex array memory used in association with the vertex program [23]) compared to the CPU-based simulation.

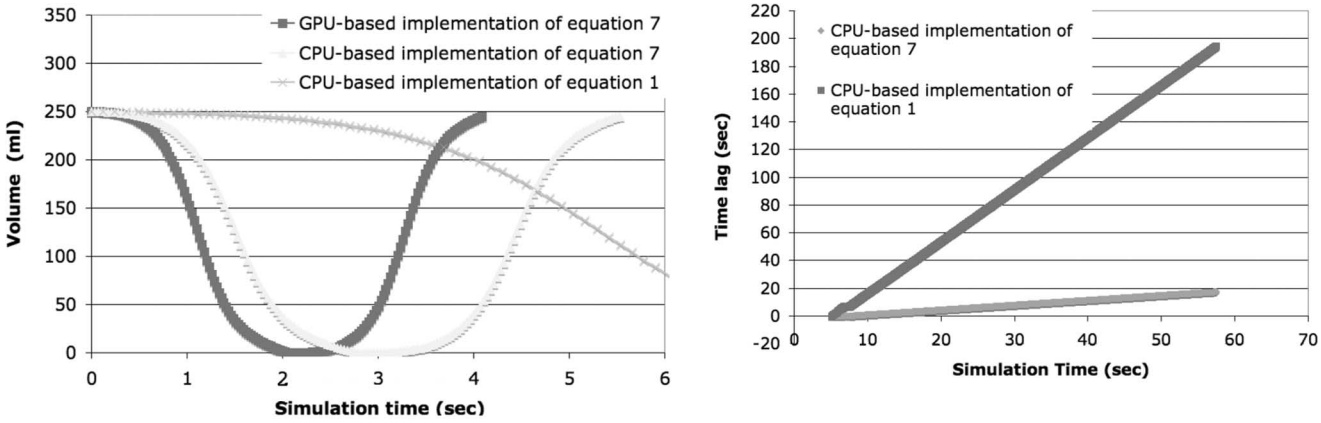
The FPS observed during the 3-D lung deformations are also traced using the learning_VAR program and are as detailed in Table II. In this paper, we have used a Nvidia's shader, cg_simple [23] to simulate lighting conditions. For simulation purposes, both generic point-based and polygon-based rendering approaches were considered. While the polygon-based rendering approach has been extensively discussed in the graphics field, the point-based rendering approach is currently being addressed for high-resolution rendering and tailored for VR/AR applications. Also, for a point-based rendering approach, the vertex size was set to be five for occlusion purposes. The relation between the vertex size and the FPS is later discussed in this section. It can be seen that the FPS observed for point-based rendering is much higher as compared to the polygon-based rendering, which stems from the low computational complexity of point-based rendering for data traversal, occlusion culling, and lighting steps. In the case of a point-based rendering approach, it can be seen that the FPS for a GPU-based implementation of (7) is approximately 1.4 and 10 times the FPS of CPU-based implementation of (7) and CPU-based implementation of (1), respectively. In the case of a polygon-based rendering approach, GPU-based implementation of (7) is approximately 1.2 and 5 times the FPS of CPU-based implementation of (7) and CPU-based implementation of (1), respectively. Variations in the number of light sources (a rendering parameter) caused a reduction of approximately two FPS for each of the methods. Thus, a GPU-based implementation of (7) can be used in conjunction with both point-based and polygon-based rendering approaches.

From the FPS numbers discussed in Table II, it can be seen that for a GPU-based deformation approach, the FPS is almost halved when rendering is changed from mono to stereo. This reduction in FPS numbers is caused by the fact that in the GPU-based deformation approach, the 3-D lung model gets deformed for each eye (for a stereoscopic visualization). For a CPU-based deformation, the 3-D lung model is deformed only once and rendered twice. Thus, the FPS numbers in this case are not halved when rendering is changed from mono to stereo.

The GPU-based deformation approach provides less improvement over the CPU-based deformation approach when polygon-based rendering is used, which stems from the high data-structure traversal involved in a polygon rendering. Specifically, in a polygon-based rendering, each node is accessed

TABLE II
FRAME RATES OBTAINED FOR 3-D LUNG DEFORMATIONS

Approach	FPS- Point-based rendering		FPS - Polygon-based rendering	
	Mono	Stereo	Mono	Stereo
GPU implementation of equation (7)	148	75	46	23
CPU implementation of equation (1)	10	7	8	6
CPU implementation of equation (7)	76	56	28	19



(a) (b)

Fig. 3. (a) Delay caused by the usage of CPU-based lung deformation using (1) (*green*) and (7) (*yellow*) and the subsequent optimization seen in a GPU-based lung deformation (*red*). (b) Demonstration of the increase in simulation lag with the usage of the CPU-based deformations of (1) (*red*) as opposed to (7) (*blue*). A point-based rendering approach is considered for the above results.

approximately four times from the memory. A GPU-based per-vertex deformation approach would thus compute the deformation at each of the four times, which overshadows the benefit of using the GPU. Thus, a point-based rendering approach would be more suitable for high-resolution lung models in order to satisfy the requirements of the AR environment.

For the rest of the paper, we use a point-based rendering approach. Fig. 3(a) presents a comparison of the lung ventilation (volume change with time) visualized using a CPU-based implementation of (7) (2800 + AMD Athlon) with the ventilation visualized using a GPU-based implementation of (7) (Nvidia GeForce4 FX5600). For simulation purpose, a point-based rendering approach is considered to minimize the delays caused by the rendering process. It can be seen that there exists a difference (referred as lag) between the lung ventilation with a GPU-based deformation [the black square line in Fig. 3(a)] and the lung ventilation with a CPU-based deformation using (7) [the light triangle line in Fig. 3(a)]. The time lag can be explained from the offloading of the deformation computations from the CPU to the GPU. Since we deform the 3-D lung model at every frame, the time lag is directly related to the FPS. The CPU-based implementation of (7) has a time lag of approximately 1.4 s/breathing cycle. The CPU-based implementation of (1) [the gray cross line in Fig. 3(a)] has a time lag of approximately 13 s/breathing cycle. This lag in CPU-based deformation increases linearly with simulation time as shown in Fig. 3(b) for both the CPU-based implementation of (1) (the light gray line) and (7) (the dark gray line). These graphs support the computational speed-up results obtained from using the GPU.

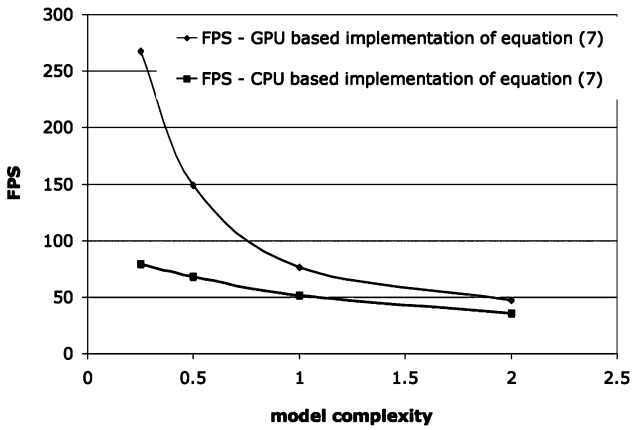


Fig. 4. The increase in the slope of the time delay with an increase in the 3-D model complexity ($1\times$, $2\times$, and $4\times$). A point-based rendering approach is considered for the above result.

The decrease in the FPS with an increase in the 3-D model complexity is shown in Fig. 4. We use a notation of "$c\times$" to refer to the increase in the 3-D model complexity by a constant c. With the increase in the 3-D model complexity ($0.25\times$, $5\times$, $1\times$, and $2\times$) the CPU-based implementation of (7) had a decrease in the FPS since the number of SH coefficients does not increase with an increase in the total number of nodes. There also exists a significant decrease in the FPS in the case of a GPU-based implementation of (7), which is caused by an increase in the number of data elements transferred from the vertex array for each rendering. Thus, for higher model complexity, improved GPU with higher vertex array bandwidth would be required.

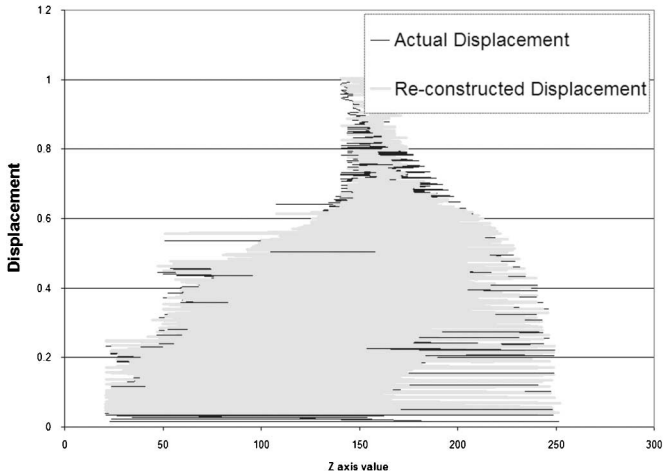


Fig. 5. Comparison of the displacement of 3-D lung nodes computed using (1) (*dark gray*) and (7) (*light gray*). The RMS difference in the displacement was computed to be less than 1 mm.

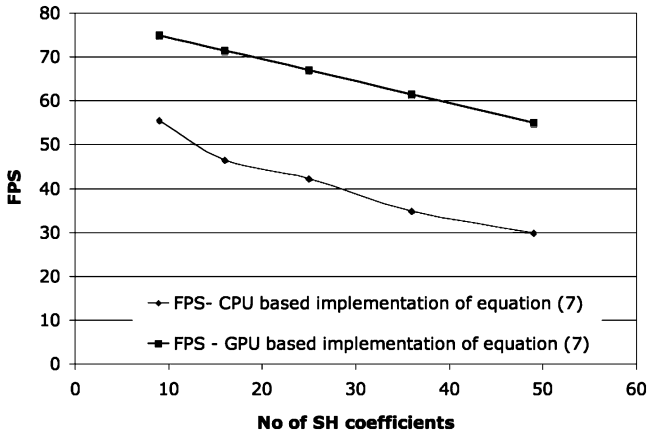


Fig. 6. Comparison of the FPS obtained for lung deformations using GPU-based implementation of (7) and CPU-based implementation of (7) for an increase in the number of SH coefficients used. A point-based rendering approach is considered for the above result.

The accuracy in using SH transformation is shown in Fig. 5. It was observed that the displacement of the nodes could be reconstructed using 16 SH coefficients with an accuracy of less than 1% RMS error, which translates to less than 1-mm RMS error. An initial validation of the deformation obtained using CPU-based computation was discussed in [41]. The difference in this change in shape and volume was negligible when the GPU-based computation was compared to the change in shape and volume obtained using the validated CPU-based computation.

The scalability of the GPU-based computation approach was compared with the CPU-based computation and is shown in Fig. 6. It can be seen that for an increase in the number of SH coefficients, the FPS of the CPU-based computation approach decreased more than the GPU-based computation approach. Such scalability will be of importance for future work in modeling patho-physiological changes in breathing caused by disease states.

The variations in the vertex size and its effect on the GPU-based and the CPU-based computation are shown in Fig. 7. It

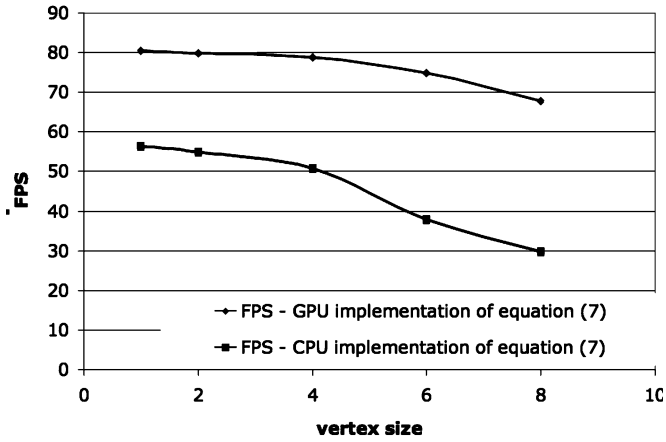
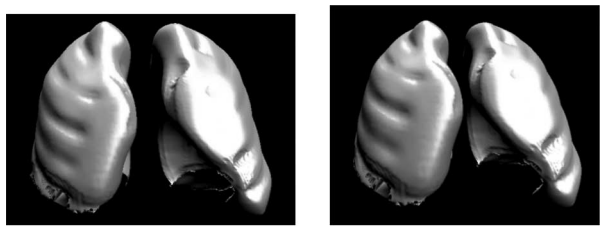


Fig. 7. Comparison of the FPS obtained for lung deformations using GPU-based implementation of (7) and CPU-based implementation of (7) for an increase in the vertex size. A point-based rendering approach is considered for the above result.



(a) (b)

Fig. 8. 3-D Lung shape (*front view*) in an upright position. (a) At the start of inhalation. (b) At the end of inhalation.

can be seen that for an increase in the vertex size, the FPS of the CPU-based computation approach decreased more than that of the GPU-based computation approach. The reduction in the FPS values for both approaches is caused by the increase in the occlusion culling computations.

A snapshot of the visualization of high-resolution 3-D lungs in the upright position at the start of an inhalation phase is shown in Fig. 8(a). The change in shape and volume at the end of the inhalation phase using a GPU-based computation is shown in Fig. 8(b). On a detailed observation one would notice the "bucket-handle" rotation of rib-cage markings in the lung. It refers to the rotation of the rib cage along the spinal-cord axis during inhalation and exhalation. A sequence of 3-D lungs at different transpulmonary pressure in a perspective view is shown in Fig. 9(a)–(d). The bounding box of the 3-D lungs at the start of inhalation is shown in each of the images.

The side-views of the lung dynamics in the upright and supine positions are shown in Fig. 10(a) and (b), respectively. Such visualization snapshots are taken during the inhalation phase with the deformed 3-D lung model (light gray lung model) overlapping the undeformed 3-D lung model (dark gray lung model). The lower the regional overlap, the higher is the visibility of the gray undeformed lung model. Such variations in the overlap visually represent the change in lung shape caused by the lung deformation. The subtle difference in the deformation of the base of the lungs in the upright and supine positions can be

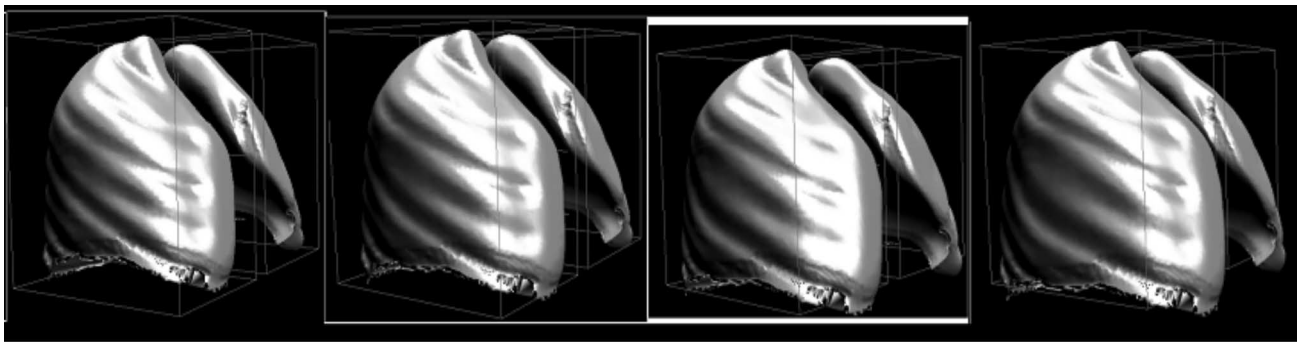


Fig. 9. 3-D lung shapes (*perspective view*) in an upright position are shown at transpulmonary pressure at (a) 5%, (b) 40%, (c) 75%, and (d) 100%.

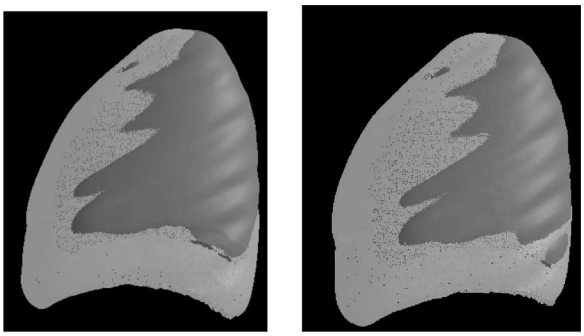


Fig. 10. Visualization of 3-D deformed lung shape (*red*) overlapped with the undeformed lung (*gray*) at the end of inhalation. (a) In an upright position. (b) In a supine position.

Fig. 11. Real-time 3-D lung dynamics. (a) When visualized through the HMD in an AR setup. (b) When superimposed over a HPS and visualized through the HMD.

seen. Specifically, the front side of the base region in the supine position displaces more when compared to the backside of the base region in the supine position, which is not observed in the upright position.

The usage of 16 SH coefficients used in the GPU-based implementation yields α to be 1.03 for lung deformations. α for values more than 16, the value of α tends to 1. These sequences of images represent the shape change that will be viewed in AR.

Finally, Fig. 11(a) represents a snapshot of the lung dynamics being visualized in an AR setup. The 3-D lung model is in an upright orientation and is projected onto the ARC screen. Such 3-D visualization can constitute a training platform for medics where a group of students or experts may visualize the 3-D deforming lung model as if the layers of the body had been peeled off to reveal the lungs and its important anatomical structures. Furthermore, a same view of the lung could also be given to each participant to train on a specific procedure such as needle insertion in the case of a pneumothorax. Fig. 11(b) represents the lung dynamics when the 3-D models are now superimposed over the HPS and visualized through a head mounted display (HMD). To take this camera view, the 3-D models were hand-positioned over the HMD and their dynamic registration was left to future work.

VI. Discussion

In this paper, we have discussed a method to compute 3-D lung deformations in a GPU for an AR environment that accounts for changes in the deformations associated with changes in the orientation of the patient or HPS. The physics and physiology-based deformation operator for lung deformations allows us to model lung deformations with variations in physics-based parameters. The GPU-based deformation approach for lung dynamics discussed in this paper closely models the change in 3-D lung shape in real time. Such an approach coupled with the PV relation (discussed in Section III-A) allows us to visualize normal 3-D lung deformations. Such visualization may play a significant role in assessing clinical interventions for a patient. The usage of high-resolution models in the visualization supports meticulous modeling of tissue degenerations.

In our approach, we have represented each lung as a single compartment. The choice of such a single compartmental model enables real-time visualization of high-resolution 3-D lung deformations. The accuracy in the usage of a single compartmental model has been validated by some of the peers. The validation however needs to be extended across a wide range of human subjects varying in age and race. Additionally, the applicability of a single compartmental model to simulate diseased lung dynamics needs to be further validated. In the case of a multi-compartmental representation of each lung, the collision of each compartment also needs to be taken into account, which adds to the computational complexity.

The creation of such a deformation operator for lung dynamics was discussed in [41]. The usage of the regional alveolar expandability as one of the parameters allowed the deformation operator to account for the physiology of normal human subjects. The method can be extended by analyzing the regional alveolar expandability for human subjects across a wide range

of age, race, and disease states. With respect to real-time 3-D lung dynamics, such variations can lead to an increase in the number of SH coefficients used for representing the deformation operator. The proposed method can be expanded in order to address this computational aspect.

The validation of the lung deformation was discussed in [41]. Through this validation, we illustrated the method to obtain physically and physiologically based lung deformations. Specifically, an estimate of the deformation kernel was made, which allowed us to compute the accurate displacement of each surface node using (1). In Section V, we showed the accuracy of the GPU-based deformation using (7) to represent the 3-D deformed lung shape using (1). Additional validations will be done using HRCT data obtained from a higher number of normal and diseased human subjects under different breathing conditions and orientations.

We have discussed the advantages of the proposed method in terms of the time lag and the FPS obtained using the proposed method in a GPU. Although the CPU–based deformation approach can be improved by using a heuristic frame-rate control method, the effectiveness of such a heuristic method needs to be carefully quantified under rapid breathing changes in the subject's physical conditions and orientation. We are currently investigating these methods and will discuss its result in future work.

Finally, variations in rendering steps (occlusion culling, lighting, texture mapping, etc.) may also reduce the frame rates of the visualization system. In this paper, we have used a Nvidia's shader, cg_simple [23] to simulate lighting conditions. Further investigation will be required in order to verify the FPS obtained using a combination of GPU-based deformation approach and state-of-the-art rendering algorithms for each rendering step that pertains to both point-based and polygon-based rendering approach.

ACKNOWLEDGMENT

The authors would like to thank Dr. W. P. Segars of The Johns Hopkins Medical Institute for providing the 3-D lung models, C. Fidopiastis for stimulating discussions in lung physiology and AR techniques, respectively, and C. Balgeman of the University of Central Florida for technical editing.

REFERENCES

[1] J. M. Kaye, F. P. J. Primiano, and D. N. Metaxas, "A three-dimensional virtual environment for modeling mechanical cardiopulmonary interactions," *Med. Image Anal.*, vol. 2, no. 2, pp. 169–195, 1998.

[2] S.-L. Tang, C.-K. Kwoh, M.-Y. Teo, N. W. Sing, and K.-V. Ling, "Augmented reality systems for medical applications," *IEEE Eng. Med. Biol. Mag.*, vol. 17, no. 3, pp. 49–58, May/Jun. 1998.

[3] J. P. Rolland, L. Davis, and F. Hamza-Lup, "Development of a training tool for endotracheal intubation: Distributed augmented reality," in *Proc. Conf. Med. Meets Virtual Reality (MMVR)*, vol. 11, 2003, pp. 288–294.

[4] A. Santhanam, C. Fidopiastis, F. Hamza-Lup, J. P. Rolland, and C. Imielinska, "Physically-based deformation of high-resolution 3D lung models for augmented reality based medical visualization," in *Proc. Med. Image Comput. Comput-Aided Intervention Conf.*, Rennes, St. Malo, Sep. 2004, pp. 21–31.

[5] W. P. Segars, "Development of a new dynamic NURBS-based cardiac-torso (NCAT) phantom," Ph.D. dissertation, Univ. North Carolina, Chapel Hill, 2002.

[6] M. Rideout, *Mathematical Modeling of Respiratory Systems, Biophysics and Engineering Series*. Englewood Cliffs, NJ: Prentice-Hall, 1990.

[7] S. Mesic, R. Babuska, H. C. Hoogsteden, and A. F. M. Verbraak, "Computer-controlled mechanical simulation of the artificially ventillated human respiratory system," *IEEE Trans. Biomed. Eng.*, vol. 50, no. 6, pp. 731–743, Jun. 2003.

[8] W. P. Segars, D. S. Lalush, and B. M. W. Tsui, "Modeling respiratory mechanics in the MCAT and the spline-based MCAT systems," *IEEE Trans. Nucl. Sci.*, vol. 48, no. 1, pp. 89–97, Feb. 2001.

[9] S. Erturk and T. J. Dennis, "Object shape deformation with spherical harmonic interpolation," *Electron. Lett.*, vol. 34, no. 17, pp. 1657–1658, 1997.

[10] S. Erturk and T. J. Dennis, "Efficient representation of 3-D human head models," presented at the 10th Brit. Mach. Vis. Conf., Nottingham, U.K., 1999.

[11] G. Lefaix, X. Riot, P. Haigron, R. Collorec, and A. Ramee, "3D modeling and deformation analysis of the vertebra with spherical harmonics," in *Proc. 19th Int. Conf. IEEE/EMB*, vol. 1, Chicago, IL, Oct. 30–Nov. 2, 1997, pp. 422–425.

[12] D. Goldberg-Zimring, I. Talos, J. G. Bhagwat, S. J. Haker, P. M. Black, and K. H. Zou, "Statistical validation of brain tumor shape approximation via spherical harmonics for image-guided neurosurgery," *Acad. Radiol.*, vol. 12, pp. 459–466, 2005.

[13] A. Hobolth, "The spherical deformation model," *J. Biostatist.*, vol. 4, no. 4, pp. 583–595, 2003.

[14] C. Hongjun, S. Fan, N. W. Sing, and S. Daming, "An initial statistical approach to deformation modeling for soft tissue," in *Proc. Conf. Intell. Robots Syst.*, Las Vegas, 2005, pp. 1278–1283.

[15] D. Terzopolous, J. Platt, A. Barr, and K. Fleisch, "Elastically deformable models," *ACM Siggraph*, vol. 21, no. 4, pp. 205–214, 1987.

[16] M. Bro-nielsen, "Finite element modeling in surgery simulation," *Proc. IEEE*, vol. 86, no. 3, pp. 490–503, Mar. 1998.

[17] D. L. James and G. K. Pai, "Multiresolution green's function methods for interactive simulation of elastostatic models," *ACM Trans. Graph.*, vol. 22, no. 1, pp. 47–82, 2003.

[18] D. DeCarlo, J. M. Kaye, D. Metaxas, J. R. Clarke, B. Webber, and N. I. Badler, "Integrating anatomy and physiology for behavior modeling," presented at the Conf. Med. Meets Virtual Reality 3, San Diego, CA, 1995.

[19] M. H. Tawhai and K. S. Burrowes, "Developing integrative computational models," *Anatom. Rec.*, vol. 275B, pp. 207–218, 2003.

[20] I. Stakgold, "Green's functions and boundary value problems," *Math. Phys.*, 1979.

[21] D. L. James and G. K. Pai, "ARTDefo:Accurate real-time deformable objects," in *Proc. Annu. Conf. Comput. Graph. Int. Tech.*, vol. 33, 1999, pp. 65–72.

[22] C. Cebanoyan, "Graphics pipeline performance," in *GPU Gems*, R. Fernando, Ed. Boston, MA: Pearson, 2004.

[23] R. Fernando and M. Kilgard, *The CG Tutorial*. Boston, MA: Pearson, 2003.

[24] R. Fernando, *GPU Gems*. Boston, MA: Pearson, 2004.

[25] E. d'Eon, "Deformers," in *GPU Gems—Programming Techniques, Tips and Tricks for Real-Time Graphics*, R. Fernando, Ed. Boston, MA: Pearson, 2004.

[26] J. Harris, "Fast fluid dynamics simulation on the GPU," in *GPU Gems—Programming Techniques, Tips and Tricks for Real-Time Graphics*, R. Fernando, Ed. Boston, MA: Pearson, 2004, pp. 637–665.

[27] R. Rumpf and R. Strzodka, "Using graphics cards for quantized FEM computations," in *Proc. IASTED Vis., Imag. Image Process. Conf.*, Marbella, Spain, 2001, pp. 193–202.

[28] A. Santhanam, C. Fidopiastis, J. P. Rolland, and P. Davenport, "A biomathematical formulation for modeling the pressure-volume relationship of lungs," *Simul. Healthcare*, to be published.

[29] A. Santhanam, C. Fidopiastis, and J. P. Rolland, "An adaptive driver and real-time deformation algorithm for visualization of high-density lung models," presented at the Conf. Med. Meets Virtual Reality, Newport, CA, 2004.

[30] J. Mead, "Measurement of inertia of the lungs at increased ambient pressure," *J. Appl. Physiol.*, vol. 2, no. 1, pp. 208–212, 1956.

[31] J. B. West, *Respiratory Physiology, The Essentials*. Philadelphia, PA: Lippincott Williams and Wilkins, 1995.

[32] S. Krishnan, K. C. Beck, J. M. Reinhardt, K. A. Carlson, B. A. Simon, R. K. Albert, and E. A. Hoffman, "Regional lung ventilation from volumetric CT scans using image warping functions," in *Proc. IEEE Int. Symp. Biomed. Imag.: Micro Nano*, vol. 1, Iowa City, IA, Apr. 2004, pp. 792–795.

[33] L. Fan, C.-W. Chen, E. A. Hoffman, and J. M. Reinhardt, "Evaluation and application of 3D lung warping and registration model using HRCT images," in *Proc. SPIE Conf. Med. Imag.*, San Diego, CA, 2001, vol. 4321, pp. 234–243.

[34] MacRobert and T. Murray, *Spherical Harmonics An Elementary Treatise on Harmonic Functions With Applications. International Series of Monographs in Pure and Applied Mathematics*. New York: Oxford Pergamon Press, 1967.

[35] G. G. Walter and X. Shen, "Wavelets and other orthogonal systems," in *Studies in Advanced Mathematics*. Boca Raton, FL: Chapman & Hall, 2001.

[36] M. Hauth, J. Gross, W. Straber, and G. F. Buess, "Soft-tissue simulation based on measured data," in *Proc. Med. Image Comput. Comput. Aided Intervention Conf.*, Montreal, QC, Canada, 2003, pp. 262–270.

[37] A. Sheffer, C. Gotsman, and N. Dyn, "Robust spherical parameterization of triangular meshes," *Computing*, vol. 72, pp. 185–193, 2004.

[38] D. Braess, *Nonlinear Approximation Theory. Series on Computational Mathematics*. Berlin, Germany: Springer-Verlag, 1986.

[39] D. Reddy, *Introductory functional analysis with applications to boundary value problems and finite systems. Texts in Applied Mathematics*. New York: Springer-Verlag, 1998.

[40] R. Ramamoorthi and P. Hanrahan, "On the relationship between radiance and irradiance: Determining the illumination from the images of complex lambertian object," *J. Opt. Soc. Amer.*, vol. 18, no. 10, pp. 2448–2459, 2001.

[41] A. Santhanam, C. Fidopiastis, J. P. Rolland, and C. Imielinska, "Modeling and simulation of real-time 3D lung dynamics," *IEEE Trans. Inf. Technol. Biomed.*, 2007, to be published.

Anand P. Santhanam, photograph and biography not available at the time of publication.

Felix G. Hamza-Lup, photograph and biography not available at the time of publication.

Jannick P. Rolland, photograph and biography not available at the time of publication.