EG UK Computer Graphics & Visual Computing (2017)
Tao Ruan Wan and Franck Vidal (Editors)

gVirtualXRay: Virtual X-Ray Imaging Library on GPU

A. Sujar[1,2,3], A. Meuleman[3,4], P.-F. Villard[5,6,7], M. García[1,2], and F. P. Vidal[3]

[1] URJC Universidad Rey Juan Carlos, España,
[2] GMRV Grupo de Modelado y Realidad Virtual
[3] Bangor University, United Kingdom
[4] Institut National des Sciences Appliquées de Rouen, France
[5] Universite de Lorraine, LORIA, UMR 7503, Vandoeuvre-les-Nancy F-54506, France
[6] Inria, Villers-les-Nancy F-54600, France
[7] CNRS, LORIA, UMR 7503, Vandoeuvre-les-Nancy F-54506, France

Abstract

We present an Open-source library called gVirtualXRay to simulate realistic X-ray images in realtime. It implements the attenuation law (also called Beer-Lambert) on GPU. It takes into account the polychromatism of the beam spectra as well as the finite size of X-ray tubes. The library is written in C++ using modern OpenGL. It is fully portable and works on most common desktop/laptop computers. It has been tested on MS Windows, Linux, and Mac OS X. It supports a wide range of windowing solutions, such as FLTK, GLUT, GLFW3, Qt4, and Qt5. The library also offers realistic visual rendering of anatomical structures, including bones, liver, diaphragm and lungs. The accuracy of the X-ray images produced by gVirtualXRay's implementation has been validated using Geant4, a well established state-of-the-art Monte Carlo simulation toolkit developed by CERN. gVirtualXRay can be used in a wide range of applications where fast and accurate X-ray simulations from polygon meshes are needed, e.g. medical simulators for training purposes, simulation of tomography data acquisition with patient motion to include artefacts in reconstructed CT images, and deformable registration. Our application example package includes real-time respiration and X-ray simulation, CT acquisition and reconstruction, and iso-surfacing of implicit functions using Marching Cubes.

Categories and Subject Descriptors (according to ACM CCS): I.3.4 [Computer Graphics]: Graphics Utilities—Application packages I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism—I.3.8 [Computer Graphics]: Applications—J.2 [Computer Applications]: Physical Sciences and Engineering—Physics

1. Introduction

The simulation of the X-ray imaging process is extensively studied in the physics community and different physically-based simulation codes are available [A*03, B*04, BSFVS95, FDLB06]. Monte Carlo (MC) simulation methods are very popular as they produce very realistic images. They use photon cross sections with probabilistic X-ray interaction models for the transport of photons in matter. However they tend to be extremely slow due to the stochastic nature of MC methods. For transmission imaging (inc. radiography, computed tomography (CT), cone beam computed tomography (CBCT), and fluoroscopy), deterministic calculation using the Beer-Lambert law (also known as attenuation law) might be considered as a sufficient description (see Section 2.1 for details). In such case, ray-tracing is often used as a fast alternative to MC methods [FDLB06]. To speed-up computations, graphics processing unit (GPU) are being utilised more and more, but often focusing on radiotherapy and voxelised data [B*12, JZJ14]. GPU capabilities also provide the ability to render realistic organ appearances. We propose here a method for both X-ray simulation and visual rendering through an open-source library: gVirtualXRay. Technical details are available in [VGF*09, VV16]. The main contribution of this new paper is to demonstrate the usefulness of the library through its use in various medical applications where speed and accuracy are both requirements, including i) radiograph generation from dynamic polygon meshes to simulate patient posing and respiration (note that voxel data and implicit modelling (e.g. metaballs and soft objects) are also supported), ii) sinogram generation (simulation of the CT acquisition process); iii) CT reconstruction (with or without artefacts); iv) fluoroscopy images (real-time X-ray images). Procedural texturing has also been added to bones, the liver and the lungs to improve the visual appearance of the corresponding 3-D meshes.

Section 2 describes how the X-ray simulation is implemented in gVirtualXRay. Section 3 lists the library's main functionalities. Section 4 discusses two of our most advanced applications: Projectional Radiography Training Tool and Real-time Respiration

DOI: 10.2312/cgvc.20171279

and X-ray Simulation. The last section provides concluding remarks including further work.

2. X-ray Attenuation Modelling

In [VGF*09], it was demonstrated that X-ray attenuation from polygon meshes can be efficiently computed on the GPU using OpenGL and the OpenGL Shading Language (GLSL). The performance significantly increased without loss of accuracy compared to a central processing unit (CPU) implementation. In this implementation, the simulations were restricted to monochromatic X-ray beams (i.e. the incident photons have the same energy) and finite point sources. The method has been deployed into a medical simulator for training fluoroscopy (real-time X-ray images) guidance of needles [V*09]. It made use of polygon meshes that are dynamically modified depending on the respiration cycle of the virtual patient. Due to the computational requirements of such an application, speed was prioritised over accuracy. It might have been sufficient to generate visually realistic fluoroscopy images and radiographs (see Fig. 1), however it was not fully physically realistic in a medical context: Using an X-ray tube, the incident photons have different energies and the X-ray source has a finite size.

(a) *Chest.* **(b)** *Foot.*

Figure 1: *Examples of radiographs simulated on GPU.*

2.1. X-ray attenuation model

The simulation corresponds to solving the Beer-Lambert law for each point of the simulated image. This law relates the absorption of light (i.e. photons) to the properties of the material through which the light is travelling. The integrated form for a monochromatic incident X-ray beam (i.e. all the incident photons have the same energy) is:

$$N_{out}(E) = N_{in}(E) \times e^{(-\int \mu(E,\rho(x),Z(x))\mathrm{d}x)} \quad (1)$$

with $N_{in}(E)$ the number of incident photons at energy E, $N_{out}(E)$ the number of transmitted photons and μ the linear attenuation coefficient (in cm^{-1}). μ can be seen as a probability of interaction by unit length. It depends on: i) E - the energy of incident photons, ii) ρ - the material density of the object, and iii) Z - the chemical element composition of the object material.

When the material composition of the scanned objects is homogeneous, as in gVirtualXRay, it can be rewritten as a discrete model (i.e. without the integral) as follows:

$$E_{out}(x,y) = E \times N_{in} \times \exp\left(-\sum_{i=0}^{i<objs} \mu(i,E) L_p(i,x,y)\right) \quad (2)$$

with $E_{out}(x,y)$ the total output energy received by the detector at the pixel (x,y), $objs$ the total number of scanned objects and $L_p(i,x,y)$ the path length of the ray in the i^{th} object (from the point source to the pixel (x,y)) and $\mu(i,E)$ the linear attenuation coefficient at energy E for the corresponding object.

In a polychromatic case, with M different energies in the incident beam spectrum, it becomes:

$$E_{out}(x,y) = \sum_{j=0}^{j<M} E_j \times N_{in}(E_j) \times \exp\left(-\sum_{i=0}^{i<objs} \mu(i,E_j) L_p(i,x,y)\right) \quad (3)$$

When the source is defined by a set of points rather than a single infinitely small point, it becomes:

$$E_{out}(x,y) = \sum_{k=0}^{k<P} \sum_{j=0}^{j<M} E_j \times N_{in}(E_j) \times \exp\left(-\sum_{i=0}^{i<objs} \mu(i,E_j) L_p(i,k,x,y)\right) \quad (4)$$

with P the number of points defining the shape of the X-ray source and taking into account changing point sources (k) in $L_p(i,k,x,y)$.

2.2. Implementation

Over recent years we have produced gVirtualXRay, a modern C++ library that is open-source [VV16]. It is available on SourceForge at `http://gvirtualxray.sourceforge.net`. The simulation pipeline has been extended to overcome the limitations mentioned above. The simulation now takes into account:

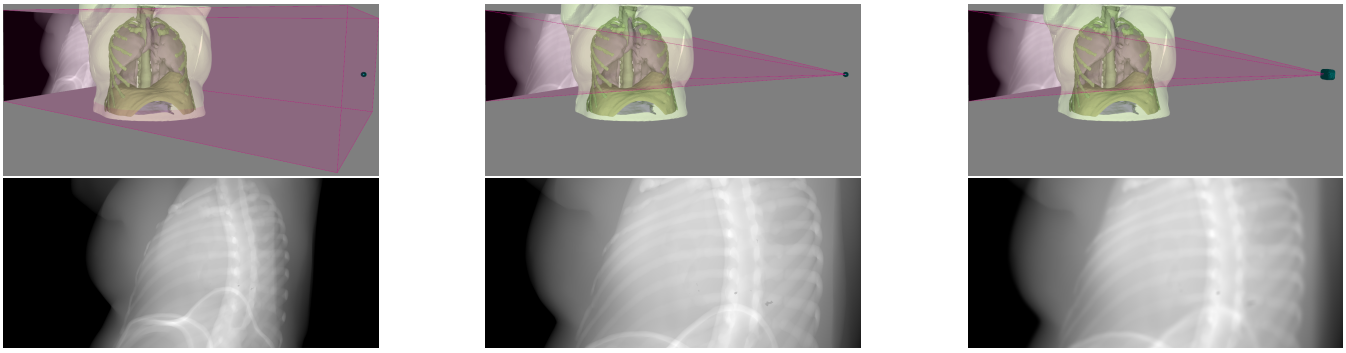
1. Parallel beams,
2. Focal spots of X-ray tubes that cause geometric unsharpness (see blur in Fig. 2), and
3. Polychromatic X-rays (i.e. the incident photons have different energies) (see Fig. 3), which can be used to simulate a CT acquisition with beam hardening.

The overall flowchart of the simulation pipeline is presented in Fig. 4. It shows how an X-ray image is produced. Grey boxes show the additional `for` loops that have been added to take into account polychromatism and focal spot. The implementation heavily relies on framebuffer object (FBO) to perform offline rendering [VGF*09].

2.3. Input parameters

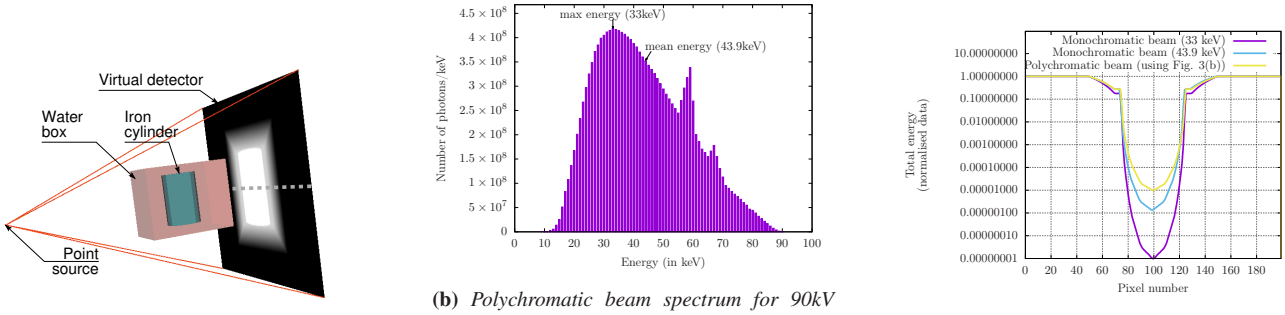
Depending on the parameters of the virtual X-ray machine, one of the three equations (2, 3, or 4) has to be solved for each point of the detector. If Eq. 2 is used, the loops highlighted in grey in Fig. 4 are executed only once per simulated image. In the other cases, the loops may be executed several times, which will generate more physically-realistic images but will increase the execution time. Three components have to be defined to produce an X-ray image:

X-ray source is defined by its shape, position, orientation, and incident photon beam. The shape can be an infinitely small point, a line, a square, a cube, or others (a set of points). It can be used



(a) *Parallel beam.* **(b)** *Infinitely small point source.* **(c)** 1^3*mm source.*

Figure 2: *Simulation with (a) parallel beam and point sources (b) without and (c) with geometrical unsharpness.*



(a) *Set up.* **(b)** *Polychromatic beam spectrum for 90kV X-ray tube peak voltage.* **(c)** *Intensity profile (dash line in Fig. 3a).*

Figure 3: *Effect of the beam spectrum.*

to simulate geometrical unsharpness (see Fig. 2). The beam is defined by its spectrum, i.e. a discrete list of photons. They can have the same energy (monochromatic case) or different energies (polychromatic case) (see Fig. 3).

X-ray detector is defined by the number of pixels along its horizontal and vertical axes, its position and orientation. Note that in most cases the detector is perpendicular to the direction of X-rays, but it does not have to be. Additionally, the pixel size along the horizontal and vertical axes has to be defined. It is often isotropic but it does not have to be.

Scanned objects are defined by their shape as polygon meshes and their material properties in term of density and chemical element composition (in percentage of atoms for each chemical element). To facilitate the development of medical applications, Hounsfield values can be assigned to each polygon mesh. A Hounsfield value can be easily converted into density and chemical element composition [SBS00]. Material properties are used to compute the attenuation coefficients for each energy in the incident beam spectrum based on the corresponding photon cross section provided by the XCOM database from the National Institute of Standards and Technology (NIST) [BHS*10].

2.4. Validation Method

The implementation of the material properties (i.e. density and mass attenuation coefficients) has been validated for different materials by comparing values computed using our code with values from the literature (see Fig. 5).

To produce a reliable platform, we have validated the results of our implementation against similar results obtained with the famous MC simulation toolkit developed by the European Organization for Nuclear Research (CERN): Geant4 [A*03]. We made available the source code and data of our GPU and MC simulations. The results are therefore fully reproducible. Fig. 6 shows an example of validation test. Object materials are defined in Hounsfield unit (HU). A cube, which has edge length of 3 cm, is made of soft tissue (HU $\sim$ 52). Inside, a cylinder is inserted. It is made of bone (HU $\sim$ 1330), its height is 3 cm and its diameter is 2 cm. The detector is made of 301×301 pixels. The size of each pixel is 0.3×0.3 mm^2. Simulation results are extremely similar: The normalised cross-corelation (NCC) between the two images is 99.747%. Other validation tests (including using a point source, a cubic source, an uncentred source, polychromatism) have been conducted and can be found on the project's homepage and in [VV16].

Further validation studies of the method are also available in [VGF*09]. They demonstrate its efficiency and scalability with respect to increasing input polygon mesh resolution and output image pixel resolution.

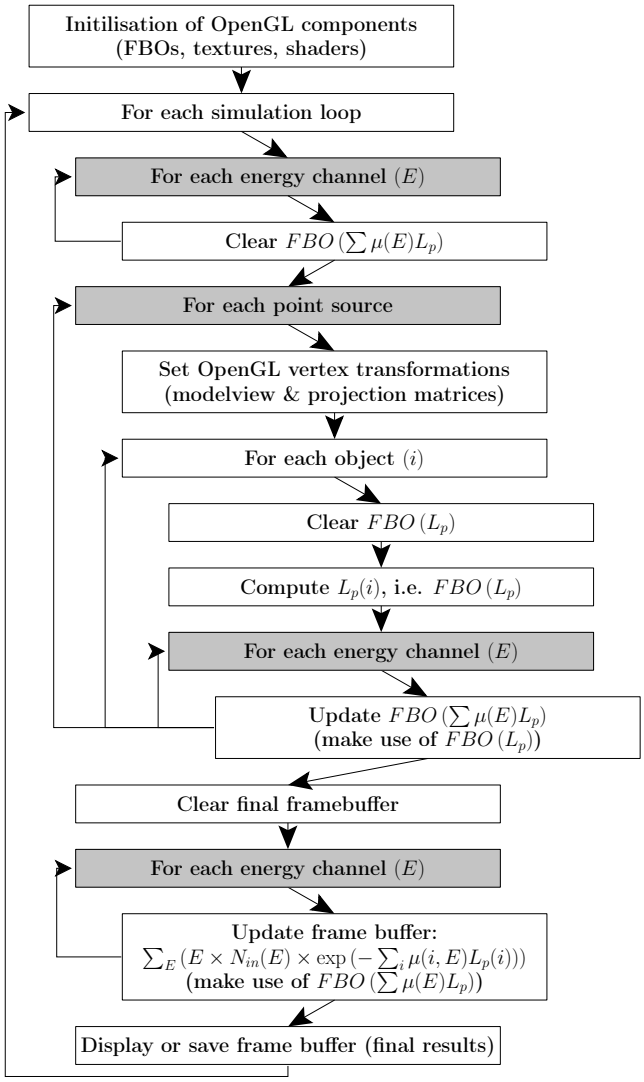
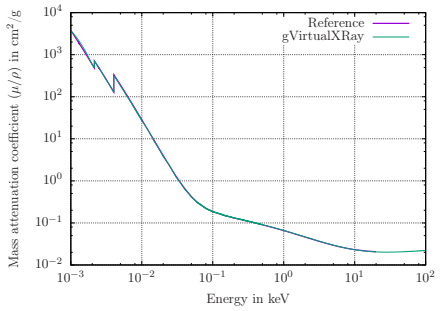
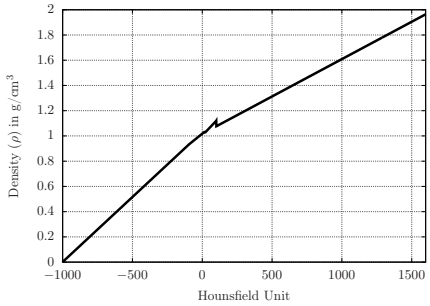


Figure 4: *Simulation flowchart.*

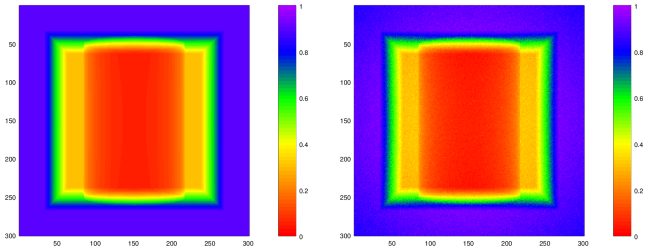


(a) *Mass attenuation coefficients for different materials.*



(b) *Density.*

Figure 5: *Material property.*



(a) *Deterministic simulated image using gVirtualXRay.*

(b) *MC simulated image using Geant4.*

Figure 6: *Example of validation test using a point source.*

3. Other functionalities

To make the library easily accessible to other developers, we make use of CMake and C++ 11. The library supports OpenGL 2.x, 3.x and 4.x and there is no use of old deprecated OpenGL functions. Utilities, such as matrix stacks and OpenGL state stacks, are included in the libraries to limit dependencies. The library also includes Marching Cubes [LC87], implicit modelling using Blobby Molecules, Metaballs and Soft Objects [Bli82], and a small imaging toolbox for tomography reconstruction, which is parallelised using OpenMP [Ope15].

More recently, we added procedural texturing of polygon meshes corresponding to the bones, lungs, diaphragm and liver (see Fig. 7). It takes care of specular and diffuse lighting as well as shadowing. The texture coordinates and texture maps are procedurally generated. Since the surface of anatomical tissues is not regular, we need to 'disturb' the surface of their polygon meshes. This is done using noise functions [EMP*02]. Normal and bump mapping is used to give a better aspect to all internal organs. Colour and texture for each type of tissues is unique. The different lobes of the lungs are not present in the polygon mesh as they are not visible in the CT images from which the geometries are extracted. They are added during the rendering phase using normal and bump mapping. Also, two types of visual appearance are visible on the diaphragm: The tendon part in white, and the muscle part in red. Again, this is not present in the CT data and it is done during the rendering step using GLSL.

4. Application examples

This section discusses the development of i) a projectional radiography training tool, and ii) a real-time respiration and X-ray

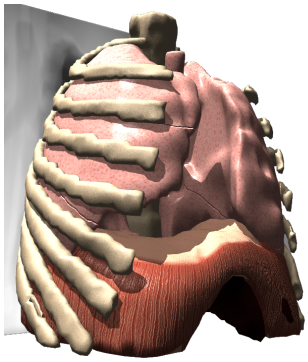


Figure 7: *Procedural texturing of bones, lungs, diaphragm and liver.*

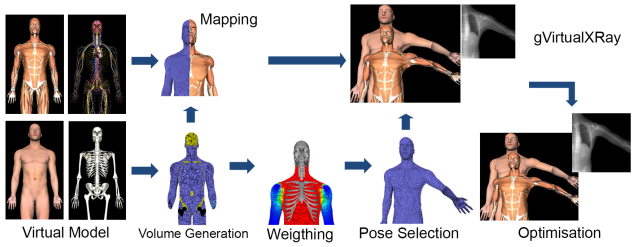


Figure 8: *Volume Generation and patient posing: i) Mapping and Weighting are performed as a pre-process step, ii) Pose selection runs at interactive rates, and iii) the Optimisation phase is performed once the final pose is selected by the user.*

simulations, which can be used in a percutaneous transhepatic cholangiography (PTC) or to simulate motion artefacts in reconstructed CT images.

4.1. Projectional Radiography

Projectional radiography is a common medical imaging diagnostic tool that can be used in almost every part of the patient's body. Each procedure requires a specific patient position and set up of the X-ray machine (incident beam of the source and the detector). These two steps are extremely important in actual medical procedures to acquire the best X-ray image and reduce the patient radiation dose. A virtual reality simulator allows to train projectional radiography in a safe environment without using real patients. This is actually an important feature as ionising radiations such as X-rays are harmful to patients.

The X-ray simulation is performed using gVirtualXRay. It requires a realistic human model with body structures of interest modelled using polygon meshes. Commercially available virtual human models, such as *ZygoteBody*^{TM} [Zyg], are usually given in a specific pose. Besides, some researchers have proposed the use of real patient data to build such models [SBLD15, SHD15]. The real patient data is usually captured using medical imaging techniques (such as CT, MRI or US) in a specific subject position. The problem is therefore to adapt the initial position to the pose required in the trained medical procedure [AHLG*13]. To tackle this problem, our simulator allows the user to modify interactively the virtual patient pose. It must be pointed out that the posing module and gVirtualXRay work together achieving interactive frame rates.

To achieve real-time performance, our posing module follows a purely geometrical approach, instead of a fully physically-based simulation algorithm such as finite element methods (FEMs). Nowadays, in the computer graphics industry, skeletal animation is the most used technique for animating articulated virtual characters. In classical skeletal animation, the rigging phase (a virtual skeleton is built) and the weighting phase (the influence of each bone in the mesh vertices is computed) are performed manually. Previous work, such as in [BTST11], tried to automate some stages of the skeletal animation but fully automatic processing still remains a challenge. Keeping in mind the need for a fully automatic procedure, we extended the technique proposed by Baran and Popovic [BP07] to deal with the internal tissues. Our algorithm defines a deformation field inside the model. It is used to transfer the movement to all the tissues. Since the pose selection process must run interactively and is supervised by a user, the most computationally expensive tasks must be moved to a pre-process step.

Fig. 8 shows the stages in which our algorithm is divided. Firstly, our technique generates a volumetric mesh. We will use this mesh to define the internal deformation field. Then, we compute each bone influence in the internal volume. Once the volumetric mesh is computed, the internal tissues are mapped into the volumetric mesh. These three stages have to be run once for each virtual patient. In our model, they are implemented in a pre-process stage.

During the Pose Selection phase, the virtual skeleton transfers its motion to the volumetric mesh. It generates a deformation field, which is used to transform the tissue meshes. Currently, our system allows transforming the virtual skeleton using direct kinematics: It can load a pre-recorded pose, or use the Microsoft Kinect [SSK*13] to capture the user's pose and transfer it to the virtual patient.

The pose selection phase is implemented to run on GPU. To speed up the simulator prototype, the pose selection and gVirtualXRay share the same virtual patient data in GPU memory. Fig. 9 shows the simulator interface. The simulator allows to configure and place the X-ray source.

Our algorithm implements *Dual Quaternion Skinning (DQS)* [KCŽO07] to compute the deformation field. DQS solves most of the *Linear Blending Skinning* problems. Even though, our system allows refining the initial solution using a physically based optimisation phase. The main goal of this step is to guarantee the volume conservation, solving the DQS volume gains (see Fig. 10). This optimisation step may be performed in a post-process stage.

4.2. Simulation of respiration motion

In this application, we aim at simulating the respiration motion and corresponding X-ray images. To simulate the respiration process, we studied the behaviour of each organ separately after having summarised the general process of the respiration. The state-of-the-art on simulating the respiration includes

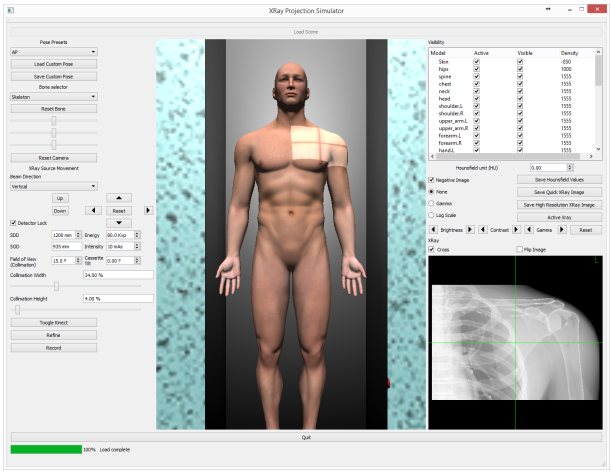


Figure 9: *Graphical user interface of our projectional radiography simulator.*

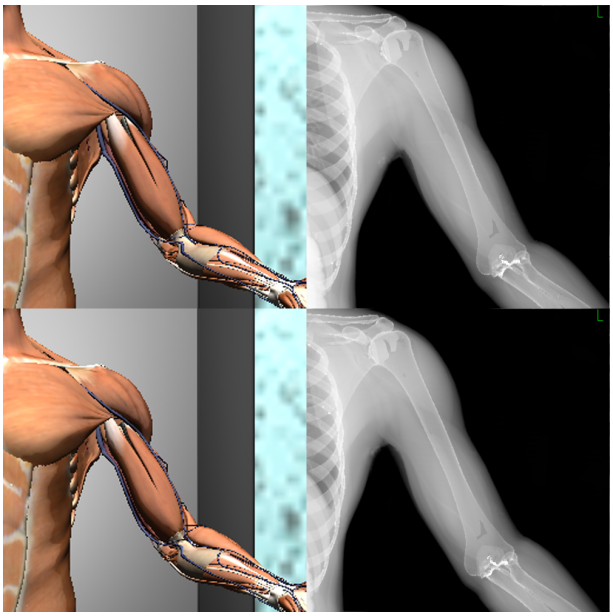


Figure 10: *: Comparison between DQS (top row) skinning and DQS+Optimisation (bottom row) result.*

real-time simulators and radiotherapy treatment planning. The first topic deals with fast models that are not accurate or pre-computed [MWF*16] where the physiological behaviour is extracted from 4-D CT scans. The second topic deals with very accurate models but are time-consuming [HDM*17].

4.2.1. Main principles of the respiration

The aim of the respiration is to inflate and deflate the lungs with air [BC74]. The lungs are passive organs: They are following muscles through a negative pressure inside the pleura. The lungs remain in contact with the muscles thank to the pleura. The main muscles involved in the respiration are the intercostal muscles and

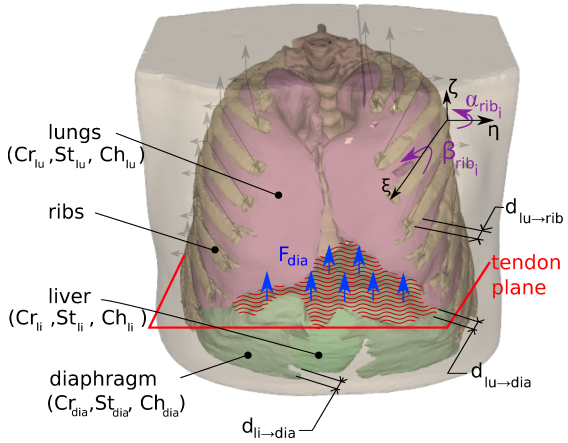


Figure 11: *Thorax anatomy and respiration model parameters*

the diaphragm. The intercostal muscles open and close the ribcage. The diaphragm contracts and relaxes applying an up and down motion that respectively pulls and pushes the bottom of the lungs. The same motion influences all the organs below the diaphragm, particularly the liver that will follow the up and down motion while being pressed by the lower ribs. To simulate all of these behaviours we start by modelling the active organs (the muscle influence) and then the passive organs. The whole anatomy is presented in Fig. 11.

Each organ is geometrically defined by a triangular mesh extracted from a segmentation of patient CT scans: initial meshes obtained by the Marching Cube algorithm are decimated to reduce the number of polygons and maintain a regular sampling rate over the mesh [LL10].

The ribcage motion is provoked by the intercostal muscles. While it is possible to model the action of each muscle tendon like in [ZCCD04], we chose to rather simulate the consequences of their action: the rib motion. We used the rib kinematic model presented in [WLGT01]. It consists in modelling each rib motion by a kinematic model defined by two rotations. Each rib rib_i is contained in a plan that has a certain orientation relative with the original coordinate system . They defined a new coordinate system (ξ,η,ζ), where ξ is the intersection of the plane of the rib and the sagittal midplane, η the intersection of the plane of the rib and the perpendicular of the sagittal midplane, and ζ is perpendicular to the plane of the rib Fig. 11. Values α_{rib_i} and β_{rib_i} are the angles between the new axis and the original ones. α_{rib_i} is called the "pump handle" angle, and β_{rib_i} is called the "bucket handle" angle. The rotation of the ribs was thus decomposed into two distinct rotations. The rotation angles are given by Equation (5) where $\alpha^{max}_{rib_i}$ and $\beta^{max}_{rib_i}$ are the maximum rotation angles for α_{rib_i} and β_{rib_i} respectively, f is the breathing frequency and t is the current simulation time-step.

$$\begin{aligned} \alpha_{rib_i} &= \alpha^{max}_{rib_i} * (0.5 * cos(t \times f/\pi - \pi) + 1 \\ \beta_{rib_i} &= \beta^{max}_{rib_i} * (0.5 * cos(t \times f/\pi - \pi) + 1 \end{aligned} \tag{5}$$

The diaphragm is mainly composed of two parts: A muscular part that contacts and relaxes, and an almost rigid part (the central

tendon). We modelled the separation of these two components using the Cartesian equation of a plane. We called it "tendon plane" in Fig. 11. A displacement vector F_{dia} is applied to all the vertices above this plan defined by a sinusoidal movement given by Equation (6). F_{dia}^{max} is the maximum displacement of the central tendon. During inhalation, the central tendon has a downward movement, which is synchronous with the expanding ribcage.

$$F_{dia} = F_{dia}^{max} * (0.5 * cos(t \times f/\pi - \pi) + 1 \quad (6)$$

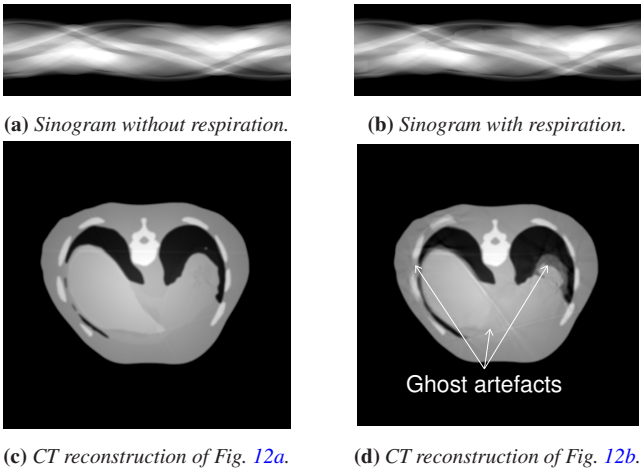
The muscular part is elastic in order to stretch during diaphragm relaxation. Various techniques exist to model soft tissue deformation behaviour [Mt05]. We chose the Chainmail method [Gib97] because it is fast enough for our context of real-time simulation while being linked to a physical meaning with its three parameters: the compression (here Cp_{dia}), the stretching (here: St_{dia}) and the shearing (here: Sh_{dia}). When a ChainMail object is manipulated, it can stretch or contract according to strict displacement rules between a point and its neighbours. These rules are based on minimum and maximum distances that are allowed between a point and its neighbours. Thus the worst cases for a point and its neighbours are either maximal compression or maximal stretch. The Chainmail method has recently proven its efficiency in medical application [RLA16].

The liver has both a large motion and a large deformation during respiration. The deformation is modelled in a similar way as the muscular part of the diaphragm (i.e. using Chainmail). Three parameters Cp_{li}, St_{li} and Sh_{li} are tuned to customise the model to a given patient. The liver should be attached to the diaphragm as it is the case in reality through the falciform ligament. This is modelled as follow: we defined a minimum distance of attachment $d_{li \to dia}$. A set of points is extracted so that the euclidean distance from the diaphragm is inferior to $d_{li \to dia}$. This set of points follows the displacement vector F_{dia}. The vertex position of the other points of the liver are computed with Chainmail, just like the diaphragm.

The lungs' behaviour is linked to both the ribcage and the diaphragm. The lungs are also deformable. Similarly to the liver, two distances of attachment are defined: $d_{lu \to dia}$ and $d_{lu \to rib}$. One to attach the lungs to the ribs and one to attach the lungs to the diaphragm. If two vertices belong to both organs, an average displacement is computed. The displacement of the other vertices ave the expected "chain reaction" according to the ChainMail rules similarly to the other two deformable organs.

4.2.2. Respiration parameter tuning

Simulating the respiration needs the fine tuning of all the parameters of the organ models defined above. The aim is to fit the mathematical model on real data. As in curve fitting, the parameters can be automatically optimised. We use artificial evolution due to the complexity of the model and data [VVL12]. The optimisation method relies on an adaptive real-coded genetic algorithm defined by elitism, mutation, crossover and new blood. The fitness function (also called objective function) is based on an error measurement between the respiration simulation and ground-truth extracted from a real 4-D (3-D + time) CT scan of patients.



(a) *Sinogram without respiration.* **(b)** *Sinogram with respiration.*

(c) *CT reconstruction of Fig. 12a.* **(d)** *CT reconstruction of Fig. 12b.*

Figure 12: *Motion artifacts.*

4.2.3. Applications

The library enables us to generate real-time X-ray images with respiration simulation [VV16]. It can be used in medical training simulators for percutaneous transhepatic cholangiography [V*09] and to easily add realism in the visual appearance of inner organs. It is also possible to simulate a complex computed tomography acquisition process to produce realistic sinograms and reconstruct CT slices with motion artifacts (see Fig. 12) [VV15]. Our framework can be used to evaluate CT reconstruction algorithm with motion artifact correction in a controlled environment. Demos and videos are available on the project's page and on YouTube at http://tinyurl.com/gqrdgvm.

5. Conclusion

We have developed a modern OpenSource library for C++ to simulate X-ray transmission images on GPU using the Beer-Lambert law with polychromatism and taking into account the shape of the source. The library is both fast and accurate. It can be used in different application contexts, ranging from medical training simulators, to simulate the imaging chain in non-descructive testing (NDT) and CT acquisition process.

Future work includes plugin developments for SOFA (an open-source simulation framework for medical simulation) (http://www.sofa-framework.org) and H3D (an open-source haptics software development platform) (http://www.h3dapi.org/). Texturing of other soft tissues will be also added. Biding to other programming languages is also considered.

Acknowledgements

This work was partially supported by the European Commission, Grants no. 321968 and no. 610425.

References

[A*03] AGOSTINELLI S., ET AL.: Geant4–a simulation toolkit. *Nucl Instrum Methods Phys Res A 506*, 3 (2003), 250 – 303. doi:10.1016/S0168-9002(03)01368-8. 1, 3

[AHLG*13] ALI-HAMADI D., LIU T., GILLES B., KAVAN L., FAURE F., PALOMBI O., CANI M.-P.: Anatomy transfer. *ACM Trans. Graph. 32*, 6 (Nov. 2013), 188:1–188:8. doi:10.1145/2508363.2508415. 5

[B*04] BIELAJEW A., ET AL.: *History, overview and recent improvements of EGS4*. Tech. rep., Stanford Linear Accelerator Center (SLAC), 2004. doi:10.2172/839781. 1

[B*12] BERT J., ET AL.: Hybrid GATE: A GPU/CPU implementation for imaging and therapy applications. In *2012 IEEE Nuclear Science Symposium and Medical Imaging Conference Record (NSS/MIC)* (Oct 2012), pp. 2247–2250. doi:10.1109/NSSMIC.2012.6551511. 1

[BC74] BOUCHET A., CUILLERET J.: *Anatomie topographique, descriptive et fonctionnelle: le thorax - première et deuxième partie*. Simep Éditions, 1974. ISBN: 2853341763. 6

[BHS*10] BERGER M. J., HUBBELL J. H., SELTZER S. M., CHANG J., COURSEY J. S., SUKUMAR R., ZUCKER D. S., OLSEN K.: *XCOM: Photon Cross Section Database*. Tech. Rep. NBSIR 87-3597, National Institute of Standards and Technology, Gaithersburg, MD, 2010. URL: http://physics.nist.gov/xcom. 3

[Bli82] BLINN J. F.: A generalization of algebraic surface drawing. *ACM Trans Graph 1*, 3 (July 1982), 235–256. doi:10.1145/357306.357310. 4

[BP07] BARAN I., POPOVIĆ J.: Automatic rigging and animation of 3d characters. *ACM Trans. Graph. 26*, 3 (July 2007). doi:10.1145/1276377.1276467. 5

[BSFVS95] BARÓ J., SEMPAU J., FERNÁNDEZ-VAREA J. M., SALVAT F.: PENELOPE: An algorithm for Monte Carlo simulation of the penetration and energy loss of electrons and positrons in matter. *Nucl Instrum Methods Phys Res B 100*, 1 (1995), 31–46. doi:10.1016/0168-583X(95)00349-5. 1

[BTST11] BHARAJ G., THORMÄHLEN T., SEIDEL H.-P., THEOBALT C.: Automatically rigging multi-component characters. *Comp. Graph. Forum (Proc. Eurographics 2012) 30*, 2 (2011). 5

[EMP*02] EBERT D. S., MUSGRAVE F. K., PEACHEY D., PERLIN K., WORLEY S. (Eds.): *Texturing and Modeling, A Procedural Approach*, 3rd ed. Morgan Kaufmann, 2002, ch. Real-time procedural solid texturing, pp. 6–94. 4

[FDLB06] FREUD N., DUVAUCHELLE P., LÉTANG J. M., BABOT D.: Fast and robust ray casting algorithms for virtual x-ray imaging. *Nucl Instrum Methods Phys Res B 248*, 1 (2006), 175–180. doi:10.1016/j.nimb.2006.03.009. 1

[Gib97] GIBSON S. F. F.: *Linked Volumetric Objects for Physics-based Modeling*. Tech. Rep. TR97-20, Mitsubishi Electric Research Labs, 1997. 7

[HDM*17] HAN L., DONG H., MCCLELLAND J., HAN L., HAWKES D., BARRATT D.: A hybrid patient-specific biomechanical model based image registration method for the motion estimation of lungs. *Med Image Anal 39* (2017), 87–100. doi:10.1016/j.media.2017.04.003. 6

[JZJ14] JIA X., ZIEGENHEIN P., JIANG S. B.: GPU-based high-performance computing for radiation therapy. *Phys Med Biol 59*, 4 (2014), R151–R182. doi:10.1088/0031-9155/59/4/R151. 1

[KCŽO07] KAVAN L., COLLINS S., ŽÁRA J., O'SULLIVAN C.: Skinning with dual quaternions. In *Proceedings of the 2007 symposium on Interactive 3D graphics and games* (2007), ACM, pp. 39–46. 5

[LC87] LORENSEN W. E., CLINE H. E.: Marching cubes: A high resolution 3D surface construction algorithm. *SIGGRAPH Comput Graph 21*, 4 (Aug. 1987), 163–169. doi:10.1145/37402.37422. 4

[LL10] LÉVY B., LIU Y.: Lp centroidal voronoi tessellation and its applications. *ACM Trans Graph 29*, 4 (2010), 119:1–119:11. doi:10.1145/1833349.1778856. 6

[Mt05] MEIER U., *et al.*: Real-time deformable models for surgery simulation: a survey. *Comput Methods Programs Biomed 77*, 3 (2005), 183–197. doi:10.1016/j.cmpb.2004.11.002. 7

[MWF*16] MASTMEYER A., WILMS M., FORTMEIER D., SCHRÖDER J., HANDELS H.: Real-time ultrasound simulation for training of US-guided needle insertion in breathing virtual patients. In *Medicine Meets Virtual Reality 22: NextMed/MMVR22* (2016), vol. 220 of *Stud Health Technol Inform*, IOS Press, pp. 219–226. 6

[Ope15] OPENMP ARCHITECTURE REVIEW BOARD: OpenMP application programming interface version 4.5, Nov. 2015. URL: http://www.openmp.org/wp-content/uploads/openmp-4.5.pdf. 4

[RLA16] RODRIGUEZ A., LEON A., ARROYO G.: Parallel deformation of heterogeneous chainmail models: Application to interactive deformation of large medical volumes. *Comput Biol Med 79* (2016), 222–232. doi:10.1016/j.compbiomed.2016.10.012. 7

[SBLD15] SERRURIER A., BÖNSCH A., LAU R., DESERNO T. M.: Mri visualisation by digitally reconstructed radiographs. In *SPIE Medical Imaging* (2015), International Society for Optics and Photonics, pp. 94180I–94180I. 5

[SBS00] SCHNEIDER W., BORTFELD T., SCHLEGEL W.: Correlation between CT numbers and tissue parameters needed for Monte Carlo simulations of clinical dose distributions. *Phys Med Biol 45*, 2 (Feb. 2000), 459–78. 3

[SHD15] SERRURIER A., HERRLER A., DESERNO T.: Towards realistic patient-specific human models for virtual reality regional anaesthesia simulation. In *Proceedings of 34th ESRA Congress* (2015), pp. 02–05. 5

[SSK*13] SHOTTON J., SHARP T., KIPMAN A., FITZGIBBON A., FINOCCHIO M., BLAKE A., COOK M., MOORE R.: Real-time human pose recognition in parts from single depth images. *Communications of the ACM 56*, 1 (2013), 116–124. 5

[V*09] VILLARD P., ET AL.: Simulation of percutaneous transhepatic cholangiography training simulator with real-time breathing motion. *Int J Comput Assist Radiol Surg 4*, 9 (Nov. 2009), 571–578. doi:10.1007/s11548-009-0367-1. 2, 7

[VGF*09] VIDAL F. P., GARNIER M., FREUD N., LÉTANG J. M., JOHN N. W.: Simulation of X-ray attenuation on the GPU. In *Theory and Practice of Computer Graphics (TCPG'09)* (June 2009), Eurographics, pp. 25–32. doi:10.2312/LocalChapterEvents/TPCG/TPCG09/025-032. 1, 2, 3

[VV15] VIDAL F. P., VILLARD P.-F.: Simulated motion artefact in computed tomography. In *Eurographics Workshop on Visual Computing for Biology and Medicine* (2015), Bühler K., Linsen L., John N. W., (Eds.), The Eurographics Association, pp. 213–214. doi:10.2312/vcbm.20151228. 7

[VV16] VIDAL F. P., VILLARD P.-F.: Development and validation of real-time simulation of X-ray imaging with respiratory motion. *Comput Med Imaging Graph 49* (Apr. 2016), 1–15. doi:10.1016/j.compmedimag.2015.12.002. 1, 2, 3, 7

[VVL12] VIDAL F. P., VILLARD P.-F., LUTTON E.: Tuning of patient specific deformable models using an adaptive evolutionary optimization strategy. *IEEE Transactions on Biomedical Engineering 59*, 10 (Oct. 2012), 2942 – 2949. doi:10.1109/TBME.2012.2213251. 7

[WLGT01] WILSON T. A., LEGRAND A., GEVENOIS P., TROYER A.: Respiratory effects of the external and internal intercostal muscles in humans. *J Physiol 530*, 2 (2001), 319–330. doi:10.1111/j.1469-7793.2001.0319l.x. 6

[ZCCD04] ZORDAN V. B., CELLY B., CHIU B., DILORENZO P. C.: Breathe easy: Model and control of simulated respiration for animation. In *ACM SIGGRAPH/Eurographics symposium on Computer animation* (July 2004), pp. 29–37. doi:10.1145/1028523.1028528. 6

[Zyg] ZYGOTE MEDIA GROUP: ZygoteBODY. URL: https://www.zygotebody.com/. 5