Ultrasound simulation with animated anatomical models and on-the-fly fusion with real images via path-tracing

Journal Article

Author(s):
Starkov, Rastislav; Tanner, Christine; Bajka, Michael; Goksel, Orcun

Publication date:
2019-08

Permanent link:
https://doi.org/10.3929/ethz-b-000345177

Originally published in:
Computers & Graphics 82, https://doi.org/10.1016/j.cag.2019.05.005

Funding acknowledgement:
179116 - Imaging Soft Tissue Elasticity (SNF)

Ultrasound Simulation with Animated Anatomical Models and On-the-Fly Fusion with Real Images via Path-Tracing

Rastislav Starkov, Christine Tanner, Michael Bajka, Orcun Goksel

Abstract—**Ultrasound is an essential imaging modality in clinical screening and diagnosis, for reducing morbidity and improving quality of life. Successfully performing ultrasound imaging, however, requires extensive training and expertise in navigating a hand-held probe to a correct anatomical location as well as subsequently interpreting the acquired image. Computer-generated simulations can offer a safe, flexible, and standardized environment to train such skills. Data-based simulations display interpolated slices from a-priori-acquired real ultrasound volumes, whereas generative simulations aim to reproduce the complex ultrasound interactions with comprehensive, geometric anatomical models, such as using ray-tracing to mimic acoustic propagation. Although sonographers typically focus on relatively smaller structures of interest in ultrasound images, the fidelity of the background anatomy may still play a role in contributing to the realism of a generated US image; e.g. when imaging a relatively smaller fetus within large abdominal background. It was proposed earlier to compose ray-traced images with acquired volumes in a preprocessing step. Despite its simplicity, this prevents any view-dependent artifacts and interactive model changes, such as those induced by animations, which can, for instance, model fetal motion. To fully leverage the flexibility of the model-based generative approach, we propose herein an on-the-fly image fusion, based on the two techniques, by moving the interpolation stage within the ray-tracer, such that the pre-acquired image data can be referred to in the background, while the acoustic interactions with the model can be resolved in the foreground. This allows for animated anatomical models, which we realize during simulation runtime via scene-hierarchy *subtree switching* between precomputed acceleration structure graphs. We demonstrate our proposed techniques on ultrasound sequences of fetal and heart motion, where only animated models can afford to meet realism requirements entailed by the temporal domain.**

I. Introduction

Ultrasound (US) is an imaging modality that offers safe, low-cost and real-time means of clinical examinations. However, its low signal-to-noise ratio and direction-dependent imaging artifacts complicate image interpretation for diagnosis. Furthermore, navigating the imaging probe requires certain skills such as excellent hand-eye coordination with respect to anatomical knowledge. Such skills can make a difference between identifying a cancerous tumor or missing one during screening, which may have critical consequences for the patient. Thus, both acquisition and interpretation of ultrasound image information heavily relies on the experience and skill of a sonographer. This accordingly necessitates an extensive training to ensure the success of clinical outcomes.

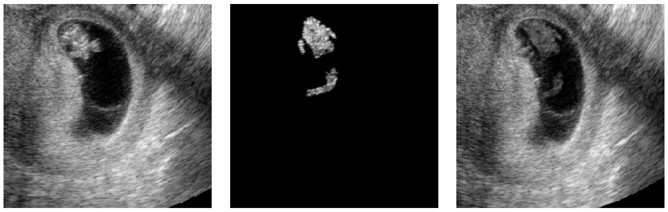
Training is possible with tissue-mimicking phantoms, cadavers, and volunteers, which all have associated ethical and realism issues. Furthermore, with these aforementioned methods, it is also not possible to cover a sufficiently wide range of training scenarios for a comprehensive learning experience. That motivates the need for computer-based simulations, where arbitrary content can be created and incorporated in a safe, repeatable training environment. Such simulations can also allow for experiencing rare cases, which are unlikely to be encountered in regular clinical routine for live training [1].

Data-based US simulations can provide relatively high image realism [2, 3, 4, 5, 6], where image slices are interpolated during simulation time from a-priori-acquired US volumes, and can even be altered by means of interactive tissue deformations [7]. However, acquisition of rare cases with representative diversity, crucial for comprehensive training, is a major challenge. For instance, in obstetrics it is infeasible to collect US volumes of fetuses at all gestational ages, at different position/orientations, with different anatomical variations, and all combinations thereof with standardized image quality. Moreover, a substantial variety of pathological appearance introduces another degree of freedom, which altogether pose a major limitation for image-based methods for the purposes of extensive training. Lastly, prepossessing and storage of acquired US volumes further complicates the implementation of data-based approaches.

Alternatively, model-based techniques [8, 9, 10] aim to generate US images by means of simulating US interaction with an anatomical representation; for instance, by using ray-tracing techniques on triangulated surface models of anatomy [11, 12, 13]. Firstly, such approaches require that the acoustic propagation and tissue interaction are simulated realistically for rendering believable images; secondly, the scene geometry needs to be modeled true to the human anatomy. Given accurate anatomical models, state-of-the-art methods using Monte-Carlo path-tracing techniques are shown to produce highly realistic ultrasound images at interactive frame rates [13]. Therefore, model-based methods are limited not by prior image acquisition, but rather by the anatomical modeling effort; i.e., any anatomical variation that can be modeled, can potentially be simulated. Unfortunately, precise modeling at high, discernible detail is a time-consuming effort and can only be afforded in small regions of actual clinical

Rastislav Starkov · Christine Tanner · Orcun Goksel are with Computer-assisted Applications in Medicine, Computer Vision Lab, ETH Zurich, Switzerland, e-mail: {*rastislav.starkov,tanner,ogoksel*}@vision.ee.ethz.ch

Michael Bajka is with Department of Gynecology, University Hospital of Zürich, Switzerland, e-mail: *michael.bajka@hin.ch*

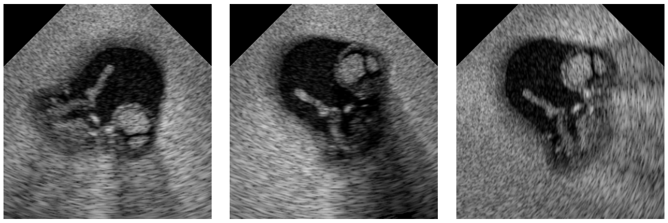


(a) Original volume (b) Ray-traced (c) Fused volume

Fig. 1: Example of the approach in [14]: original volume (a) and ray-traced fetus/volume (b) are fused via image *stitching* (c).

interest. For instance, common obstetric examinations during pregnancy focus on the fetus itself. For their training, a fetus as well as any physiological and pathological variation thereof can be modeled in detail based on anatomical literature and expertise. Although the surrounding anatomy of the mother, e.g. abdominal structures such as the intestines, is also visible in large parts (background) of the exam images, it is often not the focus of interest for the examination and it would be counter-productive to spend modeling effort and time for its realistic simulation. Nevertheless, omitting such background image appearance in a simulation would still affect the realism of the entire image and could hinder the authentic experience of performing an US examination.

In earlier work [14] we introduced a combination of data- and model-based approaches as a compromise between the two, where the region of clinical interest is modeled in detail and simulated by means of Monte-Carlo ray-tracing, and then fused in a preprocessing step with lower-interest background images collected in-vivo, as presented in Fig. 1. This reduces modeling efforts, but requires fusing rendered and acquired volumes offline, to be later used in real-time in an *image-based* simulation where planar slices can be interpolated, e.g. using [7]. However, such offline fusion relies on a lengthy and tedious preprocessing pipeline, before the volumes can actually be used for ultrasound simulation. Furthermore, the fused volumes are eventually used in an image-based simulation fashion, which imposes limitations on faithfully reproducing imaging-direction and model-state dependent artifacts, as illustrated in Fig. 2; e.g. the ultrasonic shadows would still cast as in the preprocessed imaging direction, but not in the interactive imaging direction, and any changes in the model such as its movement or animation cannot be represented in the images. In this work we propose an on-the-fly fusion of prerecorded background anatomy and simulated content incorporated into US ray-tracing, as shown in Fig. 3. This eliminates the need for the aforementioned offline prepossessing as well as permits to faithfully represent direction-dependent ultrasound artifacts and create simulations with moving and animated models. For realizing animations with (sub)structures of complex models at interactive frame-rates, we employ a simple scene subgraph switching scheme with precomputed acceleration structures for Monte-Carlo path-tracing.



(a) 0°, ray-traced (b) 90°, ray-traced (c) 90°, image-based

Fig. 2: Comparison of view-dependent imaging characteristics between different simulation approaches. It can be seen that if the simulated image (a) is simply rotated, e.g. by 90°, to achieve the orientation in (b), using image-based techniques such as in [7, 14], then the acoustic shadowing artifact would be pointing in a completely wrong and unrealistic direction (c). Furthermore, orientations of speckles would similarly be incorrect. Therefore, interactive ray-tracing and rendering as proposed in this work are necessary for realistic images.

II. Methods

Below we first summarize Monte-Carlo ray-tracing for simulating ultrasound for the sake of completeness. We then propose the incorporation of dynamic, animated models. In particular, we discuss computational and storage limitations posed by naive approaches and how we propose to overcome these using node-switching via *selectors* between prebuilt acceleration structures at animation keyframes. Next, we introduce runtime compounding of the ray-traced output with pre-acquired volumes from clinical examinations, within our path-tracing framework.

A. Path-tracing for Ultrasound Simulation

For ultrasound imaging, a transducer first emits an acoustic wave as a short modulated pulse of a few wavelengths. As this acoustic wavefront travels within the tissue at certain speed of sound $\mathbf{c}$, it may reflect and refract at anatomical boundaries based on the acoustic impedance ($\mathbf{Z}$) difference between respective tissues. The wavefront is meanwhile also omni-directionally scattered by (uncountably) many subwavelength structures in the tissues, such as micro vasculature, cell nuclei, organelles, and large proteins. Scattering and reflections send a portion of the acoustic energy back to the transducer, which listens, samples and records these echo signals digitally. After an image-formation process called *beamforming*, such a radio-frequency (RF) signal goes through a post-processing pipeline with time-gain compensation (to account for depth-based acoustic attenuation), signal demodulation (to determine the echo amplitude to display), and log-compression (to compress the large dynamic range of a signal to grayscale image values). This yields a single column (scanline) of an image as in Fig. 4. The procedure is repeated multiple times for a moving window of transducer elements, and all the scanlines are subsequently transformed into screen Cartesian coordinates (scan-conversion), to create an ultrasound image frame known as (brightness) *B-mode* image.

Monte-Carlo integration. Ray-based methods model wave propagation with a collection of rays that are cast from the

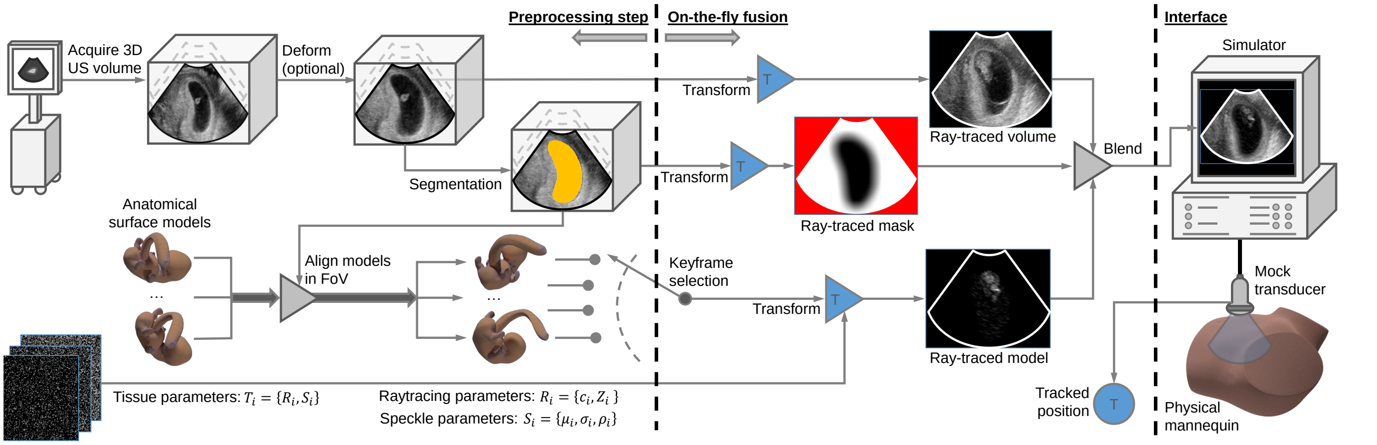
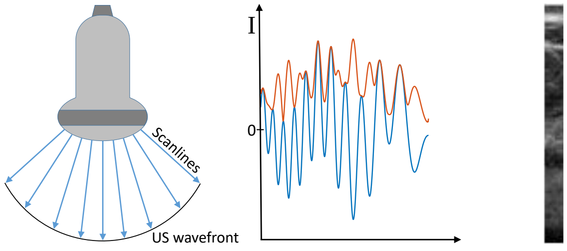
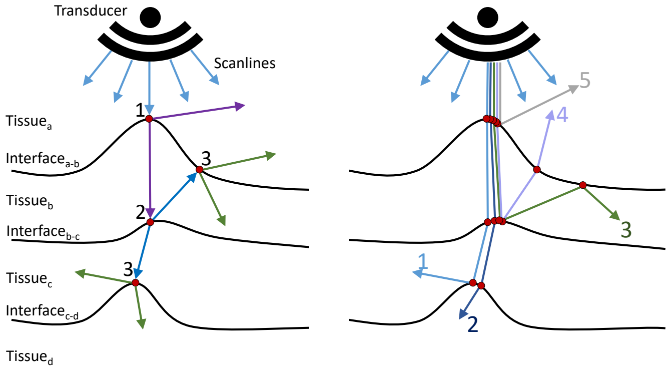


Fig. 3: Overview of the on-the-fly fusion workflow. Segmented surfaces S, e.g. a gestational sac, provide location information that can be used to position triangulated surface models (static or animated) in a desired location in the volume. Such segmented surfaces can also act as masking geometry and/or outermost anatomy for the simulated foreground model. Transducer transformation T is provided on-the-fly either by the user in a GUI or by a position tracking sensor attached to an interactively manipulated mock transducer. The complete set of tunable (ray-tracing) parameters R can be found in our earlier work [13].



(a) Transducer plane (b) RF signal (c) Scanline

Fig. 4: Convex transducer (a) sends an US wave-front. Recorded signal (b) for an individual scanline is postprocessed to produce a corresponding image (c).

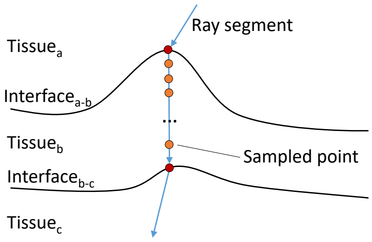


(a) Deterministic (binary tree) (b) Stochastic

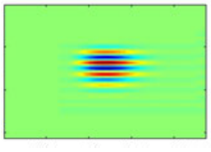
Fig. 5: Rays generated by a deterministic ray-tracer always split into reflected and refracted rays, gradually producing deeper recursion levels labeled by progressively larger numbers in (a), whereas rays cast by a stochastic Monte-Carlo ray-tracer take arbitrary (both in terms of the number and the orientation of constituent ray segments) paths (marked by numbers in (b)) while travelling through tissue.

surface of a transducer [11, 12], producing echo signal along scanlines, as they travel through anatomy. The amplitude values $I(t)$ of an RF signal are then found by integrating over all contributions [12] recorded by the transducer at time t. Similarly to US, ray-surface intersections (tissue interfaces) may result in a reflection or a refraction. Note that in the context of ray-tracing an intersection point is parameterized by a differential area that defines a distribution of all possible outgoing rays. Then to account for the exitant energy, it is required to cast secondary rays across the entire set of directions defined by the distribution function. Stochastic Monte-Carlo ray-tracing is well equipped to process such multidimensional integrals [15]. Furthermore, even if each surface acted as a perfect mirror (the normals in the differential area face the same direction) [11], necessitating only one reflected and one refracted ray at every surface intersection, as shown in Fig. 5, a deterministic ray-tracer would suffer from an exponential growth of secondary rays. In contrast, a stochastic Monte-Carlo ray-tracer can reduce computation time while providing similarly converged values by probabilistically sampling each time either a reflected or a refracted ray. Alternatively, at every intersection point a surface normal is sampled, according to the provided distribution function, which is then used in conjunction with Snell's law and Fresnel equations to compute the direction and the energy of an outgoing ray (reflected or refracted). Moreover, Monte-Carlo integration enables a number of importance sampling strategies [16], which can reduce the variance of numerical integration. These techniques, which are described in greater detail in [13], account for a part of what constitutes the visual appearance of ultrasound imaging, namely anatomical boundaries.

Speckle texture. The characteristic ultrasound texture is composed of a speckle pattern, which is a major factor in the perception of US images, as well as in clinical imaging and



(a) Ray-sampling



(b) Typical PSF

Fig. 6: Simulating speckle texture. (a) Each ray segment between anatomical boundaries (enclosed within tissue) is sampled according to the given temporal sampling rate of a simulated ultrasound system and the speed-of-sound in a particular tissue, which together define the axial resolution of a transducer. (b) Scatterers around sampled locations are then convolved with a kernel – point-spread function (PSF).

diagnostics. Speckles are the result of wave interferences produced by ultrasound interacting with innumerable "scatterers" in the body. Likewise, to generate a physically-based speckle pattern when simulating US, rays should similarly interact with scatterers as they propagate through tissues. We use well-established convolution-based methods for generating speckle patterns [17]. Assuming axially-separable Point-Spread Function (PSF), fast convolutions can be computed [9]. This method was tailored for use in ray-tracing-based simulations in [11, 12, 13], where a tissue-specific spatial distribution of scatterers is represented by a voxel grid. The grid elements are defined by the density ρ of the scatterers inside the tissue as well as their amplitudes, which are obtained by sampling from a Gaussian distribution (μ_s, σ_s). Then given stationary anatomy, the speckle pattern is produced by first generating equally spaced samples on ray segments, as illustrated in Fig. 6, where for each successive sample, the ray intensity is attenuated according to Beer-Lambert law. Then a static, anatomy-specific distribution of scatterers around these sampled points is convolved with the PSF of the simulated transducer.

Image formation. To combine the convolution-based speckle pattern generation method with the ray-tracing routine, echo amplitudes from processing both the anatomical boundaries and the scatterers (from the bulk tissue between the boundaries) are summed up. Thereafter, convolution is performed on the combined intensities to obtain an RF signal. Subsequently, the RF signals are postprocessed to produce the final ultrasound image.

B. Animation Pipeline

In addition to probe manipulation by an examiner, in-vivo US examination often involves a wide range of anatomical motions and displacement, for instance due to haphazard fetal muscle activity or periodic movements, such as the beating heart or breathing. Omitting these dynamic effects (e.g., as in [2, 5] and even when they are user-interactive as in [7, 14]) results in lifeless static anatomy, impairing the realism of the produced simulations. Incorporating animated models into ray-tracing-based methods, however, is not a straightforward task.

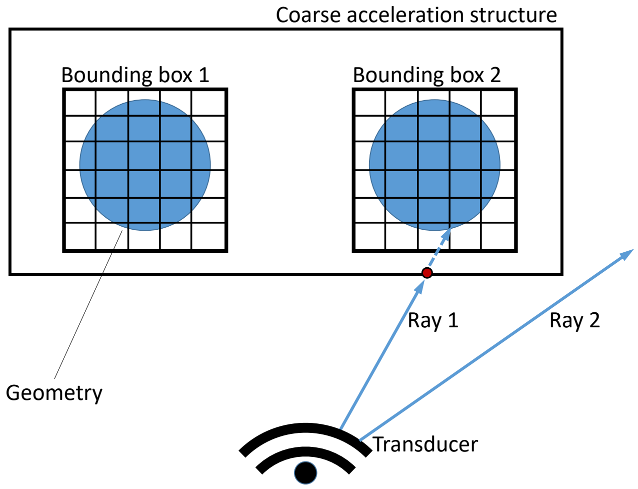


Fig. 7: Once Ray 1 encounters the initial intersection with the coarse acceleration structure, it examines the two underlying bounding boxes to find the next intersection and then proceeds to traverse the fine acceleration structure referenced by Bounding Box 2. Meanwhile, Ray 2 fails the intersection check with the union bounding box and is immediately terminated.

Ray-based methods rely heavily on *acceleration structures* [18] in one form or another, so that ray-geometry intersections can be calculated in an efficient fashion and also a large number of rays can be terminated early, as illustrated in Fig. 7. Accordingly, it significantly reduces computation time spent on processing redundant ray-surface intersections, and makes such ray-tracing possible in the first place. Every level of a generated acceleration structure graph contains bounding boxes (the leaves of which are geometric primitives, e.g. triangles), that enable efficient graph traversal in the scene hierarchy. Typically, these graphs are precomputed once and reused throughout the simulation. However, when non-static geometry is introduced into the scene, existing acceleration structures are invalidated. Consequently, they need to be rebuilt or refit [19] for every animation frame, which can be a significant computational burden for rendering at interactive frame rates.

We herein use two methods to circumvent the above scene-graph update problem. One way is to keep the entire animated sequence, along with its prebuilt acceleration structures, on the GPU. Then, specific geometries can be extracted from the device memory depending on which animation key frame needs to be visualized. Alternatively, for certain types of simplistic motion that can be represented by affine transformations, such as displacement and rotation of the fetus within the gestational sac or palpation of arteries, we use a method similar to instancing [20]. Without loss of generality, below we describe these approaches to enable animated models using the nomenclature of the ray-tracing framework NVIDIA OptiX [21], which is the framework that we employ for our GPU-based ultrasound simulator. The methods, nevertheless, are generic and can be adopted to similar constructs available in other ray-tracing frameworks.

Rebuilding acceleration structures. Consider a high-level node (i.e., the so-called *group* in NVIDIA OptiX) of the scene hierarchy, which has a number of children nodes (*geometry groups*) that contain actual geometric data (*geometry instances*) and respective acceleration structures. By swapping one of the children nodes with another geometry group (or simply alternating geometry instances of a particular group) that is currently not part of the hierarchy, we can ray-trace different scenes (i.e. node exchange) at different frames, thus producing animated sequences. However, a naive implementation of such an approach requires, at each animation step, re-uploading the geometry to GPU and rebuilding its complete acceleration structure. Interactive ray-tracing can greatly benefit from using object hierarchy subdivision, instead of spatial subdivision [19], for which specialized fast builders [22] can be employed, albeit at the cost of reduced scene traversal efficiency. Consequently, performance of ray-tracing in animated scenes can be noticeably improved by caching keyframe models on the GPU, such that when they are swapped, only the coarse acceleration structures of the higher scene-graph level nodes need to be updated. Nevertheless, the particular implementation of this in OptiX is not clear to us, e.g. whether or not a geometry node is kept in the device memory when it is explicitly removed from the scene tree when exchanging with another keyframe model. Therefore, instead of a fallback to completely rebuilding acceleration structures, we resort to a guaranteed strategy that keeps the loaded geometry on the GPU, as described below.

Selector nodes. Instead of re-uploading the geometry for every subsequent animation frame, it is possible to store the complete set of surface models that comprise the animation, along with their respective acceleration structures, on the GPU, and then simply swap between them when required by means of *selector nodes* [21]. Such an approach permits the use of a particular class of acceleration structure builders, termed as Split Bounding Volume Hierarchy (SBVH), tailored for a fast scene traversal at the expense of longer build times [22, 23]. To be able to traverse an acceleration structure of a scene that contains a selector node, the bounding box of the latter is chosen to be the union of all the bounding boxes of its children, such that it encompasses the span of any possible geometry which can be picked during ray-tracing. Hence, the acceleration structures of the higher scene-graph nodes are not updated when a selector swaps keyframe models. The downside is that such an approach results in a higher number of false intersections, as shown in the scene configuration presented in Fig. 8, as compared to rebuilding acceleration structures. Note that in the latter case an acceleration structure update happens only once per frame. In contrast, the number of false intersections encountered during a single execution of a ray-tracing kernel is unbounded and grows based on the number of samples generated per scanline/elevation layer [13], as well as the depth of path-tracing recursion. Thus, the choice between using selectors and exchanging children nodes comes down to a trade-off between the time it takes to rebuild acceleration structures vs. computations spent on processing false intersections.

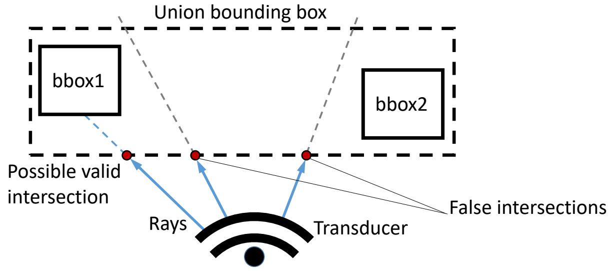


Fig. 8: Two objects contained within bounding boxes 1 and 2 correspond to two different key frames of an animated sequence. Substantial difference in the positions of the two objects results in a large union bounding box. Consequently, regardless of which geometry is currently picked by the selector, a significant portion of the rays will report intersections with the union bounding box, even though further processing will show that no actual ray-surface intersection takes place.

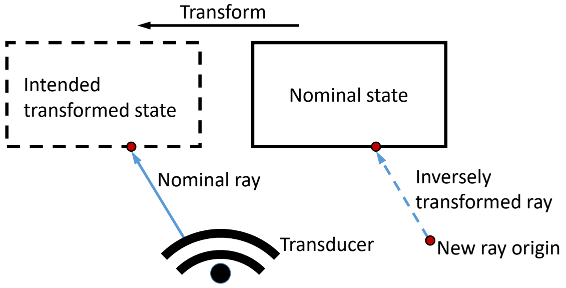


Fig. 9: Consider an affine transformation which results in a translation of an object from its nominal state to an intended translated state. Once the translation is performed, the nominal ray will be able to report an intersection with the surface of the transformed object. However, if the inverse translation is instead applied to the nominal ray, leaving the geometry unaltered, the ray will intersect the geometry in the nominal state at the same point.

Transform nodes. The simplest way of introducing animation relies on applying affine transformations to the geometry that is to be animated. For instance, palpating an artery produces a deformation that can be represented as a non-uniform scaling. Another example is displacement of the fetus (or a part thereof, for instance an arm or a leg motion) within the gestational sac, which can be achieved by translating/rotating these structures. The benefit of using this approach is that instead of actually applying a transformation to the geometry, which would be computationally very inefficient, an inverse transformation [20] is applied to the rays via *transform nodes* [21], leaving the original geometry unchanged, cf. Fig. 9. Hence, only the coarse high-level acceleration structure needs to be updated [24].

C. On-the-Fly Fusion

Having proposed an efficient ray-tracing approach above for certain types of animated anatomies, the question remains

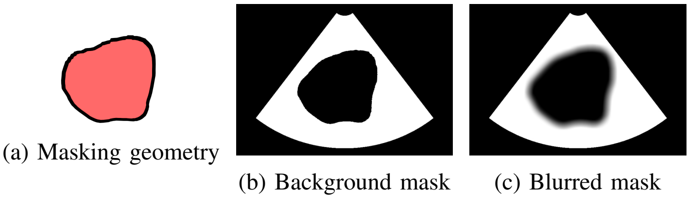

(a) Masking geometry (b) Background mask (c) Blurred mask

Fig. 10: (a) Gestational sac is used as masking geometry, which is also the outer most anatomy simulated during ray-tracing. (b) White pixels within the image plane of a transducer in the scanconverted mask are taken from the acquired image data, whereas pixels within the mask are sampled from the ray-tracing simulation. The border between these two regions is blurred (c) to have a smooth transition between the different contents.

how to fuse ray-traced synthetic images with the data-based anatomical background during the execution of the simulation. For this purpose, we propose to move the image fusion step into the US ray-tracer as described below.

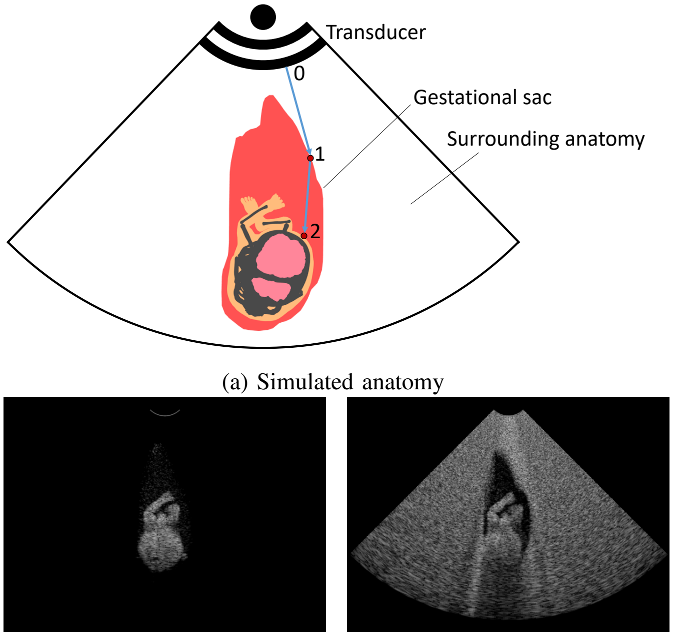
Firstly, if desired, in an offline step [14] the original image volume may be deformed to allow for inserting synthetic anatomy of a different size. Subsequently, the region of the prerecorded background data that is redundant and to be replaced by the simulated content is marked by a mask as shown in Fig. 10. We then use this mask during the on-the-fly fusion step where the values within the foreground are taken from the simulated ray-tracing output, whereas the remaining values are extracted from the US volume. This is akin to removing this masked content from the volume (as in [14]), however the fusion is achieved during simulation runtime, therefore obviating the need for an actual image-content deletion.

The masking geometry is processed by a specific ray-tracing kernel, which simply casts rays along scanlines, ignoring any reflection/refraction effects, hence simply transmitting secondary rays in the same traversal direction at every intersection point. For every ray-surface intersection, ray segments are then examined to determine whether they are located outside or inside the masking geometry. Furthermore, to save computations the same ray-tracing kernel is also used to look-up for the voxel values of the strode US image data. Produced ray segments are discretized identically to the ray-marching step for scatterer convolution in the ultrasound ray-tracing kernel, i.e. based on the temporal sampling rate of the simulated machine and the speed-of-sound of enclosing anatomy. Then, for every generated sample both registered mask values and US volume values are placed in prescan buffers.

Next, the ultrasound ray-tracing kernel is executed in such a way that no volume processing is performed for ray segments outside of the outermost anatomical structure of the region of interest, as shown in Fig. 11. The output of the ultrasound simulation is similarly written in a prescan buffer. Subsequently, the scanconversion step produces 3 different B-mode sized images, namely the sampled US volume, the ray-traced simulation and the mask, which are used jointly to merge the background anatomy with the simulated output via the mask. Without any blending, there would be a sharp transition between the sampled image values outside the masking geometry, and the simulated values inside. Similarly to [14], we blur the scanconverted mask with a box filter as shown in Fig. 10c, such that the transitional values at the border of the masking geometry range between 0 and 1. We can then produce a weighted combination of the background and the foreground with weights provided by the blurred mask, e.g. for linear blending:

$$I(\mathbf{x}) = M(\mathbf{x})I_b(\mathbf{x}) + (1 - M(\mathbf{x}))I_f(\mathbf{x})$$

where M is the blurred mask, $\mathbf{x}$ an image location, and I_b and I_f are the scanconverted sampled volume (background) and the simulated output (foreground) respectively.



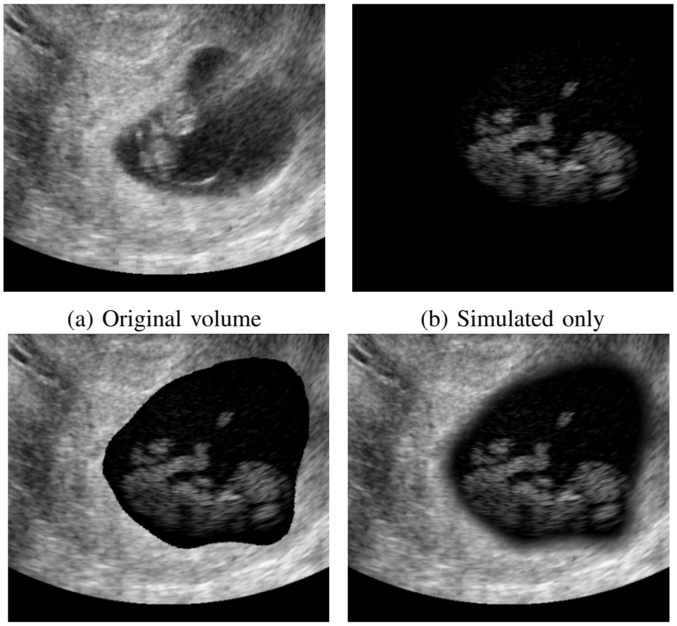
(a) Simulated anatomy

(b) No surroundings (c) With surroundings

Fig. 11: Processing of ray segment 0-1 (a) involves minimal amount of computations, such as updating the current depth value (number of axial samples) and the absorbed energy, according to how far the ray travelled before intersecting anatomy. Conversely, a complete US simulation is performed for ray segment 1-2, including ray-marching and sampling of the scatterer texture. The resulting B-mode image (b) lacks the texture surrounding the simulated anatomy, as opposed to performing a complete US simulation for both the foreground and the background (c). Notice that in the latter case ultrasound waves also interact with the surrounding anatomy, which may result in an altered appearance of the content within the gestational sac.

III. Results

The core code functionality is written in C++, whereas the NVIDIA OptiX 5.0.1 ray-tracing framework is used to produce prescan outputs via executing a number of ray-tracing kernels, namely for ultrasound wavefront propagation, volume



(a) Original volume

(b) Simulated only

(c) No blending

(d) With blending

Fig. 12: Progressive sequence of steps (a, b, d) demonstrating the on-the-fly fusion between an US scan extracted from a preacquired volume and a simulated US image. Notice that simple composition of the two images (a) and (b) exhibits sharp, unrealistic borders (c).

sampling, and mask creation. We used NVIDIA CUDA v8.0 to implement the ultrasound postprocessing pipeline and the on-the-fly fusion. All simulations in this paper, including the baseline ray-tracing comparisons, were run on a Windows PC with Intel® Core™ i7-6800K @ 3.4 GHz CPU, 32 GB RAM and NVIDIA GTX 970 GPU with 4 GB VRAM.

A. On-the-Fly Fusion

We use one of the original volumes presented in [14] and extract a 3D region which serves as the background. It is then fused with the simulated anatomy of an 11 week old fetus [25], along with its gestational sac. Example results for the entire sequence of steps are presented in Fig. 12. The baseline ray-tracing is performed using the adaptive sampling scheme [13], which proved to be the most computationally efficient ray-tracing strategy. We herein used a linear blur box kernel of 32 pixels in size, determined empirically.

The results presented in Tab. I show that the on-the-fly fusion method is more efficient than performing an US simulation on the entire scene, despite having to run two extra ray-tracing kernels, as well as additional image formation operations, namely two more scanconversions for the sampled volume and the mask followed by blurring and fusion stages. Note that a large part of computational resources is taken up by volume processing operations, such as convolutions in the sampling of high- and low-frequency scatterer textures, accounting for ray energy due to attenuation and other operations performed at each sample during marching along ray segments. During the on-the-fly fusion, a much-simplified US ray-tracing kernel is needed with no ray-marching required for ray segments outside of the simulated anatomy, i.e. in areas where the background image is simply sampled via interpolation.

Imaging Parameters	
Image depth (mm)	100
Transducer freq. (MHz)	7
Transducer FOV (°)	90
Scanlines	192
Axial samples	2048
Elev. layers	7
Axial res. (mm)	0.05

Method	Simulated	Fusion
Triangles (thousand)	926	929
Performance (ms)	95.3	85.4
Performance (FPS)	10.5	11.6

TABLE I: Ray-tracer input and performance comparison for producing a final simulated image output, between simulating the entire scene by the ultrasound ray-tracing pipeline vs. our proposed on-the-fly fusion. Fusion is seen to save runtime by avoiding unnecessary computations outside the anatomical region of interest.

B. Evaluation

Note that our simulated images may not always be consistent with pre-acquired in-vivo (background) images with which they are fused, therefore potentially hindering a realistic simulation outcome. In order to study the hypothesis that background image fusion boosts realism, we have conducted a user study in the form of a questionnaire for an expert sonographer and practicing gynecologist.

For 3 different *scenarios* from [14], two US exams of normal pregnancy (Fig. 12 and 13) and a case of ectopic pregnancy (Fig. 14), 3 *slices* with different transducer positions were simulated, using 3 background *methods*: (i) a pre-acquired in-vivo volume (i.e. the proposed on-the-fly fusion), (ii) a homogeneous background with Gaussian scatterers as in [13], and (iii) no background. In order to make results robust to any potential suboptimal simulation parametrization, we devised a study setting by providing the assessor many images with different parameters for each of the (3 scenarios × 3 slices × 3 methods=) 27 cases. The assessor then selected the best image for each scene, to follow on to compare them for the three different methods given above. For each simulation scenario (cf. Figs. 12-14), we first optimized visually the 11 major simulation parameters in a graphical interface for known imaging parameters of an ideal resemblance to the pre-acquired in-vivo background image. Later each simulation parameter was perturbed around its subjective optimal value that produced noticeable differences in the visual appearance of the simulated US images, cf. Fig. 13. This resulted in (1+2×11=) 23 simulated US images, which are presented to the assessor on a single page as randomly shuffled. The assessor was not explicitly told the differences between the US images and was asked to first select out of the 23 parametrizations the image that in his opinion ”is the most realistic for educational purposes in gynecology training”. Next, the assessor rated in

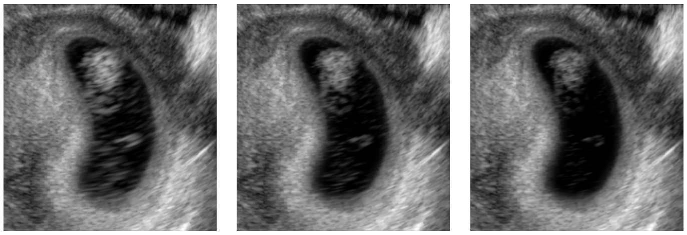
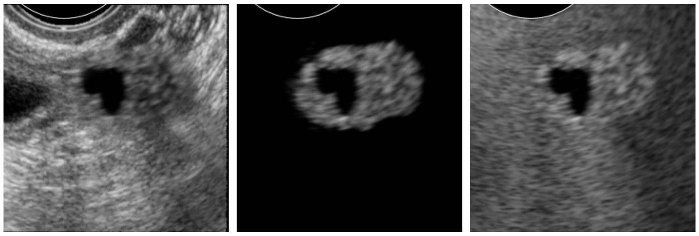


Fig. 13: US images produced using different simulation parameters (imaging frequency of 5, 7, and 9 MHz from left to right) fused on the same in-vivo background image.



(a) Preacquired data (b) No background (c) Scatterer texture

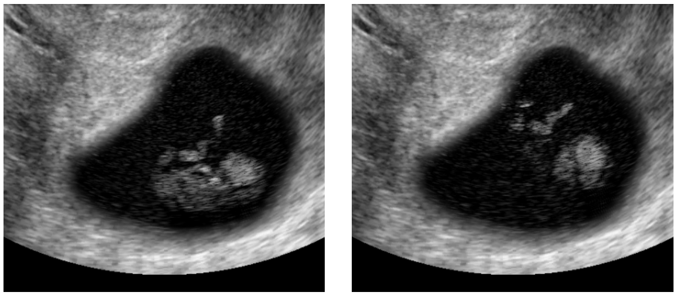
Fig. 14: US images with different backgrounds showings results of the on-the-fly method (a) and the original ray-tracing framework (b, c). Images are obtained with identical foreground anatomy.

a 5-point Likert scale the agreement (i.e. ”strongly agree”, ”agree”, ”neither agree nor disagree”, ”disagree”, ”strongly disagree”) to the statement that ”this above chosen image is sufficiently realistic for educational purposes in gynecology training”. Finally, for the same evaluation criterion the assessor ranked the images from three compared methods above (i-iii).

The results of the evaluation revealed that the clinician consistently gave a rank of 1 to US images generated via the on-the-fly fusion approach, i.e. with preacquired background anatomy; while at the same time expressing strong confidence that the selected images are sufficiently realistic for educational purposes in gynecology training. Note that the incongruity of the speckle patterns of the fused regions after only a single round of parameter adjustment is virtually unavoidable. Regardless, US scans obtained by ray-tracing the entirety of the simulation scene in the scatterer space only received consistently a rank of 2, despite that they demonstrate seamless blending between the foreground and the background.

C. Animated Sequences

As mentioned in Sec. II-B, certain types of affine transformations, such as movement of a fetus as in Fig. 15, can be represented by transform nodes. For instance, using a rigid transformation for the node of the fetus, it can be ray-traced in different positions and orientations, hence providing an impression of motion with minimal impact on the ray-tracing runtime, as demonstrated in Tab. II. Transformations can be either produced algorithmically or read from a file that contains trajectories and orientations. Regulating the frequency of the updates supplied to the transform nodes or providing animation key frames at certain time intervals allows for fine control over the produced sequences, with minimal GPU storage space for caching prebuilt acceleration structures.



(a) Nominal (b) Transformed

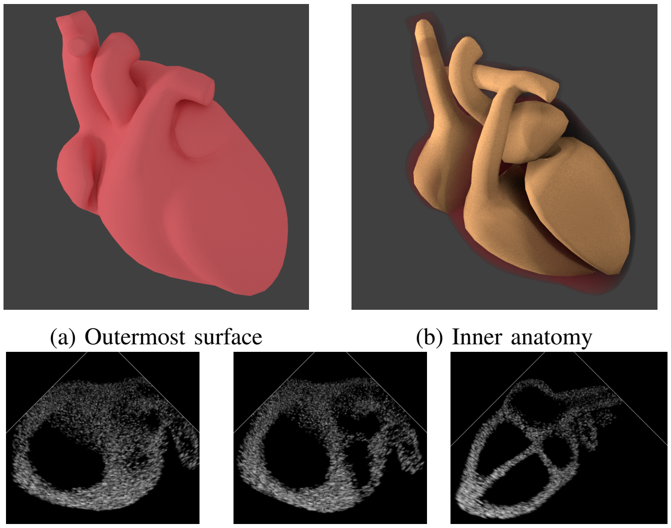
Fig. 15: Fetus in the nominal state (a) is translated and rotated such that in a different frame it is ray-traced in the transformed state (b).

Method	(a) Fusion (static)	(b) Fusion (animated)
Triangles (thousand)	934	934
Performance (ms)	85.6	87.7
Performance (FPS)	11.6	11.4

TABLE II: Performance comparison for producing a final simulated image between a static scene (a) and animated with a transform node (b), to demonstrate the minimal runtime impact of affine transforms for simple animations. In both cases the on-the-fly fusion is used for ultrasound simulation, and all imaging parameters are kept the same as in Table I.

Alternatively, using selector nodes, arbitrarily complex animations can be produced, e.g., the complex beating of the heart as demonstrated in Fig. 16. The animated cardiac sequence used in this example involves 8 keyframes with corresponding anatomically deformed versions of a triangulated model. Note that since the simulation runtime depends on the number and the complexity of surfaces traversed during ray-tracing, for a fair comparison the simulation performance for the static geometry was measured separately for every keyframe across the entire sequence, and then averaged to present in the evaluation result. The measurements presented in Tab. III demonstrate that introducing the animated cardiac sequence has only minor impact on the runtime performance. This is due to the fact that the range of bounding boxes spanned by the constituent key frame models for the heart animation is relatively minor. Notice that simulating animated anatomy with selectors turned out to be slightly less computationally demanding. This is because animation runs through every keyframe model, some of them requiring less processing time than others; whereas static geometry is represented by one of the keyframes during the entire simulation runtime, which may happen to produce the largest number of ray-surface intersections.

Animated sequences from the fetal and heart models can be seen with interactive probe motion in the supplementary video clip.



(a) Outermost surface (b) Inner anatomy

(c) Keyframe i (d) Keyframe j (e) 4-chamber view

Fig. 16: Outermost anatomical surface (a) contains two surface models (b), which represent 4 chambers of the heart. B-mode images (c) and (d) correspond to different keyframes of a cardiac sequence. (e) A different view of the heart anatomy illustrating the four heart chambers.

Method	(a) Simulated (avg. static)	(b) Simulated (animated)
Triangles (thousand)	7.3	7.3
Performance (ms)	65.0 ± 0.63	64.1
Performance (FPS)	15.4 ± 0.15	15.6

TABLE III: Performance comparison for producing a final image output, between a static ray-traced heart and animated with a selector node, to demonstrate the minimal impact of cached acceleration structure switches on the simulation runtime. Baseline ultrasound simulation uses the original ray-tracer, with our on-the-fly fusion stage disabled. All imaging parameters are the same as in Table I.

IV. Discussion

Using the method we present herein, versatile and plausible ultrasound images can be simulated, by virtue of combining real-life image data as background with a model-based ultrasound simulation pipeline. This also enables animated models, which can be efficiently ray-traced using the methods demonstrated in this work. In contrast to [14], this method produces a merged output on the fly. This increased degree of interactivity, however, comes with some limitations, some of which were addressed by the offline hybrid method [14].

To prevent visual mismatch between simulated and background speckle patterns, the imaging field-of-view must be simulated from a similar direction as the original (background) image such that the speckle patterns are directed and smear in the same way. For this, one can place the simulated probe origin near the acquired image location. For a visual similarity, the simulation parameters must also be tuned carefully, which we performed empirically by manually setting the parameters in a GUI with real-time visual feedback. Our evaluation study results in Sec 3.2 indicate that our parametrization herein was relatively successful, without any disturbing artifacts. Nevertheless, such manual tissue and imaging parametrization is still quite tedious, and in the future it may be possible to use scatterer estimation techniques [26, 27, 28] to approximate background textures, for a natural speckle blending with consistent imaging settings entirely in the scatterer space. Consequently, on-the-fly fusion herein requires careful adjustment of the speckle pattern within the simulated content, whereas the method of Tanner et al. [14] is capable of filling [29] the newly created anatomy with a texture that is consistent with the background. This merged output is further perfected by refining mask borders via a complex seamless image stitching procedure [30, 31] using local image intensity and gradients in a graphcut technique, while the on-the-fly fusion herein can only handle simple blending strategies along mask borders. The aforementioned methods used in [14] would be infeasible in an interactive implementation, and are not straightforward or possible in a directional ray-tracing scheme.

View-dependent characteristics are taken into account for model-based ray-traced regions. However, similarly to [14], the background is not affected by ultrasound physics, as it is simply extracted from an existing ultrasound image volume. For instance, the full volume simulation shown in Fig. 11 produces acoustic shadows below the fetus, which is not possible when using image based interpolation, given that volumes only store the recorded intensities at each voxel. These intensities could be artificially modified based on traced ray energies in the fusion step, but achieving an artifact-free ad-hoc method would be quite challenging using such heuristics.

We examined different approaches to animation, in particular in the context of NVIDIA OptiX. Selector nodes have been demonstrated to present good performance for types of animations, where the union bounding box of the underlying surface models is strongly similar to each of their individual bounding boxes, such as in the case of cardiac sequences. Although this seems advantageous for several practical animation scenarios one can imagine, it is also conceivable to craft counter-example scenarios for which animated sequences consist of largely dissimilar individual keyframes, as in the case of erratic limb movements: given such motion, selector nodes may result in unnecessary computation time spent on processing false intersections, whereas rebuilding acceleration structures may, however costly they are, speed up ray-surface intersection routines to a greater extent. Therefore, the methods herein are not a one-size-fits-all solution, but should be considered given the type of anatomical motion to be simulated and the complexity of the triangulated surfaces. Conversely, the motion and the anatomy to be simulated could be approximated or simplified to enable the use of the presented methods.

Given static scatterer textures, the transformation of dynamic anatomy will visually manifest in its moving parts simply sliding through the speckle pattern, as seen in the example of the beating heart. A possible solution could revolve around introducing a finite element method (FEM) [7] mesh, representing current (possibly deformed) state of anatomy, such that the FEM mesh can account for differences between keyframe models and refer back to correct scatterer amplitude values in the nominal state using shape element functions; to be investigated in future work.

Animations could alternatively be achieved by generating multiple simulated volumes and prefusing them into a single 4D volume, such that it can be subsequently sampled with respect to spatial and temporal coordinates. It is a computationally lightweight method, however as a result of its data-based nature, image anisotropy would be lost. Furthermore, additional 3D texture memory, even in the smaller part of animated regions, e.g. the heart, would be prohibitive for the GPU storage capacity.

Motion blur introduced in OptiX 5.0 can be a potential technique in the future, to provide interpolated meshes between keyframes if no changes in topology take place. It should be noted that methods shown herein are not necessarily specific to OptiX, and can be tailored for the use in any reasonably well-developed ray-tracer that incorporates conventional computer graphics concepts, such as acceleration structures and instancing.

V. Conclusion

We have presented an on-the-fly fusion method, capable of merging data-based and model-based simulation outputs during simulation runtime. This enables all the advantages of the model-based ray-tracing approach, allowing to keep interactive view-dependent ultrasound imaging characteristics while introducing animated models. Efficient animation techniques by simple transformations as well as using scene-graph acceleration structure node switching have been described in this work and exemplified using fetal and heart motion models. The methods that we have demonstrated enable interactive simulation of ultrasound images with dynamic scenes, as a major step towards realistic image simulation for medical training. The realism of the resulting US images was confirmed qualitatively in a study by a gynecologist.

Acknowledgments

Funding provided by Innosuisse MoCaFrame project and the Swiss National Science Foundation grant #179116.

References

[1] T. Blum, A. Rieger, N. Navab, H. Friess, and M. Martignoni, “A review of computer-based simulators for ultrasound training,” *Simul Healthc*, vol. 8, pp. 98–108, 2013.

[2] H. Ehricke, “SONOSim3D: a multimedia system for sonography simulation and education with an extensible case database,” *Eur J Ultrasound*, vol. 7, no. 3, pp. 225–300, 1998.

[3] W. Arkhurst, A. Pommert, E. Richter, H. Frederking, S.-I. Kim, R. Schubert, and K. H. Höhne, “A virtual reality training system for pediatric sonography,” *Int Congr Ser*, vol. 1230, pp. 483–487, 2001.

[4] H. Maul, A. Scharf, P. Baier, M. Wüstemann, H. Günter, G. Gebauer, and C. Sohn, “Ultrasound simulators: experience with the SonoTrainer and comparative review of other training systems,” *Ultrasound Obstet Gynecol*, vol. 24, pp. 581–585, October 2004.

[5] A. M. Tahmasebi, P. Abolmaesumi, and K. Hashtrudi-Zaad, “A haptic-based ultrasound training/examination system (HUTES),” in *Proc IEEE Int Conf Robot Autom (ICRA)*, pp. 3130–3131, 2007.

[6] S. Sclaverano, G. Chevreau, L. Vadcard, P. Mozer, and J. Troccaz, “BiopSym: a simulator for enhanced learning of ultrasound-guided prostate biopsy,” *Stud Health Technol Inform*, vol. 142, pp. 301–306, 2009.

[7] O. Goksel and S. E. Salcudean, “B-mode ultrasound image simulation in deformable 3-D medium,” *IEEE Trans Med Imag*, vol. 28, no. 11, pp. 1657–1669, 2009.

[8] T. Reichl, J. Passenger, O. Acosta, and O. Salvado, “Ultrasound goes GPU: real-time simulation using CUDA,” in *Proc SPIE Med Imag*, p. 726116, 2009.

[9] H. Gao, H. F. Choi, P. Claus, S. Boonen, S. Jaecques, G. H. Van Lenthe, G. Van der Perre, W. Lauriks, and J. D’Hooge, “A fast convolution-based methodology to simulate 2-D/3-D cardiac ultrasound images,” *IEEE Trans Ultrason Ferroelectr Freq Control*, vol. 56, no. 2, pp. 404–409, 2009.

[10] M. Salehi, S.-A. Ahmadi, R. Prevost, N. Navab, and W. Wein, “Patient-specific 3D ultrasound simulation based on convolutional ray-tracing and appearance optimization,” in *Proc MICCAI*, pp. 510–518, 2015.

[11] B. Bürger, S. Bettinghausen, M. Radle, and J. Hesser, “Real-time GPU-based ultrasound simulation using deformable mesh models,” *IEEE Trans Med Imag*, vol. 32, no. 3, pp. 609–618, 2013.

[12] O. Mattausch and O. Goksel, “Monte-Carlo ray tracing for realistic ultrasound training simulation,” in *Proc Eurographics Workshop Vis Comput Biomed (EG VCBM)*, pp. 173–181, 2016.

[13] O. Mattausch, M. Makhinya, and O. Goksel, “Realistic ultrasound simulation of complex surface models using interactive Monte-Carlo path tracing,” *Comput Graph Forum*, vol. 37, no. 1, pp. 202–213, 2018.

[14] C. Tanner, R. Starkov, M. Bajka, and O. Goksel, “Framework for fusion of data- and model-based approaches for ultrasound simulation,” in *Proc MICCAI*, pp. 332–339, 2018.

[15] J. T. Kajiya, “The rendering equation,” *ACM SIGGRAPH Comput. Graph*, vol. 20, no. 4, pp. 143–150, 1986.

[16] E. Veach, *Robust Monte Carlo Methods for Light Transport Simulation*. PhD thesis, Stanford University, Stanford, CA, USA, 1998.

[17] J. C. Bamber and R. J. Dickinson, “Ultrasonic B-scanning: a computer simulation,” *Phys Med Biol*, vol. 25, no. 3, p. 463, 1980.

[18] J. H. Clark, “Hierarchical geometric models for visible surface algorithms,” *Commun. ACM*, vol. 19, no. 10, pp. 547–554, 1976.

[19] I. Wald, S. Boulos, and P. Shirley, “Ray tracing deformable scenes using dynamic bounding volume hierarchies,” *ACM Trans. Graph.*, vol. 26, no. 1, 2007.

[20] S. M. Rubin and T. Whitted, “A 3-dimensional representation for fast rendering of complex scenes,” *ACM*

SIGGRAPH Comput. Graph., vol. 14, no. 3, pp. 110–116, 1980.

[21] S. G. Parker, J. Bigler, A. Dietrich, H. Friedrich, J. Hoberock, D. Luebke, D. McAllister, M. McGuire, K. Morley, A. Robison, and M. Stich, "OptiX: A general purpose ray tracing engine," *ACM Trans. Graph.*, vol. 29, no. 4, pp. 66:1–66:13, 2010.

[22] T. Karras and T. Aila, "Fast parallel construction of high-quality bounding volume hierarchies," in *Proc High-Perform Graph (HPG)*, pp. 89–99, 2013.

[23] M. Stich, H. Friedrich, and A. Dietrich, "Spatial splits in bounding volume hierarchies," in *Proc High-Perform Graph (HPG)*, pp. 7–13, 2009.

[24] J. Lext and T. Akenine-Möller, "Towards rapid reconstruction for animated ray tracing," in *Proc Eurograph Short Present*, pp. 311–318, 2001.

[25] P. Loughna, L. Chitty, T. Evans, and T. Chudleigh, "Fetal size and dating: charts recommended for clinical obstetric practice," *Ultrasound*, vol. 17(3), pp. 160–166, 2009.

[26] O. Mattausch and O. Goksel, "Image-based reconstruction of tissue scatterers using beam steering for ultrasound simulation," *IEEE Trans Med Imag*, vol. 37, no. 3, pp. 767–780, 2018.

[27] O. Mattausch and O. Goksel, "Scatterer reconstruction and parametrization of homogeneous tissue for ultrasound image simulation," in *Proc IEEE Eng Med Biol Conf (EMBC)*, pp. 6350–6353, 2015.

[28] A. Al Bahou, C. Tanner, and O. Goksel, "ScatGAN for reconstruction of ultrasound scatterers using generative adversarial networks," in *Proc IEEE Int Symp Biomed Imag (ISBI)*, 2019.

[29] A. Efros and T. Leung, "Texture synthesis by non-parametric sampling," in *Proc IEEE Int Conf Comput Vis (ICCV)*, pp. 1033–1038, 1999.

[30] V. Kwatra, A. Schödl, I. Essa, G. Turk, and A. Bobick, "Graphcut textures: image and video synthesis using graph cuts," *ACM Trans Graph*, vol. 22, no. 3, pp. 277–286, 2003.

[31] B. Flach, M. Makhinya, and O. Goksel, "PURE: Panoramic ultrasound reconstruction by seamless stitching of volumes," in *Proc MICCAI Workshop Simul Synth Med Imag (SASHIMI)*, pp. 75–84, 2016.