Direct Visuo-Haptic 4D Volume Rendering using Respiratory Motion Models

Dirk Fortmeier, *Student Member, IEEE,* Matthias Wilms, Andre Mastmeyer, and Heinz Handels, *Member, IEEE*

Abstract—This article presents methods for direct visuo-haptic 4D volume rendering of virtual patient models under respiratory motion. Breathing models are computed based on patient-specific 4D CT image data sequences. Virtual patient models are visualized in real-time by ray casting based rendering of a reference CT image warped by a time-variant displacement field, which is computed using the motion models at run-time. Furthermore, haptic interaction with the animated virtual patient models is provided by using the displacements computed at high rendering rates to translate the position of the haptic device into the space of the reference CT image. This concept is applied to virtual palpation and the haptic simulation of insertion of a virtual bendable needle. To this aim, different motion models that are applicable in real-time are presented and the methods are integrated into a needle puncture training simulation framework, which can be used for simulated biopsy or vessel puncture in the liver. To confirm real-time applicability, a performance analysis of the resulting framework is given. It is shown that the presented methods achieve mean update rates around 2000 Hz for haptic simulation and interactive frame rates for volume rendering and thus are well suited for visuo-haptic rendering of virtual patients under respiratory motion.

Index Terms—Visuo-haptics, Volume Rendering, Breathing motion, Needle insertion simulation

1 Introduction

Visuo-haptic simulation of needle insertion is an active field of research. Its aim is to provide a virtual environment in which a surgeon can train an intervention without harming real patients or relying on costly and non-reusable tissue phantoms. Scenarios that are often simulated are insertions of needles into the liver [1], [2], [3] and the contained vessels, into blood vessels for anesthetization or catheter placement [4], [5] or prostate brachytherapy [6], [7].

To provide such a simulation, different components are needed. The core problem is to provide a realistic model of a patient and display this model in a convincing and immersive virtual environment. For this, methods for visual rendering as well as methods for the interaction with the virtual patient model are needed. The latter generally is carried out using a haptic input device that enables the user to steer virtual tools and receive realistic force-feedback simulated by a haptic algorithm.

The behavior of needles inserted in the abdominal region can be highly influenced by respiration-induced organ motion. For instance, breathing influences and displaces especially the shape of parts of the liver that are close to the diaphragm. The displacement of the diaphragm close to the liver can reach up to 5 cm between full inspiration and full expiration [8]. Other organs as for instance the gall bladder or intestines are affected in the same way. For some needle intervention techniques, flexible needles are used that bend under the influence of the deformed surrounding tissue. In contrast, stiff needles influence the deformation of surrounding tissues and the breathing causes tilting of the needle outside of the patient's body. Both effects have to be included in a realistic simulation of needle insertion.

Real-time simulation of breathing motion for the lung has been performed by [9] for visualization purposes in an augmented reality framework. A model for the motion of abdominal organs is presented in [10], which is fast to compute for organ surfaces and was suggested to support radiotherapy by providing a real-time capable prediction of tumor motion based on depth imaging of the patients skin. In [1], [11], visuo-haptic simulation of respiratory motion of a virtual patient model was presented for liver biopsy and methods for parameter estimation for this simulation are given in [12], [13]. Another visuo-haptic simulation approach is proposed in [2], where the motion of the liver was integrated in an ultrasound simulation. Both approaches are limited to a simplified model of respiratory motion based on a sinusoidal displacement and do not incorporate breathing effects in the modeling of haptic forces. Similarly, [14] presents a simplified breathing model based on a sinusoidal geometric transformations in a virtual reality application for the visual simulation of angiography without haptic interaction.

Visualization of medical imaging data in visuo-haptic environments is typically performed by indirect volume rendering, i.e. classical rasterization of polygonal surface models of isosurfaces extracted from 3D image data. Another option is direct volume rendering, for which a thorough overview of methods and implementation is given in [15]. This method is especially suited for static volume data as for example used in [2]. To introduce deformations of the rendered volume images, methods for direct rendering using unstructured meshes have been given by [16], [17] and more recently by [18] and [19]. As for computation of soft tissue behavior, mesh based approaches come with different advantages as reduced computational load and drawbacks as a complicated mesh generation procedure.

- *D. Fortmeier is with the Institute of Medical Informatics and the Graduate School for Computing in Medicine and Life Sciences, University of Luebeck, Luebeck, Germany.*
  E-mail: dirk.fortmeier.DE@IEEE.org
- *M. Wilms, A. Mastmeyer, and H. Handels are with the Institute of Medical Informatics, University of Luebeck, Luebeck, Germany. This work is supported by the German Research Foundation (DFG, HA 2355/11-2).*

Instead of using a mesh based approach, in this work, we adapt the basic idea to use deformed rays during ray casting [20], and use respiratory motion models to visualize a globally deformed 3D CT image under breathing motion.

A central part of this work is about the insertion of needles into soft tissue. Using the finite element method, resulting deformations of tissue and interacting needle have been presented i.a. in [6], [21], [22], [23] and [24]. Also, haptic simulation of bending needles using the angular spring method was used in [25], [26], which is supposed to be an accurate simplification and is fast to compute. Alternatives to this concept are elastic rod methods [27], [28], as for instance used in [6].

Nowadays, spatio-temporal 4D CT image data acquired during free breathing [29] can be used to analyze the patient-specific motion of internal structures caused by respiration to improve radiation therapy of tumors in the thorax and abdomen. Recently in this context, a lot of effort has been put into the development of non-linear image registration techniques that enable a precise estimation of respiratory motion based on 4D image data. Furthermore, the development of subsequent respiratory motion analysis and modeling approaches [30] have been adressed. In this work, those techniques are employed to enable highly realistic rendering of virtual patient models under respiratory motion.

This work builds on a visuo-haptic framework [3] and methods from image registration and modeling of respiratory motion [31], [32]. It contributes a novel real-time capable method to display a virtual patient model under the influence of breathing motion and integrates the breathing motion into haptic algorithms. It introduces realistic models that include natural variations of the breathing motion. The key idea is to use a time varying displacement field describing the breathing motion to modify the sampling positions during ray casting for direct volume rendering and use the displacement field to translate the haptic device position into a reference space. In comparison to existing methods, our visuo-haptic rendering approach enables direct volume rendering, haptics, realistic models of complex local breathing and models based on actual patient data in a single framework.

The article is divided into three main parts: First, methods for interactive visual and haptic rendering of a breathing patient are introduced. These are based on a model of respiration that is represented by a displacement field function, which is detailed in the second section. In the third part, the methods are demonstrated and evaluated based on an prototypical implementation of a needle insertion intervention.

2 Visuo-haptic Rendering

A 4D CT data set usually consists of a sequence of 3D CT images, which represent the patient's anatomy at different phases of a single breathing cycle. Therefore, a direct approach of using a 4D CT data set for a visuo-haptic simulation would be to periodically step through the sequence of CT images. However, this has major drawbacks: When only rendering a single cycle of images (a single 4D CT data set), no inter-cycle variations of the breathing motion (e.g., breathing depth) can be represented. Additionally, since the number of images is usually very low (usually 7-14 phases are reconstructed) compared to the length of a breathing cycle (~2-7s), it is not possible to provide a frame rate of at least 24 Hz for the visualization as needed by a human observer to get the impression of a fluid animation. The situation is even more delicate for haptic rendering, where an update rate of at least 1000 Hz is required. These requirements motivate new methods that can quickly compute a visual and haptic representation for any given time point in between the time points of the acquired image sequence.

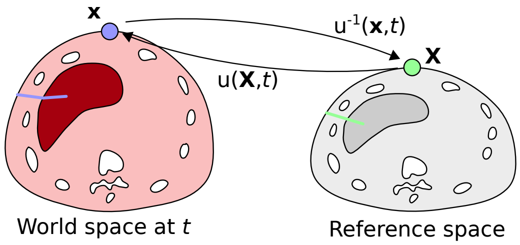


Figure 1: Schematic axial slice of a virtual patient model. For each point in the world space, positions $\mathbf{x}$ can be mapped to $\mathbf{X}$ in the static reference space and vice versa. Here, this is demonstrated for a point on the surface of the breathing patient and points along a needle path.

The utilized simulation framework is based on direct visuo-haptic rendering, i.e. a volumetric voxel image is used for visual and haptic rendering of a virtual patient. Without a motion model, the virtual patient model is static. In the following, the associated space to this state will be called the reference space. Using a motion model and the voxel image associated to the reference space for rendering is a key idea of our methods. In this chapter, a function $u(\mathbf{X}, t) : \Omega \rightarrow \mathbb{R}^3 (\Omega \subset \mathbb{R}^3)$ that yields a displacement for any given point $\mathbf{X} \in \Omega$ in the image domain and for any simulation time point $t \in \mathbb{R}$ is used. This represents a mapping of each position $\mathbf{X}$ in the reference space to the corresponding point $\mathbf{x}$ in space and time of the virtual world space as shown in Fig. 1. In general, this function is required to be bijective, i.e. an inverse function $u^{-1}(\mathbf{x}, t)$ exists. A simple approxiation of this function to represent breathing motion would be a sinusoidal displacement $u(\mathbf{X}, t) = (0, a{\cdot}\sin(t), 0)^\top$, where a is the maximal amplitude in mm. This of course does not suffice to represent the complex local motion of organs under respiratory breathing in detail. An in-depth presentation of other possible motion models follows in section 3.

2.1 Visual Rendering using Motion Fields

Visual rendering of the patient image data deformed by breathing motion is based on a ray casting volume rendering method [15] with non-linear rays, for example used in [20] where ray deflectors are used to locally bend the rendering space. We build on this basic idea and adapt it so each ray that is cast through the image volume can be considered to be warped by a function representing the breathing motion. Here, we perform the warping as follows.

For each sample position $\mathbf{x}$ on a viewing ray, the position $\mathbf{X} = \mathbf{x} + u^{-1}(\mathbf{x}, t)$ in reference space could be computed directly with the inverse displacement $u^{-1}(\mathbf{x}, t)$, which is not available for the motion models used. To solve this problem, we combine ray casting and computation of the inverse displacement on the fly: Starting from the camera position, the sample position $\mathbf{x}$ in the world space and the corresponding position along the warped viewing ray $\mathbf{X}$ are iteratively updated as

$$\begin{aligned} \mathbf{x}^{+} &= \mathbf{x} + \mathbf{d} && (1) \\ \mathbf{X}^{+} &= \mathbf{x}^{+} - u(\mathbf{X} + \mathbf{d}, t) && (2) \end{aligned}$$

while sampling along the viewing ray with ray direction vector $\mathbf{d}$ and sample distance $\|\mathbf{d}\|$. This is inspired by the fixed-point inversion approach from [33] and needs further explanation. In [33], it is shown that for an invertible displacement field, it is possible to compute the inverse by applying an iterative algorithm for each element of the field. In practice, we use Eq. (6) from [33] and substitute the deformation fields by the corresponding terms including the displacements, giving the sequence $\mathbf{X}_n$=$\mathbf{x} - u(\mathbf{X}_{n-1})$. This corresponds to our Eq. 2 for a given and fixed location $\mathbf{x}^{+}$. Instead of applying this algorithm several times for the same point (the authors state that normally a number smaller ten is sufficient), it is evaluated once for each ray casting sampling position and the value from the previous sampling position along the viewing ray is used as an initial solution.

After trilinear interpolation of the volumetric reference image data at $\mathbf{x}^{+}$, application of a transfer function follows for each sampling step. Label data defined in the reference space are used for tagged volume rendering of a deformed volume in the same fashion by using nearest neighbor interpolation. In the same way, ray casting based X-ray simulation and ultrasound imaging simulation is performed. Multiplanar reformations (MPRs) are generated by applying the fixed-point algorithm with several iterations per point to compute the inverse at each sampling position without ray traversal.

In our framework, it is possible to only use direct volume rendering, but nevertheless the methods are applicable to surface based rending as well and thus also interesting for fully surface based simulation frameworks. Using the function $u(\mathbf{X}, t)$, vertices of polygonal surface models can be displaced and used to render surface models of organs.

2.2 Haptic Rendering for Virtual Palpation and Ultrasound Probing

Haptic rendering of virtual palpation and ultrasound probing is similar to our previous approaches [34]: Multiple virtual contact nodes are placed on the surface of the virtual tools and each of the positions $\mathbf{x}$ representing a node is connected to a virtual proxy position $\mathbf{p}$ by a spring to calculate friction forces. Structure repulsion forces are calculated by non-linear functions relating palpation depth to forces. To introduce breathing motion into the haptics computation, the device position is transformed from the world space to the reference space by the motion model.

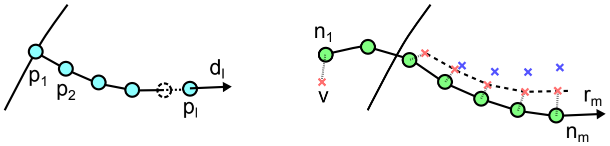


Figure 2: Left: Needle path discretized by nodes $\mathbf{p}_i$, only the tip node is adjusted during insertion. Right: Nodes $\mathbf{n}_j$ representing the discretized needle are connected to proxies (red) by springs. These proxies are displaced by the breathing motion (blue).

To achieve this, again the inversion scheme of [33] is applied iteratively on the haptic device position $\mathbf{x}$, giving the position in reference space as $\mathbf{X}^{+} = \mathbf{x} - u(\mathbf{X}, t)$ in each haptic rendering frame. Haptic force and torque computation then takes place in the reference space. Computing the absolute orientation [35] for the set of proxy positions and the set of proxy positions displaced by $u(\mathbf{x}, t)$ gives a rotational transformation that can be used to translate the resulting force back to the world space. In practice, we found the angle of the rotation to be less than $5°$ and thus it might be neglected.

To visualize local deformations at the palpation site, the methods presented in [3] resp. [36] for the needle are applied. Notice, that the resulting deformations and their visualization do not affect the force output of the haptic algorithms.

2.3 Haptic Rendering of Needle Insertion

For the modeling of needle insertion into soft tissue, several methods exist. Here, we build on the needle insertion algorithm presented in [3] and adapt it to enable curved insertion paths. In this method, the soft tissue behavior is modeled by linear springs, so no tetrahedral mesh is needed as for example in [24].

During needle insertion, the needle algorithm uses two sets of nodes to compute haptic forces based on breathing motion (Fig. 2) and provide a visualization of the bended needle:

- $P = \{\mathbf{p}_i \in \mathbb{R}^3\}_{1 \le i \le l}$: nodes representing the path of the needle during insertion
- $N = \{\mathbf{n}_j \in \mathbb{R}^3\}_{1 \le j \le m}$: nodes representing the complete discretized needle

New path nodes are placed along the insertion path in the reference space. Thus, the number of these nodes varies:

- $|P| = 0$: needle is outside the body,
- $|P| = 1$: needle tip is on the skin surface of the patient,
- $|P| \ge 2$: needle is inside the patient.

The nodes are equidistantly distributed along the insertion path and their position does not change during insertion except for the tip node, which is explained later. In Alg. 1, an overview of the haptic algorithm is given.

At first, the direction of the needle tip $\mathbf{d}_l$ in the reference space is computed (`getTipOrientInRefSpace`) by

$$\mathbf{R} = \begin{bmatrix} u(\mathbf{p}_l + (1,0,0)^\top, t)^\top - u(\mathbf{p}_l, t)^\top \\ u(\mathbf{p}_l + (0,1,0)^\top, t)^\top - u(\mathbf{p}_l, t)^\top \\ u(\mathbf{p}_l + (0,0,1)^\top, t)^\top - u(\mathbf{p}_l, t)^\top \end{bmatrix}^\top + \mathbf{I} \qquad (3)$$

Algorithm 1 In each iteration of the haptic loop, the algorithm computes new node positions P^+ and N^+ as well as the haptic force $\mathbf{f}$ and torque $\mathbf{t}$.

```
input: $\mathbf{x} \leftarrow$ haptic device position
input: $\mathbf{X} \leftarrow$ old device position in reference space
input: $\mathbf{q} \leftarrow$ haptic device orientation
input: $P \leftarrow$ set of path nodes from prev. step
input: $N \leftarrow$ set of needle nodes from prev. step
$\mathbf{d}_l \leftarrow$ getTipOrientInRefSpace($P, N$)
$\mathbf{X}^+ \leftarrow \mathbf{x} + u(\mathbf{X}$,t)
$P^+ \leftarrow$ updatePathNodes($P, \mathbf{X}^+, \mathbf{d}_l$ )
if ( $|P^+| = 0$ ) then
    $N^+ \leftarrow$ simpleNNodePlacement($\mathbf{x}, \mathbf{q}$)
    $\mathbf{f} \leftarrow 0$, $\mathbf{t} \leftarrow 0$
else
    $N^+ \leftarrow$ updateNNodes($P^+, N, \mathbf{x}, \mathbf{q}$)
    $\mathbf{f} \leftarrow$ computeForces($P^+, N^+, \mathbf{x}, \mathbf{X}^+$)
    $\mathbf{t} \leftarrow$ computeTorque($P^+, N^+, \mathbf{x}, \mathbf{X}^+$)
end if
return $\{N^+, P^+, \mathbf{X}^+, \mathbf{f}, \mathbf{t}\}$
```

and then $\mathbf{d}_l = \mathbf{R}^{-1}\mathbf{r}_m / \left\|\mathbf{R}^{-1}\mathbf{r}_m\right\|$ with $\mathbf{r}_m$ being the needle tip node's direction vector. Together with the position of the haptic device tip in the reference space $\mathbf{X}$, this is used to update the set of path nodes (updatePathNodes). In case the needle is outside the patient's body, P remains empty. On first contact with skin, a node is placed at the contact position. In case the needle has entered the patient's body, a chain of nodes is placed as follows: The tip node $\mathbf{p}_l$ moves in direction $\mathbf{d}_l$ in case $\mathbf{X}$ moves forward relatively to its position from the previous iteration of the algorithm or otherwise towards the node behind the tip. If the distance $\Delta l = \|\mathbf{p}_{l-1} - \mathbf{p}_l\|$ becomes smaller than $0.5 \cdot \Delta p$ with Δp being the fixed spacing between path nodes, $\mathbf{p}_{l-1}$ is removed from the set. In case $\Delta l > 1.5 \cdot \Delta p$, a new node is placed on the line between the nodes in a way that its distance to $\mathbf{p}_{l-1}$ is Δp. For the tip node and newly placed nodes, the material properties stiffness k_j, cutting stiffness coefficients $a_{1,j}$ and $a_{2,j}$, cutting force f_j^{cut} and maximal friction force R_j are obtained for the position of the node via transfer functions relating CT image data to material properties or, if available, by using a labeling mask. To simulate a cutting force, forward movement of the path tip node is prohibited as long as the spring force connecting device position and the path tip node in direction of the needle is below the cutting force threshold:

$$a_{2,l}\Delta d^2 + a_{1,l}\Delta d < f_l^{\text{cut}} \tag{4}$$

with $\Delta d = (\mathbf{X} - \mathbf{p}_l) \cdot \mathbf{d}_l$ being the distance between device position and path tip node. For $a_{2,l} \neq 0$, non-linear cutting forces as measured for example in [37] can be produced. Given these considerations, thus the path tip node is updated by

$$\mathbf{p}_l^+ = \mathbf{p}_l + \mathbf{d}_l \cdot \begin{cases} \Delta d & \text{if } \Delta d \leq 0 \\ 0 & \text{else if Eq. 4 holds} \\ \Delta d + \gamma_1 - \sqrt{\gamma_1^2 + \gamma_2} & \text{else if } a_{2,l} > 0 \\ \Delta d - \gamma_2 & \text{else} \end{cases}$$

with $\gamma_1 = \frac{1}{2}\frac{a_{1,l}}{a_{2,l}}$ and $\gamma_2 = \frac{f_l^{\text{cut}}}{a_{2,l}}$.

No. Nodes	Force [N]	Deflec. [mm]	Diff. [mm]
15	0.5	8.6	-3.8
	1.0	17.1	-8.8
25	0.5	13.2	0.7
	1.0	25.8	-0.2
50	0.5	29.2	16.7
	1.0	53.2	27.2

Table 1: Simulated deflection of the needle tip and difference to measurements for a 16 gauge needle. The stiffness parameter has been tuned for 25 nodes.

As long as the needle is not in contact with the skin, no forces have to be computed and no physical needle model has to be employed to determine the needle node positions. Instead, in this case these are placed along the ray defined by haptic device position $\mathbf{x}$ and orientation quaternion $\mathbf{q}$ (simpleNNodePlacement).

Otherwise, the positions of the needle nodes N are computed by a discretized needle model implemented using the Bullet Physics library [38], which uses a Projected Gauss-Seidel solver. Each needle node is represented by a single rigid body and connected to its successor by a ball and socket joint and a rotational spring. The needle is forced to comply to the insertion path in the following way. Each pair of path and needle nodes representing a part of the needle that is inside the tissue is connected by a spring using the material stiffness k_j of the path node. Except for the tip node, the spring force is projected onto the plane perpendicular to the needle tangent at the node.

The needle is discretized by 25 elements, which is a trade-off between computation time and accuracy. Furthermore, the Bullet based physics simulation of the needle is not run only once per haptic simulation frame, but several times with a fixed time step of 1 ms to faster reach a static state. The actual number of iterations depends on the remaining available processing time of the haptic frame.

We calibrated the stiffness for a 16 gauge needle with a length of 150 mm: To this aim, we first measured the deflection caused by a lateral force at the tip of a needle fixed at the base for 0.5 N and 1.0 N resp., which we found to be 12.5 mm resp. 26.0 mm. Then the stiffness parameter for the rotational spring were adjusted to match this measurements in the simulation using 25 nodes. For a different number of elements, the resulting simulated deflection differ, see Tab. 1. In terms of convergence, we found that in the calibration setup, the mean node movement converges to a value below 10^{-5} mm after roughly 1200 iterations.

An additional spring is used to connect the base node position $\mathbf{n}_1$ of the needle with a position $\mathbf{v}$ to induce bending to the needle in case the user angulates the haptic device handle. This position is depending on the insertion point of the needle in world space, which is $\mathbf{s} = \mathbf{p}_1 + u(\mathbf{p}_1)$. Given this surface point, the position that is used for the spring is $\mathbf{v} = \mathbf{s} - \bar{l}\mathbf{q}_z$. Here, $\bar{l}$ is the length of the needle part that is still outside the tissue. The direction vector of the haptic device $\mathbf{q}_z$ is computed from the haptic device orientation quaternion $\mathbf{q}$ by using the quaternion rotation operator applied to the pure quaternion pointing in z direction $\mathbf{q}_z = \mathbf{q} \cdot (0,0,1)^\top \cdot \mathbf{q}^{-1}$.

Now, with a new needle path $\mathbf{P}$ and needle configuration, the forces and torques that will be displayed by the haptic device are computed. The force is the sum of:

1) A resistance force to cutting in needle direction at the tip

$$\mathbf{f}_{\text{cut}} = -(a_{2,l}\Delta d^2 + a_{1,l}\Delta d)\mathbf{d}_l \tag{5}$$

2) Friction force along the needle shaft. For each path node, an offset $\Delta x_i \in \mathbb{R}$ is updated in each iteration:

$$\Delta x_i^+ = \begin{cases} \Delta x_i + \Delta x & \text{if } |\Delta x_i + \Delta x| < \frac{R_i}{k_i} \\ \frac{R_i}{k_i}\operatorname{sgn}(\Delta x_i + \Delta x) \end{cases} \tag{6}$$

with Δx being the relative movement of the haptic device in insertion direction. From these offsets, the total friction force is computed as

$$\mathbf{f}_{\text{fric}} = \sum_{2\le i\le l} k_i \frac{\|\mathbf{p}_i - \mathbf{p}_{i-1}\|}{\Delta p}\Delta x_i \tag{7}$$

For larger numbers of path nodes n and high values of k_i this can act like a very stiff spring. To prevent unstabilities, i.e. overshooting associated to mass-spring systems with large stiffness, the change of the friction force is limited in each haptic frame by a fixed value.

3) A base force that results from the offset of haptic device position and the computed needle base projection onto the plane perpendicular to the insertion vector at the beginning of the needle path:

$$\mathbf{f}_{\text{base}} = \kappa \cdot [(\mathbf{n}_1 - \mathbf{x}) - (\mathbf{n}_1 - \mathbf{x}) \cdot \mathbf{d}_1] \tag{8}$$

with κ being the stiffness of a spring, which couples the virtual needle base and device position.

Torque is computed by using a rotational spring between the haptic device orientation and the tangential direction of the simulated needle at the air and tissue interface. Rotational forces around the needle z-axis are set to zero.

3 4D Motion Models

The motion model function $u(\mathbf{X}, t)$ was described as being a time varying function without further specification. This section will present different approaches to model and implement respiratory motion. We utilize existing methods from image registration and motion modeling of the lung (mainly [31], [32]) and introduce how these can be used in a real-time visuo-haptic liver puncture simulation with varying breathing cycles.

Our visuo-haptic simulation of a breathing virtual patient is based on displacement fields estimated from a 4D CT data set. This data set consists of a sequence of n 3D CT images $I_{j\in\{1,\ldots,n\}} : \Omega \to \mathbb{R}$ and represents a single respiratory cycle from maximum inspiration over expiration back to total inspiration. In the following, the states of this cycle are labeled as end inspiration (EI), mid inspiration (MI), end expiration (EE) and mid expiration (ME).

For each phase j from the sequence, a non-linear transformation $\varphi_j : \Omega \to \Omega$ exists, which describes the respiratory motion between an arbitrary reference phase and phase j. Here, this phase corresponds to the static state of the virtual patient. These transformations can be represented by $\varphi_j = id + u_j$ with displacement fields $u_j : \Omega \to \mathbb{R}^3$ that assign a displacement vector to each voxel. Estimation of respiratory motion based on 4D CT image data by non-linear image registration techniques has been a very active area of research in recent years. However, most publications on this topic deal with the special case of lung motion estimation [39], [40], [41]. In contrast, the realistic visuo-haptic simulation of a breathing patient requires an estimation of breathing-induced motion of all thoracic and abdominal structures displayed to the user. Estimating the motion of multiple structures is a challenging problem, as most registration approaches inherently assume the underlying motion to be smooth. This assumption is obviously wrong at interfaces of structures sliding along each other (e.g., lung/liver and its surrounding tissue [31]). We, therefore, employ the registration approach proposed by Schmidt-Richberg et al. [31], which is able to handle those discontinuities.

As transformations between the reference image and each phase j are computed independently, noise or image artifacts can result in a temporally inconsistent motion estimation. Temporal inconsistencies can lead to visible motion artifacts during simulation and we, therefore, perform a PCA-based filtering of the estimated displacement fields to remove them, which is a common approach in respiratory motion modelling [42].

For the transformation from any phase j to the reference phase, the inverse of u_j is needed, which we prevent by inversion during run-time. Also, it is necessary to provide a transformation between the phases for each point in time. Modeling and a memory conserving implementation of this function $u(\mathbf{X}, t)$ is crucial. Theoretically, using inversion of the displacement fields or diffeomorphic registration, inverse displacement fields are available, but no satisfying inverse motion model could be build based on the inverse fields directly. In the following, different approaches for modeling the function are represented: The first one is a key frame based approach, in which all u_j are considered to be key frames in a sequence; in between two key frames, $u(\mathbf{X}, t)$ is computed by linear interpolation of adjacent u_j and u_{j+1}. The second approach is similar to the first, but also introduces extrapolation of out-of-sample predictions, i.e. states that are not captured in the 4D data. A third approach relies on the modeling of the breathing motion driven by a so-called surrogate signal.

3.1 Full Cycle Key Frame Approach

Similar to computer animation methods, each displacement field u_j in the sequence of images I_j is considered to be a key frame, and in between key frames, interpolation of the displacement fields takes place. All key frames are placed on a normalized motion cycle time line at $\tau_j \in [0,1)$, as shown in Fig. 3. This yields a displacement $u(\mathbf{X}, t) = b(\mathbf{X}, \tau)$ at position $\mathbf{X}$ in reference space and the current normalized time $\tau \in [0,1)$ using adjacent u_i and u_{i+1}, which are interpolated linearly based on adjacent τ_i and τ_{i+1}:

$$b(\mathbf{X}, \tau) = (1 - \alpha(\tau))u_i(\mathbf{X}) + \alpha(\tau)u_{i+1}(\mathbf{X}) \tag{9}$$

with $\alpha(\tau) = \frac{\tau - \tau_i}{\tau_{i+1} - \tau_i}$.

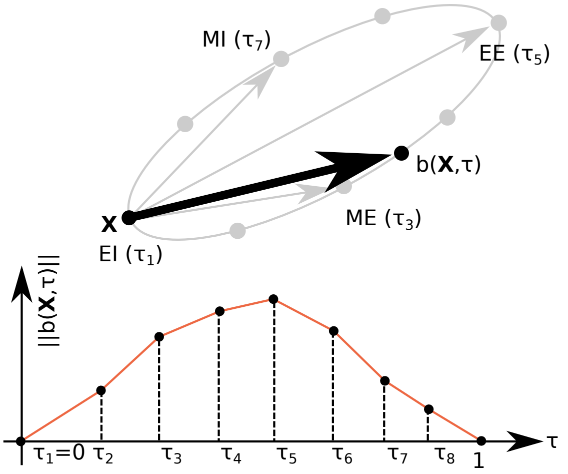
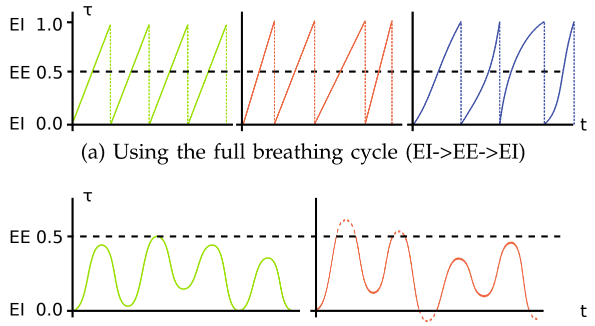


Figure 3: Upper: Plot of displacement for a fixed $\mathbf{x}$ during the cycle. Lower: Magnitude of the displacement $\|b(\mathbf{X},\tau)\|$ for normalized cycle time τ. Here, $n = 8$ key frames placed at $\tau_{i=1..8}$ are used. Between two displacements given by adjacent key frames, interpolation takes place.



(a) Using the full breathing cycle (EI->EE->EI)

(b) Using only one half cycle (EI->EE)

Figure 4: Functions for cycling through the image sequence (EI = end of inspiration, EE = end of expiration).

During the simulation, the value of τ is changed to run through the sequence. As depicted in Fig. 4a, several functions $\tau = f(t) : \mathbb{R} \to [0,1)$ could be applied: (1) The obvious choice is to cycle through the sequence by a sawtooth shaped function with fixed period. (2) This can be augmented by introducing randomness to change the length of each period, but keeping $f'(t) = \alpha$ constant. (3) Furthermore, randomness can be introduced by varying $f'(t)$ during the cycle. This randomness introduces variations of the breathing to represent the natural variation of breathing.

The used number of key frames should reflect the variations and non-linearities in the breathing motion of the patient. At least, four key frames that correspond EI, ME, EE and MI are needed for a full cycle including hysteresis. The maximum number is limited by the available memory of the graphics hardware and the resolution of the displacement fields. We found that with high resolution displacement fields, the resulting amount of memory easily exceeds the available space when using the full sequence of available phases.

3.2 Half Cycle Key Frame Approach

The major disadvantage of the full cycle model is that only a single breathing cycle can be looped. The cycle length and the temporal distance between adjacent key frames can be varied, but maximum inspiration and expiration is fixed to the available maximum and minimum state captured in the 4D CT image sequence, limiting the approach to in-sample predictions. Since maximum inspiration and expiration varies naturally with each cycle, it is desirable to model this aspect.

To also include out-of-sample predictions, only half of the phases (e.g. only EI->ME->EE) can be used, which we call a half cycle approach. Oscillating between maximum expiration and maximum inspiration from the full cycle is used and beyond EI or EE, extrapolation takes place. Extrapolation is not possible with the full cycle approach, since all the key frames form a closed loop. The upper part of Fig. 4b visualizes a function that oscillates τ to produce an oscillation between EI and EE. The lower part represents a function that includes values of τ that do not fit into the range of maximal and minimal inspiration. Using extrapolation, the displacement fields for states beyond maximal and minimal inspiration can be approximated. We do not use adjacent key frames for the extrapolation using Eq. 9, but the key frames associated to maximum and minimum expiration. This gives the following extrapolation expressions for out-of-sample (OOS) predictions:

$$b_{\mathrm{oos}}(\mathbf{X},\tau) = (1-\alpha_{\mathrm{oos}}(\tau))u_{\mathrm{EI}}(\mathbf{X}) + \alpha_{\mathrm{oos}}(\tau)u_{\mathrm{EE}}(\mathbf{X}) \qquad (10)$$

with $\alpha_{\mathrm{oos}}(\tau) = \frac{\tau-\tau_{\mathrm{EI}}}{\tau_{\mathrm{EE}}-\tau_{\mathrm{EI}}}$.

As drawback of this method, hysteresis, i.e. different paths between EE->EI and EI->EE, is not included in this model.

3.3 Surrogate Signal-based Motion Model

The conceptual limitation of the previous two approaches is that either states beyond minimal and maximal inspiration from the sequence (full cycle approach) or hysteresis cannot be represented (half cycle approach). To overcome these limitation, real surrogate signals are used. These represent patient-specific variations of the breathing, which we consider a valuable contribution to realistic visuo-haptic simulation of respiration. Basically, a surrogate signal can be a 1D signal of the patient's breathing measured by a surrogate, for example a spirometry device or an abdominal belt. It is assumed that a linear correspondence exists between the measured surrogate signal and the actual motion of internal organs of the patient. For 4D CT imaging, surrogate signals are recorded for reconstruction and do not have to be acquired in an extra step.

Here, our surrogate signal-based motion model is used, which has been evaluated for diffeomorphic motion modeling of the lung [32]. Generally, a surrogate signal can be a signal of higher dimension; the aim is to increase accuracy of the resulting model. For instance, a depth image acquired by a time-of-flight camera could be used. We use a dimensionality of $n_{\mathrm{sur}} = 2$ by combining a spirometry signal and its time derivative and denote it by $\hat{\mathbf{z}}(t) = (g(t), g'(t))^T : \mathbb{R} \to \mathbb{R}^{n_{\mathrm{sur}}}$. This is done to combine the measured value of the spirometry signal with information indicating expiration resp. inspiration and thus enabling modeling of hysteresis.

Using surrogate signals $\mathbf{z}_j$ associated to the sequence of non-linear transformations φ_j, a linear motion model can be learned by multivariate regression under the assumption of a linear correspondence between surrogate signal and respiratory motion. First, the transformations φ_j are reformulated as column vectors $\mathbf{b}_j \in \mathbb{R}^{3m}$ with m being the number of image voxels of I_j. The model then relies on the mean motion vector $\bar{\mathbf{b}}$, a learned system matrix $\mathbf{B}\in \mathbb{R}^{3m\times 2}$ and the surrogate signal $\hat{\mathbf{z}}(t)$:

$$\mathbf{m} = \bar{\mathbf{b}} + \mathbf{B}\hat{\mathbf{z}}(t) \tag{11}$$

for which $\mathbf{B}$ can be estimated by multivariate regression. With given $\mathbf{B}$, Eq. 11 can be rewritten as the linear combination of three column vectors $\mathbf{a}_{1..3} \in \mathbb{R}^{3m}$

$$\mathbf{m} = \mathbf{a}_1 g(t) + \mathbf{a}_2 g'(t) + \mathbf{a}_3 \tag{12}$$

Reinterpreting the column vectors as functions $a_{1..3} : \mathbb{R}^3 \rightarrow \mathbb{R}^3$ by rearranging the elements into a 3D image structure and using linear interpolation, computation of a single displacement now is defined as

$$u(\mathbf{X}, t) = a_1(\mathbf{X})g(t) + a_2(\mathbf{X})g'(t) + a_3(\mathbf{X}) \tag{13}$$

and can be used in the algorithms for visual and haptic rendering.

4 Evaluation & Results

To support the applicability of the proposed methods, we first demonstrate forces computed by the needle insertion algorithm for given parameters. Then, we present the implementation of a demonstration system and give results of the visualization[1]. In the end, a performance analysis of the needle algorithm and the ray casting is presented.

4.1 Needle Force and Parameter Evaluation

Our needle insertion algorithm can model the measurements presented in [37] for ex-vivo needle insertion into bovine liver, see Fig. 5. To achieve this behavior, the parameters $a_1 = 0.048\frac{N}{mm}$ and $a_2 = 0.0052\frac{N}{mm^2}$ have been set according to [37] and $f^{\text{cut}} = 2.5N$ for the capsule of a virtual liver tissue phantom. Inside the liver, the cutting force has been set to $f^{\text{cut}} = 1N$ and friction parameters to $R = 0.025N$ and $k = 0.5\frac{N}{mm}$.

To reflect our experience with liver puncture in PTCD simulation [3], the parameters were set to $a_1 = 1.2\frac{N}{mm}$, $a_2 = 0$, $f^{\text{cut}} = 0.3N$ and the friction parameters as above. A resulting force plot is given in Fig. 5, for which the needle was steered along a predefined path into the liver of the virtual patient in caudal-crandial direction one time with respiratory motion enabled and one time without it. First, the needle was inserted into the liver over 10 seconds, then it was stopped, followed by an additional insertion further into the liver tissue. After an additional pause, the needle was retracted.

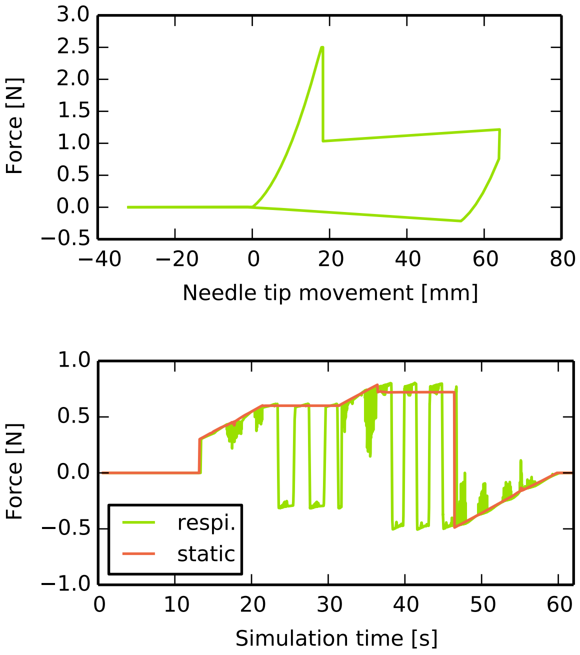


Figure 5: Forces acting on the haptic device in insertion direction for needle insertion along predefined insertion paths: (upper) Modeling of measurements given in [37] (lower) needle insertion into a virtual patient with and without simulation of respiratory movements.

4.2 Demonstration System

In this section, the capabilities of our 4D visuo-haptic rendering framework are demonstrated for a liver puncture scenario[2]. For this purpose, a low-dose 4D CT data set (14 phases, 512 × 512 × 460 voxels) was used to generate a sequence of displacement fields as described in section 3. As the reference phase, maximum inspiration was selected. Surrogate signal data of a spirometry device acquired for 4D reconstruction was available. Using this data, the motion models with different resolutions were created (64^3, 128^3 and 256^3 voxels). Additionally, the reference phase image data was resampled with a resolution of 256^3 voxels. Since this data set is based on low-dose CT data, the contrast of structures inside the liver is very low. For providing a training scenario of liver puncture, we artificially added the structures liver blood vessels, bile ducts and lesion. First, segmentation masks of the three structures were created by a rough registration of a segmentation of a similar patient image with label data and manual refinement of the result. Then, image values in the CT data where adjusted using the masks to represent typical values. Additionally, using the segmentation masks, surface models have been created using the Marching Cubes algorithm [43] followed by surface smoothing and decimation. This was also performed for threshold based segmentations of the patient's skin surface and bone structures. By this means, we are able to analyze the behavior of both ray casting based visualization and surface based visualization and show that our method is also well suited for surface based rendering.

[1] A video sequence is available in the supplementary material.

[2] Remarks on implementation are given in the appendix

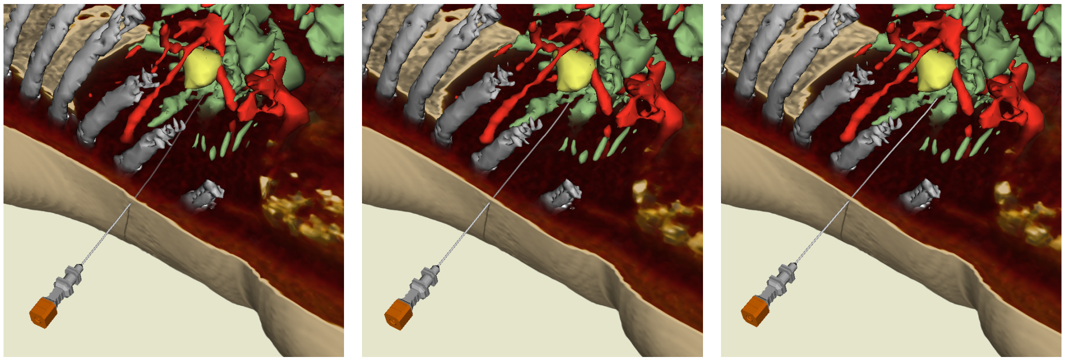


Figure 6: Sequence of mixed surface and volume rendering with surface model of a lesion (yellow), bile ducts (green), liver blood vessels (red) and bony ribs (white).

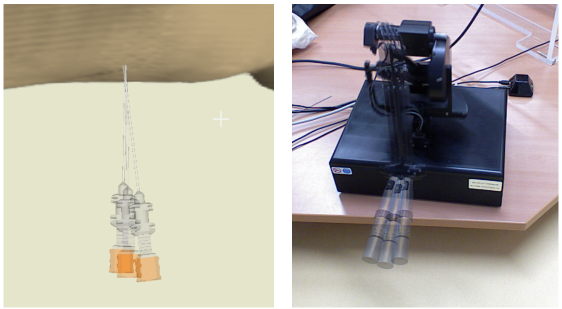


Figure 7: Overlayed image and photo sequence of virtual needle and haptic device handle with movement and rotation induced by breathing motion.

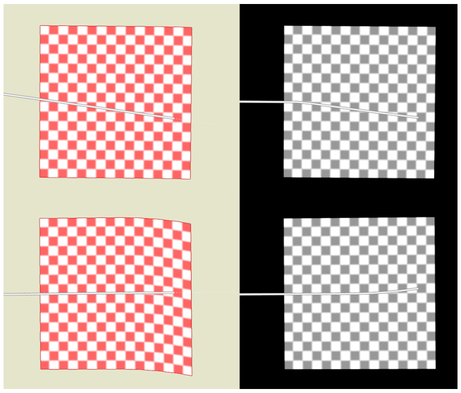


Figure 8: Clipped tissue phantom with effect of needle angulation during needle insertion (upper) and effect of breathing induced deformation of tissue at the needle tip on the needle path (lower). Left side: world space. Right side: reference space.

4.3 Visual Results of the Motion Modeling

Fig. 6 shows a combined volume and surface rendering with a needle inserted towards a mock up tumor model. The breathing induced motion of the lung, liver and inside-liver structures can clearly be seen. Bending of the needle is also well visible. Forces resulting from the lever-like behavior of the needle can be clearly perceived at the haptic device handle. Fig. 7 shows the movement of the haptic device handle under free movement in the same scenario.

In Fig. 8, the effect of needle angulation by the user and breathing motion on the needle path is demonstrated. For the first case, the needle was inserted perpendicular to the surface for ca. 10 mm into a virtual tissue phantom without breathing motion. Afterwards the needle was hold, angulated and further insertion took place. This results in a slightly bended needle. For the second case, the needle has been inserted into the phantom under displacement of parts of the phantom. The needle remains nearly without bending, but a non-linear needle path in reference space can be perceived.

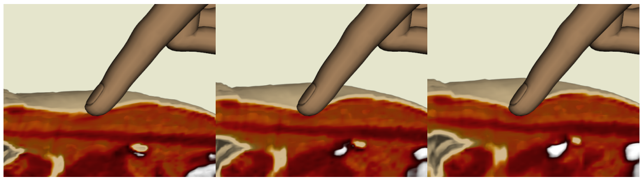


Figure 9: Visualization of the influence of a palpating finger with fixed position on the breathing patient model.

To demonstrate the rendering of local deformations combined with the breathing induced global displacement, Fig. 9 shows a sequence with a virtual finger that is fixed to a position close to the patient model's skin surface.

Resulting displacement fields are visualized using color coding of the projected direction in Fig. 10. Also, Fig. 11 presents different settings of the inversion scheme for a fixed breathing phase. For ray casting, it can be seen that using only a single iteration and not including the result from previous sampling points does not create a smooth rendering. With it, a smooth rendering is created and with twice the number of sampling points, only interpolation rendering artifacts vanish without affecting the overall result in a noticeable way. For the computation of a slice using the fixed point scheme, it is shown that a single iteration does not suffice, but 5 iterations is visually nearly indistinguishable from the result given by 50 iterations.

Fig. 12 shows the drawback of using low resolution displacement fields; in regions where discontinuities in the displacement fields are present, i.e. at the sliding lung boundary, smearing artifacts arise. The artifacts are present between lung and ribs, which is not a punctured region, making them not relevant for the haptic simulation.

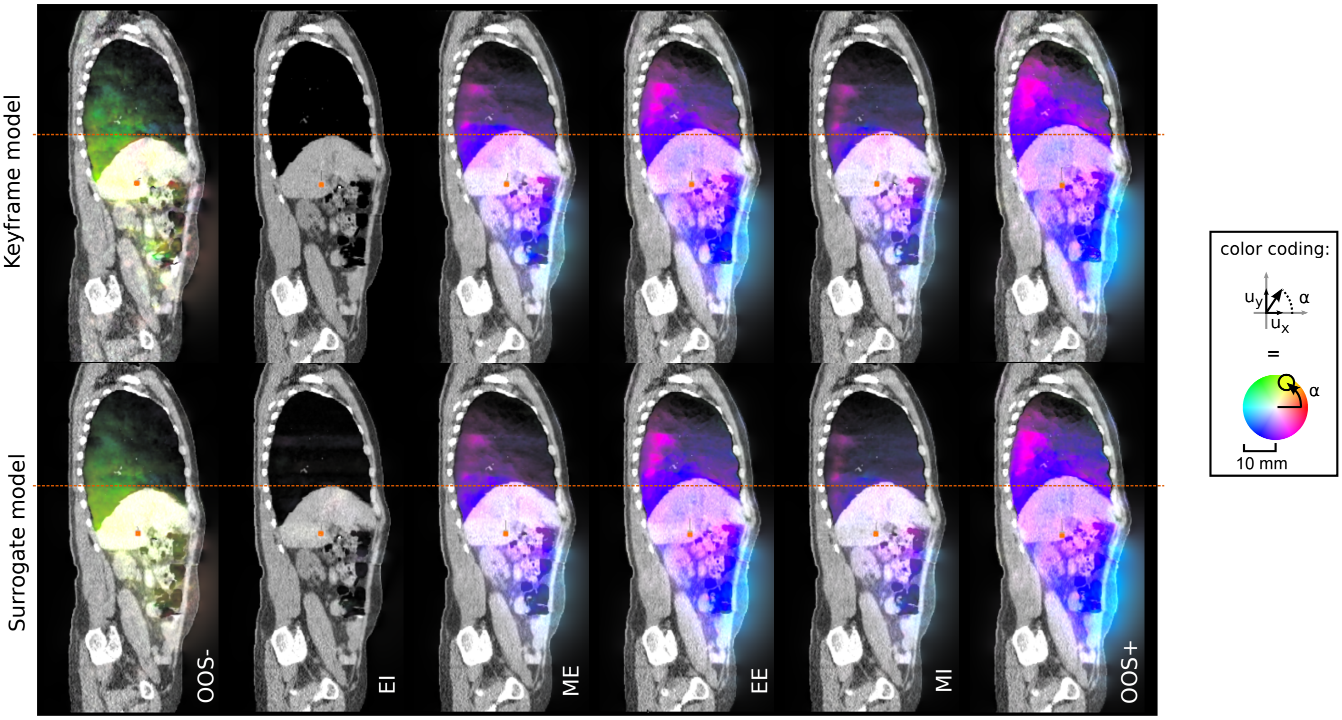


Figure 10: Resampled sagittal slices of the virtual patient CT data during the breathing sequence EI->ME->EE->MI and out-of-sample (OOS) predictions for both the key frame and the surrogate model. The direction of the displacements projected onto the slice are color code by the HSV color wheel, amplitude is indicated by opacity of the overlay (total opaqueness corresponds to 10 mm).

4.4 Performance Analysis

Run time behavior of the implementation in Nvidia Cuda is analyzed using a Nvidia GTX 680 with 3Gb of RAM. First, we analyzed the processing times for the rendering of a single frame by ray casting while using the presented methods for different image sizes and different displacement field sizes. The results are shown in Tab. 2. The resolution of the rendering viewport was set to 1224×1014 pixels. Note that both key frame approaches can rely on the same code for rendering and only differ in the determination of the interpolation parameter and interpolated key frames, which takes place before the ray casting.

Updating the position of vertices of surface models using Cuda is very fast: For surface models of bone (430,578 vertices), skin (145,648 vertices) and liver blood vessels (12,026 vertices), update times of 770.5±40.7 μs, 418.1±16.8 μs and 141.1±20.6 μs, resp. have been measured (252 samples).

Also, a performance analysis of the needle insertion algorithm for different numbers of needle nodes (15, 25 and 50) was performed on a PC with an Intel i7 CPU 970 @ 3.20GHz and 24 Gb of RAM. Fig. 13 shows box-plotted rendering times of the parts of Alg. 1 grouped into (1) update of displacements, (2) time stepping the physics simulation (3) total computation time. Furthermore, for a number of 25 needle nodes, the mean processing time for one step of the physics simulation has been measured, which is 14.5±15.7 μs, giving a mean number of 15.1±6.7 iterations of the physics simulation in each haptic simulation frame.

	Image size		Field size		N	Mem	Rendering	
Model	Voxel	Mb	Voxel	Mb		Mb	mean	stdev
none	small	151	0	0	0	151	16.36	0.10
none	large	1,085	0	0	0	1,085	15.99	0.12
key frame	small	151	64^3	4	14	210	35.74	0.16
key frame	small	151	128^3	34	14	621	36.83	0.22
key frame	small	151	256^3	268	6	1,762	37.52	0.33
key frame	large	1,085	64^3	4	14	1,144	35.90	0.16
key frame	large	1,085	128^3	34	14	1,555	36.05	0.15
key frame	large	1,085	256^3	268	4	2,159	36.21	0.32
surrogate	small	151	64^3	4	3	164	27.08	0.13
surrogate	small	151	128^3	34	3	252	27.48	0.90
surrogate	small	151	256^3	268	3	956	27.63	0.95
surrogate	large	1,085	64^3	4	3	1,098	26.32	0.13
surrogate	large	1,085	128^3	34	3	1,186	26.53	0.13
surrogate	large	1,085	256^3	268	3	1,891	26.55	0.20

Table 2: Ray casting rendering times in ms for the different motion models with a low (256^3 voxels) and high ($512 \times 512 \times 460$ voxels) resolution of the image data.

5 Discussion

The previous chapters of this article showed the general applicability of our presented methods and the performance of our implementation. It was shown that the influence of breathing motion modeled by the given approaches can be used for visuo-haptic interaction and creates realistic bending of a flexible needle. Besides this, bending and breathing motion influence the path taken by the needle through the virtual patient's tissue. Regarding haptic force feedback, the resulting forces can be perceived distinctively.

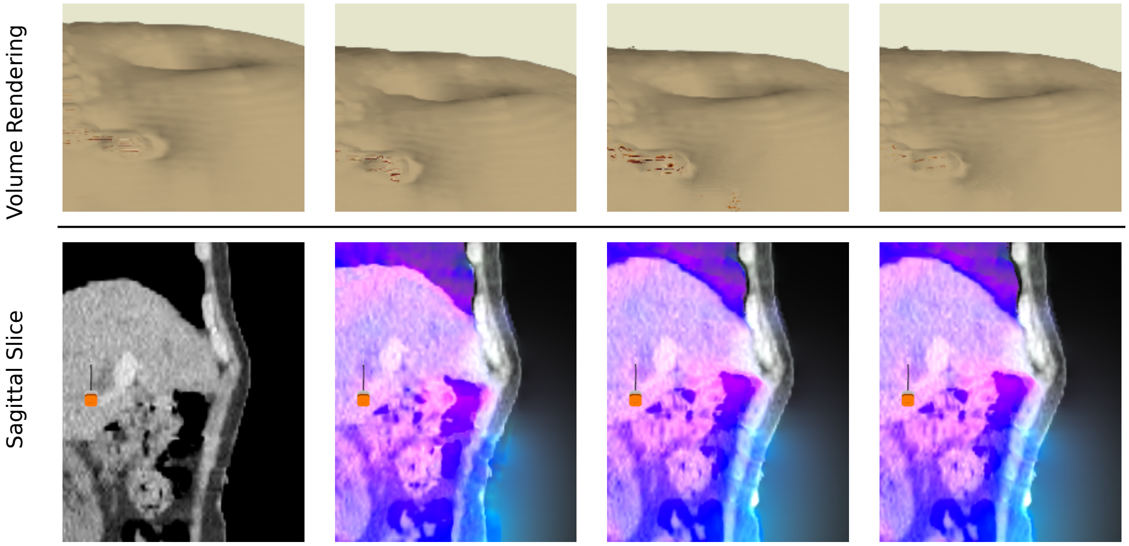


Figure 11: Effects of different settings of the inversion scheme. Upper row (left to right): Volume rendering of the navel without inversion, without incorporation of the result of previous samples, the scheme as described in the article and with doubled number of sampling points. Lower row: Sagittal slice with 0, 1, 5 and 50 iterations of the inversion scheme.

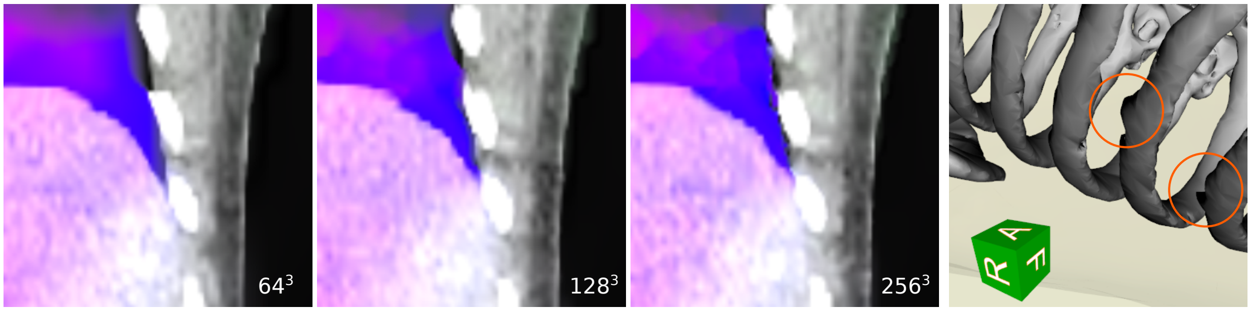


Figure 12: Left: Low resolutions of the displacement field lead to smearing artifacts at sliding interfaces. Right: Surface model of ribs are affected by this at a resolution the displacement field of 64^3 elements.

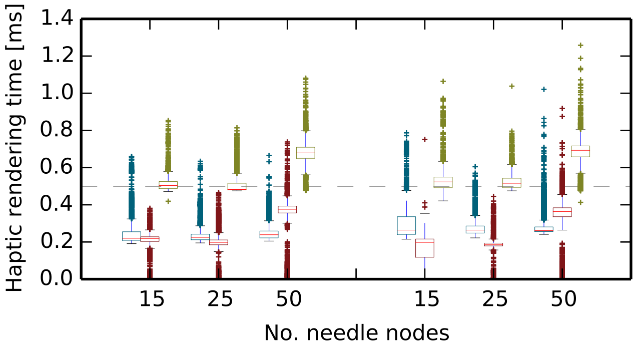


Figure 13: Box-plotted rendering timings of needle algorithm for the key frame based approach (left), and the surrogate signal based approach (right). Each pair of three contains of update of displacements (blue), physic simulation (red) and total computation time (green).

For the palpation simulation, the angle of rotational back transformation of the resulting palpation force was small for the patient model under consideration. Probably, this is caused by the fact, that nearly no rotation is present in $u(\mathbf{x}, t)$ near the skin surface. The computed force and torque could directly be displayed to the user without the transformation. In general, this assumption might not be true for every patient model. Nevertheless it might be interesting to investigate if the effect of small changes of force direction can be perceived significantly and in which extent the type of haptic device has influence on this.

For the estimation of haptic parameters, we adjusted the parameters to reflect the behavior to be similar to previous versions of the simulation framework (e.g., [26], [3]) that did not include breathing simulation. We showed that the needle insertion algorithm is capable of simulating forces as measured in [37]. Also, we demonstrated that the stiffness parameters of the needle can be adjusted to resemble behavior of a 16 gauge needle. Further parameter estimation and evaluation should be performed in future work in case the framework is applied to a full feature training simulator and should include the assessment of face and construct validity.

The presented demonstration scenario uses a virtual patient with an artificially introduced lesion. Thus, the used surrogate signal and computed displacement function might not fully reflect reality. For patient data including a real lesion, the methods will incorporate the small changes in tissue behavior. Generally, the motion methods are from the domain of radiation therapy of lesions under respiratory motion, where an accurate prediction of the movement of the lesion is desired.

For the visualization, regions, in which large discontinuities of the resulting displacement field are present, smearing artifacts arise. These are especially visible at locations where a high gradient of image values exists. An example for this are the regions between ribs (high image values) and lungs (very low image values). For the puncture scenarios simulated by our framework, the lungs or structures inside of it are not relevant for haptic simulation and thus these artifacts only remain as an visualization issue. We consider the interpolation between voxels along the discontinuities at the interfaces of sliding structures as the source for this. By using displacement fields with a larger resolution, this will be mitigated in the future.

Concerning rendering performance, it was shown that, as expected, the introduction of motion models increases the time needed for volume rendering, but interactive frame rates are still easily achievable. Remarkably, the resolution of the displacement field does influence the performance only very little. Updating the positions of the vertices for surface rendering is negligible. For haptic rendering, we found that 25 nodes gave us the best results as a trade-of between plausible visualization of the discretized needle and a stable simulation.

For our method to work, we only need a simple displacement function and perform its inversion on-the-fly and only for needed elements. This is advantageous since it is straightforward to implement such a function. Also, the iterative fixed point inversion is always computed only once per sampling point and relies on accurate results from previous sampling steps. This applies for both the haptic rendering as well as the visual rendering by ray casting. For haptic rendering, the accuracy of the inversion method depends on two factors: First, the temporal sampling rate, i.e. the haptic update rate, which is very high. Second, the change of sampling location, which depends on the movement of the haptic device. For ray casting, the sampling rate, i.e. the spatial distance between sampling points on a ray influences the accuracy of the computed inverse displacement. Also, it depends on the rate of change of the displacement function, which is also dependent on the smoothness of the displacement field.

A central concern of our simulation framework is the goal of avoiding unstructured mesh models for haptic and visual rendering for the purpose of being able to omit the mesh generation process for new patients. Nevertheless, the motion models presented here could be integrated into coarse grid or mesh based frameworks by applying the displacement function to mesh nodes. For FEM based methods, it might be reasonable to apply it to static or boundary nodes.

Overall, we consider the introduction of the presented methods a very valuable contribution to visuo-haptic simulation of needle puncture interventions. The methods enable for the first time realistic visuo-haptic needle insertion simulation into a virtual breathing patient model based on a patient specific 4D image data set and real breathing signals. Especially the transfer of methods from image registration to visuo-haptic rendering enables a highly realistic model of respiratory motion.

6 Conclusion and Future Work

A visuo-haptic simulation framework for needle insertion capable of rendering virtual patients under the influence of breathing motion was presented. The motion models are fast to compute, making them suitable for visual rendering using ray casting or surface rendering, and can be used for haptic interaction with the virtual patient model under respiratory motion.

Here, we used a recorded spirometry surrogate signal. In the future, the source of the signal could be based on surrogate data recorded at run-time of the simulator. This way, training of an intervention could be augmented by having a training partner control the breathing of the virtual patient model. Alternatively, a realistic synthetic surrogate signal could be simulated during run-time [44].

Future work will include the investigation of methods for estimation of breathing motion for patients for which only 3D images are available. This way, motion models could be generated for the simulation without having to obtain a full 4D CT.

Additionally, the framework could be applied to other needle insertion interventions. If necessary, bevel-tip needle behavior could be easily integrated. Furthermore, the application of the presented respiratory models is not limited to (multi-)proxy based haptic algorithms and needle insertion and could be integrated in surgery simulation frameworks that rely on mesh based methods.

Appendix

Implementation of our 4D framework is split into a CPU component that deals with the haptic computations and a GPU component that is responsible for visualization. The ray casting based visual volume rendering of the breathing patient is implemented entirely using Nvidia Cuda. The rendering process writes to a pixel buffer object shared with the OpenGl environment of a VTK based rendering pipeline. Deformable triangular surfaces are also stored in shared pixel buffer objects. Additionally, VTK is used for the visualization of tools in the scene. Compared to the main memory of a modern workstation, the amount of available GPU memory is limited. Medical images and especially displacement fields are relatively large. In our Cuda-based GPU implementation, we store each image voxel by 9 bytes (4 bytes gray value, 4 bytes for a second channel, 1 bytes for the label set). To use fast texture interpolation, it was necessary to use single precision floats (4 bytes). Each element of the displacement field is stored by 16 bytes (4 bytes for x-, y- and z-component, 4 unused bytes). This way, a single image with a resolution of $512 \times 512 \times 460$ voxels and four displacement fields used for the key frame method at a resolution of 256^3 voxels already accounts for approx. 2 Gb of memory, which already fills a huge part of the available GPU memory on our reference hardware (Nvidia GTX 680 with 3Gb of RAM). Concerning the CPU based implementation of the haptic algorithms, it is worth mentioning that instead of performing look up and interpolation in the displacement fields just-in-time, all needed values are fetched all at once before performing Alg. 1 to save time. Also, the needle haptic algorithm was designed in a way that reduces random access to elements of the available image data; during insertion, only values at the needle path tip are updated.

References

[1] P. F. Villard, F. P. Vidal, L. Ap Cenydd, R. Holbrey, S. Pisharody, S. Johnson, A. Bulpitt, N. W. John, F. Bello, and D. Gould, "Interventional Radiology Virtual Simulator for Liver Biopsy," *Int. J. Comput. Assist. Radiol. Surg.*, pp. 1–13, 2013.

[2] N. Dong, Y. Wing, and Q. Jing, "A Virtual Reality Simulator for Ultrasound-guided Biopsy Training," *IEEE Comput. Graph. Appl.*, vol. 31, no. 2, pp. 36–48, 2010.

[3] D. Fortmeier, A. Mastmeyer, J. Schröder, and H. Handels, "A Virtual Reality System for PTCD Simulation using Direct Visuo-haptic Rendering of Partially Segmented Image Data," *IEEE J. Biomed. Heal. Informatics*, 2015.

[4] S. Ullrich, IEEE, and T. Kuhlen, "Haptic Palpation for Medical Simulation in Virtual Environments," *IEEE Trans. Vis. Comput. Graph.*, vol. 18, no. 4, pp. 617–620, 2012.

[5] T. T. R. Coles, N. N. W. John, D. A. Gould, and D. D. G. Caldwell, "Integrating Haptics with Augmented Reality in a Femoral Palpation and Needle Insertion Training Simulation," *IEEE Trans. Hapt.*, vol. 4, no. 3, pp. 199–209, 2011.

[6] N. Chentanez, R. Alterovitz, D. Ritchie, L. Cho, K. K. Hauser, K. Goldberg, J. R. Shewchuk, and J. F. O'Brien, "Interactive Simulation of Surgical Needle Insertion and Steering," *ACM Trans. Graph.*, vol. 28, no. 3, pp. 1–10, 2009.

[7] O. Goksel, K. Sapchuk, W. J. Morris, and S. E. Salcudean, "Prostate Brachytherapy Training with Simulated Ultrasound and Fluoroscopy Images," *IEEE Trans. Biomed. Eng.*, vol. 60, no. 4, pp. 1002–12, 2013.

[8] P. Keall, T. Yamamoto, and Y. Suh, "Introduction to 4D Motion Modeling and 4D Radiotherapy," in *4D Modeling and Estimation of Respiratory Motion for Radiation Therapy*, ser. Biological and Medical Physics, Biomedical Engineering, C. Ehrhardt, Jan and Lorenz, Ed., 2013, pp. 1–21.

[9] A. P. Santhanam, C. Imielinska, P. Davenport, P. Kupelian, and J. P. Rolland, "Modeling Real-time 3-d Lung Deformations for Medical Visualization," *IEEE Trans. Inf. Technol. Biomed.*, vol. 12, no. 2, pp. 257–70, 2008.

[10] A. Hostettler, S. A. Nicolau, Y. Rémond, J. Marescaux, and L. Soler, "A Real-time Predictive Simulation of Abdominal Viscera Positions during Quiet Free Breathing," *Prog. Biophys. Mol. Biol.*, vol. 103, no. 2-3, pp. 169–84, 2010.

[11] P. F. Villard, P. Boshier, F. Bello, and D. Gould, "Virtual Reality Simulation of Liver Biopsy with a Respiratory Component," in *Liver Biopsy*, H. Takahashi, Ed. InTech, 2011, ch. 20.

[12] F. Vidal, P. Villard, and E. Lutton, "Automatic Tuning of Respiratory Model for Patient-based Simulation," in *International Conference on Medical Imaging Using Bio-Inspired and Soft Computing (MIBISOC)*, 2013, pp. 225–231.

[13] P. F. Villard, F. P. Vidal, F. Bello, and N. W. John, "A Method to Compute Respiration Parameters for Patient-based Simulators," *Stud. Heal. Technol. Informatics*, vol. 173, pp. 529–533, 2012.

[14] X. Wu, J. Allard, and S. Cotin, "Real-time Modeling of Vascular Flow for Angiography Simulation," *Int. Conf. Med. Image Comput. Comput. Interv. (MICCAI)*, vol. 10, no. Pt 1, pp. 557–65, Jan. 2007.

[15] K. Engel, M. Hadwiger, J. Kniss, C. Rezk-Salama, and D. Weiskopf, *Real-time Volume Graphics*. AK Peters, Ltd., 2006.

[16] C. T. Silva, J. L. D. Comba, S. P. Callahan, and F. F. Bernardon, "A Survey of GPU-Based Volume Rendering of Unstructured Grids," *Revista de informática teórica e aplicada*, vol. 12, no. 2, pp. 9–29, 2005.

[17] J. Georgii and R. Westermann, "A Generic and Scalable Pipeline for GPU Tetrahedral Grid Rendering," *IEEE Trans. Vis. Comput. Graph.*, vol. 12, no. 5, pp. 1345–1352, 2006.

[18] J. Gascon, J. M. Espadero, A. G. Perez, R. Torres, and M. A. Otaduy, "Fast Deformation of Volume Data Using Tetrahedral Mesh Rasterization," in *Proc. of the 12th ACM SIGGRAPH/Eurographics Symposium on Computer Animation*. ACM, 2013, pp. 181–185.

[19] R. Torres, J. M. Espadero, F. A. Calvo, and M. A. Otaduy, "Interactive Deformation of Heterogeneous Volume Data," pp. 131–140, 2014.

[20] Y. Kurzion and R. Yagel, "Space Deformation using Ray Deflectors," *Render. Tech. 95*, 1995.

[21] S. P. DiMaio and S. E. Salcudean, "Interactive Simulation of Needle Insertion Models," *IEEE Trans. Biomed. Eng.*, vol. 52, no. 7, pp. 1167–79, 2005.

[22] C. Duriez, C. Guébert, M. Marchal, S. Cotin, and L. Grisoni, "Interactive Simulation of Flexible Needle Insertions based on Constraint Models," *Int. Conf. Med. Image Comput. Comput. Interv. (MICCAI)*, vol. 12, no. Pt 2, pp. 291–9, 2009.

[23] O. Goksel, K. Sapchuk, and S. E. S. Salcudean, "Haptic Simulator for Prostate Brachytherapy with Simulated Needle and Probe Interaction," *IEEE Trans. Hapt.*, vol. 4, no. 3, pp. 188–198, 2011.

[24] I. Peterlik, M. Nouicer, C. Duriez, S. Cotin, and A. Kheddar, "Constraint-Based Haptic Rendering of Multirate Compliant Mechanisms," *IEEE Trans. Hapt.*, vol. 4, no. 3, pp. 175–187, 2011.

[25] O. Goksel, E. Dehghan, and S. E. Salcudean, "Modeling and Simulation of Flexible Needles," *Med. Eng. Phys.*, vol. 31, no. 9, pp. 1069–78, 2009.

[26] M. Färber, T. Dahmke, C. Bohn, and H. Handels, "Needle Bending in a VR-Puncture Training System using a 6DOF Haptic Device," in *Proceedings of MMVR 17*, vol. 142, Jan. 2009, p. 91.

[27] M. Bergou, M. Wardetzky, S. Robinson, B. Audoly, and E. Grinspun, "Discrete Elastic Rods," *ACM Trans. Graph.*, vol. 27, no. 3, p. 1, 2008.

[28] J. Spillmann and M. Teschner, "CORDE: Cosserat Rod Elements for the Dynamic Simulation of One-Dimensional Elastic Objects," *ACM SIGGRAPH 2007 Symp. Comput. Animat.*, vol. 1, 2007.

[29] J. Ehrhardt, R. Werner, D. Säring, T. Frenzel, W. Lu, D. Low, and H. Handels, "An Optical Flow based Method for Improved Reconstruction of 4D CT Data Sets Acquired during Free Breathing," *Med. Phys.*, vol. 34, no. 2, pp. 711–721, 2007.

[30] J. Ehrhardt and C. Lorenz, Eds., *4D Modeling and Estimation of Respiratory Motion for Radiation Therapy*. Springer Berlin Heidelberg, 2013.

[31] A. Schmidt-Richberg, R. Werner, H. Handels, and J. Ehrhardt, "Estimation of Slipping Organ Motion by Registration with Direction-dependent Regularization," *Med. Image Anal.*, vol. 16, no. 1, pp. 150–159, 2012.

[32] M. Wilms, R. Werner, J. Ehrhardt, A. Schmidt-Richberg, H.-P. Schlemmer, and H. Handels, "Multivariate Regression Approaches for Surrogate-based Diffeomorphic Estimation of Respiratory Motion in Radiation Therapy," *Phys. Med. Biol.*, vol. 59, no. 5, pp. 1147–64, 2014.

[33] M. Chen, W. Lu, Q. Chen, K. J. Ruchala, and G. H. Olivera, "A Simple Fixed-point Approach to Invert a Deformation Field," *Med. Phys.*, vol. 35, no. 1, p. 81, 2008.

[34] D. Fortmeier, A. Mastmeyer, and H. Handels, "An Image-Based Multiproxy Palpation Algorithm for Patient-Specific VR-Simulation," in *Medicine Meets Virtual Reality 21, MMVR 2014*, ser. Stud. Health Technol. Inform., vol. 196. IOS Press, 2014, pp. 107–113.

[35] B. K. P. Horn, "Closed-form Solution of Absolute Orientation using Unit Quaternions," *J. Opt. Soc. Am.*, vol. 4, no. 4, p. 629, 1987.

[36] D. Fortmeier, A. Mastmeyer, and H. Handels, "Image-based Soft Tissue Deformation Algorithms for Real-time Simulation of Liver Puncture," *Curr. Med. Imaging Rev.*, vol. 9, no. 2, pp. 154–165, 2013.

[37] A. M. Okamura, C. Simone, and M. D. O'Leary, "Force modeling for needle insertion into soft tissue." *IEEE transactions bio-medical engineering*, vol. 51, no. 10, pp. 1707–16, Oct. 2004.

[38] E. Coumans. Bullet Physics Library. [Online]. Available: "http://bulletphysics.org/"

[39] D. Sarrut, J. Vandemeulebroucke, and S. Rit, "Intensity-Based Deformable Registration: Introduction and Overview," in *4D Modeling and Estimation of Respiratory Motion for Radiation Therapy*, ser. Biological and Medical Physics, Biomedical Engineering, J. Ehrhardt and C. Lorenz, Eds. Springer Berlin Heidelberg, 2013, pp. 103–124.

[40] R. Werner, A. Schmidt-Richberg, H. Handels, and J. Ehrhardt, "Estimation of Lung Motion Fields in 4D CT data by Variational Non-linear Intensity-based Registration: A Comparison and Evaluation Study," *Phys. Med. Biol.*, vol. 59, no. 15, pp. 4247–60, 2014.

[41] K. Murphy, B. Van Ginneken, J. M. Reinhardt *et al.*, "Evaluation of Registration Methods on Thoracic CT: The EMPIRE10 Challenge," *IEEE Trans. Med. Imaging*, vol. 30, no. 11, pp. 1901–1920, 2011.

[42] J. McClelland, "Estimating Internal Respiratory Motion from Respiratory Surrogate Signals Using Correspondence Models," in *4D Modeling and Estimation of Respiratory Motion for Radiation Therapy*, ser. Biological and Medical Physics, Biomedical Engineering, J. Ehrhardt and C. Lorenz, Eds. Springer Berlin Heidelberg, 2013, pp. 187–213.

[43] W. E. Lorensen and H. E. Cline, "Marching Cubes: A High Resolution 3D Surface Construction Algorithm," in *ACM SIGGRAPH Computer Graphics*, vol. 21, no. 4, 1987, pp. 163–169.

[44] M. Wilms, J. Ehrhardt, R. Werner, M. Marx, and H. Handels, "Statistical Analysis of Surrogate Signals to Incorporate Respiratory Motion Variability into Radiotherapy Treatment Planning," in *SPIE Medical Imaging 2014, Image-Guided Procedures, Robotic Interventions, and Modeling*, Z. R. Yaniv and D. R. Holmes, Eds., no. 1, 2014, p. 90360J.