Virtual Reality Training Simulator for Percutaneous Needle-Based Interventions with Haptic Feedback

Noah Bertholon, Claire Martin, Erik Pernod, Hadrien Courtecuisse

Virtual Reality Training Simulator for Percutaneous Needle-Based Interventions with Haptic Feedback

Noah Bertholon[1], Claire Martin[1], Erik Pernod[2], and Hadrien Courtecuisse[1]

[1]iCube Laboratory, UMR 7357, CNRS - University of Strasbourg, Strasbourg, France
[2]InfinyTech3D, Nice, France

Abstract

This paper proposes a comprehensive virtual reality (VR) training simulator for needle-based interventions such as radiofrequency tumor ablation of the liver. As such, needle flexibility is simulated as well as an exhaustive representation of the abdominal cavity, including breathing motion. A modular unified framework is proposed, allowing the coexistence of solutions to complex and heterogeneous challenges within an asynchronous system. This paper presents a novel high-rate force rendering model based on an existing asynchronous approach. In addition, a complete visual rendering pipeline in VR, including real-time imaging guidance from interactive fluoroscopic images will be present. The strategic use of numerical simulation methods achieves high computational performance in spite of the complexity of the simulation, allowing user interaction through the haptic interface.

Keywords: needle insertion, finite element method, haptic, training simulator, virtual reality

1 Introduction

Liver cancers represent about 10% of cancer deaths worldwide. Among the possible treatments, partial liver resection has been widely performed. However, this procedure presents the drawbacks of an open surgery for the patient in terms of tissue damage and recovery time. Less invasive procedures exist, such as needle-based treatments, which interact with the tissues locally. Although benefiting the patient, such procedures remain complex without direct visibility of internal anatomical structures. Imaging guidance systems are used to help the practitioner, mainly ultrasound (US) or X-ray imaging systems. Nonetheless, leading the needle to the desired location while minimizing insertions and withdrawals to avoid damaging the tissue is a complex endeavor. Consequently, needle-based procedures require efficient training, often involving cadavers, animals, or mannequins. Those approaches can be limited in terms of mechanical behavior and possible scenarios since tissues do not always react as human ones and degrade over time. Numerical training simulators are emerging to provide practitioners with more flexible training tools. Thanks to VR devices and haptic interfaces, the user can be immersed in a virtual world. They allow for multiple scenarios, including particular medical complications, with the possibility to use patient-specific data.

In order to be seen as an efficient training tool, a needle insertion simulator must present some characteristics considered as essential by medical doctors. First, realism is key for an effective training session. Organs' mechanical properties should mimic real ones, the feeling of the needle puncturing or rubbing organs should be captured, and the needle flexibility has to be modeled. Indeed, practitioners use this flexibility to route the needle through organs to reach the desired location. Second, the simulator has to be permissive

by allowing users to learn from their mistake. They should be able to insert the needle from anywhere and lead it where they want. Last, it should be immersive, with smooth sensorial feedback, requiring around a 60 Hz refresh rate for visual rendering and at least 600 Hz for haptics [1]. The immersion is reinforced by taking context into account. From representing the whole Operating Room (OR) to modeling the breathing motion, those are key to feel more involved in a real procedure [2].

All of those requirements demand heterogeneous, often non-linear models to be simulated, raising complexity in the simulation. In this work, a modular and unified framework is proposed for a numerical needle insertion training simulator meeting those requirements. It offers both a haptic rendering and a full visual immersion in 3D with VR, representing the complete operating room environment, coupled with a simulation of the entire abdomen. Existing simulators and simulation methods from the literature will be exposed in Section 2. Then, Section 3 briefly introduces the chosen physical models used in the proposed simulator. Finally, Section 4 presents our contributions regarding haptic and visual rendering. Results will eventually be presented and discussed in Section 5.

2 Numerical Training Simulators from State-of-the-Art

Needle insertion simulators and user immersion : Several simulators exist in the literature, each one of them having its own methods to compute the needle insertion simulation and to render the scenario and environment to the user. In a recent survey performed by Corrêa et al. [3], emerging rendering methods for numerical needle insertion training simulators were discussed. While Augmented Reality (AR) allows for interaction with virtual and physical objects, it presents challenges in the superimposition of these two worlds. VR, on the other hand, appears to be the preferred method for visual rendering in needle insertion simulators. Realism in simulators depends on the adopted models and the coupling methods between simulation and rendering. Wu et al. [4] developed a VR needle insertion simulator coupled with a haptic environment and a Novint Falcon haptic interface for force rendering. Visual output is displayed on a screen. Sutherland et al. [5] presented an AR simulator that integrates a physical mannequin with a virtual ultrasound (US) probe and a haptic device. The camera tracks the US probe and dummy, and ultrasound images are computed from patient-specific CT slices. The CT images are overlaid onto the mannequin images, leading to challenges related to the superimposition of real and virtual information. Alamilla et al. [6] presented a VR simulator with two haptic devices to model the needle and a US probe, respectively providing haptic and visual (2D and 3D) renderings, thus avoiding the tracking process. The simulation is coupled with the haptic devices using the *Boost* libraries by sharing the poses of the devices with the virtual scene. Although the simulator benefits from a hand support for better haptic rendering, visual information is displayed on a screen, limiting the visual immersion of the simulator.

Simulation and contact models : Deformations of anatomical structures are key features to model in such simulators. Although several modeling methods exist, such as Position-Based Dynamics (PBD) [7], Projective Dynamics (PD) [8, 9], or meshless methods [10], Finite Element (FE) models remain the gold standard in mechanics computation due to their accuracy and the increased computational power that allows the computation of complex models in real-time.

In addition to individual organ deformations, collisions between the components must be modeled. The specific problem of contact can be handled with constraint-based methods with Lagrange multipliers with accuracy for large time steps [11, 12]. Iterative methods allow for the resolution of the generated Linear Complementarity Problem (LCP) for large-scale simulations [13, 14, 15], while direct methods are computationally expensive to provide exact solutions. More recently, Incremental Potential Contact has shown great robustness, stability, and accuracy in simulating interactions between bodies with complex mechanical behaviors, including friction [16]. It is based on a continuous collision detection method and

a potential barrier energy and results in intersection-free and inversion-free contact resolutions, even if its application to needle insertion is still to be studied.

The specific case of needle-tissue interactions is critical in medical procedures. The related forces are often split into stiffness, friction, and cutting forces [17], each of them being defined by a specific model. However, other models include the mechanical behavior of the needle and the tissue to compute these particular interactions. Roesthuis et al. [18] used springs to simulate the interaction between needles and tissues in their interactive modeling approach. Chentanez et al. [19], on the other hand, employed the Timoshenko beam theory to predict the needle deflection within soft tissues. In [20], Wang et al. rely on a cantilever beam model coupled with hyperelastic material for the tissue. Duriez et al. [21] proposed a constraint-based method to control the relative movements of needle models, composed of beam elements with six degrees of freedom, and volume models, consisting of tetrahedra or hexahedra. This approach does not require any costly remeshing operation and allows for dynamical creation of constraints along the needle path. This approach has been extended to various applications, including real-time simulations of complex interactions such as punctures and fractures, as well as robotic control [22, 23, 24]. Constraint-based simulations can also provide haptic feedback, allowing users to feel simulated tissue deformation and to respond accordingly.

Update rate requirements and asynchronism : Computational performance is crucial to allow user interaction through the haptic interface, while simplifications and approximations tend to decrease accuracy and realism. Therefore, one of the main challenges in large-scale numerical training simulators is to develop methods allowing them to reach high computational performance while using models that are accurate enough so that the simulator can be considered an efficient training tool. As stated before, a smooth rendering requires update rates of at least 600 Hz and 60 Hz for the haptic and visual loops, respectively. With their methods, Wu et al. reached a 144 Hz haptic update rate in [4], inciting them to consider GPU computations in future work to improve the haptic realism of their simulator. Asynchronous solutions are often proposed to answer requirements from the different loops. The haptic and visual rendering loops are separated in [5] and, for a satisfactory haptic rendering of 1000 Hz, Sutherland et al. could reach a visual update rate of 20 Hz with an FE model of 2000 nodes. Courtecuisse et al. computed mechanics and haptic rendering at different rates in [25] by using separated threads, allowing them to reach about 1000 Hz and 30 Hz for the haptic and the simulation loops, respectively, while considering hyper-elastic models. Similarly, the mechanics of a rod were computed at high rates (1 kHz) based on the motions of the haptic interface to which the rod was related in [26]. The simulation was performed at lower rates (about 50 Hz), and only mechanical information was shared with the high-rate loop in order not to prevent the haptic thread from reaching high update rates.

Computation of force feedback : Force rendering can be computed using different strategies. Analytical models are widely developed due to their generally low computational cost and their ability to model different physical phenomena. Okamura et al. define in [27] the needle insertion forces as the sum of stiffness, friction, and cutting forces. Similarly, Sutherland et al. consider forces resulting from tissue displacement, friction, and needle bending in [5]. Alamilla et al. also model orthogonal clamping forces and hard contact in the context of intra-articular needle insertion [6]. These models mainly depend on parameters such as needle insertion depth or velocity. In order to consider contact coupling, which is relevant if multiple objects are colliding, Peterlik et al. used a constraint-based approach with FE models to compute haptic forces from a deformable rod in [26]. While this paper is, to our knowledge, one of few that computes the deformation of the needle at haptic rate, their method is quite cumbersome and does not seem to reach 1 kHz on heavier scenes.

3 Background

The main focus of the simulator is to behave realistically, to better immerse the student and hopefully make them learn better. This implies modeling the whole abdominal cavity so as to prevent the sensation of a free-floating liver and to correctly model breathing motion. The latter is produced by the diaphragm, while the stomach, the large and small intestines are neighboring organs of the liver, as illustrated in Figure 1. Therefore, all of these organs are simulated in the proposed framework, as well as a skin patch that the needle will pierce before reaching the liver. Organ meshes were generated from segmented medical images, fetched on the IRCAD website [28]. The respiratory motion was also gathered on real patient data, using 2D fluoroscopic imaging. The diaphragm plays this motion, pushing on organs, which in turn deforms and pushes other organs. Additionally, a set of spring-like constraints is applied to approximate ligament positions on the organs.

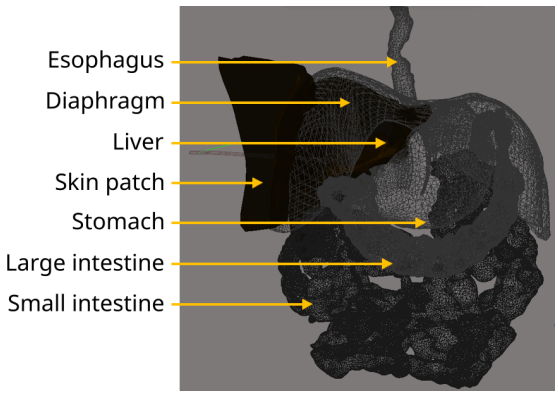

Figure 1: The abdominal cavity.

After a brief introduction of the bottleneck in Finite-Element (FE) simulation, three categories of numerical models used to compute organ mechanics are introduced. The strength of each model is detailed to address the mentioned FE bottleneck.

3.1 Finite Element Method

While fast computation is required to simulate the whole abdominal cavity in real-time, simulation stability is also a key requirement to provide sensible haptic feedback. A first step toward stability is made by using Euler Implicit time integration, which offers better simulation stability than explicit methods. Furthermore, all objects in the scene are based on the FE method, which is heavily documented. Briefly, in the FE method, the following Linear Complementarity Problem (LCP) is solved:

$$\begin{cases} \mathbf{A}\mathbf{x} = \mathbf{b} + h\mathbf{H}^T\lambda & (1) \\ \delta_{t+h} = \delta_{free} + h^2\mathbf{W}\lambda & (2) \end{cases}$$

This equation will not be detailed here; the interested reader might refer to [29] for further discussion. The main takeaway of Equation 1 is that on one side, $\mathbf{A}\mathbf{x} = \mathbf{b}$ computes the free position of the object being simulated, $\mathbf{x}$ being unknown. On the other side, $h\mathbf{H}^T\lambda$ adds constraints (i.e. collisions) to the system, λ being another unknown. While λ describes the strength of the forces to apply to resolve constraint violations δ, $\mathbf{H}$ stores the contact directions of those violations.

The main challenge in solving this LCP is to compute the compliance matrix $\mathbf{W}$, which describes the coupling between all simulated constraints and is used to compute the unknown λ using Equation 2 :

$$\mathbf{W} = \sum \mathbf{H}_i\mathbf{A}_i^{-1}\mathbf{H}_i^T \quad (3)$$

It requires the inversion of the system matrix $\mathbf{A}$, a matrix as large as the number of Degrees Of Freedom (DOFs) in the system. Note that at some point, Equation 1 could also benefit from the inversion of the system matrix to solve the free motion part.

Although the system is very large considering all the simulated organs, [30] shows that the system matrix of a heterogeneous system can be represented as the sum of smaller matrices representing each

object. Likewise, the associated compliance matrix can also be computed on individual objects and summed together. This allows using dedicated methods and solvers for each object to lower their simulation time, then assembling the full system and compliance matrix at a lower cost. 3 categories of models have been established to ensure low computation time and stability : Fully integrated volumes represent organs that interact directly with the needle, namely the liver and the skin, requiring precise computation. The needle is in its own category thanks to its simple mesh. Finally, reduced models [31] are used for organs that are in contact with the liver but should not be touched by the needle thus can be approximated. Those 3 categories will be annotated with subscripts v, n, and r, respectively.

Fully Integrated Volumes : The liver relies on linear tetrahedral elements, each with 4 nodes and 3 DOFs per node, to which the corotational method, described in [32], is added. This approach breaks down large displacements into rotations and small deformations. Linear deformations are computed within a rotated frame; the resulting forces are subsequently transformed back to the global frame using the transpose of the rotation matrix. This method allows modeling non-linearity while managing computational demands and stability. However, $\mathbf{A}_v$ remains large, and its inversion cannot be precomputed as the volume deforms under user interaction. Thankfully, to compute $\mathbf{W}_v$, constraints generally impact few DOFs, making $\mathbf{H}_v$ sparse, which particularly can be exploited to improve computational performance. Previous work in the literature optimizes the related computations [33, 30]. The work presented in [25] and [29] is especially interesting. The former exposes an asynchronous pre-conditioner computing the inverse of the system matrix aside from the main simulation loop. The second method benefits from the sparsity of $\mathbf{H}$ to compute $\mathbf{W}$ at a much lower cost.

The Needle : In the simulator, the needle model consists of beam elements, which are segments with 2 nodes and 6 DOFs per node (translations and rotations) that follow the beam theory [34]. Therefore, each node is related to at most two adjacent nodes, resulting in a Block-Tridiagonal (BTD) matrix for $\mathbf{A}_n$, whose size is small given the size of the needle mesh itself. The inversion of $\mathbf{A}_n$ can therefore benefit from Thomas' algorithm to be performed at low cost. $\mathbf{H}_n$ is denser but remains small: costs are therefore limited. $\mathbf{W}_n$ is the product of small matrices, themselves computed at low cost, leading to fast computations.

Reduced Models : A reduction approach is considered for the colon, stomach, and esophagus to limit the computational cost of their simulation, as they will not directly interact with the needle. Each organ is simulated using a snapshot-Proper Orthogonal Decomposition (POD) [31] coupled with a hyperreduction [35]. The former reduces the size of the system by using a smaller space but requires computing the full mechanics of the model. The latter helps with this computation by filtering out all parts of the mesh that do not actually impact the simulation. This results in a sparse system matrix, which is then compacted in a small dense matrix, making the assembly of $\mathbf{W}_r$ much faster.

4 Method

The models mentioned above address the computational weight of the simulated system by isolating each object in its own solver. The proposed asynchronous framework further alleviates the computational pressure by spreading it on multiple threads and machines, allowing for real-time simulation while keeping a smooth sensorial rendering for the user. After introducing its asynchronous architecture, the benefits in the $\mathbf{W}$ computation for each chosen model are detailed. Then, the high-rate haptic rendering model is presented to conclude by the visual rendering pipeline.

4.1 Asynchronous Architecture

Figure 2 illustrates the asynchronous architecture based on the work of Courtecuisse et al. [25], and will be discussed throughout Section 4. The main thread runs the simulation, which owns the mechanical data and is responsible for sending them to the different modules. The VR module is fully autonomous, as long as the simulation sends meshes information when their simulation is complete. However, the haptic thread requires synchronization with the main thread to refresh its mechanical data and to provide the simulation with the new needle position. While waiting for this synchronization, the haptic thread exchanges with the haptic device the position of the end-effector and the forces to apply on the latter. Those forces are updated at a haptic rate using lingering mechanical data from the simulation. Finally, the asynchronous preconditioner computes the inverse of the system matrix with data given by the simulation. When completed, the inverted matrix is passed to the simulation, which directly sends new data to repeat the cycle.

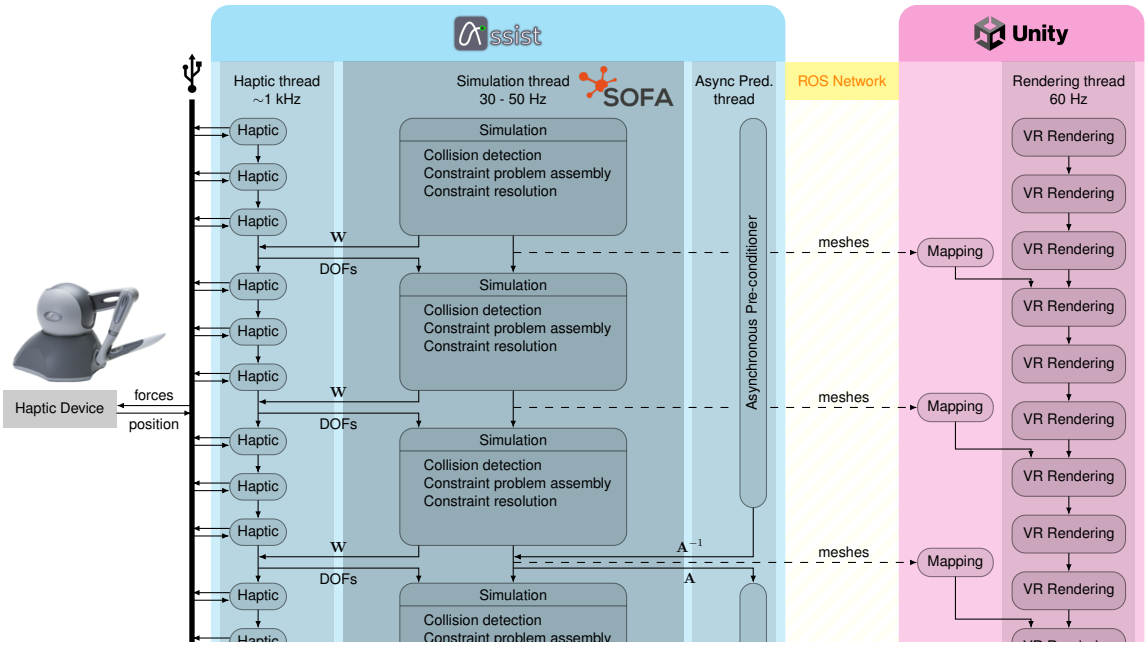

Figure 2: On the first computer, the simulation runs in the main thread and is responsible for synchronizing with the haptic thread and sending data to the VR computer. At slower rate, the asynchronous preconditioner computes the inverse of the system matrix. Meanwhile, the haptic thread runs in a dedicated thread to compute contact and reaction forces at a haptic rate.

4.2 Quick Assembly of the Compliance Matrix in a Needle Insertion Context

Asynchronous preconditioner : The fully integrated volumes require assembling their compliance matrix $\mathbf{W}_v$ accordingly to Equation 3, which in turn requires the inversion of their system matrix $\mathbf{A}_v$. Courtecuisse et al. [25] propose to do this computation in a separate thread, to be updated at lower rates from an asynchronous pre-conditioner $\mathbf{P}$. If $\mathbf{A}_{v,t}$ is the system matrix computed at a time step t, $\mathbf{P}$ is built as a factorization of $\mathbf{A}_{v,t}^{-1}$:

$$\mathbf{P} = \mathbf{A}_{v,t}^{-1} = (\mathbf{L}\mathbf{D}\mathbf{L}^{\mathrm{T}})^{-1} \tag{4}$$

with $\mathbf{D}$ and $\mathbf{L}$ a diagonal matrix and an inferior triangular matrix, respectively. This factorization generally takes a few simulation time steps, during which the previously available system matrix is used in the simula-

tion and considered constant. The assumption being that between consecutive time steps, $\mathbf{A}_v$ is subjected to small variations, making $\mathbf{P}$ a good approximation of $\mathbf{A}_v^{-1}$ in the few steps following its computation. This is particularly relevant in the case of needle insertion, where liver and skin deformations mainly occur locally around the needle shaft. Once the new factorization is available in the asynchronous thread, it is used in the simulation for the following computations of $\mathbf{W}_v$:

$$\mathbf{W}_v = \mathbf{H}_v(\mathbf{L}\mathbf{D}\mathbf{L}^{\mathrm{T}})^{-1}{\mathbf{H}_v}^{\mathrm{T}} \tag{5}$$

IsoDOFs : The needle insertion context allows to fully exploit the idea behind the *IsoDOFs* presented in [29]. Indeed, because contacts occur locally, the sparsity of $\mathbf{H}$ is enhanced, which is the key optimization from their paper. Furthermore, the needle's constraint points are generated on the surfaces of the organs' tetrahedral elements, and those remain until the needle withdrawal. Effectively constraining the same DOFs during the whole insertion as illustrated in Figure 3. [29] proposes a new formulation for the constraint Jacobian matrix :

$$\mathbf{H}_v = \hat{\mathbf{H}}_v\bar{\mathbf{I}}_v \tag{6}$$

where $\bar{\mathbf{I}}_v$ is a partial identity matrix selecting the DOFs impacted by the constraints (which the authors called *IsoDOFs*), and $\hat{\mathbf{H}}_v$ contains the contact directions applied to the *IsoDOFs*. The preservation of IsoDOFs ensures minimal changes to $\bar{\mathbf{I}}_v$ during the needle insertion, allowing for the efficient computation of the product $\mathbf{L}^{-1}{\bar{\mathbf{I}}_v}^{\mathrm{T}}$ (see [29]). Coupled with the asynchronous pre-conditioner, it results in:

$$\mathbf{W}_v = \hat{\mathbf{H}}_v\bar{\mathbf{I}}_v(\mathbf{L}\mathbf{D}\mathbf{L}^{\mathrm{T}})^{-1}{\bar{\mathbf{I}}_v}^{\mathrm{T}}{\hat{\mathbf{H}}_v}^{\mathrm{T}} \tag{7}$$

The IsoDOFs method simplifies $\mathbf{W}_v$ into the product of a small matrix ($\hat{\mathbf{H}}_v$) with a term that either grows or shrinks steadily ($\bar{\mathbf{I}}_v$), significantly reducing the overall computational cost.

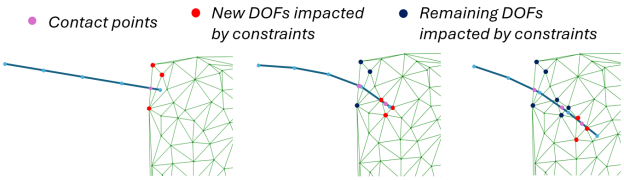

Figure 3: The needle path being constant during insertion, previously impacted DOFs calculation can be reused until needle withdrawal, during which IsoDOFs will be simply removed of $\bar{\mathbf{I}}_v$.

4.3 Haptic Rendering

In the simulation, the haptic device's end-effector is modeled as a point. Thus, needle motions have to be expressed from that point, and forces applied along the needle shaft have to be transferred to that point.

A virtual spring relates the interface's end-effector (in the real world) to the needle's base (in the simulation). As the user moves the interface, the needle follows its motion, according to the defined stiffness of the spring, effectively used as a penalty force. Since the interface has no obstacle in the real world, the spring prevents the needle from clipping through organs if an impossible movement is made. The end-effector's position is fed to the simulation during the synchronization phase. A Gauss-Seidel solver recovers the forces λ and provides the $\mathbf{W}$ matrix to the haptic thread.

In the haptic thread, $\mathbf{W}$ has all the mechanical information required to update the λ at high rates while taking into account the interface movements. This is done with another Gauss-Seidel solver, but in the haptic

thread. However, the interaction forces λ are expressed at the n_c constraint points of the needle; thus, they have to be transferred to the needle base by a transformation matrix:

$$\mathbf{H}_{n,base} = \mathbf{H}_n \mathbf{H}_{base} \tag{8}$$

The $\mathbf{H}_n$ matrix is provided at a simulation rate and relates constraint points to the needle's DOFs. $\mathbf{H}_{\text{base}}$ related the latter to the needle's base. In practice, although the needle is modeled as a deformable object, the curvatures that occur during insertion are generally small. It is therefore reasonable to assume that these deformations have only a limited influence on the transmission of forces to the base. Under this assumption, $\mathbf{H}_{base}$ is precomputed from the initial configuration in which the needle is considered straight, and to keep it constant throughout the simulation (see Figure 4). This approximation allows transferring forces to the base without accounting for torques, which is consistent with the limitations of the haptic interface. Indeed, the latter only accounts for 3 Degrees of Force Feedback (DOFFs), neglecting torque.

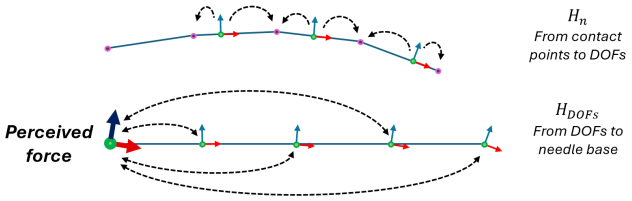

Figure 4: Forces computed on contact points using the deformable model are propagated to the DOFs through the $\mathbf{H}_n$ matrix updated at low-rate. Then are propagated to the needle base as if the needle was straight the whole time, allowing to precompute $\mathbf{H}_{base}$.

4.4 Visual Rendering

In order to maximize visual immersion of the user, VR is used to display a 3D scene of the complete Operating Room environment as shown in Figure 5. The first challenge of VR is the calibration between what is seen and what is simulated. This issue is addressed by using the position of one controller as a reference point to render the scene from. Placed next to the haptic interface, the integrated tracking system helps locate the haptic interface in the VR world. Then, to lighten the computational pressure, everything in the scene is fixed, apart from the organs and the needle. The simulation only sends mechanical DOFs $\mathbf{p}_m$ through the network, which have to be mapped to the visual DOFs $\mathbf{p}_v$ to be rendered. This mapping is done using a precomputed matrix $\mathbf{J}$, following this simple calculation : $\mathbf{p}_v = \mathbf{J}\mathbf{p}_m$. Since $\mathbf{p}_m$ is much smaller than $\mathbf{p}_v$, this allows to greatly reduce the bandwidth required to transfer positional data, thus reducing the network overhead. Furthermore, the rendering thread runs independently of the main thread so as to provide a smooth refresh rate, even if the simulation does not meet the 60 Hz requirement. This is important to avoid motion sickness since a VR headset is used instead of a simple screen. The rendering thread is storing data until fresh ones arrive.

A C-arm is also modeled in the VR scene to provide the user with visual guidance through 2D fluoroscopic images. Images are produced by a method inspired by [36], where rays are shot per pixel from the C-arm. The depth traveled in each organ is measured to compute the absorption of each ray, following the Beer-Lambert law. This absorption level is then mapped to a grayscale value, which produces the final image, which is then rendered to a screen in the VR environment as shown in Figure 5.

Figure 5: Inside the simulator : The user moving the C-arm (left). The OR environment with the fluoroscopic imagery on the right screen, and the patch of skin at the bottom. (right)

5 Results and Discussions

This section evaluates the computational performance of the simulator and the generated force rendering. As illustrated in Figure 2, the core of the simulation runs on Sofa [37] on one machine, while the visual rendering is performed on a separate machine by Unity [38]. The two machines communicate through a network using ROS [39] to update the visual models of both the organs and the needle. Assist [40], an open-sourced framework, helps coordinate everything by handling all the data transfer between all threads and the associated synchronization. Results are measured on the simulation computer, using a Ryzen 9 5900x 12-core processor × 24 (CPU), 32.0 Gio (RAM), and NVIDIA Corporation GA104 [GeForce RTX 3070 Ti] as GPU. Section 5.1 presents the general performance of the complete pipeline in comparison with the required update rates for both visual and haptic rendering smoothness. Then, Section 5.2 will evaluate and discuss the novel haptic rendering model.

5.1 Computation Time : General Performance

This section presents the general performance of the complete simulator pipeline, assessing the simulation, visual rendering, and haptic loops. A needle insertion simulation is performed with a recorded motion of the needle (initiated with the haptic interface) so as to be able to reproduce the same movement several times. The number of nodes composing each mesh is exposed in Table 1.

Table 1: Number of node per meshes.

Meshes	# nodes
Needle	11
Diaphragm	60
Liver	1197
Skin	142
Large intestine*	16
Small intestine	60
Stomach-Oesophagus*	7

* reduced with the MOR method discussed in Section 3

The general performance of the complete simulator was measured and averaged over 100 time steps during the numerical needle insertion. Results are presented in Table 2 and gather simulation steps, data communication with ROS, and the visual and haptic update rates. Although the simulation is complex, including several deformable models in contact with each other, one simulation step is, on average, close to what is required to reach real-time for a smooth visual rendering. The resulting update rate in the VR rendering system also reaches real-time performance, allowing the user to recover a fluid visual display. The haptic loop also achieves real-time performance in terms of haptic fluidity. Data communication with ROS is not computationally costly when considering the size of the chosen meshes. A study could be performed in future works to assess the limit number of node positions to be transmitted above which data communication speed becomes insufficient.

Table 2: Measured performance over 100 time steps for the simulation (first column), visual rendering (middle column), and haptic (last column) sides.

Simulation computer		**Visual rendering computer**		**Haptic rendering**	
Simulation frequency	44 Hz	Visual display frequency	48 - 72 Hz	Update rates	1021 Hz
Sending data with ROS	0.52 ms	Receiving data from ROS	0.49 ms		

5.2 Evaluation of the Haptic Rendering

This section provides an evaluation of the haptic rendering as exposed in Section 4.3. The aim of this method is to provide a force rendering that is close to what would be recovered by updating $\mathbf{H}_n$ at each step of the high-rate haptic loop, in a shorter amount of time. To assess the validity of the asynchronous design, the force rendering of 3 different implementations were compared :

- **Low-rate** : Forces are computed at low-rates at the end of a simulation step.
- **Mixed (our model)** : The computation of the constraint violations and the resulting force is done at high-rate while mechanical information ($\mathbf{A}_i$, $\mathbf{H}_i$, $\mathbf{W}$) are updated at low-rate.
- **High-rate** : Similar to the low-rate approach, but running at higher frequency with more (simulated) haptic positions. Representing ground truth.

To properly compare those implementations, the organization of the simulation and haptic loops were replicated in a synchronous series of steps, while still using the asynchronous preconditioner and *IsoDOFs* methods to build $\mathbf{A}_v$ and $\mathbf{W}_v$. Indeed, separating the haptic thread from the simulation could introduce errors in result comparisons and interpretation depending on the speed of the threads.

The trajectory in the mixed and high-rate implementation is defined as an interpolation of what is considered for the low-rate one. Indeed, the needle insertion simulation pipeline in the large-scale scenario involving organs was running at about 44 Hz, while the haptic thread was running at 1021 Hz. Being about 25 times faster than the simulation, the needle haptic trajectory interpolates the simulation motion with 25 times the number of points for both the mixed and high-rate implementations. For the latter, the time step needs to be reduced by a factor of 25.

As explained in Section 4.3, the mapping $\mathbf{H}_{base}$ (relating the needle DOFs and base) is considered constant throughout the entire scenario. While this could introduce errors in constraint violation calculation, in a real scenario the needle should never bend by a large amount, according to clinicians. The transfer of forces, mainly torques, to the needle base is also impacted by this approximation, but since a 3-Degrees-of-Force-Feedback (DOFFs) haptic interface is used, this is not an issue. This approximation should be reconsidered in case a 6 DOFFs interface were to be used.

The test scenario used for the measurements is split into 4 sections : the straight insertion of the needle in a cube, the bending of the needle, its release, and its withdrawal. This simple scenario aims at introducing minimal bias from other simulation phenomena and is defined numerically. The strength of the force applied to the base of the needle was measured for each section in each implementation. The results are shown in Figure 6; each section of the trajectory has its own plot, and each implementation has its own color. It can be observed that the proposed implementation closely follows the ground truth, and while it is not a perfect overlap, it is as good as state-of-the-art implementation. Zooming on the force profile shows a stair-like shape on the proposed model since mechanics, particularly $\mathbf{H}_n$, are constant between simulation updates. However, the observed shape is not perceived with the haptic interface as steps remain small.

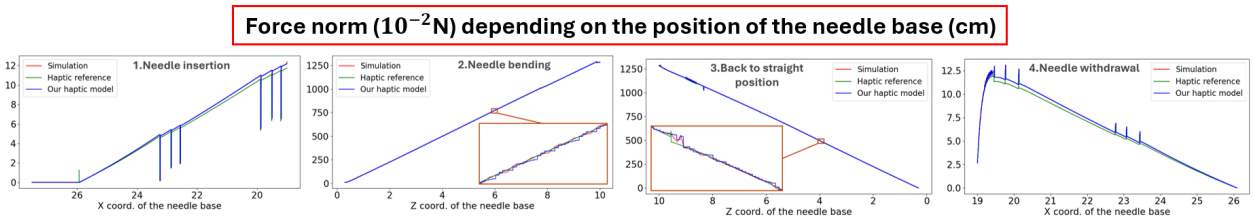

Figure 6: Comparison of the norm of the force rendering for the 3 scenarios, in the 4 sections of the needle trajectory.

Table 3: Average computation time over 10 sec. to update force rendering in the 3 implementations.

Scenario	Force computation time	Time step
Low-rate full step	7.78 ms	10 ms
High-rate full step	9.01 ms	0.4 ms
Our haptic force update	0.0075 ms	10 ms

Now, let's review the computation time benefit of the proposed method. Since the low and high rate implementations require the whole simulation to complete before computing forces, the time for the full simulation step is measured. For the proposed method, the time required to perform the force calculation was measured and is presented in Section 4.3. The considered computation performance was averaged over 10 seconds, at the end of the needle insertion, before the bending section, results are exposed in Table 3.

The low-rate implementation reaches real-time performance, while being however not efficient enough for haptics, even in such a simple scene, highlighting the need for a specific treatment for haptic rendering. The performance of the ground truth is similar to the low-rate one, since the performed computations are identical, thus remains insufficient for haptic rates. Meanwhile, the proposed model, consisting of few steps, is fast enough to reach haptic rates, while providing a force that is close to the chosen ground truth.

6 Conclusion

In this paper, a unified framework of a complete needle insertion training simulator was presented. It combines a numerical simulation loop performing the chosen scenario, as well as haptic rendering to provide the user with force sensation, and a novel visual rendering loop to immerse the trainee within a complete virtual operating room. The different loops share information independently of the way they were computed, making the simulator highly modular and flexible. The proposed coupling of the simulation and rendering systems also allows splitting the computational burden between the different entities, enabling the latter to fully exploit the local computational resources. The overall pipeline reaches performance close to real-time, allowing user interaction through the haptic interface. Such performance was reached in a large-scale scenario involving multiple colliding organs with breathing motions to reproduce a realistic scenario. The simulation itself performs a unified mechanical resolution of heterogeneous models. Moreover, the pipeline satisfies some recommendations from clinicians regarding the flexibility of the simulator (realistic, permissive, and immersive). Eventually, a force rendering model was presented, which allows reaching high haptic update rates while providing a force that closely follows a slower ground truth model where mechanical information is updated at each time step.

Further pipeline optimizations can still be performed to reach greater performance so as to improve the stability of the overall system. Indeed, tasks can be parallelized on the simulation side (e.g. displaying data and sending information with ROS while the simulation is running for a new time step) to increase the update rate instead of sequentially performing all the software tasks (including, but not limited to the simulation

itself). The simulated scenario could also be improved to match the reality more closely by considering ultrasound or 3D images from a CT scan instead of 2D fluoroscopic images as a guiding system and starting the procedure with trajectory planning.

Future work includes considering expert feedback in the simulator in order to improve the scenario of the biopsy and RFA procedures. Metrics should also be defined to quantify the performance of the user and provide constructive feedback, as expected from a training tool. Regarding the simulation, numerical methods could be developed to improve performance and stability while using fine meshes and focusing on high-frequency updates of contact information to recover a finer haptic rendering.

References

[1] F. G. Hamza-Lup, K. Bergeron, and D. Newton, "Haptic systems in user interfaces – state of the art survey." Association for Computing Machinery, Inc, 4 2019, pp. 141–148.

[2] C. D. Vece, C. Luciano, and E. D. Momi, "Psychomotor skills development for veress needle placement using a virtual reality and haptics-based simulator," *International Journal of Comp.Assisted Radiology and Surgery*, vol. 16, pp. 639–647, 4 2021.

[3] C. G. Corrêa, F. L. Nunes, E. Ranzini, R. Nakamura, and R. Tori, "Haptic interaction for needle insertion training in medical applications: The state-of-the-art," *Med. Engineering and Physics*, vol. 63, pp. 6–25, 1 2019.

[4] H. Wu, C. Chen, Y. Zhou, J. Wang, and Y. Xie, "VR-based haptic simulation for dynamic needle insertion," in *2019 IEEE/ASME International Conf. on Advanced Intelligent Mechatronics (AIM)*, 2019, pp. 924–929.

[5] C. Sutherland, K. Hashtrudi-Zaad, R. Sellens, P. Abolmaesumi, and P. Mousavi, "An augmented reality haptic training simulator for spinal needle procedures," *IEEE Trans. on Biomedical Engineering*, vol. 60, pp. 3009–3018.

[6] M. A. Alamilla, C. Barnouin, R. Moreau, F. Zara, F. Jaillet, H. T. Redarce, and F. Coury, "A virtual reality and haptic simulator for ultrasound-guided needle insertion," *IEEE Trans. on Med. Robotics and Bionics*, vol. 4, pp. 634–645, 8 2022.

[7] M. Müller, B. Heidelberger, M. Hennix, and J. Ratcliff, "Position based dynamics," *Journal of Visual Communication and Image Representation*, vol. 18, pp. 109–118, 4 2007.

[8] S. Bouaziz, S. Martin, T. Liu, L. Kavan, and M. Pauly, "Projective dynamics," *ACM Trans. on Graphics (TOG)*, vol. 33, 7.

[9] M. Ly, J. Jouve, L. Boissieux, and F. Bertails-Descoubes, "Projective dynamics with dry frictional contact," *ACM Trans. on Graphics (TOG)*, vol. 39, 7.

[10] Q. Cheng, P. X. Liu, P. Lai, S. Xu, and Y. Zou, *Journal of healthcare engineering*.

[11] M. Jean, "The non-smooth contact dynamics method," *Comp.Methods in Applied Mechanics and Engineering*, vol. 177, pp. 235–257, 1999.

[12] Y. Renard, "Generalized newton's methods for the approximation and resolution of frictional contact problems in elasticity," *Comp.Methods in Applied Mechanics and Engineering*, vol. 256, pp. 38–55.

[13] C. Duriez, F. Dubois, A. Kheddar, and C. Andriot, "Realistic haptic rendering of interacting deformable objects in virtual environments," *IEEE Trans. on Visualization and Comp.Graphics*, vol. 12, pp. 36–47, 4.

[14] H. Courtecuisse and J. Allard, "Parallel dense gauss-seidel algorithm on many-core processors," *2009 11th IEEE International Conf. on High Performance Computing and Communications, HPCC 2009*, pp. 139–147, 2009.

[15] M. Macklin, K. Erleben, M. Müller, N. Chentanez, S. Jeschke, and V. Makoviychuk, "Non-smooth newton methods for deformable multi-body dynamics," *ACM Trans. on Graphics*, vol. 38, 10.

[16] M. Li, Z. A. H. Ferguson, T. Schneider, T. Langlois, D. Zorin, D. Panozzo, C. Jiang, and D. M. Kaufman, "Incremental potential contact: Intersection- and inversion-free, large-deformation dynamics," *ACM Trans. on Graphics*, vol. 39, 7 2020.

[17] C. Yang, Y. Xie, S. Liu, and D. Sun, "Force modeling, identification, and feedback control of robot-assisted needle insertion: A survey of the literature," *Sensors (Switzerland)*, vol. 18, 2018.

[18] R. J. Roesthuis, Y. R. van Veen, A. Jahya, and S. Misra, "Mechanics of needle-tissue interaction," *International Conf. on Intelligent Robots and Systems (IROS 2011)*, pp. 2557–2563, 9.

[19] 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. on Graphics*, vol. 28, p. 1, 7.

[20] L. Wang, D. Gao, J. Fu, Y. Luo, and S. Zhao, "Simulation of coupling process of flexible needle insertion into soft tissue based on abaqus," *Computers, Materials & Continua*, vol. 64, pp. 1153–1169, 6.

[21] C. Duriez, C. Guébert, M. Marchal, S. Cotin, and L. Grisoni, "Interactive simulation of flexible needle insertions based on constraint models," *Lecture Notes in Comp.Science*, vol. 5762 LNCS, pp. 291–299.

[22] Y. Adagolodjo, L. Goffin, M. D. Mathelin, and H. Courtecuisse, "Robotic insertion of flexible needle in deformable structures using inverse finite-element simulation," *IEEE Trans. on Robotics*, vol. 35, pp. 697–708, 2019.

[23] P. Perrusi, P. Baksic, and H. Courtecuisse, "Interactive finite element model of needle insertion and laceration," *EUROGRAPHICS 2021 (Short Paper)*, vol. l, pp. 1–4, 2021.

[24] C. Martin, Z. Zeng, and H. Courtecuisse, "Efficient Needle Insertion Simulation using Hybrid Constraint Solver and Isolated DOFs," in *Eurographics 2023 - Short Papers*, V. Babaei and M. Skouras, Eds. The Eurographics Association, 2023.

[25] H. Courtecuisse, Y. Adagolodjo, H. Delingette, and C. Duriez, "Haptic rendering of hyperelastic models with friction," *IEEE International Conf. on Intelligent Robots and Systems*, vol. 2015-Decem, pp. 591–596, 9 2015.

[26] I. Peterlik, C. Duriez, and S. Cotin, "Asynchronous haptic simulation of contacting deformable objects with variable stiffness," *IEEE/RSJ IROS*, pp. 2608–2613.

[27] A. M. Okamura, C. Simone, and M. D. O'Leary, "Force modeling for needle insertion into soft tissue," *IEEE Trans. on Biomedical Engineering*, vol. 51, pp. 1707–1716, 10 2004.

[28] IRCAD, "Liver segmentation 3D-IRCADb-01," https://www.ircad.fr/research/data-sets/liver-segmentation-3d-ircadb-01/, 2026-01-24.

[29] Z. Zeng, S. Cotin, and H. Courtecuisse, "Real-time FE simulation for large-scale problems using precondition-based contact resolution and isolated dofs constraints," *Comp.Graphics Forum*, 2022.

[30] H. Courtecuisse, J. Allard, P. Kerfriden, S. P. Bordas, S. Cotin, and C. Duriez, "Real-time simulation of contact and cutting of heterogeneous soft-tissues," *Med. Image Analysis*, vol. 18, pp. 394–410.

[31] O. Goury and C. Duriez, "Fast, generic, and reliable control and simulation of soft robots using model order reduction," *IEEE Trans. on Robotics*, vol. 34, pp. 1565–1576, 12.

[32] C. Felippa, "A systematic approach to the element-independent corotational dynamics of finite elements," *Center for Aerospace Structures*.

[33] I. Peterlík, M. Nouicer, C. Duriez, S. Cotin, and A. Kheddar, "Constraint-based haptic rendering of multirate compliant mechanisms," *IEEE Trans. on Haptics*, vol. 4, pp. 175–187, 2011.

[34] M. Petyt, *Theory of matrix structural analysis*. New York, McGraw-Hill [1967, ©1968].

[35] D. Ryckelynck, "A priori hyperreduction method: an adaptive approach," *Journal of Computational Physics*, vol. 202, pp. 346–366, 1 2005.

[36] F. Vidal, M. Garnier, N. Freud, J. Létang, and N. John.

[37] Sofa, "Simulation Open Framework Architecture," https://www.sofa-framework.org/, 2026-01-24.

[38] Unity, "Unity," https://unity.com/, 2026-01-24.

[39] ROS, "Robot Operating System," https://docs.ros.org/en/foxy/index.html, 2026-01-24.

[40] Assist, "Advanced Simulation and System Integration Software Toolbox," https://assist.cnrs.fr/, 2026-01-24.

Author Biographies

NOAH BERTHOLON is a Ph.D. candidate at the University of Strasbourg, he obtained his master degree at the Université Claude Bernard Lyon 1 in 2025. His research interests include stability in numerical simulation, GPU parallelism and real-time training simulation. His email address is nbertholon@unistra.fr
CLAIR MARTIN is a Ph.D. graduated from the University of Strasbourg in 2025 and received her M.Sc. degree in biomedical engineering from Telecom Physique Strasbourg, France. Her research focuses on numerical simulation for medical applications and haptics. Her email address is claire.martin2@etu.unistra.fr
ERIK PERNOD is an engineer at InfinyTech3D, graduated from the ISITIV in 2008. He started as research engineer at the French research institute Inria, Erik then worked as technical officer at Belcurves company and created in 2018 his own company InfinyTech3D. His work focus on mechanical and 3D numerical modeling through Virtual Reality application. His email address is erik.pernod@infinytech3d.com.
HADRIEN COURTECUISSE is a research scientist at CNRS. His research spans multi-physics modeling, medical simulation, robotics, and biomechanical registration, with a focus on integrating experimental data. During his Ph.D., he concentrated on GPU parallelization and real-time models for contact and interaction simulations. In 2013, he joined the CNRS robotics team in Strasbourg, where he applied his expertise in computer vision, robotics, and medical simulation to develop robotic surgical assistance systems. He obtained his HDR in 2021. His email address is hcourtecuisse@unistra.fr.