This paper appears in in Simulation Modelling Practice and Theory https://doi.org/10.1016/j.simpat.2026.103365

3D Just-in-Time Modelling of Real-Time Volumetric Tissue Deformation for Ultrasound-Guided Percutaneous Surgery

Kian Zalzalah[a,*], Carlos Rossa[a]

[a]*Department of Systems and Computer Engineering, Carleton University, 1125 Colonel By Drive, Ottawa, ON K1S 5B6, Canada*

Abstract

Real-time prediction of tissue deformation is essential for interactive ultrasound-guided percutaneous surgery simulation, but high-fidelity physics-based models often exceed practical online computational limits. This paper presents a 3D Just-in-Time (JIT) modelling framework for real-time prediction of volumetric tissue deformation during ultrasound-guided percutaneous procedures. An offline historical database of needle insertion poses and corresponding volumetric deformation fields is generated using a simulation engine. During runtime, the proposed method retrieves a gated neighbourhood of geometrically similar insertions, computes curvature-aware constrained interpolation weights, and predicts the deformation field in a reduced output space obtained through principal component analysis (PCA). The full volumetric displacement field is then reconstructed and used to deform a baseline ultrasound volume for slice extraction. Across the tested grid resolutions, the proposed PCA-JIT framework achieved millisecond-scale prediction times and is up to 90 times faster than the simulation engine used to generate the database. Prediction errors remained low across all tested mesh resolutions, and the predicted fields enabled deformation-consistent B-mode slice rendering from arbitrarily posed imaging planes. The proposed framework provides an accurate and computationally efficient surrogate for volumetric tissue deformation prediction and is well suited for interactive surgical simulation,

*Corresponding author

Email addresses: kianzalzalah@cmail.carleton.ca (Kian Zalzalah), carlosrossa@cunet.carleton.ca (Carlos Rossa)

ultrasound-based training, and future image-guided intervention systems.

Keywords: Just-in-time modelling, Ultrasound, Tissue deformation, Volumetric imaging, Surgical simulation, Principal component analysis

1. Introduction

Ultrasound-guided percutaneous surgery (usPS) uses thin instruments to reach targets in the tissue through small punctures in the skin. Procedures such as biopsy, brachytherapy, nephrolithotomy, and tumour ablation commonly rely on ultrasound as the primary imaging modality. Unlike computed tomography and magnetic resonance imaging, ultrasound is portable, radiation-free, widely accessible, and safe for vulnerable patients [1, 2]. Despite its benefits over other types of interventions, usPS has a steep learning curve and requires specialized training to achieve surgical proficiency. This is often attributed to the high level of coordination and dexterity needed to simultaneously image the tool while steering it towards a target, as well as tool-induced tissue and target deformation that shift anatomical structures in the images [3]. In nephrolithotomy, for example, the learning curve only plateaus after 60 interventions and surgical excellence requires more than 100 trials [4]. Such steep learning curves limit widespread adoption and increase the risk of complications. Ultrasound-guided intervention training, ultrasound image simulation, and real-time soft-tissue deformation remain active research areas. For example, recent work has explored mixed-reality support for ultrasound-guided procedures [5], reviewed emerging ultrasound simulation and image-generation methods for training and guidance [6], and investigated real-time compliant or constraint-based deformation models for interactive soft-tissue simulation [7].

Two-dimensional ultrasound (2D-US) is widely used in usPS because it provides B-mode image slices of a region of interest (ROI) and the surgical tool in longitudinal or transverse planes. In longitudinal imaging, the interventionist inserts the tool parallel to the imaging plane so that it appears as a line in the 2D images. In transverse imaging, the tool shaft crosses the imaging plane orthogonally and appears instead as a bright spot. To visualize and guide the tool tip toward the target, the interventionist must continuously sweep the ultrasound probe back and forth and map the volume by mentally combining planar images. In contrast, three-dimensional ultrasound (3D-US) scans a volume and provides panoramic views of the ROI, helping determine

if the tool is placed correctly in a target or located peripherally [8, 9]. 3D-US enables more intuitive recognition of the 3D spatial relationship between the tool and the target, and easier real-time adjustments of the tool path [10]. However, 3D-US has low frame rates and does not eliminate the learning curve associated with spatial coordination between the tool steering and imaging.

To overcome learning curves, usPS can strongly benefit from simulation training. Nowadays, physical, cyber-physical, or virtual simulators are a fundamental component of surgical training programs, allowing interventionists to practice and hone their skills in a risk-free environment [11, 12]. State-of-the-art simulators include dry labs, virtual reality, cadaver-based wet labs, and animal tissue training [13]. Robotic systems, such as the widely adopted da Vinci Surgical robot, primarily use dry lab simulations and built-in scoring systems to evaluate trainee performance [14]. In interventional radiology and image-guided surgery, trainers offering haptic and visual feedback have been shown to improve targeting precision and reduce procedural errors [15, 16]. Tissue and image deformation models can teach the interventionist how to minimize tissue damage, interpret US images, and ensure accurate tissue targeting [17, 18].

Cyber-physical usPS simulators often require accurate modelling of the tool and tissue interaction to update the resulting images in real-time [19, 14, 20]. Traditionally, these models involve detailed biomechanical models of tool-tissue interactions that often exceed real-time computational constraints on standard hardware, thereby limiting their usability in portable simulators [21, 22]. The biomechanical response of soft tissue to external forces, such as those caused by needle insertion and tissue cutting, has primarily been simulated using Finite Element Models (FEM). FEM relies on solving complex partial differential equations describing tissue deformation. It has been extensively used to simulate needle insertion, including for needle guidance and tissue trauma assessment in the liver [23, 24], for needle steering in brachytherapy [25], and general 2D and 3D image deformation [26]. While accurate, FEM models are computationally expensive and large models often require specialized hardware to run in real-time [27, 28, 29]. Furthermore, FEM models often struggle with non-linear tissue response, particularly with large deformations commonly seen in needle-based procedures. Despite advancements in computer power and optimized solvers, the use of real-time standard FEM simulation in a clinical setting and surgical simulators remains limited [30, 31]. A cyber-physical usPS simulator must update in

real-time the needle-induced tissue deformation and the corresponding ultrasound image feedback. Online solves, tool/tissue contact or interaction update, and volumetric image resampling step can exceed the visual update budget, particularly for dense 3D deformation fields on standard hardware. This motivates shifting deformation generation offline and replacing repeated online simulation with a fast surrogate prediction stage based on neighbour retrieval, constrained weighting, reduced-space prediction, and full-field reconstruction.

Several non-FEM and data-driven approaches have been proposed to improve the runtime suitability of tissue deformation simulation. Examples include position-based dynamics and mass-spring-type models for interactive surgical simulation, meshless methods for large deformation, and surrogate models trained from precomputed deformation data. Machine-learning surrogates have been used for real-time prediction of breast tissue deformation from FEM-generated data [32], Kriging-based metamodels have been applied to needle-insertion deformation prediction [33], and neural-network surrogates have been investigated for rapid visualization of tissue stress during physical examination training [34]. These approaches show that offline data can be used to reduce online computational cost, but they introduce trade-offs in terms of training requirements, interpretability, memory cost, and generalization outside the sampled data distribution. Simple nearest-neighbour or inverse-distance interpolation methods are fast and easy to implement, but they do not enforce local consistency constraints and may be less reliable in nonlinear regions of the pose space. Global learning-based surrogates can provide rapid online prediction after training, but they typically require fitting a global nonlinear mapping and may require architecture selection, hyperparameter tuning, and retraining when the deformation engine, tissue model, or database changes.

This paper proposes a data-driven alternative to computationally expensive simulation engines using Just-in-Time (JIT) modelling. JIT modelling is a regression approach that builds a local predictive model on-the-fly using only the training samples that best match the query point [35]. In the proposed algorithm, the data samples are precomputed deformations of a mesh representing the tissue, for a given needle insertion position in the mesh. To simulate mesh deformation for a new needle insertion query, the JIT algorithm retrieves a local neighbourhood of mesh deformations that best approach the needle insertion query and computes interpolation weights based on their geometric proximity using a curvature-aware local surrogate

model under unbiasedness and first-order consistency constraints. The resulting mesh deformation is then computed as a weighted recombination of the selected deformation samples, thereby achieving low latency prediction while preserving spatial coherence of the reconstructed deformation. Unlike traditional simulation engines, JIT performs online predictions using the precomputed data, dramatically reducing computation time while maintaining simulation accuracy. Any simulation engine can be used to build the database, allowing JIT to leverage existing models without relying on a specific solver.

While JIT is effective for moderate-sized datasets, extending it to 3D volumetric tissue deformation introduces a significant scalability challenge because each database sample corresponds to a dense displacement field over the full mesh. For a grid with $M \times N \times P$ nodes, each sample contains $3MNP$ displacement components, so both storage and online recombination costs grow cubically with spatial resolution.

For dense volumetric grids, direct full-field storage and online recombination become increasingly expensive because each database sample contains $3MNP$ displacement values. This motivates the use of PCA compression so that online interpolation can be performed in a reduced coefficient space before reconstructing the full volumetric displacement field. Surrogate modelling and output-space reduction have been explored in computational mechanics to accelerate prediction of high-dimensional simulation outputs while reducing online evaluation cost [36].

This paper addresses this limitation by integrating principal component analysis (PCA) with JIT and compressing each volumetric deformation sample into a low-dimensional coefficient vector prior to online prediction. JIT and PCA have previously been combined in other application domains, including adaptive monitoring of multivariate physiological time-series, where JIT is used to select relevant data samples and PCA is used to construct a local monitoring model and compute statistical indicators for abnormality detection [37]. This differs from the current algorithm, where PCA is not used to form a local monitoring model, but instead as an output-space compression strategy for dense volumetric displacement fields. The historical deformation fields are projected once into a retained PCA basis, JIT interpolation is performed on the resulting coefficient vectors, and the full displacement field is reconstructed only after the reduced-space prediction is complete. Online interpolation is therefore performed directly in the reduced coefficient space, after which the predicted coefficients are back-projected to reconstruct the

full displacement field. This substantially lowers the computational burden of online prediction while preserving the dominant spatial deformation structure inherited from the selected reference deformation engine. Consequently, the runtime stage reduces to neighbour retrieval, constrained weight computation, and reduced-space coefficient prediction followed by full-field reconstruction, enabling real-time update rates for volumetric tissue deformation prediction.

To the best of our knowledge, this work is the first to apply Just-in-Time (JIT) modelling to real-time prediction of volumetric tissue deformation.

The main contributions of this work are summarized as follows:

- A 3D PCA-JIT framework is introduced for real-time prediction of dense volumetric tissue deformation during ultrasound-guided percutaneous needle insertion.
- PCA-based output compression is integrated with JIT modelling so that online interpolation is performed in a reduced coefficient space rather than directly over full volumetric displacement fields.
- A curvature-aware constrained interpolation strategy is developed to compute local JIT weights using geometric neighbour selection, unbiasedness and first-order consistency constraints, and Hessian-based curvature regularization.

The proposed framework is evaluated against the selected PHS reference deformation engine and a FULL-JIT ablation to quantify prediction accuracy, runtime, scalability, and the effect of PCA output compression. The remainder of this paper is organized as follows. Section 2 presents the proposed 3D-JIT framework, including the offline PCA compression and effective curvature training, the online neighbourhood selection strategy, and the curvature-aware constrained weight computation used for real-time volumetric field prediction. Section 3 describes the offline historical database construction using a 3D polyharmonic spline (PHS) simulation engine and the pose sampling strategy used to provide structured coverage of the insertion space. Section 4 defines the evaluation criteria and error metrics used to compare 3D-JIT predictions against the PHS reference deformation, and Section 5 reports simulation results including accuracy, runtime, and speedup across volumetric grid resolutions. Section 6 then details how mesh deformations are converted into an ultrasound volume and sliced to produce B-mode

images from the deformed volume. Section 7 discusses the main limitations of the proposed framework, and Section 8 concludes the paper and outlines future directions.

2. 3D Just-in-Time Modelling

The objective of the proposed 3D-JIT framework is to predict in real-time a dense volumetric displacement field for ultrasound volume deformation and slice rendering. A needle-tissue interaction simulator can accurately generate such fields, but often not at real-time rates. JIT addresses this limitation by leveraging a historical dataset of deformations generated offline, retrieving a neighbourhood of similar needle poses online, and computing interpolation weights under consistency constraints to synthesize the deformation corresponding to the current query.

At runtime, the proposed model solves the real-time prediction through a sequence of lightweight operations. The current needle pose is first used to retrieve geometrically similar historical insertions. Curvature-aware constrained weights are then computed from the selected neighbours, and the weighted prediction is performed in the reduced PCA coefficient space. The predicted coefficients are finally back-projected to reconstruct the full 3D displacement field, which is then used for ultrasound volume deformation and slice extraction. Thus, the expensive deformation generation is performed offline, while the online stage only requires retrieval, constrained weighting, reduced-space interpolation, and reconstruction.

Consider a tissue volume discretized as a regular grid with $M \times N \times P$ nodes, where $i = 1, \ldots, M$, $k = 1, \ldots, N$, and $\ell = 1, \ldots, P$. Let $\mathbf{G} \in \mathbb{R}^{M \times N \times P \times 3}$ denote the undeformed reference mesh coordinates, with

$$\mathbf{G}(i, k, \ell, p) = \begin{cases} x_i, & p = 1, \\ y_k, & p = 2, \\ z_\ell, & p = 3. \end{cases} \tag{1}$$

For each deformation sample j resulting from a needle insertion, the simulation engine returns a displacement tensor $\mathbf{G}_j \in \mathbb{R}^{M \times N \times P \times 3}$, where $\mathbf{G}_j(i, k, \ell, p)$ gives the displacement of node (i, k, ℓ) along Cartesian direction p. Thus, each tensor $\mathbf{G}_j$ represents the volumetric displacement field associated with insertion sample j, defined with respect to the undeformed reference mesh $\mathbf{G}$.

For simplicity, a flattened displacement vector is defined as

$$\mathbf{g}_j = \mathrm{vec}(\mathbf{G}_j) \in \mathbb{R}^{3MNP}, \tag{2}$$

where the operator $\mathrm{vec}(\cdot)$ stacks all nodal displacement components into a column vector.

Each mesh deformation results from inserting a needle at coordinates $\mathbf{n}_s = [x_s \ y_s \ z_s]^T$, and advancing it to a tip position $\mathbf{n}_t = [x_t \ y_t \ z_t]^T$. We define the 6D needle pose vector as

$$\mathbf{x}_j = [x_{s,j} \quad y_{s,j} \quad z_{s,j} \quad x_{t,j} \quad y_{t,j} \quad z_{t,j}]^T \in \mathbb{R}^6. \tag{3}$$

Assume that a simulation engine creates an offline database of needle pose vectors and the resulting deformation, containing N_h samples. The database is defined as:

$$\mathcal{S} = \{(\mathbf{x}_j, \mathbf{g}_j)\}_{j=1}^{N_h}. \tag{4}$$

For a new needle pose query $\mathbf{x}_0$, the JIT predictor selects the N_{opt} database samples whose needle poses are closest to $\mathbf{x}_0$. The resulting mesh deformation is then predicted as the affine weighted combination

$$\hat{\mathbf{g}}(\mathbf{x}_0) = \sum_{q=1}^{N_{\text{opt}}} (w_q \mathbf{g}_q) \ \in \mathbb{R}^{3MNP}, \tag{5}$$

where w_q are the interpolation weights associated with the q^{th} selected neighbour, to be determined later in Subsections 2.1 and 2.3 such that

$$\sum_{q=1}^{N_{\text{opt}}} w_q = 1,$$

and $\mathbf{g}_q$ denotes the corresponding displacement field, locally re-indexed from the historical database as defined in (2). While the full historical database is indexed by $j = 1, \ldots, N_h$, the selected neighbourhood is locally re-indexed by $q = 1, \ldots, N_{\text{opt}}$.

For large meshes, directly evaluating (5) is computationally expensive because the database $\mathcal{S}$ has $3 \times M \times N \times P \times N_h$ scalar values. This motivates integrating PCA to compress the database samples $\mathbf{g}_j$ into a lower dimensional space, performing the weighted combination above on this reduced space, before reconstructing the weighted mesh. This process is explained in detail in the next subsection, followed by a description of how the optimal number of neighbours N_{opt} and weights w_q are calculated.

2.1. Offline Compression and Local Curvature

Each database sample has dimension $3MNP$, which makes online regression impractical. We therefore compress volumetric deformation fields offline into a low-dimensional coefficient space using PCA, and then estimate an effective curvature matrix, where second-order information about the pose-to-deformation mapping is used to regularize neighbour weights.

PCA seeks a rank- k subspace that captures the dominant variability in the historical deformation fields. Let $\mathbf{g}_j \in \mathbb{R}^{3MNP}$ denote the vectorized 3D displacement field of the j^{th} database sample, where MNP is the number of grid nodes and the factor 3 corresponds to the three Cartesian displacement components. The mean deformation vector is $\boldsymbol{\mu} \in \mathbb{R}^{3MNP}$, and $\mathbf{V} \in \mathbb{R}^{3MNP\times k}$ is the retained PCA basis whose columns are the eigenvectors associated with the k largest eigenvalues of the output covariance matrix. Each database output is encoded by its PCA coefficient vector $\mathbf{c}_j \in \mathbb{R}^k$:

$$\mathbf{c}_j = (\mathbf{V})^T (\mathbf{g}_j - \boldsymbol{\mu}) \in \mathbb{R}^k, \tag{6}$$

so that each mesh may be approximated by

$$\mathbf{g}_j \approx \boldsymbol{\mu} + \mathbf{V}\mathbf{c}_j. \tag{7}$$

For a new query pose $\mathbf{x}_0$, the interpolation weights $\mathbf{w}$ are computed from neighbouring database samples with similar needle poses, but the weighted recombination is performed in the compressed PCA space rather than directly on the full mesh. Let $\mathbf{c}_q$ denote the PCA coefficient vector of the q^{th} selected neighbour. The predicted PCA coefficients for the new query are obtained as:

$$\hat{\mathbf{c}}(\mathbf{x}_0) = \sum_{q=1}^{N_{\text{opt}}} w_q \, \mathbf{c}_q \in \mathbb{R}^k. \tag{8}$$

The full mesh corresponding to the query is then reconstructed using PCA back-projection as

$$\hat{\mathbf{g}}(\mathbf{x}_0) = \boldsymbol{\mu} + \mathbf{V}\,\hat{\mathbf{c}}(\mathbf{x}_0) \in \mathbb{R}^{3MNP}. \tag{9}$$

In this formulation, the online weighted combination is carried out over N_{opt} vectors in $\mathbb{R}^k$ rather than over N_{opt} full deformation vectors in $\mathbb{R}^{3MNP}$, which substantially reduces the computational cost of online prediction. Finally, the volumetric displacement tensor is obtained via a reshape operator $\mathcal{R}_{M,N,P}(\cdot)$:

$$\hat{\mathbf{G}}(\mathbf{x}_0) = \mathcal{R}_{M,N,P}\big(\hat{\mathbf{g}}(\mathbf{x}_0)\big) \in \mathbb{R}^{M\times N\times P\times 3}. \tag{10}$$

Algorithm 1 Offline Training: PCA Compression and Smoothed Local Hessians

Input: Historical set $\mathcal{S}$ in (4), PCA rank k, curvature-training parameters
Outputs: PCA model $(\boldsymbol{\mu}, \mathbf{V})$, coefficients $\{\mathbf{c}_j\}$, stored Hessians $\{\mathbf{H}_j\}$
Train PCA on $\{\mathbf{g}_j\}$ to obtain $\boldsymbol{\mu}$ and $\mathbf{V}$
Compute $\mathbf{c}_j$ for all samples using (6)
for $j = 1, \ldots, N_h$ **do**
  Fit local quadratic models for $\{c_m(\mathbf{x})\}_{m=1}^{k}$ around $\mathbf{x}_j$
  Compute $\widetilde{\mathbf{H}}_j^{(m)}$ using (14)
  Form raw Hessian $\widehat{\mathbf{H}}_j$ using (15)
  Smooth neighbouring raw Hessians to obtain $\mathbf{H}_j$ using (16)
end for
Return $(\boldsymbol{\mu}, \mathbf{V})$, $\{\mathbf{c}_j\}$, and $\{\mathbf{H}_j\}$

For a given query pose $\mathbf{x}_0$, the remaining task is to determine the neighbourhood and interpolation weights used in (8).

2.2. Online Optimal Neighbourhood Selection

For a query pose $\mathbf{x}_0$, candidate database samples are first filtered using geometric gating constraints, after which the N_{opt} nearest neighbours are selected. The purpose of this gating stage is to ensure that retrieved neighbours correspond to geometrically similar insertions, entry location, and overall pose consistency with the structured database sampling.

Because tissue deformation depends strongly on how far and where the needle is inserted, we first require each candidate neighbour to have a tip position similar to that of the query:

$$\sqrt{(x_{t,0} - x_{t,j})^2 + (y_{t,0} - y_{t,j})^2 + (z_{t,0} - z_{t,j})^2} \leq \gamma_{\text{in}}, \tag{11}$$

where γ_{in} is a predefined insertion gate threshold. We also require candidate neighbours to originate from a similar entry location in the mesh by enforcing

$$\sqrt{(x_{s,j} - x_{s,0})^2 + (y_{s,j} - y_{s,0})^2 + (z_{s,j} - z_{s,0})^2} \leq \gamma_{\text{en}}, \tag{12}$$

where γ_{en} is another predefined entry gate threshold. This constraint prevents selecting neighbours whose deformation fields are spatially shifted to a different region of the volume.

For the reported experiments, the gating thresholds are

$$\gamma_{\text{in}} = 0.05, \qquad \gamma_{\text{en}} = 0.15,$$

for all reported grid resolutions.

Let $\mathcal{F}(\mathbf{x}_0) \subseteq \{1, \ldots, N_h\}$ denote the set of database indices that satisfy the gating constraints in (11) and (12). Among these feasible samples, similarity to the query is ranked using Euclidean distance in the 6D pose space:

$$d_j^2 = \|\mathbf{x}_j - \mathbf{x}_0\|_2^2, \qquad j \in \mathcal{F}(\mathbf{x}_0). \tag{13}$$

The N_{opt} feasible samples with the smallest values of d_j are then selected and locally re-indexed as $q = 1, \ldots, N_{\text{opt}}$, as required in (8). If fewer than N_{opt} samples satisfy the gating constraints, the nearest feasible samples are used, or the gate thresholds may be relaxed in implementation to ensure a sufficient neighbourhood size.

In the reported experiments, the neighbourhood size is fixed at $N_{\text{opt}} = 40$. This value has been selected through preliminary neighbourhood-size tuning using held-out query poses from the admissible pose range. Smaller neighbourhoods produce more local predictions but can be more sensitive to database spacing and constrained-weight stability, whereas larger neighbourhoods increase the online cost of the KKT weighting step and PCA coefficient interpolation. Therefore, $N_{\text{opt}} = 40$ is used as a compromise between local accuracy, weight stability, and online runtime.

2.3. Curvature-Aware Regularized Weights

Once the neighbours have been selected, their interpolation weights $\mathbf{w} = [w_1, w_2, \ldots, w_{N_{\text{opt}}}]^T \in \mathbb{R}^{N_{\text{opt}} \times 1}$ must be computed before they are combined in the local linear predictor. The weights are constrained to satisfy unbiasedness and first-order consistency. Here, first-order consistency means that the weighted neighbour offsets around the query cancel, so that any locally affine dependence of the output on the pose vector is reproduced exactly at the query. In practice, the unbiasedness constraint preserves constant outputs, while the first-order constraint removes linear bias with respect to the local pose offsets.

The purpose of the Hessian term is to identify where the local affine assumption is less reliable. It estimates how rapidly the deformation response

changes as the needle pose changes. When the pose-to-deformation relationship is nearly linear, neighbouring samples can be combined reliably using affine interpolation. However, near localized deformation regions, such as the needle tip or shaft, two neighbours that are similarly close to the query in pose space may not be equally reliable. A neighbour located in a region with stronger second-order variation can introduce larger interpolation error if it is weighted too strongly. The curvature-aware regularization reduces the influence of neighbours whose stored Hessian predicts stronger local nonlinear variation relative to the query.

To implement this idea, we introduce a curvature-aware regularization based on stored second-order descriptors of the local pose-to-PCA-coefficient mapping. Specifically, Hessian descriptors are computed from the second-order variation of the retained PCA coefficients with respect to changes in the needle pose. The resulting weights therefore depend not only on geometric proximity to the query, but also on a local estimate of nonlinearity in the pose-to-deformation map. Because the full output dimension $3MNP$ is too large for direct second-order modelling, curvature is estimated in the retained PCA coefficient space. In this reduced space, the local dependence of each coefficient on the 6D needle pose can be approximated by a quadratic model, from which sample-specific Hessian matrices are obtained and stored offline.

For the nonlinear mapping from the 6D needle pose $\mathbf{x} \in \mathbb{R}^6$ to the reduced PCA coefficient vector $\mathbf{c}(\mathbf{x}) \in \mathbb{R}^k$, a local quadratic model is fitted offline around each database insertion. Let $\mathbf{x}_j \in \mathbb{R}^6$ denote the pose of the j^{th} database sample, $j = 1, \ldots, N_h$. For each retained PCA coefficient $c_m(\mathbf{x})$, $m = 1, \ldots, k$, this local quadratic fit yields a coefficient-wise Hessian matrix

$$\widetilde{\mathbf{H}}_j^{(m)} = \left[\frac{\partial^2 c_m(\mathbf{x})}{\partial x_a \, \partial x_b} \right]_{\mathbf{x}=\mathbf{x}_j,\, a,b=1}^{6} \in \mathbb{R}^{6\times 6}. \tag{14}$$

Thus, each database insertion is initially associated with k Hessian matrices in the reduced output space. These are first combined into a single raw insertion-level Hessian descriptor,

$$\widehat{\mathbf{H}}_j = \sum_{m=1}^{k} \alpha_m \, \widetilde{\mathbf{H}}_j^{(m)}, \tag{15}$$

where the aggregation weights are chosen proportional to the empirical variance of each retained PCA coefficient across the historical database. Hence,

coefficients associated with larger database variability contribute more strongly to the raw insertion-level curvature descriptor.

To improve robustness, these raw Hessian descriptors are then smoothed offline over the database pose space. For each database insertion j, let $\mathcal{N}_H(\mathbf{x}_j) \subseteq \{1, \ldots, N_h\}$ denote the set of database indices in the local Hessian neighbourhood around $\mathbf{x}_j$. Then, for each $\ell \in \mathcal{N}_H(\mathbf{x}_j)$, the index ℓ refers to a neighbouring database insertion included in the smoothing process. The stored Hessian associated with insertion j is defined by Gaussian averaging of neighbouring raw Hessians,

$$\mathbf{H}_j = \frac{\sum\limits_{\ell \in \mathcal{N}_H(\mathbf{x}_j)} \omega_{j\ell}\, \widehat{\mathbf{H}}_\ell}{\sum\limits_{\ell \in \mathcal{N}_H(\mathbf{x}_j)} \omega_{j\ell}}, \tag{16}$$

over $\ell \in \mathcal{N}_H(\mathbf{x}_j)$ where the weights depend on similarity in pose space according to:

$$\omega_{j\ell} = \exp\left(-\frac{\|\mathbf{x}_\ell - \mathbf{x}_j\|_2^2}{h_H^2}\right), \qquad h_H > 0, \tag{17}$$

where, h_H is selected directly from the local spacing of the normalized 6D pose database. For each database insertion, the squared Euclidean distances to the samples in its Hessian-smoothing neighbourhood $\mathcal{N}_H(\mathbf{x}_j)$ are computed, and h_H is set as the square root of the median squared distance. This data-driven choice adapts the smoothing scale to the database density and avoids introducing an additional manually tuned tissue parameter. In this way, each database insertion stores a single smoothed local curvature descriptor $\mathbf{H}_j \in \mathbb{R}^{6\times 6}$. In aggregation, the purpose of this curvature-aware regularization is to reduce the influence of neighbours for which local affine interpolation is expected to be less accurate. Variance-weighted aggregation emphasizes dominant deformation modes and Gaussian smoothing improves the robustness of local Hessian estimates.

Let $\mathbf{x}_0 \in \mathbb{R}^6$ denote the query pose. For notational convenience, these selected neighbours are locally re-indexed as $q = 1, \ldots, N_{\text{opt}}$, with corresponding pose vectors $\mathbf{x}_q$ and stored Hessians $\mathbf{H}_q$. The pose offset of the q^{th} selected neighbour relative to the query is then

$$\Delta\mathbf{x}_q = \mathbf{x}_q - \mathbf{x}_0, \qquad q = 1, \ldots, N_{\text{opt}}. \tag{18}$$

The corresponding scalar curvature penalty is computed using the stored

Hessian carried by that neighbour,

$$v_q = \frac{1}{2}\Delta\mathbf{x}_q^T \mathbf{H}_q \Delta\mathbf{x}_q. \tag{19}$$

Since $\mathbf{H}_q$ is a second-order descriptor and need not be positive semidefinite, the sign of v_q is not used directly. Instead, the magnitude of the quadratic response is incorporated through the regularization matrix

$$\mathbf{R} = \operatorname{diag}\Big(v_1^2 + \varepsilon_H, \ldots, v_{N_{\text{opt}}}^2 + \varepsilon_H\Big), \qquad \varepsilon_H > 0, \tag{20}$$

where ε_H is a small positive regularization constant for numerical stabilization.

Let

$$\mathbf{D} = \begin{bmatrix} (\Delta\mathbf{x}_1)^T \\ \vdots \\ (\Delta\mathbf{x}_{N_{\text{opt}}})^T \end{bmatrix} \in \mathbb{R}^{N_{\text{opt}} \times 6}, \tag{21}$$

and $\mathbf{1} \in \mathbb{R}^{N_{\text{opt}}}$ denote the all-ones vector. The final weights are obtained by solving the regularized Karush–Kuhn–Tucker system

$$\begin{bmatrix} 2\mathbf{R} & \mathbf{1} & \mathbf{D} \\ \mathbf{1}^T & 0 & \mathbf{0}^T \\ \mathbf{D}^T & \mathbf{0} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{w} \\ \eta_0 \\ \boldsymbol{\eta} \end{bmatrix} = \begin{bmatrix} \mathbf{0} \\ 1 \\ \mathbf{0} \end{bmatrix}, \tag{22}$$

where $\eta_0 \in \mathbb{R}$ is the Lagrange multiplier associated with the unbiasedness constraint, and $\boldsymbol{\eta} \in \mathbb{R}^6$ contains the multipliers associated with the first-order consistency constraints. The enforced conditions are

$$\mathbf{1}^T\mathbf{w} = 1, \qquad \mathbf{D}^T\mathbf{w} = \mathbf{0}. \tag{23}$$

The first condition ensures that the interpolation weights form an affine combination, while the second guarantees exact reproduction of first-order local behaviour at the query pose. The curvature-aware regularization then penalizes neighbours whose stored local Hessian predicts stronger second-order variation relative to the query, thereby favouring neighbours for which the local affine approximation is expected to be more reliable. In this way, the final weight computation combines geometric proximity, affine consistency, and neighbour-specific curvature information within a single constrained optimization problem.

Algorithm 2 Online 3D-JIT Prediction

Inputs: Query pose $\mathbf{x}_0$, database $\mathcal{S}$, stored local Hessians $\{\mathbf{H}_j\}_{j=1}^{N_h}$, and N_{opt}
Output: Predicted 3D displacement field $\hat{\mathbf{G}}(\mathbf{x}_0)$
Compute the gating constraints in (11)–(12)
Select the N_{opt} nearest feasible database samples and define $\mathcal{N}_{\text{opt}}(\mathbf{x}_0)$
Locally re-index the selected neighbours as $q = 1, \ldots, N_{\text{opt}}$, with associated poses $\mathbf{x}_q$ and stored Hessians $\mathbf{H}_q$
Form $\Delta\mathbf{x}_q = \mathbf{x}_q - \mathbf{x}_0$ for $q = 1, \ldots, N_{\text{opt}}$
Construct $\mathbf{D}$ using (21)
Compute the curvature penalties v_q using (19)
Construct $\mathbf{R}$ using (20)
Solve the regularized KKT system (22) subject to (23)
Compute predicted coefficients $\hat{\mathbf{c}}(\mathbf{x}_0)$ using (8)
Reconstruct the displacement vector $\hat{\mathbf{g}}(\mathbf{x}_0)$ using (9)
Reshape to the volumetric field $\hat{\mathbf{G}}(\mathbf{x}_0)$ using (10)
return $\hat{\mathbf{G}}(\mathbf{x}_0)$

3. Historical Database Construction

The 3D-JIT framework requires an offline database generated using any simulation engine that can return a volumetric displacement field on an $M \times N \times P$ grid. In this paper, we use a 3D polyharmonic spline (PHS) model to simulate mesh displacement resulting from needle insertion, following our previously developed spline-based deformation framework [38]. In our previous work, the underlying spline-based deformation framework is compared against FEM simulations and ex-vivo ultrasound speckle-tracking measurements, providing supporting evidence that the generated displacement fields reproduce the dominant deformation trends considered in this work. Unlike FEM, the PHS model does not directly use Young's modulus or Poisson's ratio as constitutive parameters. Instead, it generates smooth volumetric displacement fields from prescribed needle-induced landmark displacements, with behaviour controlled by phenomenological parameters such as shaft displacement, inward pull, tip displacement, influence radii, and spline regularization. These parameters are calibrated in the prior PHS study to reproduce the dominant deformation trends of an equivalent FEM reference model. Although the PCA-JIT formulation is independent of the deformation engine

used to generate the historical database, the trained database itself is specific to the tissue response represented by the selected reference engine, interaction parameters, geometry, and boundary conditions. Consequently, a database generated for one homogeneous tissue response should not be expected to generalize directly to other types of tissues without recalibration, retraining, and database regeneration.

To provide structured coverage of the needle pose space, insertions are sampled according to three factors: (i) entry points on the entry face, (ii) insertion directions, and (iii) insertion depth. For simplicity, entry points are placed on the plane $x = 0$ and sampled on a uniform $E_y \times E_z$ grid over the entry face, with coordinates

$$\mathbf{n}_s = [\, 0 \;\; y_s \;\; z_s \,]^T, \qquad \mathbf{n}_t = [\, x_t \;\; y_t \;\; z_t \,]^T. \tag{24}$$

For each sampled entry point $\mathbf{n}_s$, a needle insertion is generated by selecting a tip position $\mathbf{n}_t$ inside the unit volume, where x_t represents insertion depth and (y_t, z_t) determine the lateral deviation of the trajectory. The insertion direction is therefore defined by the vector

$$\mathbf{d} = \mathbf{n}_t - \mathbf{n}_s. \tag{25}$$

For implementation, this direction may also be expressed in slope form as

$$s_y = \frac{y_t - y_s}{x_t}, \qquad s_z = \frac{z_t - z_s}{x_t}, \tag{26}$$

with bounds chosen so that $\mathbf{n}_t \in [0, 1]^3$ for all sampled depths up to $x_{t,\max}$.

Within these bounds, the insertion direction is sampled by discretizing the slope variables along the two lateral axes. Specifically, for each entry point $\mathbf{n}_s$, we sample U_y uniformly spaced values of s_y and U_z uniformly spaced values of s_z. Each pair (s_y, s_z) defines one insertion direction.

For each sampled direction, the insertion depth is then parameterized by the tip coordinate x_t, which is sampled over the end-depth band $x_t \in [x_{t,\min}, x_{t,\max}]$ using K uniformly spaced values, including both endpoints. Once $\mathbf{n}_s$, (s_y, s_z), and x_t are fixed, the corresponding tip point is

$$\mathbf{n}_t = [\, x_t \;\; y_s + s_y x_t \;\; z_s + s_z x_t \,]^T. \tag{27}$$

Accordingly, the database is constructed from $E_y E_z$ sampled entry points, $U_y U_z$ sampled directions per entry point, and K sampled depths per direction, yielding

$$N_h = (E_y E_z)(U_y U_z) K. \tag{28}$$

In the current implementation, the entry grid is set to $E_y = 10$ and $E_z = 10$, giving 100 uniformly sampled entry points on the entry face. For each entry point, the insertion direction is sampled using $U_y = 10$ and $U_z = 10$ slope values along the two lateral axes, giving 100 admissible directions per entry point. The insertion depth is sampled using $K = 4$ values over the end-depth interval $x_t \in [0.80, 0.95]$. A boundary margin of 0.08 is used to define the admissible entry and tip regions so that the needle path remains away from the lateral boundaries. These values produce $N_h = 40{,}000$ historical samples per grid resolution.

The selected sampling strategy is chosen to balance pose-space coverage and offline database cost. The structured grid provides uniform coverage of entry position, insertion direction, and near-target insertion depth while keeping the number of historical samples finite. The end-depth interval is selected to focuses on near-target configurations, where tissue deformation is most relevant for ultrasound-guided needle placement and deformation-consistent slice rendering. The database should therefore be interpreted as covering this admissible straight-insertion pose space, rather than the full continuum of possible needle trajectories from shallow to deep insertion. Unseen-pose performance is evaluated using continuous held-out query poses sampled from the same admissible ranges but excluded from the stored database samples.

Each sampled insertion therefore produces one pose-output pair $(\mathbf{x}_j, \mathbf{g}_j)$ in the historical database, where $\mathbf{x}_j$ encodes the entry and tip coordinates and $\mathbf{g}_j$ is the corresponding flattened volumetric displacement field. Once the database is created, the algorithm can be evaluated and the interpolation results compared against the reference deformation.

4. Evaluation Criteria

To evaluate the proposed 3D-JIT framework, we compare its predictions against reference volumetric deformations generated by the selected simulation engine. In this paper, PHS is used as the reference deformation engine; however, the same evaluation procedure could be applied to databases generated using FEM, mass-spring methods, meshless methods, phantom experiments, ex-vivo measurements, or other deformation sources. Accordingly, the reported errors quantify how closely PCA-JIT reproduces the selected PHS reference field for held-out query poses, rather than establishing independent biomechanical ground truth. For each test query $s \in \{1, \ldots, N_t\}$, the JIT

model predicts $\hat{\mathbf{g}}_s \in \mathbb{R}^{3MNP}$ using the procedure described in Section 2. We then compare the predicted displacement to the reference displacement field $\mathbf{g}_s \in \mathbb{R}^{3MNP}$, computed using the reference engine for the same query. We report the mean absolute displacement error per grid node, obtained by summing the absolute errors across the three Cartesian displacement components and averaging over all nodes and test queries:

$$\epsilon_{\mathrm{MAE}} = \frac{1}{N_t} \sum_{s=1}^{N_t} \frac{1}{MNP} \left\| \hat{\mathbf{g}}_s - \mathbf{g}_s \right\|_1 , \tag{29}$$

To characterize large local deviations, we report the mean of the per-query maximum absolute errors.

$$\epsilon_{\max} = \frac{1}{N_t} \sum_{s=1}^{N_t} \left\| \hat{\mathbf{g}}_s - \mathbf{g}_s \right\|_\infty , \tag{30}$$

and the mean 95^{th} percentile error across queries,

$$\epsilon_{95} = \frac{1}{N_t} \sum_{s=1}^{N_t} \mathrm{prctile}(\left| \hat{\mathbf{g}}_s - \mathbf{g}_s \right|, 95) , \tag{31}$$

where $\mathrm{prctile}(\cdot, 95)$ is evaluated over the component-wise absolute displacement errors for each query.

To ensure that interpolation is evaluated on configurations not stored in the database, we sample continuous query poses from the same admissible pose ranges used to construct the database while excluding the discrete poses stored in $\mathcal{S}$. In the reported experiments, $N_t = 100$ held-out queries are used for each grid resolution. This evaluation is therefore an in-distribution interpolation test. The held-out queries are unseen poses sampled from the same admissible pose ranges as the historical database, rather than out-of-distribution insertions. Consequently, the reported errors quantify interpolation accuracy within the covered pose space and should not be interpreted as evidence of generalization outside the sampled insertion range, straight-insertion assumption, or selected tissue-response model.

To assess real-time suitability, we measure the per-query prediction latency of the 3D-JIT model. For each test query we report the speedup relative to the PHS engine as

$$S = \frac{t_{\mathrm{PHS}}}{t_{\mathrm{JIT}}}, \tag{32}$$

where T_{PHS} is the average runtime required to generate a reference deformation field using PHS for a query pose, and T_{JIT} is the average 3D-JIT prediction runtime.

5. Simulation Results

This section evaluates the proposed 3D-JIT framework against the PHS simulation engine used as reference deformation. We begin with representative insertion examples to illustrate the spatial agreement between PCA-JIT and PHS for individual queries, and then report aggregate prediction accuracy and runtime across multiple volumetric grid resolutions.

As validation context for the selected spline-based reference model, Fig. 1 summarizes an ex-vivo porcine tissue insertion experiment in which PHS is compared with FEM and ultrasound speckle-tracking measurements. In this experiment, an 18G needle is inserted into the tissue under B-mode ultrasound imaging, and tissue motion in a small region of interest ahead of the needle tip is estimated using ultrasound speckle tracking. The experimentally measured displacement is compared with the corresponding model-predicted displacement extracted along the ultrasound imaging plane, with FEM used as an additional reference model. The resulting errors show that the tuned spline-based model follows the experimentally observed tissue motion with accuracy comparable to the FEM reference. These results support the use of the spline/PHS model as a computationally efficient reference deformation engine for the present PCA-JIT study. For the purpose of this work, the PCA-JIT experiments reported here should still be interpreted as surrogate reproduction of the selected PHS reference field rather than full clinical or biomechanical validation.

All reported results use a fixed PCA rank of $k = 100$. This value has been selected based on a diagnostic analysis performed on a representative historical database by examining reconstruction error and projection/reconstruction cost as functions of the retained rank, as shown in Fig. 2. As the retained rank increases, the reconstruction MAE decreases, but the improvement becomes progressively smaller beyond $k = 100$, while the total projection/reconstruction cost generally increases.

All simulations were implemented in MATLAB R2025b on a desktop workstation running Windows 11. The workstation is equipped with an Intel Core i9-12900KF CPU and 128 GB RAM. No GPU acceleration or parallel computing was used for the reported PCA-JIT online prediction times unless

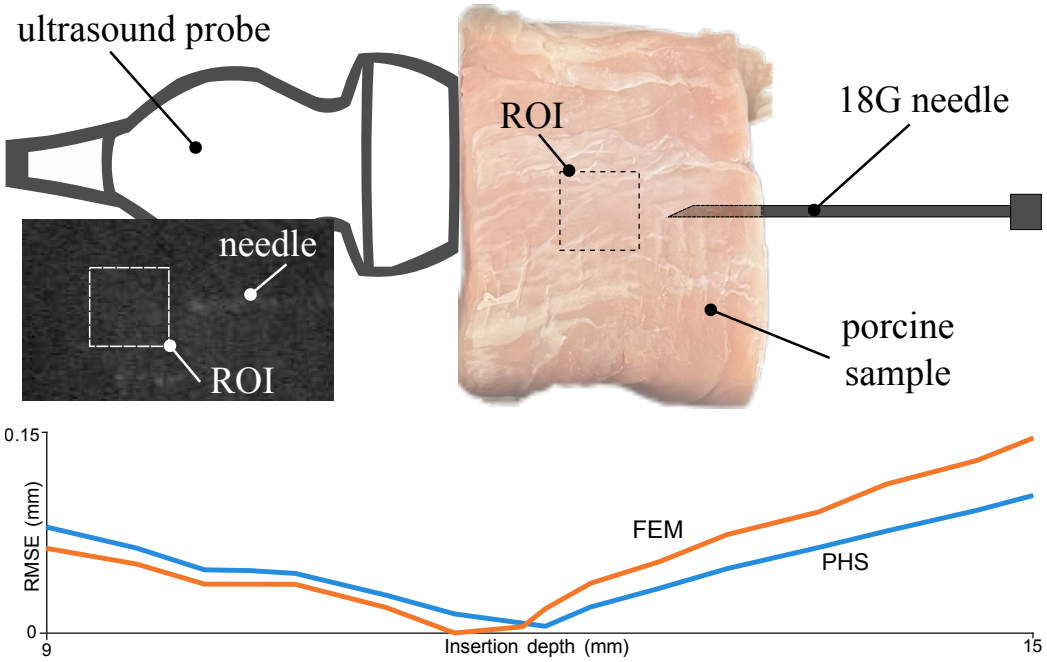


Figure 1: Validation context for the spline-based reference deformation model. The experimental setup consists of an 18G needle inserted into an ex-vivo porcine tissue sample under ultrasound imaging, with tissue motion estimated in a region of interest ahead of the needle tip. The RMSE between the experimentally measured motion and the corresponding model-predicted displacement is shown as a function of insertion depth for PHS and FEM.

otherwise stated. Reported runtimes correspond to single-query execution under the same hardware and software environment for both PHS and PCA-JIT, so that the measured speedups reflect their relative online computational cost under identical conditions.

To first illustrate a single prediction result before moving to aggregate metrics, Fig. 3 presents two representative needle insertion scenarios together with their reconstructed deformation fields and corresponding neighbours selected from the historical database. For scenario 1, panels (a)–(d) show that the PCA-JIT prediction closely matches the PHS reference field, while the largest discrepancies remain spatially localized near the main deformation region. For scenario 2, panels (e)–(h) show the same comparison together with the slice plane used for planar inspection, and panels (i)–(k) present the corresponding 2D slice views extracted from the 3D fields. In both cases, the dominant neighbours correspond to geometrically similar insertion poses

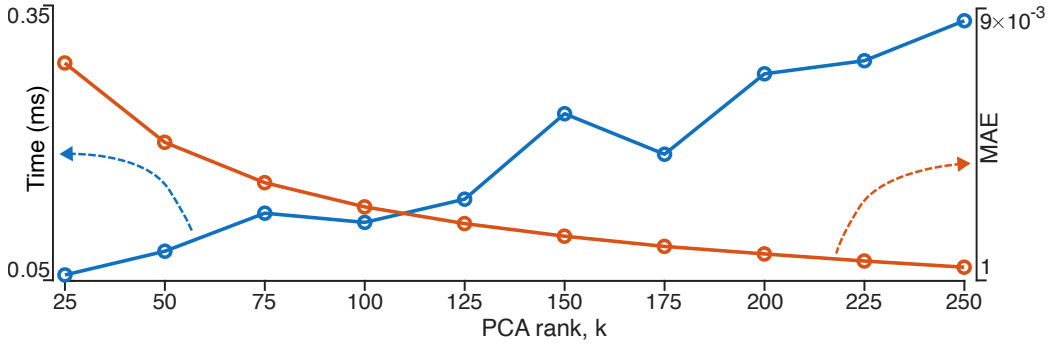


Figure 2: Offline PCA diagnostic analysis used to select the retained output rank. Reconstruction mean absolute error (MAE) versus PCA rank k and total projection and reconstruction cost versus PCA rank k.

within the admissible gating constraints, and the final prediction is influenced primarily by a small subset of neighbours with large weight magnitudes. The overlaid error histograms in panel (l) further confirm that most grid points are associated with low error values, with only a small tail of higher-error outliers. These examples provide an interpretable qualitative verification that the proposed model preserves the main deformation structure of the PHS reference while interpolating from nearby historical insertions.

Having established representative field-level agreement for individual insertions, we next quantify the speed–accuracy trade-off introduced by PCA output compression. Fig. 4 summarizes the prediction error across volumetric grid resolutions ranging from 24^3 to 42^3, with all values averaged over 100 held-out insertions per resolution. To isolate the effect of PCA compression, PCA-JIT is compared with a FULL-JIT ablation using the same historical database, held-out query poses, neighbourhood-selection procedure, and curvature-aware constrained weights. The only difference between the two methods is the prediction space: FULL-JIT applies the weights directly to the full volumetric displacement vectors in $\mathbb{R}^{3MNP}$, whereas PCA-JIT applies the same weights to the reduced coefficient vectors in $\mathbb{R}^k$ and then reconstructs the full field using the retained PCA basis. Therefore, Fig. 4(a) provides a direct accuracy comparison for the proposed PCA integration. The close agreement between PCA-JIT and FULL-JIT shows that the retained PCA basis preserves the dominant deformation modes required for local JIT interpolation, while introducing only a small additional approximation error relative to full-space interpolation. Fig. 4(b) reports the PCA-JIT error

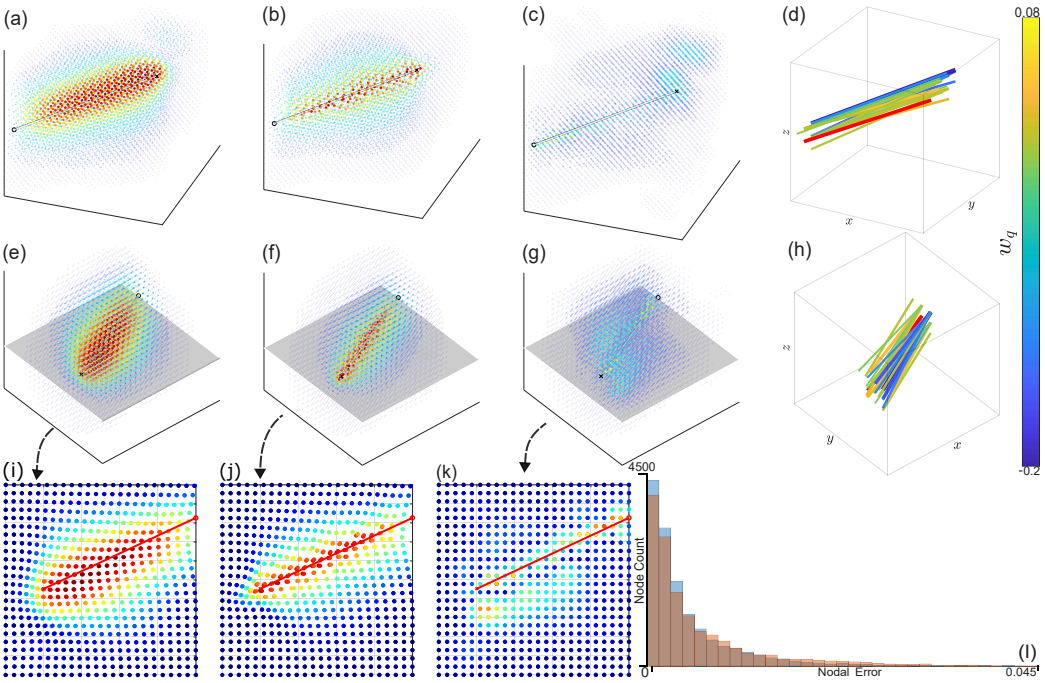


Figure 3: Qualitative and quantitative comparison of two representative needle insertion scenarios. (a)–(d) Scenario 1: PCA-JIT prediction, PHS reference deformation, nodal error, and the neighbours selected for the insertion query. (e)–(h) Scenario 2: PCA-JIT prediction, PHS reference deformation, nodal error, and influential neighbours with the slice plane used for 2D inspection indicated in 3D. (i)–(k) Corresponding 2D slice views from scenario 2 for the PCA-JIT prediction, PHS reference deformation, and nodal error. (l) Overlaid histograms of nodal error magnitude for the two representative scenarios.

statistics ϵ_{MAE}, ϵ_{95}, and $\epsilon_{\max}$, all of which remain small as the grid resolution increases, indicating that the proposed framework preserves overall field fidelity relative to the PHS reference deformation. The maximum error metric in (30) is computed by first taking, for each query, the largest absolute component-wise deviation over all voxels and then averaging this value across the 100 test queries. Thus, $\epsilon_{\max}$ reflects worst-case local discrepancies, whereas ϵ_{MAE} and ϵ_{95} summarize typical prediction accuracy.

The online computational performance of the proposed method over the same set of test queries is then evaluated. Fig. 5(a) compares the mean per-query runtime of PCA-JIT against the PHS simulation engine as the volumetric grid resolution increases from 24^3 to 42^3, again averaged over 100 held-out insertions per resolution. Across all tested resolutions, PCA-

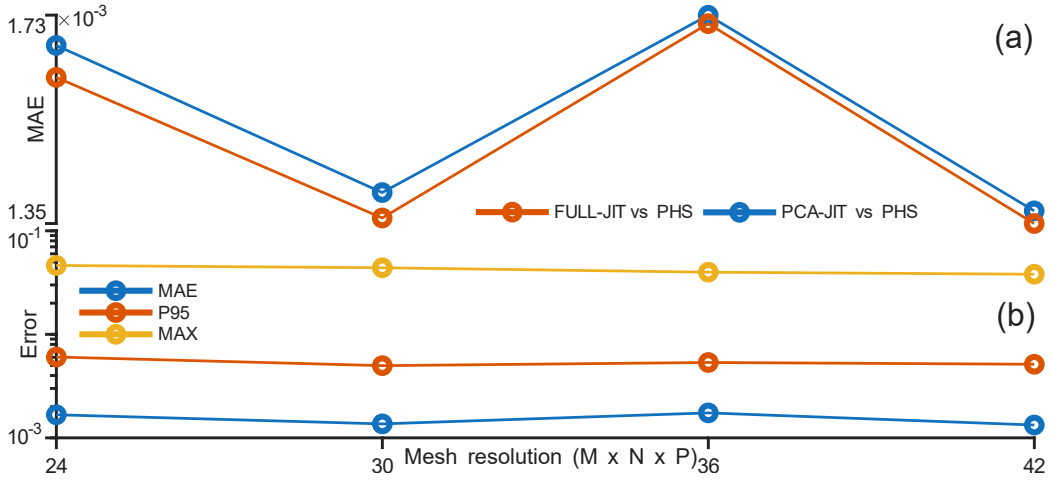


Figure 4: Prediction error versus grid resolution, averaged over 100 held-out insertions per resolution. (a) Mean absolute error of PCA-JIT and FULL-JIT relative to PHS. (b) PCA-JIT error statistics: mean absolute error ϵ_{MAE}, 95^{th} percentile error ϵ_{95}, and maximum error $\epsilon_{\max}$.

JIT achieves millisecond-scale prediction latency, whereas PHS requires hundreds of milliseconds per-query. The corresponding speedup, computed as $S = T_{\mathrm{PHS}}/T_{\mathrm{JIT}}$, increases with resolution and reaches $89.0\times$ at 42^3, indicating that the computational advantage of the proposed approach becomes more pronounced as the displacement-field dimension grows. Fig. 5(b) further breaks down the mean online runtime into its main computational components. The neighbour-retrieval and weight-calculation stage remains comparatively stable across grid resolutions, since it depends mainly on the fixed local neighbourhood size rather than the full mesh dimension. In contrast, the full-mesh weighted prediction cost increases as the output dimension $3MNP$ grows. This trend identifies full-field recombination/reconstruction as the main scalability bottleneck and highlights the benefit of performing the weighted prediction in the reduced PCA coefficient space before reconstructing the volumetric displacement field.

Table 1 consolidates the quantitative accuracy and runtime results across all tested resolutions. Consistent with the representative examples and aggregate plots, the proposed PCA-JIT framework maintains low prediction error while substantially reducing online computation time relative to the PHS reference model. Although the error varies slightly across resolutions, it remains small overall, indicating that the reduced PCA representation preserves the

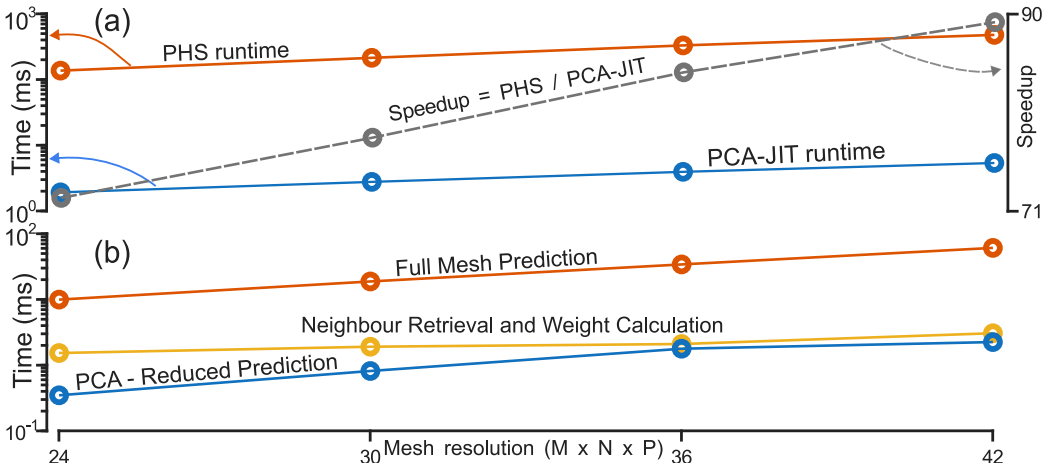


Figure 5: Runtime performance versus grid resolution, averaged over 100 held-out insertions per resolution. (a) Mean per-query runtime of PCA-JIT and PHS, together with the corresponding speedup $S = T_{\mathrm{PHS}}/T_{\mathrm{JIT}}$. (b) Measured component-wise contribution to the PCA-JIT online runtime, separated into neighbour retrieval and weight calculation, PCA reduced weighted prediction, and full-mesh weighted prediction.

dominant deformation behaviour. At the same time, the speedup increases with volumetric grid resolution, showing that the benefit of operating in the reduced PCA space becomes more significant as the displacement-field dimension grows. This trend is particularly relevant for interactive ultrasound simulation, where increasing spatial resolution is desirable for image realism but would otherwise make direct full-field interpolation or repeated reference-engine evaluation prohibitively expensive. At the largest evalu-

Table 1: Prediction error between PCA-JIT and PHS reference deformation across volumetric grid resolutions, together with mean per-query runtimes t_{JIT} and t_{PHS}. All values are averaged over 100 held-out insertions per resolution. Errors are reported for displacement fields defined on the normalized computational domain $[0,1]^3$, and runtimes are given in ms. Speedup is computed as $T_{\mathrm{PHS}}/T_{\mathrm{JIT}}$ using mean runtimes.

Grid	ϵ_{MAE}	ϵ_{95}	$\epsilon_{\max}$	t_{JIT}	t_{PHS}	Speedup
24^3	1.672×10^{-3}	6.040×10^{-3}	4.628×10^{-2}	1.92	136.6	71.1
30^3	1.365×10^{-3}	4.984×10^{-3}	4.397×10^{-2}	2.77	214.0	77.3
36^3	1.743×10^{-3}	5.347×10^{-3}	3.983×10^{-2}	3.94	330.5	83.9
42^3	1.331×10^{-3}	5.140×10^{-3}	3.806×10^{-2}	5.37	477.8	89.0

ated resolution, 42^3, storing the full single-precision displacement database would require approximately 35.6 GB, whereas PCA compression with $k = 100$ reduces the stored representation, including the PCA basis, coefficient database, mean field, pose database, and stored Hessian descriptors, to approximately 0.11 GB. The reported speedups, including the 89.0× speedup at 42^3, refer to online prediction time only and compare one PCA-JIT prediction against one PHS reference evaluation for the same query. Offline costs, including database generation, PCA computation, coefficient projection, Hessian fitting, and Hessian smoothing, are one-time preprocessing costs and are not included in the reported per-query speedup.

To interpret the primary error metric in physical units, the normalized mean absolute error can be multiplied by the physical field-of-view length. For the representative 40×40 mm ultrasound field of view considered in this work, a normalized displacement error of 10^{-3} corresponds to approximately 0.04 mm. Using this scale, the mean absolute errors in Table 1 correspond to approximately 0.053–0.070 mm, indicating sub-millimetre average deformation differences relative to the PHS reference field. There is no universal clinical threshold for acceptable full-field tissue deformation error because the acceptable tolerance depends on the procedure, target size, anatomy, imaging resolution, needle diameter, and safety margin. As a point of reference, [39] reported that targeting errors below approximately 2.5 mm may be clinically insignificant in MRI-guided prostate biopsy simulation because this value is comparable to the needle-tip diameter.

To directly evaluate the contribution of the curvature-aware weighting term, an internal curvature-free ablation is performed. The curvature-free variant uses the same PCA-JIT pipeline as the proposed method, including the same database, held-out queries, PCA rank, gating thresholds, neighbourhood size, and first-order consistency constraints. Only the Hessian-based regularization term is removed and replaced with a distance-based weighting term. The results are reported in Table 2, which show modest but consistent improvement from the hessian inclusion over the curvature-free variant. The curvature-aware method gives lower ϵ_{MAE}, ϵ_{95}, and $\epsilon_{\max}$ across all tested resolutions. Runtime remains essentially unchanged between the two variants, indicating that the Hessian-based weighting term does not introduce a meaningful online computational penalty. These results support the curvature-aware term as a useful internal regularization component, while suggesting that its benefit may become more pronounced for databases with stronger local nonlinearity representing non-homogeneous tissue.

Table 2: Internal ablation of curvature-aware weighting. The curvature-aware PCA-JIT method and the curvature-free constrained PCA-JIT variant use the same historical database, held-out query poses, PCA rank, gating constraints, neighbourhood size, and first-order consistency constraints. The curvature-free variant replaces the Hessian-based regularization with a distance-based regularization. Errors are reported on the normalized computational domain $[0,1]^3$, and runtimes are given in ms.

Curvature-aware PCA-JIT

Grid	ϵ_{MAE}	ϵ_{95}	$\epsilon_{\max}$	t_{JIT}
24^3	1.672×10^{-3}	6.040×10^{-3}	4.628×10^{-2}	1.92
30^3	1.365×10^{-3}	4.984×10^{-3}	4.397×10^{-2}	2.77
36^3	1.743×10^{-3}	5.347×10^{-3}	3.983×10^{-2}	3.94
42^3	1.331×10^{-3}	5.140×10^{-3}	3.806×10^{-2}	5.37

Curvature-free PCA-JIT

Grid	ϵ_{MAE}	ϵ_{95}	$\epsilon_{\max}$	t_{JIT}
24^3	1.856×10^{-3}	6.481×10^{-3}	4.965×10^{-2}	1.90
30^3	1.459×10^{-3}	4.996×10^{-3}	4.443×10^{-2}	2.79
36^3	1.757×10^{-3}	5.774×10^{-3}	4.040×10^{-2}	3.95
42^3	1.354×10^{-3}	5.563×10^{-3}	3.928×10^{-2}	5.34

6. Ultrasound Volume Deformation and Slice Rendering

The predicted displacement field can be used to deform a baseline 3D ultrasound volume and extract deformation-consistent slice views for visualization. Let $I_0(x, y, z)$ denote the undeformed ultrasound intensity volume defined on a voxel grid over the imaging domain. For a query needle pose $\mathbf{x}_0$, the 3D-JIT model predicts a vectorized displacement field $\hat{\mathbf{g}}(\mathbf{x}_0)$, which is reshaped into the tensor

$$\hat{\mathbf{G}}(\mathbf{x}_0) \in \mathbb{R}^{M \times N \times P \times 3}, \tag{33}$$

where the last dimension stores the predicted displacement components in the x-, y-, and z-directions. The corresponding deformed mesh coordinates are given by

$$\mathbf{G}^{\mathrm{def}}(\mathbf{x}_0) = \mathbf{G} + \hat{\mathbf{G}}(\mathbf{x}_0). \tag{34}$$

To apply this deformation to the ultrasound volume, the predicted displacement field on the simulation mesh is interpolated onto the ultrasound

voxel grid. Let

$$\mathbf{d}(x, y, z) = \begin{bmatrix} \delta x(x, y, z) \\ \delta y(x, y, z) \\ \delta z(x, y, z) \end{bmatrix} \tag{35}$$

denote the interpolated displacement vector at voxel location (x, y, z). The deformed ultrasound volume is then obtained by resampling the baseline volume using this displacement field. Under a pullback formulation, the deformed intensity is written as

$$\mathbf{I}_{\text{def}}(x, y, z) = \mathbf{I}_0 \left(x + \delta x, \quad y + \delta y, \quad z + \delta z\right) \tag{36}$$

where the δx, δy, and δz are functions of (x, y, z) but omitted for notational simplicity.

Trilinear interpolation is used to evaluate both the displacement field on the ultrasound voxel grid and the intensity values sampled from the undeformed volume. Samples mapped outside the available image domain are assigned the nearest valid boundary value. The quantitative validation of the displacement field is therefore provided by the PCA-JIT versus PHS reference comparison in Section 5, whereas the ultrasound rendering example below is a qualitative demonstration of how the predicted field can be coupled to volume warping and slice extraction.

To qualitatively demonstrate the ultrasound volume deformation and rendering pipeline, we apply the method to a $200 \times 200 \times 87$ voxel ultrasound volume acquired from a Renal Biopsy Ultrasound Phantom (Blue Phantom). The original volume is deformed using predicted displacement fields from two representative needle insertions. No measured deformed ultrasound volume is available for these examples; therefore, the comparison is intended to show deformation-consistent resampling and slice extraction rather than quantitative ultrasound image validation.

These sample results are shown in Fig. 6. The same intensity normalization and rendering settings are used in all panels to preserve visual consistency. In addition to the volumetric rendering, we report three slice views for three representative configurations: undeformed tissue, a straight needle insertion through the middle of the volume, and an angled insertion. For the angled configuration, the XZ view is extracted using a tilted plane aligned with the needle shaft so that the full shaft remains visible within the slice. By contrast, in the YZ view the needle appears only weakly because the slicing plane is largely cross-sectional to the insertion direction. These examples

illustrate how the volumetric ultrasound data can be sampled to visualize needle pose and its image-plane appearance under different insertion geometries.

Algorithm 3 Online Ultrasound Volume Deformation and Slice Extraction

Input: Undeformed mesh $\mathbf{G}$, reference volume I_0, query insertion $\mathbf{x}_0$, trained 3D-JIT model

Outputs: Deformed volume I_{def}, orthogonal slice views, and oblique slice view

Predict the displacement field $\hat{\mathbf{G}}(\mathbf{x}_0)$ using the 3D-JIT model
Compute the deformed mesh coordinates using eq (34)
Interpolate $\hat{\mathbf{G}}(\mathbf{x}_0)$ from the simulation mesh onto the ultrasound voxel grid using trilinear interpolation
for each voxel location (x, y, z) in the deformed output grid **do**
    Read the interpolated displacement components $\delta x(x, y, z)$, $\delta y(x, y, z)$, and $\delta z(x, y, z)$
    Sample the undeformed ultrasound volume at the corresponding pull-back location using trilinear intensity interpolation
    If the mapped location lies outside the image domain, assign the nearest valid boundary value
    Assign the deformed intensity using Eq. (36)
end for
Extract orthogonal and oblique slice views from I_{def}
return I_{def} and the extracted slice views

7. Limitations

The proposed framework is developed under several assumptions. The current database focuses on straight needle insertions parameterized by entry and tip coordinates. This choice provides a controlled setting for evaluating volumetric PCA-JIT interpolation, but does not yet include curved needle steering, bevel-tip deflection, needle bending, or dynamic steering trajectories. The tissue volume is represented as a homogeneous normalized block. This simplifies database generation and isolates the deformation-prediction problem, but does not yet explicitly include tissue heterogeneity, anisotropy, fracture, cutting, or patient-specific anatomy and material properties. Future

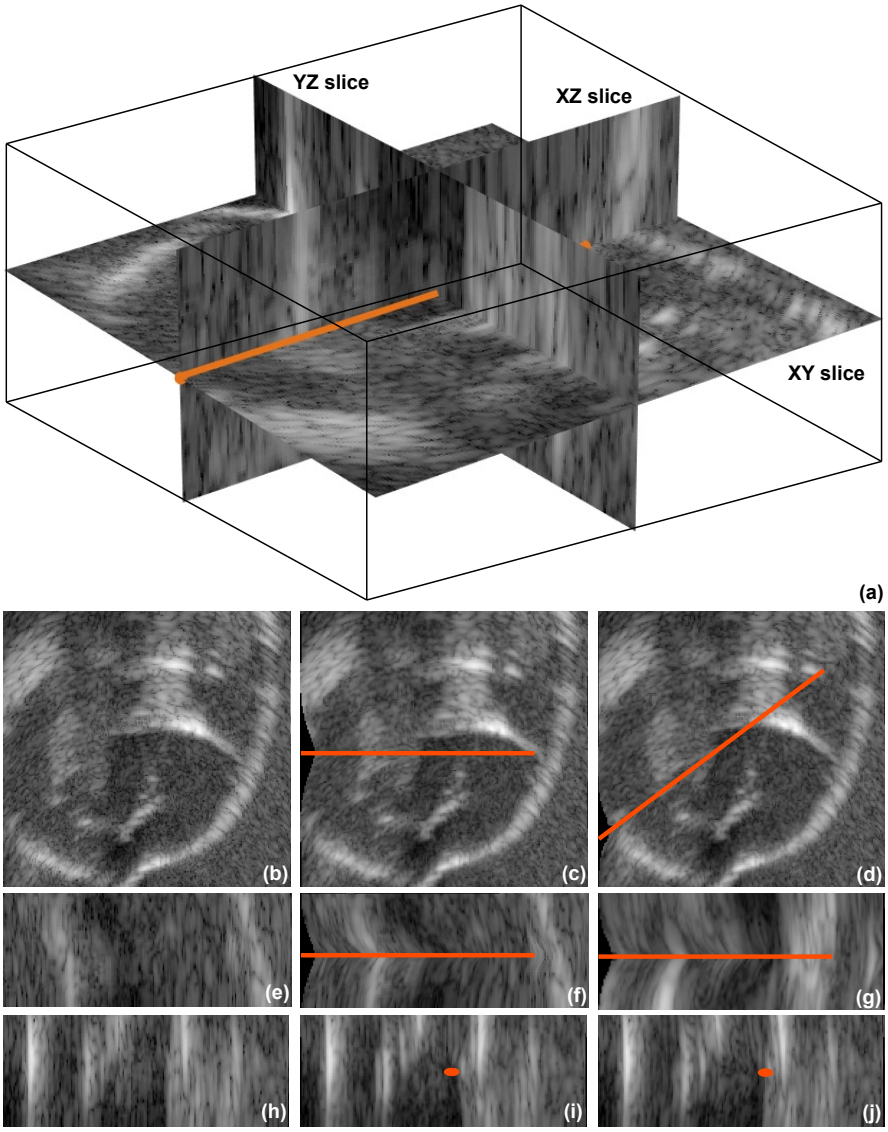


Figure 6: Qualitative illustration of volumetric ultrasound image deformation and slice rendering for different needle insertions. (a) Volumetric image inside the rectangular volume, together with the slice planes and the needle shaft (orange) for the first configuration. (b)–(d) XY slices for three configurations: undeformed tissue, a straight insertion through the centre of the volume (configuration 1), and an angled insertion (configuration 2). (e)–(g) Corresponding XZ slices for the same three configurations. For the angled insertion, the XZ slice is rotated into a tilted plane to fully visualize the needle shaft. (h)–(j) Corresponding YZ slices for the same three configurations. The red overlays indicate the needle trajectory.

databases can incorporate these effects through expanded input parameterization, tissue-specific calibration, or training data generated from more detailed deformation engines. PHS is used as the reference deformation engine for database generation and evaluation. Although PCA-JIT is independent of the chosen deformation engine, the trained model naturally reflects the assumptions and characteristics of the engine used to generate the database. Thus, the reported errors quantify how closely PCA-JIT reproduces the selected PHS reference field, while independent biomechanical or clinical validation remains a direction for future work. The current algorithm is not intended as a complete benchmark against all possible surrogate models. Its objective is to evaluate whether PCA-JIT reproduces deformation fields generated by the selected database-generation engine while reducing online computation time. Accordingly, the main comparisons are made against the PHS reference deformation fields and against a FULL-JIT ablation that isolates the effect of PCA output compression.

The purpose of the comparison is therefore not to determine which deformation model is globally superior, but to evaluate whether PCA-JIT can act as a time-efficient surrogate for the selected database-generation engine. PHS is used as the reference engine because it generates the historical deformation fields used in this study, while the FULL-JIT comparison isolates the effect of PCA output compression within the same JIT framework. Broader benchmarking against FEM, mass-spring models, position-based dynamics, Kriging, Gaussian-process regression, neural-network surrogates, or curvature-free nearest-neighbour interpolation would require separate implementation, training, tuning, and validation of each method, and is left for future work.

Finally, the ultrasound volume deformation example demonstrates deformation consistent volume warping and slice extraction. This example is used as a qualitative visualization of the rendering pipeline, while quantitative validation of ultrasound image appearance against measured deformed volumes remains future work.

8. Conclusion

This paper presents a 3D Just-in-Time modelling framework for real-time prediction of volumetric tissue deformation during ultrasound-guided percutaneous interventions. The proposed method uses an offline database of reference deformation fields, retrieves geometrically similar needle inser-

tions online, computes curvature-aware constrained interpolation weights, and predicts the deformation field in a reduced PCA coefficient space before reconstructing the full volumetric displacement field.

Across the tested grid resolutions, PCA-JIT maintains close agreement with the selected PHS reference deformation while achieving millisecond-scale prediction times. The speedup increases with volumetric grid resolution, reaching $89.0\times$ at 42^3, demonstrating the benefit of reduced-space prediction for dense volumetric outputs. The predicted displacement fields are also coupled to ultrasound volume warping and slice extraction, illustrating how the framework supports deformation-consistent visualization in interactive ultrasound simulation.

Future work focuses on broader experimental validation, expanded database coverage, tissue-specific and patient-specific parameterization, comparison with additional surrogate baselines, and integration with force-feedback and haptic simulation.

Funding

This work was supported by the Natural Sciences and Engineering Research Council of Canada (NSERC) [RGPIN-2025-05468].

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

The data supporting the findings of this study are not publicly available.

Code availability

The code used in this study is not publicly archived as part of an ongoing research framework.

References

[1] D. L. Miller, A. Abo, J. S. Abramowicz, T. A. Bigelow, D. Dalecki, E. Dickman, J. Donlon, G. Harris, J. Nomura, Diagnostic Ultrasound Safety Review for Point-of-Care Ultrasound Practitioners, Journal of Ultrasound in Medicine 39 (6) (2020) 1069–1084. doi:10.1002/jum.15202.

[2] P. Birowo, P. A. R. Raharja, et al., X-ray-free Ultrasound-guided Percutaneous Nephrolithotomy in Supine Position Using Alken Metal Telescoping Dilators in a Large Kidney Stone: A Case Report, Research and Reports in Urology 12 (2020) 287–293. doi:10.2147/RRU.S259941.

[3] H. Mazdarani, A. Cotton, C. Rossa, 2D Ultrasound-Guided Visual Servoing for In-Plane Needle Tracking in Robot-Assisted Percutaneous Nephrolithotomy, in: 2023 IEEE International Conference on Systems, Man, and Cybernetics (SMC), 2023, pp. 1786–1791.

[4] Y. Song, Y. Ma, Y. Song, X. Fei, Evaluating the Learning Curve for Percutaneous Nephrolithotomy under Total Ultrasound Guidance, PloS One 10 (8) (2015) e0132986. doi:10.1371/journal.pone.0132986.

[5] M. Trojak, et al., Mixed Reality Biopsy Navigation System Utilizing Markerless Needle Tracking and Imaging Data Superimposition, Cancers 16 (10) (2024) 1894. doi:10.3390/cancers16101894.
URL https://www.mdpi.com/2072-6694/16/10/1894

[6] C. M. Solano-Cordero, N. Encina-Baranda, M. Pérez-Liva, J. L. Herraiz, Recent advances in b-mode ultrasound simulators, Applied Sciences 15 (23) (2025). doi:10.3390/app152312535.
URL https://www.mdpi.com/2076-3417/15/23/12535

[7] Y. Guo, et al., A real-time virtual liver simulator using position based dynamics with compliant constraints for medical applications, Scientific Reports 15 (1) (2025) 26557.

[8] Q. Huang, Z. Zeng, A Review on Real-Time 3D Ultrasound Imaging Technology, BioMed Research International 2017 (1) (2017) 6027029. doi:10.1155/2017/6027029.

[9] S. Xia, et al., A learning curve analysis of ultrasound-guided in-plane and out-of-plane vascular access training with Perk Tutor, in: Medical Imaging 2018: Image-Guided Procedures, Robotic Interventions, and Modeling, Vol. 10576, SPIE, 2018, pp. 512–519. doi:10.1117/12.2293789.

[10] S. Selladurai, et al., Towards 3D-dense ultrasound image simulation from 2D CT scans for ultrasound-guided percutaneous nephrolithotomy: a progressive training approach from basic to advanced simulator complexity, Medical & Biological Engineering & Computing (Jan. 2026). doi:10.1007/s11517-025-03502-y.

[11] A. Lelevé, et al., Haptic Training Simulation, Frontiers in Virtual Reality 1 (Jul. 2020). doi:10.3389/frvir.2020.00003.

[12] B. Sainsbury, et al., Evaluation of a Virtual Reality Percutaneous Nephrolithotomy (PCNL) Surgical Simulator, Frontiers in Robotics and AI 6 (Jan. 2020). doi:10.3389/frobt.2019.00145.

[13] K.-S. Choi, et al., A virtual training simulator for learning cataract surgery with phacoemulsification, Computers in Biology and Medicine 39 (11) (2009) 1020–1031. doi:10.1016/j.compbiomed.2009.08.003.

[14] A. M. Rahimi, et al., Training in robotic-assisted surgery: a systematic review of training modalities and objective and subjective assessment methods, Surgical Endoscopy 38 (7) (2024) 3547–3555. doi:10.1007/s00464-024-10915-7.

[15] M. Selim, et al., Experimental evaluation of virtual needle insertion framework with enhanced haptic feedback, International Journal of Computer Assisted Radiology and Surgery 20 (8) (2025) 1643–1652. doi:10.1007/s11548-025-03420-2.

[16] S. Azher, et al., Findings Favor Haptics Feedback in Virtual Simulation Surgical Education: An Updated Systematic and Scoping Review, Surgical Innovation 31 (3) (2024) 331–341. doi:10.1177/15533506241238263.
URL https://doi.org/10.1177/15533506241238263

[17] Y. Adagolodjo, et al., Inverse real-time Finite Element simulation for robotic control of flexible needle insertion in deformable tissues, in: 2016

IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2016, pp. 2717–2722. doi:10.1109/IROS.2016.7759422.

[18] S. U. Gjerald, R. Brekken, et al., Interactive development of a CT-based tissue model for ultrasound simulation, Computers in Biology and Medicine 42 (5) (2012) 607–613. doi:10.1016/j.compbiomed.2012.02.006.

[19] C. Rossa, M. Tavakoli, Issues in closed-loop needle steering, Control Engineering Practice 62 (2017) 55–69. doi:10.1016/j.conengprac.2017.03.004.

[20] A. Gani, et al., Impact of haptic feedback on surgical training outcomes: A Randomised Controlled Trial of haptic versus non-haptic immersive virtual reality training, Annals of Medicine and Surgery 83 (Nov. 2022). doi:10.1016/j.amsu.2022.104734.

[21] A. M. Tahmasebi, et al., A Framework for the Design of a Novel Haptic-Based Medical Training Simulator, IEEE Transactions on Information Technology in Biomedicine 12 (5) (2008) 658–666. doi:10.1109/TITB.2008.926496.

[22] H. V. Chorney, et al., System identification and simulation of soft tissue force feedback in a spine surgical simulator, Computers in Biology and Medicine 164 (2023) 107267. doi:10.1016/j.compbiomed.2023.107267.

[23] H. Mohammadi, et al., Finite-element modelling of interactions of needle with tympanic membrane and middle ear, Hearing Research 452 (2024) 109092. doi:10.1016/j.heares.2024.109092.

[24] Y. Kobayashi, et al., Development of an integrated needle insertion system with image guidance and deformation simulation, Computerized Medical Imaging and Graphics 34 (1) (2010) 9–18. doi:10.1016/j.compmedimag.2009.08.008.

[25] M. Khadem, C. Rossa, R. S. Sloboda, N. Usmani, M. Tavakoli, Mechanics of Tissue Cutting During Needle Insertion in Biological Tissue, IEEE Robotics and Automation Letters 1 (2) (2016) 800–807. doi:10.1109/LRA.2016.2528301.

[26] B. Burger, et al., Real-Time GPU-Based Ultrasound Simulation Using Deformable Mesh Models, IEEE Transactions on Medical Imaging 32 (3) (2013) 609–618. doi:10.1109/TMI.2012.2234474.

[27] H. Xie, J. Song, B. Gao, Y. Zhong, C. Gu, K.-S. Choi, Finite-element kalman filter with state constraint for dynamic soft tissue modelling, Computers in Biology and Medicine 135 (2021) 104594. doi:10.1016/j.compbiomed.2021.104594.

[28] W. Hou, P. X. Liu, M. Zheng, A new model of soft tissue with constraints for interactive surgical simulation, Computer Methods and Programs in Biomedicine 175 (2019) 35–43. doi:10.1016/j.cmpb.2019.03.018.

[29] J. N. Smit, K. F. D. Kuhlmann, B. R. Thomson, N. F. M. Kok, T. J. M. Ruers, M. Fusaglia, Ultrasound guidance in navigated liver surgery: toward deep-learning enhanced compensation of deformation and organ motion, International Journal of Computer Assisted Radiology and Surgery 19 (1) (2024) 1–9. doi:10.1007/s11548-023-02942-x.

[30] M. Kaya, E. Senel, A. Ahmad, O. Bebek, Visual tracking of multiple moving targets in 2D ultrasound guided robotic percutaneous interventions, in: 2017 IEEE International Conference on Robotics and Automation (ICRA), 2017, pp. 1996–2002. doi:10.1109/ICRA.2017.7989231.

[31] S. Frisken, M. Luo, P. Juvekar, A. Bunevicius, I. Machado, P. Unadkat, M. M. Bertotti, M. Toews, W. M. Wells, M. I. Miga, A. J. Golby, A comparison of thin-plate spline deformation and finite element modeling to compensate for brain shift during tumor resection, International Journal of Computer Assisted Radiology and Surgery 15 (1) (2020) 75–85. doi:10.1007/s11548-019-02057-2.

[32] A finite element-based machine learning approach for modeling the mechanical behavior of the breast tissues under compression in real-time 90.

[33] Soft Tissue Deformation Modeling in the Procedure of Needle Insertion: A Kriging-Based Method 35.

[34] A Surrogate Model Based on a Finite Element Model of Abdomen for Real-Time Visualisation of Tissue Stress during Physical Examination Training 9.

[35] A. Stenman, et al., Just in time models for dynamical systems, in: Proceedings of 35th IEEE Conference on Decision and Control, Vol. 1, 1996, pp. 1115–1120 vol.1. doi:10.1109/CDC.1996.574658.

[36] L. Greve, B. P. van de Weg, Surrogate modeling of parametrized finite element simulations with varying mesh topology using recurrent neural networks, Array 14 (2022) 100137. doi:10.1016/j.array.2022.100137.

[37] X. Li, Y. Wang, Adaptive online monitoring for ICU patients by combining just-in-time learning and principal component analysis, Journal of Clinical Monitoring and Computing 30 (6) (2016) 807–820. doi:10.1007/s10877-015-9778-4.

[38] K. Zalzalah, A. Cotton, C. Rossa, Polyharmonic spline modelling of tissue deformation for ultrasound-guided surgery, 2026.

[39] S. Avni, S. Vikal, G. Fichtinger, Design of a predictive targeting error simulator for MRI-guided prostate biopsy, in: K. H. Wong, M. I. Miga (Eds.), Medical Imaging 2010: Visualization, Image-Guided Procedures, and Modeling, Vol. 7625, International Society for Optics and Photonics, SPIE, 2010, p. 76251A. doi:10.1117/12.844476.
URL https://doi.org/10.1117/12.844476