KFUPM
Journal of Undergraduate Research International

Diamond Open Access 38

Article Computer Engineering

GPU-Accelerated Mesh-Based Deformation Reconstruction for Quality Control in Additive Manufacturing

Ahmed Malaekah and Ammar Alsheghri*

Cite https://doi.org/10.64589/juri/231621

Submitted: April 24, 2026 Revised: June 09, 2026 Accepted: July 09, 2026

ABSTRACT

In-time quality control remains a major challenge in additive manufacturing (AM) owing to the high computational cost of deformation reconstruction and difficulty in handling real-image observations during printing. Therefore, this paper presents a deformation-reconstruction framework for image-based quality control in AM, which integrates free-form deformation, differentiable rendering, and an adaptive loss formulation to enable efficient and robust mesh-to-image alignment. This framework supports both synthetic multi-view inputs and real-image observations through a unified workflow. In the synthetic test case, multi-view target images were used to evaluate the reconstruction accuracy and computational performance of the proposed framework. In the real-image test case, a segmented top-view photograph was used as the target observation, and the deformation was guided by a combined loss formulation to improve robustness against real-image imperfections. The proposed framework achieved a mean runtime of 3.81 s, whereas the Central Processing Unit (CPU)-based baseline model required 692.65 s. The former method exhibited a speedup of 181.80 times, while maintaining high reconstruction accuracy. In the synthetic test case, the framework achieved Chamfer and Hausdorff distances of 0.0111 and 0.0207, respectively. In the real-image test case, the framework achieved a normalized cross-correlation value of 0.9729, together with a Chamfer distance of 0.0013 and Hausdorff distance of 0.0035, demonstrating the successful deformation of the nominal mesh toward the observed target geometry. This study provides an effective approach for deformation reconstruction under realistic conditions and demonstrates the feasibility of periodic near-real-time quality control in AM.

Keywords: additive manufacturing; quality control; mesh deformation reconstruction; GPU acceleration; free-form deformation; differentiable rendering

1. INTRODUCTION

Additive manufacturing (AM), commonly known as 3D printing, is a method in which three-dimensional (3D) components are fabricated by sequentially depositing thin two-dimensional layers of a material [1]. Each layer is constructed on top of the previous layer, and the overall geometry gradually evolves to form the desired 3D structure. Over the past decade, AM has become considerably popular owing to the availability of affordable and compact 3D-printing machines, as well as ongoing advances in material science and related software [1,2]. Originally developed for rapid prototyping, AM has become a widely adopted manufacturing approach for prototyping, low-volume production, and increasingly functional end-use components [2]. This expansion has increased the demand for reliable and automated quality-control methods that ensure geometric accuracy and structural integrity of additively manufactured components.

However, ensuring consistent component quality remains a major challenge in AM. Defects such as pores, cracks, insufficient fusion, and dimensional inaccuracies can arise during the layer-by-layer fabrication, directly affecting the performance of the final component [3]. To address this issue, non-destructive testing and image-based monitoring techniques have been widely investigated [3,4]. However, most existing approaches rely on post-processing inspection or offline analysis, which limits their ability to detect defects early or provide feedback during the printing process [3,5,6].

Recent studies have examined in situ monitoring and image-based registration techniques for real-time quality assessment [4,7–10]. Although these methods have demonstrated promising results, they have two key limitations. First, many deformation reconstruction and image registration approaches are computationally expensive, making it difficult to integrate them within the time constraints of the printing process [4,7]. Second, existing methods are typically designed for controlled or synthetic data and may not generalize well to real-image observations, which often introduce additional challenges because of noise, segmentation errors, and viewpoint inconsistencies [1,4,8,9,11].

Hence, there is a need for computational frameworks that not only reduce the runtime of deformation reconstruction tasks but also integrate robust image-based alignment methods that can

Published under an exclusive license by KFUPM

Table 1. *Comparison of representative studies related to the proposed method*

Work	Problem Addressed	Method	Key Limitation	Difference from This Study
Neural 3D Mesh Renderer [8]	Recover 3D shape from images	Differentiable rendering for mesh optimization	Not designed for AM or real-time constraints	Applies differentiable rendering to AM quality control with periodic updates and real-image pipeline
Soft Rasterizer: A Differentiable Renderer for Image-based 3D Reasoning [9]	Image-based 3D reconstruction	Soft rasterization for gradient-based optimization	Focuses on graphics tasks, not AM workflows	Extends differentiable rendering to deformation reconstruction under AM constraints
CLAIRE: A Distributed-Memory Solver for Large Deformation Diffeomorphic Image Registration [12]	Large-scale image registration	GPU-accelerated diffeomorphic registration	Operates on volumetric data, not mesh deformation	Uses mesh-based FFD and image-driven deformation for AM quality control
Review of in-situ process monitoring and in-situ metrology for metal AM [10]	In-situ monitoring in AM	Review of monitoring and metrology methods	Focuses on sensing and detection, not geometric reconstruction	Adds deformation reconstruction for geometry estimation under real-time constraints
Review on image registration methods for the quality control in AM [4]	Image registration in AM QC	Survey of registration methods	No unified real-time reconstruction framework	Proposes a fast workflow integrating deformation, rendering, and real data
This study	In-time deformation reconstruction for AM QC	FFD + differentiable rendering + adaptive loss + GPU	———————	———————————

be applied under real-world conditions. Practical quality control systems must balance accuracy, computational efficiency, and robustness to imperfect observations, while remaining compatible with the temporal constraints of AM processes.

To position our study within the context of existing literature, Table 1 lists representative studies conducted on differentiable rendering, image registration, and AM monitoring, highlighting their limitations and relations to this study.

As listed in Table 1, existing approaches address only individual aspects of image-based quality control. They do not provide an integrated framework for efficient deformation reconstruction under real-time conditions using real-image inputs. This limitation constitutes the research gap addressed in this study, which introduces periodic reconstruction strategy and real-image pipeline under runtime constraints

The primary objective of this study is to develop an efficient framework that enables image-based deformation reconstruction in accordance with the time constraints of in-process or near-real-time quality control. This study proposes a deformation reconstruction workflow that integrates free-form deformation (FFD), differentiable rendering, and an adaptive loss formulation to enable robust mesh-to-image alignment for both synthetic- and real-image inputs.

Specifically, this study introduces a periodic group-based reconstruction strategy in which deformation updates are performed at controlled intervals rather than for every layer, ensuring that the method is computationally feasible in practice. In addition, this study develops a real-image-based deformation pipeline that combines image segmentation with adaptive weighting and regularization to improve robustness when aligning a reference mesh to real-world observations. The proposed framework is evaluated using both a synthetic multi-view benchmark and a real-image single-view test case, demonstrating its ability to achieve high reconstruction accuracy while substantially reducing computational cost.

2. METHODOLOGY

2.1. System Overview and Problem Definition. This study developed an efficient deformation-reconstruction framework compatible with the time constraints of in-process or near-real-time monitoring for quality control in AM. Although online monitoring methods aim to enable real-time process control, their practical deployment requires computational pipelines that are fast and robust to imperfect observations.

The objective of this study can be defined as follows: given a reference mesh corresponding to the nominal computer-aided design (CAD) geometry and a set of images of the printed or deformed object, estimate a deformed mesh whose rendered projections best match the observed images. This problem can be formulated as an image-based deformation reconstruction task, which is closely related to image registration, in which the geometric and image domains are aligned to quantify deformation and shape discrepancies.

The proposed framework integrates FFD, differentiable rendering, and an image-based optimization strategy to iteratively update the mesh geometry. Unlike traditional approaches that

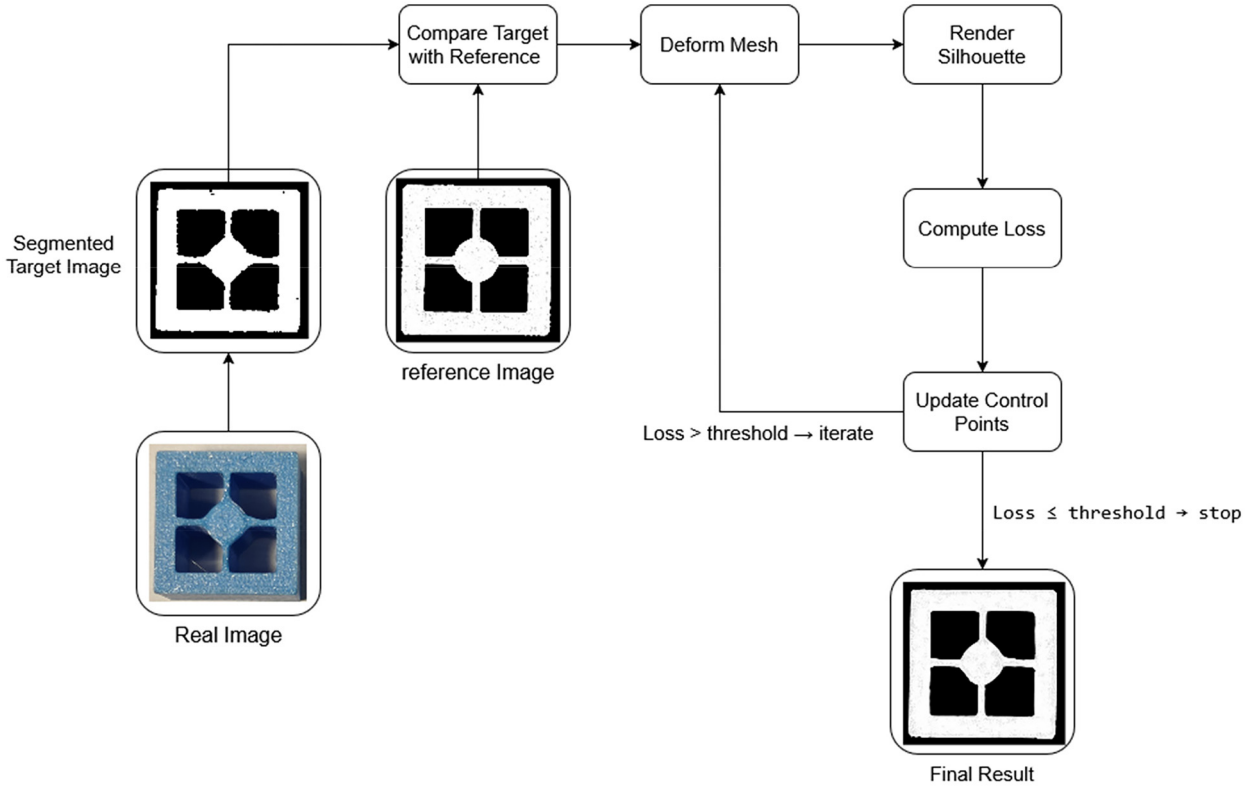


Figure 1. Workflow of the proposed deformation-reconstruction framework for real-image inputs

rely solely on similarity metrics, such as normalized cross-correlation (NCC), this framework incorporates an adaptive loss formulation to improve robustness to real-image inputs, which often compromise reconstruction accuracy due to segmentation noise and partial observations.

Two test cases were considered to evaluate the framework. The first comprised a synthetic multi-view setup, which was used to benchmark the reconstruction accuracy and computational performance of the framework under controlled conditions. The second was a real-image single-view test case in which a target mask was produced by segmenting a photograph captured using a mobile phone. Using the proposed optimization framework, the reference mesh was deformed such that its projection matched the observed image. Figurre 1 illustrates the workflow of the proposed deformation-reconstruction framework for real-image inputs.

As shown in Figurre 1 , the proposed framework uses a reference mesh and a target input, which consists of a segmented real image. The mesh is first rendered to obtain an initial projection and then iteratively deformed using FFD. At each iteration, the rendered silhouette is compared with the target input using an image-based loss function, and the control points are updated until convergence is reached. The final output is a reconstructed mesh that best matches the target image. The detailed flowchart is shown in Figure 2.

2.2. Data Generation and Image Acquisition. A synthetic experimental setup was used to validate the accuracy of the proposed framework before deploying it for monitoring real data. A reference mesh and simulated deformed mesh were loaded from the OBJ files. Multi-view silhouette images were generated by rendering the deformed mesh from four azimuth angles: 0°, 90°, 180°, and 270°. The generated images were then converted to grayscale images and normalized to the range $[0, 1]$ before similarity evaluation. A cube was deliberately selected as the reference geometry to provide a simple and controlled benchmark for developing and evaluating the proposed elastic registration and reconstruction framework. Its planar surfaces, straight edges, and defined corners facilitate the identification, visualization, and debugging of registration errors. Geometric simplicity also reduces confounding effects associated with complex surface features allowing the behavior of the proposed method to be examined more directly. The cube can be readily modeled, meshed, manufactured, and measured using commonly available tools. This makes the experimental procedure easier for other researchers to reproduce.

In addition to the synthetic test case, a real-image test case was conducted to assess the applicability of the framework to experimentally acquired images. Object B, defined in Section 3.2, was used for this experiment. The ground-truth deformation of Object B was created manually from its nominal reference geometry using Autodesk Meshmixer. The resulting deformed geometry was then 3D printed, providing a physical realization of the known ground-truth deformation. A top-view photograph of the printed Object B was captured using the main camera of a OnePlus 13 smartphone, equipped with a 50 MP Sony LYT-808 sensor with a 1/1.43-inch sensor size and an (f/1.6) aperture. During image acquisition, the camera was positioned approximately 25 cm from the object. The recorded camera settings were an (f/1.6) aperture, (1/120) s exposure time, ISO 50, 6 mm focal length, and 6.3568×digital zoom. The image metadata reported a 146 mm equivalent focal length.

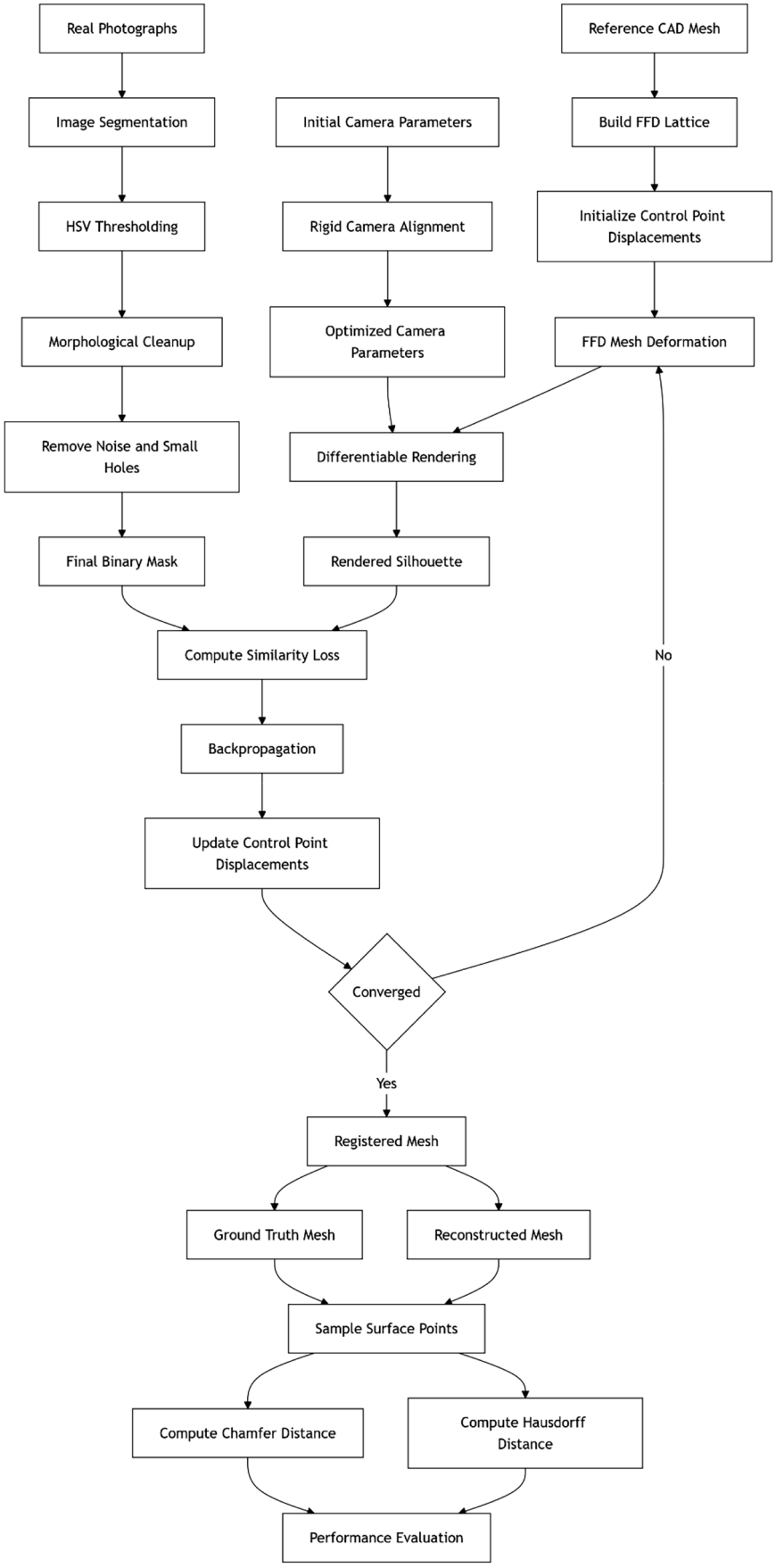


Figure 2. Detailed flowchart for the proposed framework

The original photograph had a resolution of (3076 x 4096) pixels and was cropped to (1474 x 1474) pixels to remove unnecessary background regions while retaining the printed object. The processed image had a bit depth of 24 bits. Because this experiment focused on the top face of Object B, a single-view configuration was used. The cropped photograph was segmented using hue, saturation, and value (HSV)-based thresholding, followed by morphological cleanup and hole removal to generate a binary object mask. This mask served as the image-space input for the deformation-reconstruction process. Although the ultimate objective is to achieve deformation reconstruction during printing, the present framework is designed for periodic monitoring, for example, after every 10 to 20 printed layers, to balance computational cost and monitoring benefits.

2.3. FFD Model. Mesh deformation was represented using a box-shaped FFD lattice defined by a grid of control points. Let the control grid size be (l,m,n), and let $\mathbf{P}_{ijk} \in \mathrm{R}^3$ denote the control point at index (i,j,k). The reference mesh was tightly enclosed within the lattice, and the deformation was parameterized by learnable control-point displacements $\Delta\mathbf{C}_{ijk}$.

Each mesh vertex $\mathbf{x}$ was mapped onto the normalized local coordinates (s,t,u) inside the lattice. The post-deformation vertex $\mathbf{x}''$ was computed using trivariate Bernstein blending:

$$\mathbf{x}'(s,t,u) = \sum_{i=0}^{l-1}\sum_{j=0}^{m-1}\sum_{k=0}^{n-1} B_i^{l-1}(s)\, B_j^{m-1}(t)\, B_k^{n-1}(u) \times \left(\mathbf{P}_{ijk} + \Delta\mathbf{C}_{ijk}\right), \quad (1)$$

where $\boldsymbol{B}_{\mathbf{i}}^{\mathbf{p}}(\cdot)$ are Bernstein basis polynomials.

The FFD lattice resolution was selected based on the complexity of each experimental scenario, as described in Section 3.3.

2.4. Differentiable Rendering. To enable direct gradient-based optimization based on image similarity, the silhouettes were rendered using PyTorch3D. For each camera viewpoint v, the renderer produces a predicted silhouette image $\hat{\boldsymbol{I}}_v(\Delta\mathbf{C})$ from the current deformed mesh. The images were rendered using the graphics processing unit (GPU) when available, allowing end-to-end differentiation with respect to the FFD control-point displacements.

In the synthetic test case, a reference camera configuration was set using an initial look-at configuration, after which the viewpoint was rotated around the object to generate four images at different azimuth angles $\{0°, 90°, 180°, 270°\}$. In the real-image test, only a single top-view silhouette was rendered. For this purpose, the camera pose was manually adjusted such that the rendered view was approximately aligned with the viewpoint of the captured mobile-phone image. The rendered silhouette was then compared with the segmented target mask extracted from the real photograph.

2.5. Similarity Metric and Combined Optimization Objective. For each viewpoint, the similarity between the predicted silhouette and the observed target silhouette was measured using normalized cross-correlation (NCC). NCC is suitable for image-registration applications because it measures structural similarity while being relatively insensitive to uniform intensity scaling and offset. For experiments involving multiple viewpoints, NCC was evaluated independently for each view and averaged as.

$$\mathrm{NCC}_{\mathrm{multi}} = \frac{1}{N_v}\sum_{v=1}^{N_v} \mathrm{NCC}\left(R^{(v)}, T^{(v)}\right), \quad (2)$$

where V denotes the number of views. In the synthetic test, $V = 4$.

where $R^{(v)}$ and $T^{(v)}$ denote the rendered and target silhouettes for viewpoint v, respectively, and N_v denotes the number of viewpoints. In the synthetic test case, $N_v = 4$.

For the real-image test case, NCC was retained as an evaluation metric; however, deformation optimization was driven by an adaptive weighted NCC objective combined with deformation regularization. The adaptive weighting was introduced to assign greater importance to image regions in which the rendered silhouette differed most from the segmented target silhouette.

First, the pixel-wise disagreement between the rendered and target silhouettes was calculated as

$$D_i = |R_i - T_i|, \quad (3)$$

where R_i and T_i denote the rendered and target silhouette intensities at pixel i, respectively. The disagreement map was then smoothed using a Gaussian filter:

$$D^G = G_\sigma * D, \quad (4)$$

where G_σ denotes a Gaussian kernel with $\sigma = 5$ pixels and $*$ denotes convolution. The smoothed disagreement map was normalized to the interval $[0,1]$:

$$\hat{D}_i = \frac{D_i^G - \min\left(D^G\right)}{\max\left(D^G\right) - \min\left(D^G\right) + \epsilon}, \quad (5)$$

where $\epsilon = 10^{-8}$ prevents division by zero. The adaptive weight assigned to pixel i was subsequently defined as

$$w_i = w_{\min} + (1 - w_{\min})\,\hat{D}_i, \quad (6)$$

Where $w_{min} = 0.3$. Thus, regions with relatively small disagreement retained a minimum weight of 0.3, whereas regions exhibiting the greatest disagreement received weights approaching 1.0.

Using these adaptive weights, the weighted mean intensities of the rendered and target silhouettes were calculated as

$$\mu_R = \frac{\sum_i w_i R_i}{\sum_i w_i + \epsilon}, \quad \mu_T = \frac{\sum_i w_i T_i}{\sum_i w_i + \epsilon}. \quad (7)$$

The weighted normalized cross-correlation (WNCC) was then calculated as

$$WNCC = \frac{\sum_i w_i (R_i - \mu_R)(T_i - \mu_T)}{\sqrt{\left[\sum_i w_i (R_i - \mu_R)^2\right]\left[\sum_i w_i (T_i - \mu_T)^2\right]} + \epsilon}. \quad (8)$$

To constrain physically implausible deformation of the FFD lattice, two regularization components were included. First, displacement in the z-direction was penalized to limit changes in the

thickness of the reconstructed object:

$$L_z = \frac{1}{N_c}\sum_{j=1}^{N_c}\left(\Delta C_{j,z}\right)^2, \tag{9}$$

where N_c denotes the number of FFD control points and $\Delta C_{j,z}$ denotes the z-component of the displacement of control point j.

Second, a smoothness term was applied to neighboring control-point displacements along the first two FFD lattice directions:

$$L_{\text{smooth}} = \frac{1}{2}\left[\text{mean}\left(\|\Delta C_{i+1,j,k} - \Delta C_{i,j,k}\|_2^2\right) + \text{mean}\left(\|\Delta C_{i,j+1,k} - \Delta C_{i,j,k}\|_2^2\right)\right]. \tag{10}$$

Because the optimization was formulated as a maximization problem, the final objective for the real-image test case was defined as

$$L_{\text{total}} = \text{WNCC} - \lambda_z L_z - \lambda_{\text{smooth}} L_{\text{smooth}}, \tag{11}$$

where $\lambda_z = 100$and $\lambda_{smooth} = 5$. The FFD control-point displacements were updated to maximize L_{total}.

The adaptive weight mask was periodically recomputed during optimization to reflect changes in the disagreement between the rendered and target silhouettes. It was updated every five iterations during iterations 0-49, every seven iterations during iterations 50-99, and every ten iterations thereafter. This adaptive formulation places greater optimization emphasis on regions with persistent silhouette mismatch, whereas the regularization terms constrain excessive thickness variation and spatially inconsistent control point motion.

2.6. Optimization Procedure. The control-point displacement tensor ΔC was defined as a trainable parameter and optimized using the Adam optimizer. In the synthetic test case, the optimization maximized the mean NCC across the four viewpoints, whereas in the real-image test case, it maximized the combined objective L_{total} defined in Eq. (11).

In the synthetic multi-view experiment, optimization was performed at a learning rate of 5×10^{-3}, with a maximum of 250 iterations and an early stopping criterion of $\text{NCC}_{avg} > 0.9999$. During each iteration, the reference mesh was deformed using the current control-point displacements rendered from the four predefined camera views and evaluated using the NCC averaged over all views. The resulting gradients were then backpropagated and used to update the control-point displacements.

In the real-image single-view experiment, the optimization procedure was adapted to the segmented top-view target mask. In this case, the Adam optimizer was used with a learning rate of 5×10^{-4}, and a maximum of 150 iterations. At each iteration, the deformed mesh was rendered from the calibrated top-view camera pose and compared with the segmented target mask using the combined loss described in Section 2.5. The adaptive weight mask was updated periodically during the optimization such that the loss remained focused on the most mismatched regions. At the end of the optimization procedure, the reconstructed deformed mesh was exported in the OBJ format, and the optimized FFD control-point configuration was saved for further analysis.

2.7. GPU Acceleration Strategy and Framework Efficiency. The baseline implementation was developed using a NumPy-based CPU pipeline and compared with the proposed framework implemented in PyTorch and PyTorch3D. In the proposed approach, GPU acceleration was used to support the computational requirements of the deformation-reconstruction workflow.

The framework leverages GPU-based tensor operations, differentiable rendering, and automatic differentiation to efficiently compute image-based loss functions and update deformation parameters. These design choices allowed the framework to perform iterative mesh deformation and rendering within a substantially reduced runtime compared with that of the baseline, thereby making the approach suitable for periodic execution during printing.

By combining efficient computation with adaptive loss formulation and a real-image processing pipeline, the proposed framework extends beyond synthetic reconstruction and demonstrates applicability to real-world observations. This integration enables practical deformation reconstruction under realistic conditions and supports its use in real-time quality control scenarios.

3. EXPERIMENTAL SETUP

3.1. Hardware and Software Environment. All the experiments were conducted on a high-performance workstation to evaluate both the computational efficiency and practical applicability of the proposed GPU-accelerated framework. The workstation was equipped with an Intel(R) Core(TM) i9-14900K CPU, an NVIDIA GeForce RTX 4090 GPU, and 128 GB of RAM. The framework was implemented in a Python-based environment.

The framework was implemented using PyTorch and PyTorch3D, which enabled GPU-based tensor computation, differentiable rendering, and automatic differentiation. The baseline implementation relied on a NumPy-based CPU pipeline. Both implementations were executed on the same machine to ensure fair comparisons.

3.2. Test Data and Evaluation Scenario. Two geometries were used in this study and are referred to as Object A and Object B. Object A denotes the cube-shaped geometry used in the synthetic multi-view experiment, whereas Object B denotes the sliced geometry used in the real-image single-view experiment. Their reference meshes are shown in Figures 3(a) and 3(b), respectively. Both geometries were represented using triangular surface meshes. The mesh of Object A contained 8 vertices and 12 triangular faces, whereas the higher-resolution mesh of Object B contained 3,999 vertices and 8,010 triangular faces.

In the first scenario, Object A was used in a synthetic deformation benchmark to validate reconstruction accuracy and quantify computational performance under controlled conditions. Two meshes were employed: a reference mesh representing the nominal geometry of Object A and a simulated deformed mesh representing its ground-truth deformation. Four silhouette images of the ground-truth mesh were generated from azimuth angles of (0°), (90°), (180°), and (270°), while the elevation angle was held fixed. The silhouettes were converted to grayscale and normalized to the range ([0,1]). This controlled setup enabled

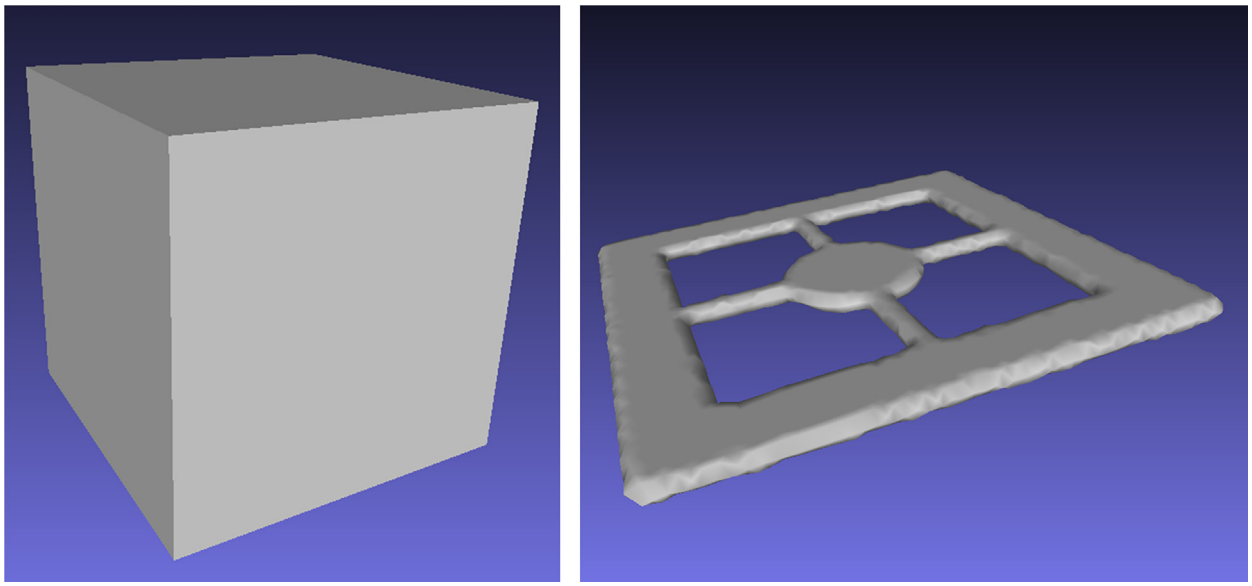


Figure 3. Reference mesh geometries used in this study: (a) Object A, used in the synthetic test case ($20mm \times 20mm \times 20mm$), and (b) Object B, used in the real-image test case ($20mm \times 20mm \times 2mm$)

direct evaluation of reconstruction accuracy, convergence behavior, and runtime without variability arising from real-image acquisition.

In the second scenario, Object B was used to evaluate the framework with experimentally acquired image data. The nominal reference geometry of Object B was manually deformed using Autodesk Meshmixer to establish a known ground-truth geometry, which was subsequently 3D printed and photographed. The proposed framework was provided with the nominal reference mesh and the segmented top-view photograph of the printed deformed object, while the Meshmixer-deformed mesh was retained as the ground truth for geometric evaluation. Chamfer and Hausdorff distances were subsequently computed between the reconstructed and ground-truth meshes.

3.3. Deformation Reconstruction Task. The experimental task in both scenarios was to reconstruct a deformed mesh by optimizing FFD control-point displacements such that the silhouettes of the reconstructed mesh matched the observed target images.

In the synthetic benchmark, the task consisted of matching the rendered projections of the deformed mesh to the four-view silhouettes generated by the simulated printer mesh. In this scenario, a coarse FFD lattice of size (2,2,2) was used, which was sufficient for a simple and controlled deformation setting.

In the real-image test case, the task was formulated as a single-view registration problem in which the rendered top-view silhouette of the deformable reference mesh was matched directly to the segmented binary mask extracted from the real target photograph. In this case, a denser FFD lattice of size (6,6,6) was employed to provide greater local deformation flexibility and to adapt the reference mesh more effectively to the target geometry observed in the real image.

3.4. Evaluation Metric. The quality of the reconstructed mesh was evaluated using image-based and geometric metrics. In the synthetic benchmark, NCC was computed independently for each of the four views, and the values were averaged to yield a single similarity score per iteration. In addition to NCC, the geometric similarity between the reconstructed and ground-truth meshes was evaluated using the Chamfer and Hausdorff distances. The Chamfer distance measures the average point-wise discrepancy between the reconstructed and target surfaces, whereas the Hausdorff distance captures the maximum geometric deviation between them.

In the real-image test case, NCC was used to evaluate the agreement between the rendered silhouette and segmented target mask. Additionally, geometric similarity was assessed using the Chamfer and Hausdorff distances computed between the reconstructed and ground-truth meshes. However, unlike in the synthetic case, the optimization was guided by a combined loss formulation rather than by NCC alone. In both scenarios, high NCC values and low geometric distances indicated high reconstruction quality.

3.5. Optimization Parameters and Iteration Definition. Both implementations used gradient-based optimization to maximize the mean NCC; however, they differ in how the optimization iteration is defined. In the synthetic CPU/GPU comparison, the proposed PyTorch-based GPU implementation evaluated all four views within a single iteration and applied one displacement update after aggregating the similarity contributions. By contrast, the original NumPy-based CPU implementation sequentially processed the four views by applying a displacement update after each view. Consequently, one iteration of the proposed method corresponds to four displacement updates in the baseline implementation. Owing to this difference, the iteration counts were not directly comparable, and performance

comparisons were therefore based primarily on the final NCC values and wall-clock execution time.

For the synthetic benchmark, the NumPy-based CPU implementation was executed at a learning rate of 2×10^{-2} for 60 iterations, whereas the PyTorch-based GPU implementation used a learning rate of 5×10^{-3} for the same number of iterations. In the real-image single-view test case, the optimization settings were adjusted to account for different observation types and denser deformation models. In that case, the PyTorch-based GPU implementation used a learning rate of 5×10^{-4}, with a maximum of 150 iterations.

3.6. Comparison Protocol. To ensure a reliable comparison between the NumPy-based CPU implementation and PyTorch-based GPU implementation in the synthetic benchmarking stage, each implementation was independently executed five times using identical input data and camera configurations. Runtime was measured as the end-to-end execution time, covering deformation updates, rendering, similarity computation, and optimization steps. The final NCC values and the total execution times were recorded for each run.

This benchmarking protocol enables the evaluation of both the numerical convergence behavior and computational speed, which are critical for assessing the feasibility of accelerated deformation reconstruction for real-time quality control. A real-image-based test case was used as a preliminary validation experiment to assess whether the proposed framework could be extended from synthetically rendered targets to experimental observations. As this stage was intended as a proof of applicability rather than a statistical benchmark, it was not included in the five-run CPU/GPU comparison protocol.

4. RESULTS AND DISCUSSION

4.1. Test Case 1: Synthetic Multi-View Reconstruction.

4.1.1. Convergence and similarity performance. Both implementations successfully increased the similarity between the reconstructed mesh projections and target silhouettes, converging to high NCC values. The NumPy-based CPU implementation achieved a final NCC value of 0.9983 in all five runs, whereas the PyTorch-based GPU framework achieved a mean final NCC value of 0.9999 with a standard deviation of 4.0×10^{-7}. All reported runs were performed using 60 iterations.

In addition to image-based similarity, the geometric reconstruction accuracy was evaluated using the Chamfer and Hausdorff distances in the synthetic test case. The proposed framework achieved Chamfer and Hausdorff distances of 0.0111 and 0.0207, respectively, indicating that the reconstructed mesh remained geometrically consistent with the target deformation while preserving a high silhouette agreement. The reported geometric distances were expressed in normalized mesh coordinates.

Although the GPU-based implementation achieved a slightly higher final NCC value, this difference should be interpreted with caution because the two implementations used different learning rates and update strategies. The NumPy baseline sequentially processed the four views and applied a deformation update after each view, whereas the proposed GPU framework evaluated all four views within a single iteration and applied only one aggregated update. Therefore, the runtime and reconstruction quality provide a more meaningful basis for comparison than the raw iteration count.

A direct quantitative comparison of the proposed approach with recent deformation reconstruction and image registration methods is challenging because existing studies often use different object geometries, datasets, sensing modalities, and evaluation protocols. Many GPU-accelerated registration approaches, such as CLAIRE [12], focus on volumetric medical image registration rather than mesh-based deformation reconstruction under AM constraints. Similarly, differentiable rendering approaches such as Neural Mesh Renderer [8] and Soft Rasterizer [9] primarily address general 3D reconstruction problems and do not consider periodic reconstruction or real-image segmentation pipelines for AM quality control. Therefore, the comparisons performed in this study were based on the original baseline implementation, and the proposed framework was conceptually positioned relative to existing approaches through a literature comparison, as presented in Table 1.

4.1.2. Runtime performance and speedup. As summarized in Table 2, the PyTorch-based GPU implementation markedly outperformed the original NumPy-based CPU implementation. The mean runtime decreased from 692.65 s to 3.81 s, corresponding to a speedup of 181.80 times. This reduction of the runtime from minutes to seconds suggests that the proposed framework substantially improves the feasibility of practical deformation reconstruction for near-real-time quality control.

4.1.3. Visual comparison between CPU and GPU reconstructions. Figure 4 visually compares the reconstructed silhouettes obtained using the original NumPy-based CPU implementation and the proposed PyTorch-based GPU implementation. The figure shows that both methods produce highly similar projections, indicating that GPU-based acceleration preserves the reconstruction quality of the original approach.

This visual agreement is consistent with the high NCC values reported previously and confirms that the proposed implementation improves computational efficiency without compromising deformation accuracy in the synthetic benchmark.

4.2. Test Case 2: Real-Image Single-View Reconstruction. To assess whether the proposed framework can be extended beyond synthetic rendered observations, a second test was conducted using a real image captured with a mobile phone. In this setup, only a single top-view photograph of the target object was used. The image was segmented using HSV-based thresholding and post-processed to produce a clean binary mask. The original reference mesh was then deformed such that its rendered top-view silhouette matched the segmented target mask.

Unlike the first test, this experiment was not intended as a benchmarking study but rather as a preliminary validation of the

Table 2. *Comparison of execution time between CPU- and GPU-based implementations.*

Method	NumPy (CPU) Time (s)	PyTorch (GPU) Time (s)
Mean	692.65	3.81
Std	33.98	0.11
Min	630.71	3.73
Max	725.51	4.02

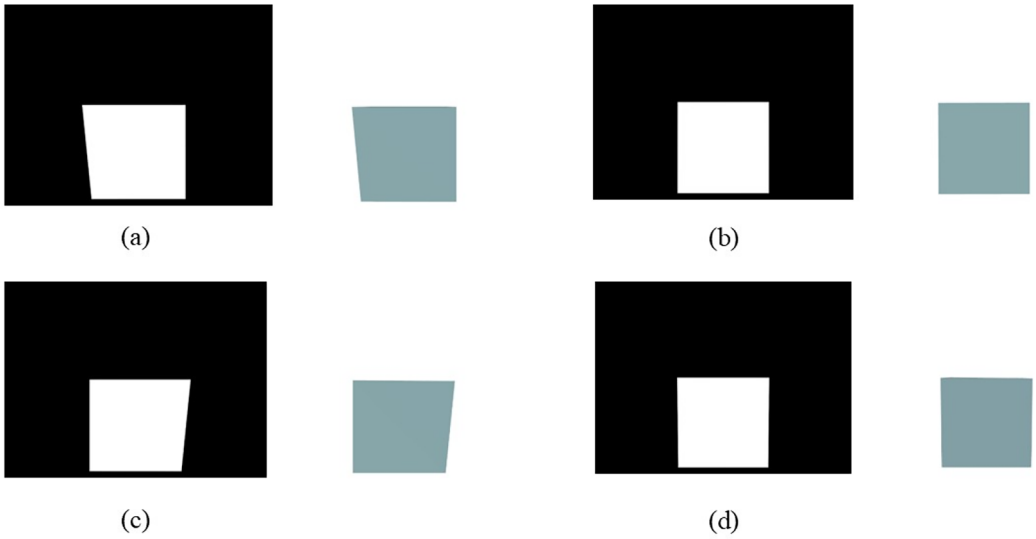


Figure 4. Comparison of reconstructed silhouettes for Test Case 1 obtained using the original NumPy-based CPU implementation (white) and the proposed PyTorch-based GPU implementation (blue-gray) at (a) 0°, (b) 90°, (c) 180°, and (d) 270°azimuth angles

applicability of the proposed framework to real images. In contrast to the synthetic benchmark, framework optimization for real-image inputs was performed using a combined loss formulation that incorporated adaptive weighting and regularization to improve robustness to real-image imperfections.

Figure 5 compares the final top-view projection of the deformed mesh with the target image. The optimization yielded an NCC value of 0.9729, indicating a high degree of similarity between the reconstructed projection and the segmented target observation. In addition, the framework achieved Chamfer and Hausdorff distances of 0.0013 and 0.0035, respectively, demonstrating that the reconstructed geometry remained close to the target shape under real-image conditions.

This result demonstrates that the proposed deformation-reconstruction framework can be extended from idealized synthetic silhouettes to experimental observations after image segmentation. Unlike traditional image-registration approaches that operate only in the image space, the proposed framework simultaneously preserves geometric consistency through mesh-based deformation and geometric regularization. Although the experiment was limited to a single real image, it demonstrated the feasibility of applying the proposed framework to real observations and highlighted the potential of extending the framework to more complex multi-view and in-situ monitoring scenarios.

4.3. Practical Implications for In-Time Quality Control. As performing deformation reconstruction after every individual printed layer remains prohibitively expensive, this study adopted a more practical deployment strategy: perform reconstruction periodically, for example, after every 10 to 20 layers. Under this strategy, the GPU runtime measured approximately 4 s, suggesting that the proposed framework is compatible with near-real-time quality-control updates.

Such periodic deformation-estimation methods can support the early identification of deformation trends and possible print

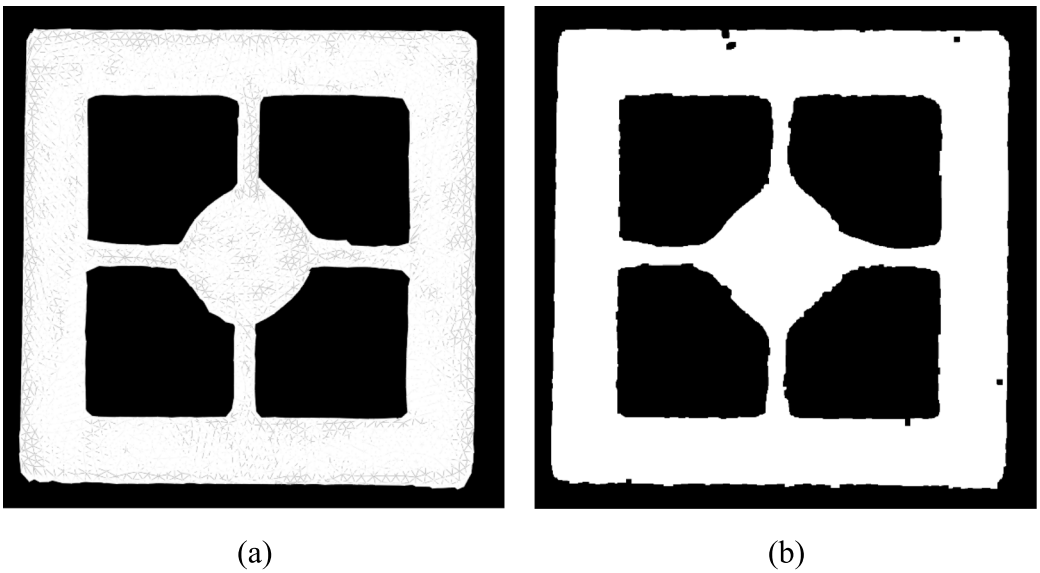


Figure 5. (a) Final top-view projection of the deformed mesh after optimization. (b) Segmented target image used for the deformation process

defects before printing is completed. This provides a realistic compromise between computational efficiency and monitoring effectiveness, particularly for AM processes in which per-layer execution is not strictly necessary.

The proposed PyTorch-based GPU implementation exhibited a higher speed than a recently reported framework for elastic registration [6], which required an execution time of 100–200 s for in-time quality control.

4.4. Limitations. The first test case provided strong quantitative evidence of acceleration and high-quality reconstruction; however, it was based on a synthetic environment with rendered silhouettes and therefore does not fully capture the variability of real imaging conditions. The second test case addressed this limitation by introducing a real captured image; however, it was restricted to a single top-view observation.

Therefore, the present validation is preliminary. Factors such as illumination variation, segmentation quality, camera calibration error, background clutter, and viewpoint mismatch may affect the reconstruction robustness in practical settings. In addition, the real-image test case in this study was limited to the top face of the object, leaving the side surface deformation unobserved. Therefore, future studies should extend the framework to multi-view real-image acquisition and full 3D deformation validation under real-world printing conditions.

5. CONCLUSIONS

This paper presents a deformation-reconstruction framework for improving the efficiency and robustness of image-based quality control in AM. The proposed framework integrates FFD, differentiable rendering, and adaptive loss formulations to enable mesh-to-image alignment under both synthetic and real-image inputs. Unlike approaches that focus solely on computational acceleration, this study emphasized the development of a unified workflow that balances accuracy, robustness, and computational efficiency.

The results from the synthetic multi-view test case demonstrated that the proposed framework considerably reduced runtime compared with that of the baseline, while maintaining high reconstruction quality. The mean execution time was reduced from 692.65 s to 3.81 s, corresponding to a 181.80-fold increment in speed. In addition, the framework achieved Chamfer and Hausdorff distances of 0.0111 and 0.0207, respectively, confirming that the reconstructed geometry remained well-consistent with the target deformation.

The real-image test case further demonstrated that the framework can be extended to experimental observations using segmentation and an adaptive loss formulation. In this case, the framework achieved an NCC value of 0.9729, together with a Chamfer distance of 0.0013 and Hausdorff distance of 0.0035, demonstrating successful deformation of the nominal mesh toward the observed target geometry under real-image conditions.

Overall, the proposed framework provides an effective approach for deformation reconstruction under realistic conditions and supports the use of periodic near-real-time quality control in AM. Future studies will focus on extending the framework to multi-view real-image data, improving the robustness to imaging variations, and validating the method on real printed parts under in-process monitoring conditions.

AFFILIATIONS AND AUTHOR DETAILS

Undergraduate Author

Ahmed Malaekah – *Department of Computer Engineering, King Fahd University of Petroleum & Minerals, Dhahran 31261, Saudi Arabia*; 0009-0000-1032-2781
Email: s202247540@kfupm.edu.sa

Corresponding Author

Ammar Alsheghri – *Research Mentor, Assistant Professor, Department of Mechanical Engineering and Interdisciplinary Research Center for Biosystems and Machines, King Fahd University of Petroleum & Minerals, Dhahran 31261, Saudi Arabia*; 0000-0001-5651-7103
Email: ammar.sheghri@kfupm.edu.sa

ACKNOWLEDGEMENTS

The authors acknowledge the support provided by the Biosystems and Machines Interdisciplinary Research Center at King Fahd University of Petroleum and Minerals. The authors also extend their appreciation to Dr. Ekrem Ekici for technical support.

REFERENCES

(1) Baumann, F. & Roller, D. Vision based error detection for 3D printing processes. *MATEC Web Conf.* **59**, 06003 (2016).

(2) Jadhav, A. & Jadhav, V. S. A review on 3D printing: An additive manufacturing technology. *Mater. Today Proc.* **62**, 2094–2099 (2022).

(3) Segovia Ramírez, I., García Márquez, F. P. & Papaelias, M. Review on additive manufacturing and non-destructive testing. *J. Manuf. Syst.* **66**, 260–286 (2023).

(4) Chand, K., Fritsch, T., Oster, S., Ulbricht, A. & Bruno, G. Review on image registration methods for the quality control in additive manufacturing. *Prog. Addit. Manuf.* **10**, 4647–4673 (2025).

(5) Valiente, J. M., López, F., Acebrón, F. & Pérez, E. AN IMAGE REGISTRATION METHOD FOR CERAMIC TILE INSPECTION PURPOSES.

(6) Alsheghri, A. *et al.* Elastic Registration for Non-Destructive Testing of Additively Manufactured TPMS Lattice Structures. *J. Nondestruct. Eval.* **45**, 52 (2026).

(7) Alsheghri, A. *et al.* Optimization of 3D network topology for bioinspired design of stiff and lightweight bone-like structures. *Mater. Sci. Eng. C* **123**, 112010 (2021).

(8) Kato, H., Ushiku, Y. & Harada, T. Neural 3D Mesh Renderer. Preprint at https://doi.org/10.48550/ARXIV.1711.07566 (2017).

(9) Liu, S., Li, T., Chen, W. & Li, H. Soft Rasterizer: A Differentiable Renderer for Image-based 3D Reasoning. Preprint at https://doi.org/10.48550/ARXIV.1904.01786 (2019).

(10) Everton, S. K., Hirsch, M., Stravroulakis, P., Leach, R. K. & Clare, A. T. Review of in-situ process monitoring and in-situ metrology for metal additive manufacturing. *Mater. Des.* **95**, 431–445 (2016).

(11) Fenta, E. W. & Alsheghri, A. Exploring 4D printing for biomedical applications: Advancements, challenges, and future perspectives. *Bioprinting* **50**, e00436 (2025).

(12) Mang, A., Gholami, A., Davatzikos, C. & Biros, G. CLAIRE: A Distributed-Memory Solver for Constrained Large Deformation Diffeomorphic Image Registration. *SIAM J. Sci. Comput.* **41**, C548–C584 (2019).