Research Square

Preprints are preliminary reports that have not undergone peer review.
They should not be considered conclusive, used to inform clinical practice,
or referenced by the media as validated information.

Dynamic triangle splatting with repaired position gradients

Samir Hossain

samir.hossain@ttu.edu

Texas Tech University

Research Article

Keywords: triangle splatting, dynamic scene reconstruction, differentiable rendering, radiance fields, novel view synthesis

Posted Date: November 23rd, 2026

DOI: https://doi.org/10.21203/rs.3.rs-11603091/v1

License: This work is licensed under a Creative Commons Attribution 4.0 International License.
Read Full License

Additional Declarations: No competing interests reported.

Dynamic triangle splatting with repaired position gradients

Samir Hossain
Department of Electrical and Computer Engineering, Texas Tech University, Lubbock, TX, USA
ORCID: 0009-0003-8986-0946. Email: samir.hossain@ttu.edu

Abstract

Triangle splatting represents radiance fields with the same primitive that meshes, game engines, and physics simulators consume. A deforming triangle soup is therefore an animated, simulation-ready mesh with no conversion step. This paper presents the first reconstruction of continuously deforming scenes with triangle splatting primitives. The attempt exposed a defect worth reporting on its own. Under the released Triangle Splatting+ rasterizer, no deformation model could learn limb-scale motion across sixteen controlled configurations. A finite-difference audit located the cause in the backward pass, which discards the interior barycentric position gradients it computes and mis-scales the remaining boundary term. Analytic vertex gradients sit two to five orders of magnitude below their true values. We repair both defects and validate the repair at three levels: agreement with finite differences, recovery of a displaced triangle where the released code fails, and a controlled comparison in which an identical training configuration gains 3.2 dB and first shows articulated motion. On the eight D-NeRF scenes the method reaches 29.84 dB at the benchmark's original resolution, above the founding baseline on four scenes, while rendering at up to 464 frames per second on one consumer GPU. Soft-body simulation runs directly on the reconstruction's vertices. We release the repaired rasterizer, training code, and audit tools.

Keywords: triangle splatting, dynamic scene reconstruction, differentiable rendering, radiance fields, novel view synthesis

1. Introduction

Splatting methods have reshaped radiance-field reconstruction. 3D Gaussian Splatting [1] replaced volumetric ray marching with rasterized primitives, and within a year the dynamic-scene ports followed [2, 3, 4]. More recently the community has revisited the primitive itself. Triangle Splatting [5] and Triangle Splatting+ [6] optimize soups of triangles by differentiable rasterization. They show that the oldest primitive in graphics is competitive for view synthesis while remaining compatible with mesh-based tooling. A follow-up wave has begun here too, with SLAM [7] and feed-forward reconstruction [8] already demonstrated.

Dynamic scenes are missing from that wave, and they are where triangles have the strongest case. A time-varying Gaussian soup requires lossy mesh extraction before use in simulation, and physics coupling requires a proxy mesh kept synchronized with the splats [9]. A deforming triangle soup needs neither. The reconstruction is the animated mesh, with shared-vertex connectivity that supports exact mesh regularizers and direct simulation (Fig. 1). This paper builds that method.

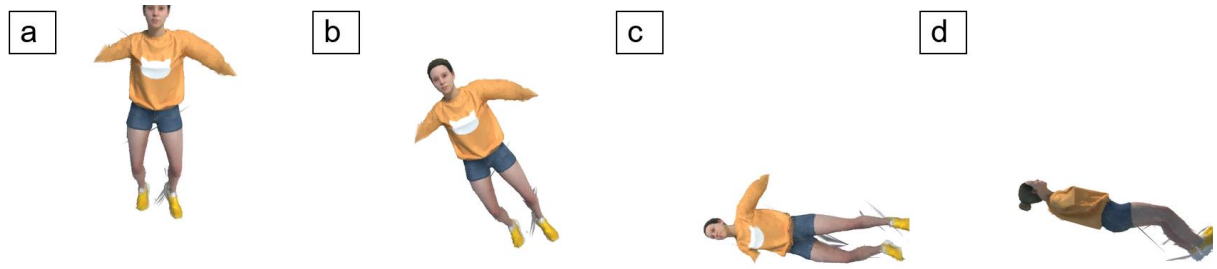

Figure 1. Simulation on the reconstruction. A scene reconstructed from monocular video drops under gravity, topples, and comes to rest (a to d). Shape-matching dynamics run on the reconstruction's own vertices and every state renders through the same rasterizer, at 24 to 52 frames per second end to end on one consumer GPU. The system, Dynamic Triangle Splatting, follows the recipe of the Gaussian dynamic literature: a canonical model trained jointly with a time-conditioned deformation field, here sparse control nodes with learned blend-skinning weights acting on mesh vertices.

The attempt surfaced a problem that matters beyond this paper. Under the released Triangle Splatting+ rasterizer, no deformation model we tried learned limb-scale motion. Sixteen controlled configurations, spanning deformation architectures, regularizer families, and schedules, stayed within a 1.7 dB band. A 500-thousand-parameter per-vertex deformation network converged to a time-invariant field with displacements three orders of magnitude below the scene's motion. Deformation learning receives its supervision entirely through vertex-position gradients. Instrumenting that path explained the record. The analytic gradients disagree with finite differences by factors of one hundred to three hundred thousand, with uncorrelated signs.

A code-level audit then located two defects in the released backward pass. The kernel that converts screen-space gradients to vertex gradients receives the accumulated interior barycentric position gradients as an argument and never reads them. The surviving boundary pathway omits a device-coordinate-to-pixel scale, which attenuates it by roughly half the image width. Neither defect is visible in static training, where densification and pruning do the geometric work and positions only fine-tune. The released repository does contain an unresolved issue (issue 25, opened 24 March 2026 and still open without a reply as of 2 November 2026) reporting that the

fixed-opacity ablation reproduces far below the published numbers, consistent with our finding. We repair both defects and validate the repair at the gradient, unit, and system level. The identical dynamic-reconstruction configuration improves from 23.98 dB to 27.18 dB, and its renders articulate limbs for the first time in our experimental record.

With working gradients, the dynamic method becomes effective. Three mesh-native components matter most. Rigidity regularizers are evaluated with the skinning weights treated as constants, which stops them from erasing motion through the weights. An alpha-mask coverage loss, balanced against a hinge penalty on extreme edge stretch, recovers fast thin geometry that photometric supervision loses. A depth-gradient damping term suppresses a drift that the audit isolates in the depth component of the repaired gradients.

Our contributions are (1) a finite-difference audit methodology that localizes two previously unreported defects in the released Triangle Splatting+ backward pass, (2) a repair validated causally at three levels and released as patches, (3) the first continuous dynamic scene reconstruction on triangle splatting primitives, with learned skinning and mesh-native losses, and (4) a controlled experimental record covering benchmark results, ablations, efficiency, simulation, and failure analysis, with negative results reported.

Gaussian systems remain 5 to 12 dB ahead on synthetic photometric fidelity, and we state that gap plainly throughout. What the triangle representation delivers at its current fidelity is a reconstruction that is already a mesh: it renders at rasterization rates and simulates without conversion.

2. Related work

2.1 Splatting primitives

3D Gaussian Splatting [1] introduced point-based differentiable rasterization for radiance fields, following neural-field formulations [10, 11, 12]. Surface-aligned variants [13], anti-aliased formulations [14], anchored layouts [15], and alternative kernels followed, including convex polytopes [16], beta kernels [17], and triangles [5, 6]. Triangle Splatting+ constrains the soup to shared vertices with opaque, hardening triangles and imports its reconstructions into a game engine directly. Concurrent work pursues adjacent goals in the static setting. TriSplat [18] produces simulation-ready triangle scenes feed-forward from sparse views, DiffSoup [19] optimizes binary-opacity soups through stochastic masking, and DMesh [20] studies differentiable mesh connectivity. ArtMesh [21] reconstructs articulated objects as part-wise rigid triangle meshes between two states from multi-view input. None of these addresses continuous deformation of full scenes from monocular video, which is the setting here.

2.2 Dynamic scene reconstruction

Deformation-field methods warp a canonical representation with a time-conditioned field. The line runs from NeRF-based formulations [22, 23, 24] through accelerated grids [25, 26, 27] to Gaussian instantiations that currently lead the D-NeRF benchmark [2, 3, 28, 29]. SC-GS [4] is the closest in design: sparse control points with learned skinning drive per-Gaussian motion, with an as-rigid-as-possible energy approximated over the unstructured point set. Our deformer adopts that design but operates on true mesh vertices. Rigidity energies are therefore evaluated exactly on the triangle graph [30, 31], and the skinned object is itself the renderable, simulatable mesh. Physics-coupled Gaussian systems either embed splats in a simulation proxy [32, 33] or maintain an extracted mesh synchronized with the splats [9]. The stated limitations of the latter, extraction error, correspondence drift, and dual-representation overhead, do not arise when the reconstruction is the mesh.

2.3 Gradients of differentiable rasterizers

Visibility makes rasterization gradients subtle. Boundary terms require edge sampling [34], reparameterization [35], analytic antialiasing [36], softening [37], or micro-edge constructions [38]. A newer thread audits the released splatting systems themselves. StopThePop [39] diagnoses and repairs sorting approximations in the 3DGS rasterizer. Mip-Splatting [14] diagnoses its sampling artifacts. An error analysis of the local-affine projection yields a corrected projection [40], AbsGS [41] shows the densification gradient suffers sign cancellation, and the gsplat library documents deviations found while rederiving the 3DGS gradients [42]. Our audit belongs to this thread with one difference of kind. The defects we report are not approximations with subtle consequences. One computed gradient buffer is never read, and a scale factor is missing, on the pathway that carries interior position information for attribute-varying primitives. Our claims concern the publicly released implementation at the commits identified in Section 4.3; we make no claim about the code used for the original paper's experiments.

3. Triangle Splatting+ preliminaries

Triangle Splatting+ [6] represents a scene as a soup of triangles with shared vertices. Each triangle carries per-vertex opacity weights, spherical-harmonic appearance, and a window parameter sigma that controls edge softness. Rendering computes a normalized barycentric quantity per pixel and raises it to a power controlled by sigma. Large sigma gives soft footprints; sigma near zero gives hard, opaque triangles. Optimization anneals sigma toward hardness while a topology engine densifies from a spatial Delaunay structure, prunes low-importance and oversized triangles on a fixed cadence, and ratchets a global opacity floor upward, so the final soup exports directly. Two properties matter later. Appearance varies over each triangle through its barycentric coordinates, so interior pixels carry genuine position gradients; this pathway

exists in the released backward mathematics. And the engine's schedules encode static-scene assumptions that interact with deformation.

4. Position gradient audit and repair

4.1 Symptom

We report the development history as evidence. Sixteen training configurations for the method of Sect. 5 varied the deformation architecture, the regularizer family and weight, the skinning parameterization, and the schedules. Test PSNR on the jumpingjacks scene stayed between 22.5 and 24.2 dB, and limbs never articulated. The most diagnostic observation was the per-vertex deformation network. With ten positional frequencies and learning rates spanning an order of magnitude, it converged to a time-invariant field with mean displacement of 0.005 scene units, on a scene whose motion spans about one unit. A deformer this dead under every architecture indicts the gradient path, not the models.

4.2 Finite-difference audit

We instrument the path directly. For a trained canonical model, a fixed view, photometric loss L, and vertex tensor v, we compare the analytic directional derivative against the central finite difference of L along random unit directions. On the released code the two disagree by factors of 100 to 300,000 with uncorrelated signs. The analytic gradient over 23 thousand vertices has total norm 3e-5, and 55 percent of its components are exactly zero. The loss surface has real slope that the backward pass does not report.

A minimal reproduction isolates the mechanism. We render one textured triangle, displace it, and ask gradient descent to recover the pose (Fig. 2). Under the released backward this fails from every initialization and window softness we tried, with an analytic-to-numeric ratio near 200 at sigma 0.3. The failure therefore lives in the per-triangle gradient path, not in occlusion, sorting, or densification.

4.3 Defects in the released backward

The audited code is the public release, pinned as follows. The rasterizer is the diff-triangle2-rasterization repository at commit 42e62aaf03f87835105ed84dd8ed513efd458716 (8 October 2025), included as a submodule of the triangle-splatting2 repository at commit fcde03dd0c3953ce1d47b210485a5e8a9e14b016 (9 October 2025). Both belong to the trianglesplatting2 GitHub organization, were cloned on 5 July 2026, and had no later commit as of 2 November 2026. The SHA-256 digest of the unmodified cuda_rasterizer/backward.cu at that

commit, also recorded in our archived repository so that the audited state can be confirmed independently of later upstream changes, is

8ceaf7c0199dd76b914b6269428cde76598e93e6371228a7c1f4008a948860aa

Reading the CUDA backward with the audit in hand locates two independent defects. First, the per-pixel backward correctly accumulates the loss gradient with respect to each triangle's projected vertex positions, arising from barycentric attribute interpolation, into a per-vertex screen-space buffer. The kernel that converts screen-space quantities into 3D vertex gradients receives this buffer as a parameter and never reads it. For attribute-varying opaque triangles, interior pixels are the dominant carrier of position information, and their entire contribution is discarded. Second, the surviving pathway differentiates the soft window through edge functions built in pixel coordinates, and its conversion to 3D omits the derivative of pixel with respect to normalized device coordinate. That factor is half the image width, about 128 at our 256-pixel unit test and 400 at benchmark resolution, matching the attenuation the audit measures.

The repair converts the interior buffer through the projection Jacobian, adds it to the vertex gradients, and restores the missing scale. It changes about sixty lines and no forward computation; rendered images are bit-identical before and after.

4.4 Validation

At the gradient level, the analytic-to-numeric ratio on the unit test moves from about 200 to between 0.8 and 2.2 at sigma 0.3, with correct signs. Residual disagreement at hard sigma is the discontinuity component that smooth backward passes cannot carry and boundary-integral methods address [34]. The interior pathway survives sigma hardening while the window pathway vanishes, so the repaired rasterizer keeps usable position gradients where the released one loses them.

At the unit level, the displaced triangle recovers its pose to a residual of 0.0005 scene units, from an initial displacement of 0.101, in 300 steps with depth components masked. Unmasked, optimization drifts along the view ray. Per-view depth is weakly observable, and an adaptive optimizer rescales its small biased gradients into full steps. This observation motivates the damping term of the method and remains a caveat of the repair.

At the system level, holding the entire training configuration fixed and swapping only the backward, jumpingjacks improves from 23.98 to 27.18 dB and limbs articulate. The sixteen-run plateau retrodicts cleanly. Architectures whose demand concentrates in position space were fully starved, while the control-node model, which aggregates thousands of vertex gradients into a few hundred transforms, extracted barely enough signal to move.

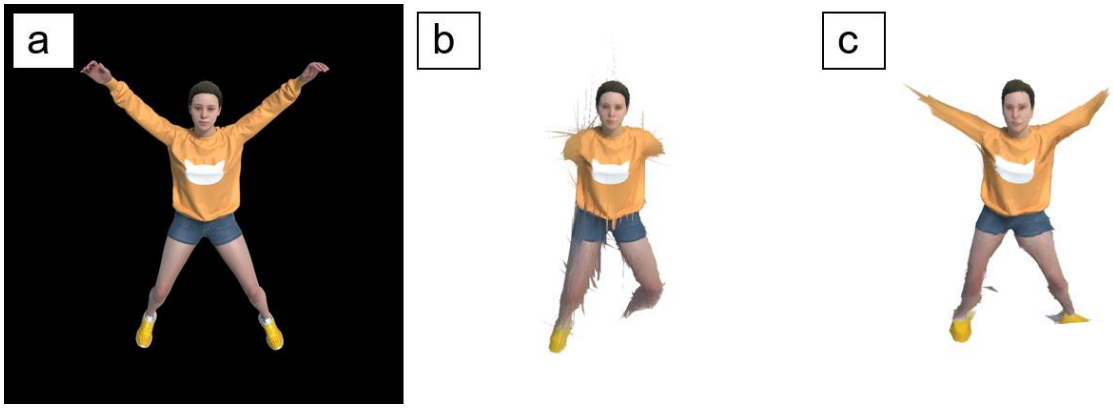

Figure 2. The gradient repair, with everything else fixed. Ground truth (a); the converged result under the released backward, where limbs never leave the mean pose (b); the same training configuration under the repaired backward (c), 3.2 dB higher and articulated.

5. Dynamic reconstruction method

Given monocular video with known cameras and timestamps t in [0, 1], we reconstruct a canonical triangle model and a deformation field mapping canonical vertices to each timestamp. The design tracks the Gaussian dynamic literature where possible, so that the representation, not an exotic pipeline, explains the results.

5.1 Joint canonical and deformation optimization

Monocular capture provides one view per timestamp, so the canonical model and the deformation must be trained jointly [2]. A warmup of 3,000 iterations fits the canonical model to all frames as if static, producing a motion-blurred average. The deformation field then activates at identity, and both are optimized together. We tested the alternative of initializing the canonical from a narrow time window, in four variants including progressive expansion of the training range. All collapse under the topology engine, whose pruning removes geometry that renders incorrectly at timestamps the deformer has not yet learned. The mean-pose warmup survives because it is moderately wrong everywhere instead of exactly right somewhere. We read this as a finding about retrofitting deformation onto static splatting engines: the maintenance machinery, not the loss, is the binding constraint.

5.2 Control-node deformation and learned skinning

The deformation field follows embedded deformation [31] and SC-GS [4]. K control nodes (K = 512), chosen by farthest-point sampling over canonical vertices, each carry a time-conditioned rigid transform produced by a shared network from a per-node latent code and a positional

encoding of time, with rotations in the continuous 6D parameterization [43]. Vertices deform by linear blend skinning over their k nearest nodes (k = 8). Skinning weights start from a radial-basis kernel and are optimized as per-vertex logits over the k candidates, which lets stretch localize at articulation boundaries. When the topology engine changes the vertex set, the node assignments and logits are rebuilt while the node transforms persist, and the logits use a separate optimizer so the rebuild does not reset the motion model's momentum. A per-vertex displacement network in the style of Deformable-3DGS is supported and evaluated in the ablations. It underperforms nodes even on repaired gradients, which we attribute to optimization competition with the directly parameterized canonical vertices.

5.3 Mesh regularization

Shared-vertex connectivity permits regularizers that point soups can only approximate. We penalize the as-rigid-as-possible energy [30] of the deformed vertices against the rest pose, with per-vertex rotations fit in closed form on true one-ring neighborhoods and treated as constants in the gradient. One routing detail proved necessary. Evaluated on the rendering forward pass, the energy's gradient reaches the skinning logits, and the optimizer discovers that uniform weights reduce relative displacement, erasing motion to satisfy the prior. All rigidity terms are therefore evaluated on a replica of the deformation whose skinning weights are detached; values are identical and gradients reach only the node transforms and vertices. Without this routing, rigidity priors tax motion itself under either skinning parameterization.

Blend skinning across node boundaries stretches some triangles into long needles. A quadratic penalty on all edge-length change suppresses motion along with the needles, and scheduling it late does not help, because it acts as a restoring force toward rest. Articulation stretches edges by less than a factor of two, while needle artifacts stretch them by one to two orders of magnitude. We therefore penalize only the excess of the deformed-to-rest edge ratio above a threshold of 2.0, quadratically. At weight 0.1 this removes needles at a PSNR cost within noise and improves LPIPS by 16 percent at fixed PSNR, a geometric cleanup that PSNR undersells.

5.4 Coverage supervision

Thin, fast geometry such as arms and tails is systematically lost. It is nearly invisible to a photometric loss against the motion-blurred warmup, and the engine prunes it. Where ground-truth alpha exists, an L1 loss between rendered and true alpha makes absent geometry produce maximal error at every pixel it should occupy. This term is a strong silhouette-seeking force. Alone it regrows limbs while exploding needle artifacts; balanced against the hinge penalty (mask 0.3, hinge 0.1) it yields articulated limbs, clean geometry, and the best accuracy of the study. We also composite training views over a random background color through the same alpha [1]. This removes a failure mode in which triangles that adopt the background color

become invisible to the photometric loss, accumulate blending importance, resist pruning, and occlude novel views.

5.5 Depth-gradient damping

The unit-level validation isolated a bias in the depth component of the repaired gradients. We damp the component of each vertex gradient along its view ray by a factor of 0.9 before it reaches the optimizer, as a hook on the deformed vertex tensor. The effect is scene-dependent in a way that supports the mechanism. Scenes with lateral or articulated motion gain, by 3.2 dB on trex and 1.7 dB on jumpingjacks, while scenes whose true motion runs along view rays lose, by 1.3 dB on bouncingballs. We report it as a characterized per-scene option, with settings disclosed in the configuration table of the released code.

5.6 Topology maintenance under deformation

Three schedule adaptations, each diagnosed from a controlled failure, make the static engine safe under deformation. Opacification and pruning must remain coupled; disabling either alone collapses training, through frozen occluders in one direction and a prune cascade from 1.9 million to 1.7 thousand triangles in the other. The prune-threshold ratchet, which grows without bound as export-time minimalism for static scenes, destroys deforming geometry once densification ends, and is capped at 0.15. Densification remains active during deformation so that error-driven growth can rebuild geometry the warmup lost, with new vertices absorbed by the skinning rebuild described above. With these settings one configuration trains all scenes.

6. Results

6.1 Protocol

We evaluate on the eight synthetic monocular scenes of D-NeRF [22] with the standard splits, training at 800 x 800 for 20,000 iterations per scene, about one hour on one RTX 5080 (16 GB). We report PSNR, SSIM, and LPIPS (AlexNet and VGG) over each scene's full test set at 800 x 800, and additionally at 400 x 400 by area downsampling, because the original benchmark numbers were reported at 400 x 400 while the Gaussian literature evaluates at 800 x 800. Comparisons are annotated with their resolution. One configuration is used for every scene, with depth damping enabled on the five scenes with predominantly lateral motion, disclosed per scene; no other per-scene tuning is performed. Every number derives from released evaluation scripts over released checkpoints, and the manuscript's tables are pinned to those outputs by tests.

6.2 Benchmark results

Table 1 reports per-scene results with the founding baseline; Fig. 3 shows qualitative pairs. At the original resolution we average 29.84 dB against D-NeRF's 30.43, and pass it on hellwarrior (+8.2 dB), standup, mutant, and lego, with a near-tie on hook. At 800 x 800 we average 29.00 dB, 0.953 SSIM, and 0.061 LPIPS. Gaussian systems lead by a wide margin at 800 x 800, with reported averages near 32.8 for TiNeuVox [25], 34 to 35 for 4DGS [3], 40 for Deformable-3DGS [2], and 42 for SC-GS [4]. We do not claim photometric parity. The base static representation itself trails 3DGS [5], and our results inherit that gap. The deficit concentrates in scenes dominated by time-varying appearance, examined in the discussion.

Table 1. D-NeRF test-set results, full test splits. Our metrics at 800 x 800 and 400 x 400; D-NeRF as reported by its authors at 400 x 400. Per-scene configurations and the remaining metrics are in the released evaluation output.

Scene	PSNR 800	SSIM 800	LPIPS-A 800	PSNR 400	D-NeRF PSNR 400
bouncingballs	29.65	0.975	0.056	30.28	38.93
hellwarrior	32.72	0.940	0.102	33.15	25.02
hook	28.22	0.939	0.067	29.07	29.25
jumpingjacks	27.43	0.957	0.062	27.98	32.80
lego	22.56	0.914	0.074	23.00	21.64
mutant	32.23	0.968	0.029	34.01	31.29
standup	32.33	0.966	0.051	33.52	32.79
trex	26.85	0.966	0.048	27.71	31.75
Average	29.00	0.953	0.061	29.84	30.43

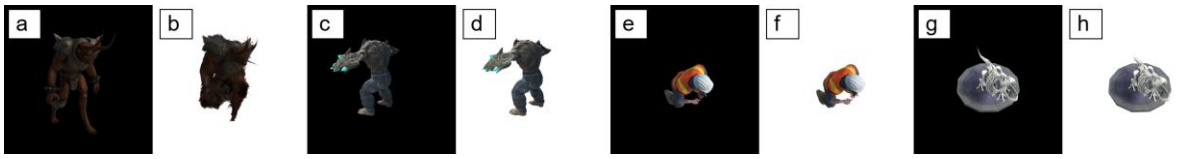

Figure 3. Ground truth and prediction pairs on held-out test views: hellwarrior (a, b), mutant (c, d), standup (e, f), trex (g, h).

6.3 Ablations

The gradient repair is the dominant factor. With the configuration held fixed and only the backward swapped, jumpingjacks moves from 23.98 to 27.18 dB, and articulation appears only under the repair. No combination of method components recovers that deficit under the released backward; sixteen configurations ceiling at 24.22 dB.

A two-by-two over skinning parameterization (fixed radial-basis against learned) and edge-length regularization (off against 0.05), run before the repair, gives 24.03, 23.98, 22.86, and 22.83 dB. The 1.15 dB cost of the edge term under either skinning identifies it as a restoring force, and motivates the hinge design of Sect. 5.3. Mask and hinge need each other (Fig. 4): hinge only reaches 28.58 dB with truncated forearms, mask with a weak hinge reaches 27.79 dB with full limbs and needle artifacts, and the balanced pair reaches 29.53 dB with both.

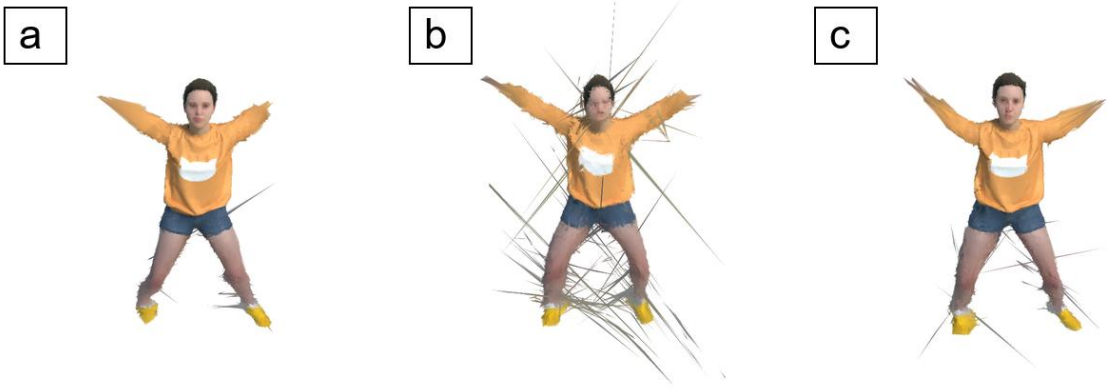

Figure 4. Coverage and stretch terms on jumpingjacks. Hinge penalty alone, clean but with truncated forearms (a); mask loss with a weak hinge, full limbs with needle artifacts (b); the balanced pair (c). The hinge is perceptually real at fixed PSNR, moving LPIPS-Alex from 0.072 to 0.060 and SSIM from 0.958 to 0.962 while PSNR changes by 0.05 dB. On repaired gradients the per-vertex network still learns no motion, at 21.8 dB with a time-invariant field, while identical training with nodes reaches 27.18 dB. With the gradient supply validated, we attribute this to optimization competition between a shared-weight field and directly parameterized canonical vertices, and leave it as an open question for per-point deformation over explicit geometry.

6.4 Efficiency

At 800 x 800 on one RTX 5080, rasterization of the reconstructed scenes runs at 238 to 464 frames per second, and the full pipeline of deformation plus rasterization at 48 to 161 frames per second. Checkpoints occupy 59 to 254 MB at 64 to 766 thousand triangles. Training is about one hour per scene, of which the warmup shared with static Triangle Splatting+ is roughly 25 minutes. For calibration, reported rates for the dynamic Gaussian systems on datacenter GPUs span tens of frames per second [2, 3]; these are consumer-hardware numbers at the standard resolution.

6.5 Simulation on the reconstruction

The deliverable that distinguishes the representation is exercised directly (Fig. 1). Taking a reconstructed scene's vertices as particles and its mesh edges as the rest structure, shape-matching soft-body dynamics [44] with gravity, ground collision, and friction runs on 51 to 228 thousand particles. Every simulated state renders through the same rasterizer, at 24 to 52 frames per second end to end on the same GPU. The subject drops, topples, and comes to rest intact, with no mesh extraction, no proxy representation, and no synchronization, in contrast to pipelines that maintain a separate simulation mesh beside the splats [9]. One negative result is instructive: pure edge-spring dynamics collapse the surface-shell soup, which has no bending or volume resistance, while shape matching is robust to soup connectivity. Fig. 5 shows the geometry buffers the mesh provides directly.

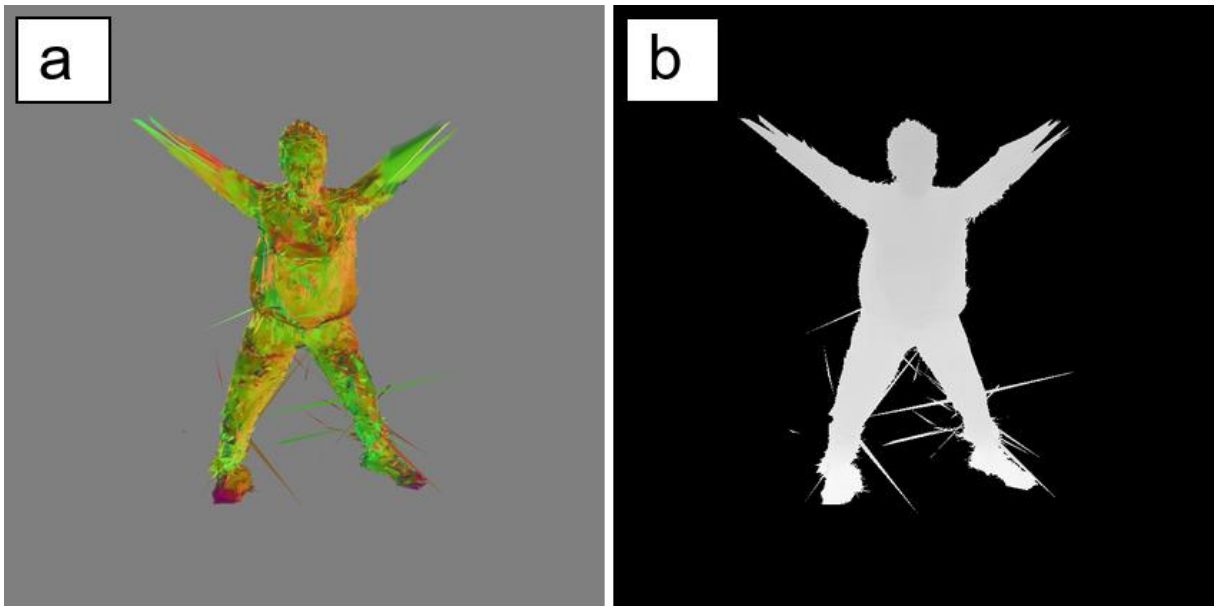

Figure 5. Geometry buffers rendered from the reconstruction at a test timestamp: per-pixel normals (a) and depth (b). The articulated pose is carried by actual surface geometry.

7. Discussion

The audit changes how the benchmark results should be read. Before the repair, every architecture converged to near-static reconstructions, and apparent differences between regularizers were second-order effects on a pipeline whose primary signal path was broken. After the repair, the same components separate cleanly, and their interactions become interpretable: rigidity priors need detached-skinning evaluation, coverage and stretch terms balance, and the topology engine needs three schedule changes to coexist with deformation. We expect the audit method to transfer. Differentiable renderers are validated by the images they produce, while their

gradients, the product every downstream method consumes, are rarely tested against ground truth. Finite differences are cheap, and the single-triangle unit test localizes a defect in minutes.

The largest deficit, bouncingballs at 8.7 dB below the baseline, is diagnostic rather than mysterious. Ball positions and plate geometry track the ground truth; the errors are moving cast shadows sweeping static geometry, which time-invariant per-vertex appearance cannot represent, and highlights beyond degree-3 spherical harmonics. The scene isolates dynamic appearance from dynamic geometry, and our method handles only the latter. A time-conditioned appearance head is the natural extension.

On real capture, the pipeline transfers without synthetic-only supervision. On the HyperNeRF vrig-3dprinter sequence [24], with structure-from-motion initialization and no masks, the method reaches 20.46 dB at quarter resolution, within the range that deformation-field radiance methods report on this capture. The scene is recognizable and the fast-moving extruder is blurred (Fig. 6). Sharpness under real-capture noise is the clear frontier.

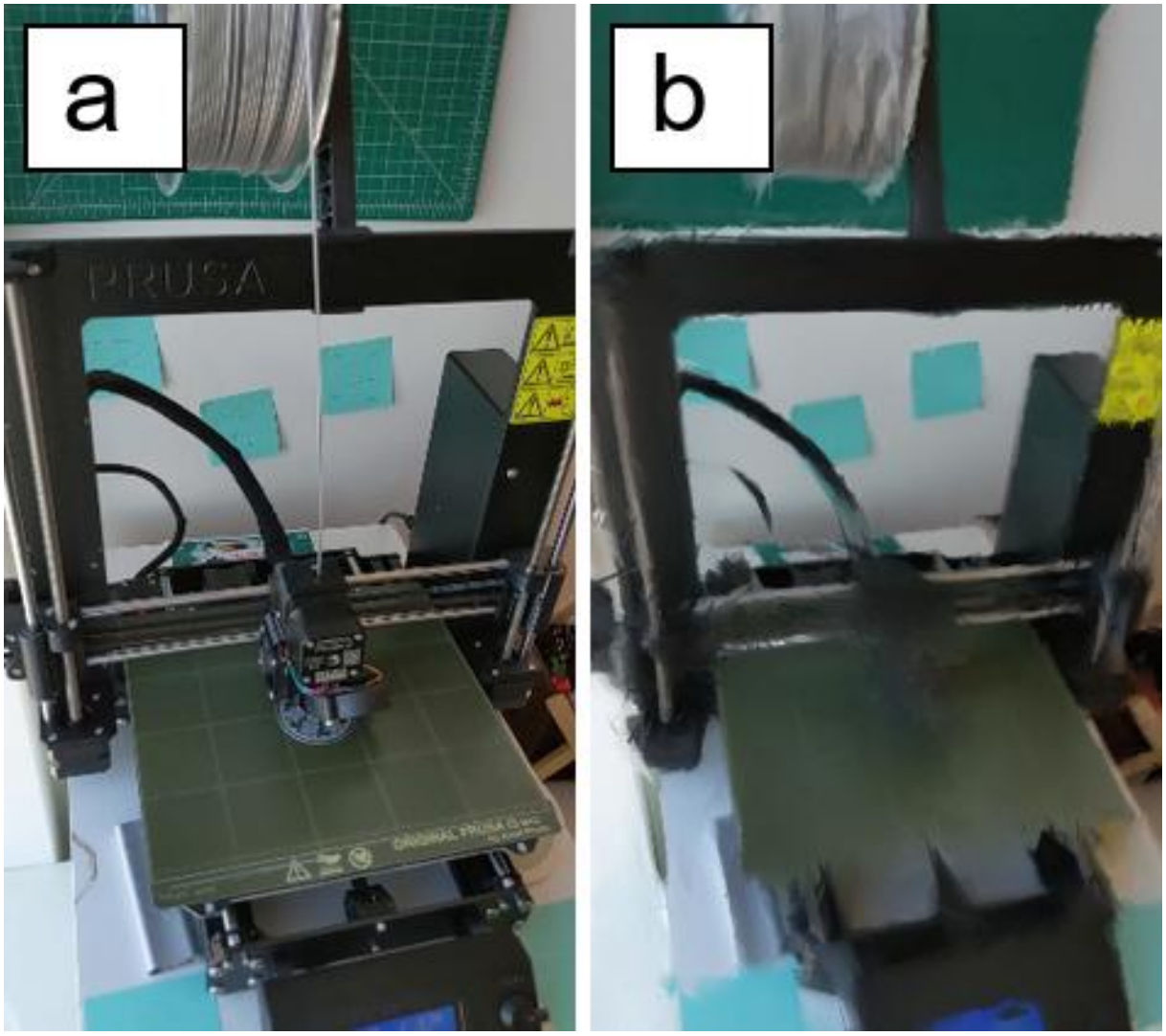

Figure 6. Real capture (HyperNeRF vrig-3dprinter). Ground-truth validation view (a) and the reconstruction at the same time and pose (b).

7.1 Limitations

Photometric fidelity trails the Gaussian systems by 5 to 12 dB on synthetic benchmarks, partly inherited from the static triangle representation and concentrated in time-varying appearance, which we do not model. The per-vertex deformation pathway remains unsolved despite repaired gradients. Depth damping is a disclosed per-scene switch, and an adaptive version is future work. Real-capture results are feasibility-grade. Topology is fixed per scene, so topological change is out of scope. The boundary-discontinuity component of the gradient remains approximate at hard sigma, and our claims about the defects are scoped to the released implementation at the audited commits.

8. Conclusions

Dynamic scenes are the setting where triangle splatting pays off most directly, and the setting that exposed its released position gradients as unusable. A finite-difference audit, two localized repairs, and three levels of validation make the representation trainable under deformation. A control-node deformer with learned skinning, mesh-native losses, and a deformation-adapted topology engine then reconstructs deforming scenes as animated meshes that render at rasterization rates and simulate without conversion. The repaired rasterizer, training code, checkpoints, audit tools, and evaluation scripts are released for reuse.

Data and code availability

The D-NeRF and HyperNeRF datasets are publicly available from their original releases. Training code, rasterizer patches, audit and evaluation scripts, and configurations are released under the MIT license at https://github.com/SamirHossain099/dtsplat and archived at https://doi.org/10.5281/zenodo.23875183. The per-scene best checkpoints are attached to the v0.1.0 release of that repository. The audited upstream commits and the SHA-256 digest of the audited backward source (Section 4.3) are recorded in that repository from release v0.1.1 onward.

Declarations

Competing interests: none. Funding: none. Generative AI: Under the author's direction Claude (Anthropic) was used for language edits for readability improvements and formatting.

References

[1] Kerbl, B., Kopanas, G., Leimkuehler, T., Drettakis, G.: 3D Gaussian Splatting for Real-Time Radiance Field Rendering. ACM Transactions on Graphics. 42(4), 1-14 (2023). https://doi.org/10.1145/3592433

[2] Yang, Z., Gao, X., Zhou, W., Jiao, S., Zhang, Y., Jin, X.: Deformable 3D Gaussians for High-Fidelity Monocular Dynamic Scene Reconstruction. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 20331-20341 (2024). https://doi.org/10.1109/cvpr52733.2024.01922

[3] Wu, G., Yi, T., Fang, J., Xie, L., Zhang, X., Wei, W., et al.: 4D Gaussian Splatting for Real-Time Dynamic Scene Rendering. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 20310-20320 (2024). https://doi.org/10.1109/cvpr52733.2024.01920

[4] Huang, Y., Sun, Y., Yang, Z., Lyu, X., Cao, Y., Qi, X.: SC-GS: Sparse-Controlled Gaussian Splatting for Editable Dynamic Scenes. Proceedings of the IEEE/CVF Conference on Computer

Vision and Pattern Recognition (CVPR). pp. 4220-4230 (2024). https://doi.org/10.1109/cvpr52733.2024.00404
[5] Held, J., Vandeghen, R., Deliege, A., Hamdi, A., Cioppa, A., Giancola, S., et al.: Triangle Splatting for Real-Time Radiance Field Rendering. arXiv preprint arXiv:2505.19175. (2025).
[6] Held, J., Vandeghen, R., Son, S., Rebain, D., Gadelha, M., Zhou, Y., et al.: Triangle Splatting+: Differentiable Rendering with Opaque Triangles. arXiv preprint arXiv:2509.25122. (2025).
[7] Fry, N., Dexheimer, E., Mazur, K., Kelly, P. H. J., Davison, A. J.: Triangle Splatting SLAM. arXiv preprint arXiv:2605.31419. (2026).
[8] Xiong, J., Li, C., Shen, J., Qi, Z., Sun, L., Zhao, D.: FTSplat: Feed-forward Triangle Splatting Network. arXiv preprint arXiv:2603.05932. (2026).
[9] Ramlal, A., Zelek, J. S.: Real-Time Physics Simulation with Dynamic Mesh-Gaussian Reconstructions. arXiv preprint arXiv:2606.00444. (2026).
[10] Mildenhall, B., Srinivasan, P. P., Tancik, M., Barron, J. T., Ramamoorthi, R., Ng, R.: NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis. Proceedings of the European Conference on Computer Vision (ECCV). pp. 405-421 (2020). https://doi.org/10.1007/978-3-030-58452-8_24
[11] Mueller, T., Evans, A., Schied, C., Keller, A.: Instant Neural Graphics Primitives with a Multiresolution Hash Encoding. ACM Transactions on Graphics. 41(4), 1-15 (2022). https://doi.org/10.1145/3528223.3530127
[12] Fridovich-Keil, S., Yu, A., Tancik, M., Chen, Q., Recht, B., Kanazawa, A.: Plenoxels: Radiance Fields without Neural Networks. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 5491-5500 (2022). https://doi.org/10.1109/cvpr52688.2022.00542
[13] Huang, B., Yu, Z., Chen, A., Geiger, A., Gao, S.: 2D Gaussian Splatting for Geometrically Accurate Radiance Fields. ACM SIGGRAPH 2024 Conference Papers. pp. 1-11 (2024). https://doi.org/10.1145/3641519.3657428
[14] Yu, Z., Chen, A., Huang, B., Sattler, T., Geiger, A.: Mip-Splatting: Alias-free 3D Gaussian Splatting. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 19447-19456 (2024). https://doi.org/10.1109/cvpr52733.2024.01839
[15] Lu, T., Yu, M., Xu, L., Xiangli, Y., Wang, L., Lin, D., et al.: Scaffold-GS: Structured 3D Gaussians for View-Adaptive Rendering. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 20654-20664 (2024). https://doi.org/10.1109/cvpr52733.2024.01952
[16] Held, J., Vandeghen, R., Hamdi, A., Deliege, A., Cioppa, A., Giancola, S., et al.: 3D Convex Splatting: Radiance Field Rendering with 3D Smooth Convexes. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 21360-21369 (2025). https://doi.org/10.1109/cvpr52734.2025.01990
[17] Liu, R., Sun, D., Chen, M., Wang, Y., Feng, A.: Deformable Beta Splatting. ACM SIGGRAPH 2025 Conference Papers. pp. 1-11 (2025). https://doi.org/10.1145/3721238.3730716

[18] Wang, W., Li, Z., Shi, J., Zhang, Z., Ye, B., Pollefeys, M., et al.: TriSplat: Simulation-Ready Feed-Forward 3D Scene Reconstruction. arXiv preprint arXiv:2605.26115. (2026).
[19] Tojo, K., Bickel, B., Umetani, N.: DiffSoup: Direct Differentiable Rasterization of Triangle Soup for Extreme Radiance Field Simplification. arXiv preprint arXiv:2603.27151. (2026).
[20] Son, S., Gadelha, M., Zhou, Y., Xu, Z., Lin, M. C., Zhou, Y.: DMesh: A Differentiable Mesh Representation. Advances in Neural Information Processing Systems (NeurIPS). pp. 12035-12077 (2024). https://doi.org/10.52202/079017-0384
[21] Yuan, S., Wang, D., Ramamoorthi, R., Cui, X.: ArtMesh: Part-Aware Articulated Mesh Fields with Motion-Consistent Dynamics. arXiv preprint arXiv:2605.16582. (2026).
[22] Pumarola, A., Corona, E., Pons-Moll, G., Moreno-Noguer, F.: D-NeRF: Neural Radiance Fields for Dynamic Scenes. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 10313-10322 (2021). https://doi.org/10.1109/cvpr46437.2021.01018
[23] Park, K., Sinha, U., Barron, J. T., Bouaziz, S., Goldman, D. B., Seitz, S. M., et al.: Nerfies: Deformable Neural Radiance Fields. Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV). pp. 5845-5854 (2021). https://doi.org/10.1109/iccv48922.2021.00581
[24] Park, K., Sinha, U., Hedman, P., Barron, J. T., Bouaziz, S., Goldman, D. B., et al.: HyperNeRF: A Higher-Dimensional Representation for Topologically Varying Neural Radiance Fields. ACM Transactions on Graphics. 40(6), 1-12 (2021). https://doi.org/10.1145/3478513.3480487
[25] Fang, J., Yi, T., Wang, X., Xie, L., Zhang, X., Liu, W., et al.: Fast Dynamic Radiance Fields with Time-Aware Neural Voxels. SIGGRAPH Asia 2022 Conference Papers. pp. 1-9 (2022). https://doi.org/10.1145/3550469.3555383
[26] Fridovich-Keil, S., Meanti, G., Warburg, F., Recht, B., Kanazawa, A.: K-Planes: Explicit Radiance Fields in Space, Time, and Appearance. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 12479-12488 (2023). https://doi.org/10.1109/cvpr52729.2023.01201
[27] Cao, A., Johnson, J.: HexPlane: A Fast Representation for Dynamic Scenes. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 130-141 (2023). https://doi.org/10.1109/cvpr52729.2023.00021
[28] Yang, Z., Yang, H., Pan, Z., Zhang, L.: Real-time Photorealistic Dynamic Scene Representation and Rendering with 4D Gaussian Splatting. International Conference on Learning Representations (ICLR). (2024). https://doi.org/10.48550/arXiv.2310.10642
[29] Li, Z., Chen, Z., Li, Z., Xu, Y.: Spacetime Gaussian Feature Splatting for Real-Time Dynamic View Synthesis. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 8508-8520 (2024). https://doi.org/10.1109/cvpr52733.2024.00813
[30] Sorkine, O., Alexa, M.: As-Rigid-As-Possible Surface Modeling. Proceedings of the Eurographics Symposium on Geometry Processing. pp. 109-116 (2007). https://doi.org/10.2312/SGP/SGP07/109-116

[31] Sumner, R. W., Schmid, J., Pauly, M.: Embedded Deformation for Shape Manipulation. ACM Transactions on Graphics. 26(3), 80 (2007). https://doi.org/10.1145/1276377.1276478
[32] Xie, T., Zong, Z., Qiu, Y., Li, X., Feng, Y., Yang, Y., et al.: PhysGaussian: Physics-Integrated 3D Gaussians for Generative Dynamics. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 4389-4398 (2024). https://doi.org/10.1109/cvpr52733.2024.00420
[33] Jiang, Y., Yu, C., Xie, T., Li, X., Feng, Y., Wang, H., et al.: VR-GS: A Physical Dynamics-Aware Interactive Gaussian Splatting System in Virtual Reality. ACM SIGGRAPH 2024 Conference Papers. (2024). https://doi.org/10.1145/3641519.3657448
[34] Li, T., Aittala, M., Durand, F., Lehtinen, J.: Differentiable Monte Carlo Ray Tracing through Edge Sampling. ACM Transactions on Graphics. 37(6), 1-11 (2018). https://doi.org/10.1145/3272127.3275109
[35] Loubet, G., Holzschuch, N., Jakob, W.: Reparameterizing Discontinuous Integrands for Differentiable Rendering. ACM Transactions on Graphics. 38(6), 1-14 (2019). https://doi.org/10.1145/3355089.3356510
[36] Laine, S., Hellsten, J., Karras, T., Seol, Y., Lehtinen, J., Aila, T.: Modular Primitives for High-Performance Differentiable Rendering. ACM Transactions on Graphics. 39(6), 1-14 (2020). https://doi.org/10.1145/3414685.3417861
[37] Liu, S., Li, T., Chen, W., Li, H.: Soft Rasterizer: A Differentiable Renderer for Image-based 3D Reasoning. Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV). pp. 7707-7716 (2019). https://doi.org/10.1109/iccv.2019.00780
[38] Pidhorskyi, S., Simon, T., Schwartz, G., Wen, H., Sheikh, Y., Saragih, J.: Rasterized Edge Gradients: Handling Discontinuities Differentiably. Proceedings of the European Conference on Computer Vision (ECCV). (2024). https://doi.org/10.48550/arXiv.2405.02508
[39] Radl, L., Steiner, M., Parger, M., Weinrauch, A., Kerbl, B., Steinberger, M.: StopThePop: Sorted Gaussian Splatting for View-Consistent Real-time Rendering. ACM Transactions on Graphics. 43(4), 1-17 (2024). https://doi.org/10.1145/3658187
[40] Huang, L., Bai, J., Guo, J., Li, Y., Guo, Y.: On the Error Analysis of 3D Gaussian Splatting and an Optimal Projection Strategy. Proceedings of the European Conference on Computer Vision (ECCV). pp. 247-263 (2024). https://doi.org/10.1007/978-3-031-72643-9_15
[41] Ye, Z., Li, W., Liu, S., Qiao, P., Dou, Y.: AbsGS: Recovering Fine Details in 3D Gaussian Splatting. Proceedings of the 32nd ACM International Conference on Multimedia. pp. 1053-1061 (2024). https://doi.org/10.1145/3664647.3681361
[42] Ye, V., Li, R., Kerr, J., Turkulainen, M., Yi, B., Pan, Z., et al.: gsplat: An Open-Source Library for Gaussian Splatting. Journal of Machine Learning Research. (2025). https://doi.org/10.48550/arXiv.2409.06765
[43] Zhou, Y., Barnes, C., Lu, J., Yang, J., Li, H.: On the Continuity of Rotation Representations in Neural Networks. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 5738-5746 (2019). https://doi.org/10.1109/cvpr.2019.00589

[44] Mueller, M., Heidelberger, B., Teschner, M., Gross, M.: Meshless Deformations Based on Shape Matching. ACM Transactions on Graphics. 24(3), 471-478 (2005). https://doi.org/10.1145/1073204.1073216

Author biography



Samir Hossain is a graduate researcher in the Department of Electrical and Computer Engineering at Texas Tech University, Lubbock, TX, USA. His research covers 3D reconstruction and differentiable rendering, together with evaluation and measurement validity for learned systems across biomedical and sensing applications. His recent work includes dynamic scene reconstruction with triangle primitives and gradient-correctness auditing of differentiable rasterizers.