Tessellation GS: Neural Mesh Gaussians for Robust Monocular Reconstruction of Dynamic Objects

Shuohan Tao[1], Boyao Zhou[2], Hanzhang Tu[2], Yuwang Wang[2], and Yebin Liu[2*]
[1]University of Cambridge, [2]Tsinghua University

Abstract

3D Gaussian Splatting (GS) enables highly photorealistic scene reconstruction from posed image sequences but struggles with viewpoint extrapolation due to its anisotropic nature, leading to overfitting and poor generalization, particularly in sparse-view and dynamic scene reconstruction. We propose Tessellation GS, a structured 2D GS approach anchored on mesh faces, to reconstruct dynamic scenes from a single continuously moving or static camera. Our method constrains 2D Gaussians to localized regions and infers their attributes via hierarchical neural features on mesh faces. Gaussian subdivision is guided by an adaptive face subdivision strategy driven by a detail-aware loss function. Additionally, we leverage priors from a reconstruction foundation model to initialize Gaussian deformations, enabling robust reconstruction of general dynamic objects from a single static camera, previously extremely challenging for optimization-based methods. Our method outperforms previous SOTA method, reducing LPIPS by 29.1% and Chamfer distance by 49.2% on appearance and mesh reconstruction tasks.

1. Introduction

Reconstruction of observed scenes has always been a major challenge in computer vision. With the spark of differentiable rendering, the task has shifted from solely relying on special equipments like depth cameras and LiDARs to leveraging more accessible multi-camera or monocular video setup. However, most methods [18, 29, 30] focus on static scenes due to the inherent under-determinacy of dynamic scene reconstruction and non-rigid deformation.

Early approaches [4, 31] employed image warping to preserve observed information for novel view synthesis or physical plausibility regularization in unobserved regions for geometry reconstruction. Recent monocular differentiable rendering methods incorporate geometric constraints alongside photometric loss with implicit NeRF [11, 15, 28, 47] or explicit Gaussian Splatting [24, 25, 44] representation. Such ”4D” methods leverages spectral-biasedness of MLP [5, 44, 47] or spatial-temporal factorization [3, 6, 33, 44] but relies on explicit or implicit (camera movement) multi-view input. As pointed out by [7], existing differentiable rendering approaches are highly susceptible to view overfitting, and using unrealistic camera motions in datasets D-NeRF [36] and DG-Mesh [25]. On the other hand, avatar-like methods [10, 13, 22, 48] enable per-pose Gaussian attributes modeling for human performance reconstruction, which heavily rely on category-specific template, *e.g.* SMPL [26]. Thus, such methods are not able to handle topological changes and category-agnostic reconstruction.

In terms of template-free method, DG-Mesh [25], TiNeuVox [6], and HexPlane [3] pose various structural and loss designs. DG-Mesh utilized Laplacian regularization to ensure smoothness of the reconstructed geometry. TiNeuVox and HexPlane factorizes spatiotemporal information into several feature planes that are decoded by MLP. Although effective for input video with enough camera movement, they do not explicitly regularize view-overfitting, and as a result, they can not recover dynamic details from general monocular video input and degrade synthesis quality for novel views far from input view, due to the lack global cue of moving object.

To bridge this gap, we propose to use a template-free geometry prior to anchor our surfel-like Gaussian attributes. To obtain such prior, we apply large reconstruction model (LRM) to extract coarse geometry frame by frame. Such geometry is in a relatively low resolution without high-frequency details and correspondence between them is unknown. Therefore, we propose to learn a deformation field with control points whose movements are defined by a motion MLP to build the correspondence between reference frame and other frames and to preserve temporal consistency by doing a sequential optimization. In this process, we design a robust Chamfer loss to overcome the flickering geometry and floating artifacts from LRM output. We

[*] Corresponding author (liuyebin@mail.tsinghua.edu.cn).

only optimize deformation field in this stage, while further anchor 2D Gaussian attributes for joint optimization of deformation and appearance in stage 2. Since the deformation field is established in stage 1, we define all Gaussian points on the geometry surface of reference frame and learn Gaussian attributes with appearance decoders and Gaussian features defined on mesh vertices. To further enhance the high-frequency appearance modeling, we propose a novel hierarchical structure of Gaussian points, which follows a structural densification mechanism instead of the gradient-based splitting in original Gaussian Splatting. Attaching surfel-like Gaussians on geometry triangles, our appearance decoders are able to optimize both geometry and appearance with solely input images. Finally, we achieve accurate motion reconstruction, temporally consistent geometry, and photo-realistic appearance modeling within 90 minutes for 500 frames.

In conclusion, our main contributions are as follows:

- Our method is capable of reconstructing dynamic object from a single monocular video under challenging camera setup, for category-agnostic objects.
- We leverage LRM to prepare per-frame coarse geometry prior and further propose a deformation MLP to build the correspondence of frames and to faithfully match the dynamic information of the input video.
- We proposed a novel mesh Gaussian structure that provides higher fidelity appearance, lower memory burden, and higher training speed. We also proposed two constraints for mesh Gaussians to avoid view-overfitting.

2. Related Work

Differentiable Rendering NeRF [23] was proposed to represent a static scene with density and color volumes defined by an MLP. The training process is time-consuming as it performs costly numerical integration along camera rays at every training iteration. Recently, a more efficient method, 3D Gaussian Splatting [18] was proposed. It represents a scene as many anisotropic 3D Gaussian volumes with tractable integrations. It achieved great speedup compared to NeRF. However, due to the anisotropic nature of 3D Gaussians, they are prone to overfitting to camera views, often elongating along the camera ray. This results in artifacts when rendered from novel viewpoints, particularly in sparse-view regions where multi-view supervision is weak. Scaffold GS [27] partially solved the overfitting issue by encoding local geometric structures in compact neural features. 2D GS [14] proposed to set one of the axis of 3D GS to have almost zero scale in order to model geometrical details more accurately. Building on top of that, mesh based GS [8, 9] anchor Gaussian Splats to mesh faces, with the Gaussian normal direction aligned with the mesh faces either through a loss term or a hard constraint. An underlying geometric representation makes them less susceptible to overfitting. Although they thrive in multi-view reconstruction tasks, they still suffer from view-overfitting in monocular reconstruction tasks, even worse on dynamic scenes.

Dynamic Reconstruction Recent works have used both NeRF and 3D GS for reconstructing dynamic scenes. NeRF based methods [3, 6, 21, 33, 34, 36] usually uses a time-conditioned or per-frame embedding conditioned NeRF to model time-varying appearance of scenes. 3D GS based methods [28, 45, 47] also uses time-conditioned deformation models to offset a canonical set of 3D GS points to respective timesteps to match the captured images. These works have all achieved incredible results on existing monocular video dataset of dynamic objects.

Nonetheless, most of these methods are not robust against novel-view rendering, and most existing datasets either provide effectively multi-view input or use testing views that's not too far from the training view, as pointed out by [7]. There are two levels of difficulties here:

- **Lack of geometric grounding.** Without an explicit geometric representation, photometric appearances remain unconstrained, reducing the reconstruction to mere colored points in space.
- **Weak coupling between geometry and appearance.** Even when a geometric representation is present, existing appearance models are not strictly constrained by it, leading to potential view-overfitting.

As a result, most successful and deployable monocular reconstruction methods either rely on class-specific templates (e.g., SMPL) [16, 20, 32, 35, 37–39, 43] or require additional depth input to compensate for missing multi-view information.

3. Method

Reconstructing both geometry and appearance of objects from slowly moving cameras is challenging due to the ambiguity in motion of unobserved region, view overfitting tendencies of differentiable rendering methods, and lack of geometric information from camera movement. Our model is able to solve the challenge by performing a 2-stage optimization process. In the first stage shown in Fig. 1 (a) and (b), we designed a robust framework to extract motion and geometry information from unstable LRM output defined by a canonical mesh and a control-point based deformation model. In the second stage in Fig. 1 (c), structured 2D GS will be initialized on the canonical mesh. We express 2D GS as functions of vertex neural features to keep an expressive and compact representation. Robustness to view-overfitting is achieved by avoiding GS occlusion and constraining GS influence to local mesh faces. We also include a carefully designed adaptive GS subdivison mechanism to automatically add new GS to regions with fine photometric or geometric details through a mesh-Gaussian quad tree.

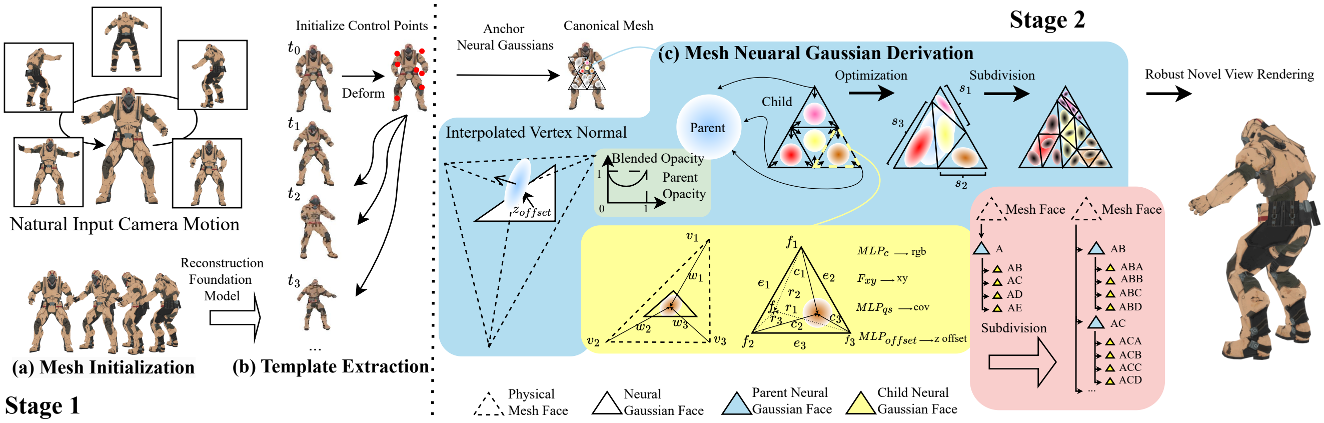

Figure 1. **Illustration of pipeline.** In stage one, we get per-frame mesh sequence from LRM by querying each frame. We fix the mesh by Taubin smoothing [40] and subdivide faces or collapse edges until the number of faces reaches our desired initial number of Gaussians. In stage two, we initialize 2D Gaussians defined by neural features on the canonical mesh. We train the neural Gaussians jointly with the deformation model. The resulting Gaussians are extremely robust to view-overfitting.

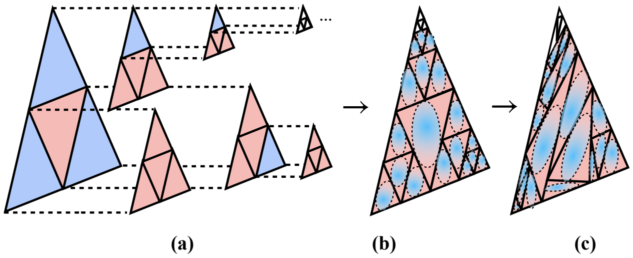

Figure 2. **Adaptive densification via mesh-Gaussian quad tree on a single mesh face.** Red triangles are leaf nodes of whose associated Gaussians will not further subdivide. Blue triangles are non-leaf nodes with no associated Gaussians. (a) and (b): the tree allows for adaptive density of Gaussians. (c): learnable subdivision ratio further improves the expressiveness.

3.1. Stage One: Data Driven Template

In the first stage, we extract 3D template and motion information from LRM [12, 42] to form a 4D template. We have LRM output corresponding meshes to each frame of the input video and choose one (in our experiments simply the first mesh) to be canonical mesh. We then perform ICP (Iterative Closest Point) between canonical mesh and each of the meshes in the sequence. Then similar to BANMo [46], our deformation model initializes control points and optimize their positions and a time-conditioned MLP to drive their motions. During this process, we jointly optimize the skinning weight of mesh vertices to drive the mesh.

$$\begin{aligned} w_{n,k} &= \mathrm{MLP}_{weight,k}(\mathbf{C}_k - \mathbf{v}_{n,t_0}) + \phi(||\mathbf{C}_k - \mathbf{v}_{n,t_0}||) \\ \mathbf{c}_{k,t} &= \mathrm{MLP}_{motion,k}(t),\ \mathbf{v}_{n,t} = \mathbf{v}_{n,t_0} + \sum_{k=1}^{K} \mathbf{c}_{k,t} w_{n,k} \end{aligned} \tag{1}$$

In the above equation where we derived the final per vertex location $\mathbf{v}_{n,t}$ at time t, MLP$_{weight,k}$ is a per control point MLP that output weights taking as input displacement from control point to canonical vertices, ϕ is an isotropic Gaussian kernel for radial basis function whose scales are determined by the average nearest neighbor distance of the control points at initialization, $w_{n,k}$ is the composite skinning weight between the n$^{\text{th}}$ vertex $\mathbf{v}_{n,t_0}$ and the k$^{\text{th}}$ control point $\mathbf{C}_k$. MLP$_{motion,k}$ is a per control point MLP that takes as input time and output control point displacement $\mathbf{c}_{k,t}$, and $\mathbf{v}_{n,t_0}$ is per vertex location at canonical time.

$$\begin{aligned} \mathcal{L}_{RCD} &= \frac{1}{|V|}\sum_{v\in V} \min\left(d^2, \min_{u\in U} \|v-u\|^2\right) \\ &+ \frac{1}{|U|}\sum_{u\in U} \min\left(d^2, \min_{v\in V} \|u-v\|^2\right) \\ \mathcal{L}_{total} &= w_{lap}\mathcal{L}_{lap} + w_n\mathcal{L}_n + \mathcal{L}_{RCD} \end{aligned} \tag{2}$$

In the above equation, we proposed robust Chamfer loss $\mathcal{L}_{RCD}$, where U and V are sets of vertices in target and source meshes, d is truncation distance, $\mathcal{L}_{lap}$ is mesh Laplacian regularization, $\mathcal{L}_n$ is normal consistency regularization, and $\mathcal{L}_{RCD}$ is our proposed robust Chamfer loss; w_{lap} and w_n are weights for respective loss terms. We optimize the deformation model against $\mathcal{L}_{total}$. Please refer to the supplementary material for more detail about stage one.

3.2. Stage Two: Tessellation Gaussian Splatting

In the second stage, we solely use the ground-truth frames to jointly optimize motion information and appearance defined by GS. Traditional GS doesn't perform well when trained on sparse views as they could overfit to training cameras. We have identified 2 major reasons. First, Gaussians could elongate along the camera rays freely without hurting photometric performance. In addition, when large scale deformation happens on non-visible regions from training views, unsupervised occluded Gaussians could appear due to the separation of the occluding Gaussians, resulting in artifacts. We show the examples of these two types of artifacts in our ablation study shown in Fig. 6. We therefore propose a novel 2D GS architecture, mesh-Gaussian quad tree, that both avoids Gaussian occlusion and scale overfitting by constraining their locations and scales to local structured triangles on mesh faces to minimize Gaussian overlapping.

3.2.1 Structured 2D Gaussians

We build mesh-Gaussian quad tree as demonstrated in Fig. 1 (c) and Fig. 2. At initialization, each mesh face is also a parent Gaussian face. Each parent face is divided into 4 smaller child faces connecting its 3 edge centers similar to Mesh-GS [8], forming a quad tree shown in Fig. 2 (a) and (b). However, instead of fixing the subdivision points like MeshGS, we introduce learnable edge point ratios s_1, s_2, and s_3 to allow these points to slide along the edges to match local textures, as in Fig. 2 (c). A large parent Gaussian is spawned at the center of the parent face, while 4 child Gaussians are spawned at the center of child faces. To maintain clarity, we refer to both parent faces and child faces as Gaussian faces, distinguishing them from the underlying physical mesh faces, whose structure remains unchanged throughout the training. This hierarchical structure enables adaptive refinement of Gaussians, resulting in a compact and expressive representation.
Each parent face has 6 learnable features $f_i \in \mathbb{R}^{128}$: 3 parent features are defined on the vertices, and 3 edge features on the edges. Each Gaussian has a learnable barycentric weight $\mathbf{r} \in \mathbb{R}^3$ to interpolate from the 3 vertices to decide their barycentric coordinate. They have a separate set of barycentric weights $\mathbf{c} \in \mathbb{R}^3$ to interpolate from the 3 vertex features to get their neural features. For both parent and child Gaussians, features are interpolated as follows:

$$f_{interpolated} = \text{softmax}(\mathbf{r}) \cdot [f_1||f_2||f_3] \tag{3}$$

where f_1, f_2, and f_3 are vertex features of the faces each Gaussian belong to, and $||$ is the concatenation operator. For parent faces, vertex features are directly stored. For child faces, for example the yellow triangular face in Fig. 1, its vertex features are calculated as:

$$\begin{aligned} f_1 &= f_{parent_1} \cdot (1 - s_1) + f_{parent_3} \cdot s_1 + f_{edge_2} \\ f_2 &= f_{parent_2} \cdot s_2 + f_{parent_3} \cdot (1 - s_2) + f_{edge_3} \end{aligned} \tag{4}$$

and f_3 equals f_{parent_3}. We interpolate parent vertex features f_{parent} along edges by our learnable shape proportion s and add the edge features f_{edge_i} on top of that. We allow child faces to share vertex features with parent face where they share a common vertex. We decode Gaussian features with 3 MLP appearance decoders:

$$\begin{aligned} \text{MLP}_{qs}([f_{interpolated}||\frac{e2}{e1}||\frac{e3}{e1}]) &= [\mathbf{q_{2D}}||\mathbf{s_{2D}}] \\ \text{MLP}_c([f||\mathbf{p}]) &= rgb \\ \text{MLP}_{offset}(f_{interpolated}) &= z_{offset} \end{aligned} \tag{5}$$

where $\mathbf{q_{2D}}||\mathbf{s_{2D}}$ is the 2D GS's rotation and scale concatenated. z_{offset} is the GS offset along the face normal direction, e_1, e_2, and e_3 are the edge lengths of each Gaussian face. To model pose dependent apperance, we encode the location of the 30 control points into a pose embedding $\mathbf{p}$, similar to Gaussian Avatar [13] and Animatable Gaussians [22]. 2D GS's normal direction is interpolated from vertex normals, and we similarly interpolate vertex colors to add to the decoded neural colors. Further GS constraints are described in Sec. 3.2.3.

3.2.2 Competitive Gaussian Opacities

We utilized Gaussian opacities for gradient-free Gaussian subdivision. Specifically, only parent Gaussians have optimizable opacities as in Fig. 3, while the four child Gaussians compete their opacities with their parents, as computed in Eq. (6), where we have heuristically set β to be 0.9. Importantly, β ensures the sum of a parent Gaussian's opacity and any of its child Gaussians' opacities never equals 1 except when the parent Gaussian's opacity is either 0 or 1, as plotted in Fig. 1. In such cases, the child Gaussians' opacities become either 0 or 1, effectively forcing binary opacity assignments.

$$\alpha_{child} = (1 - \alpha_{parent}^{\beta})^{\frac{1}{\beta}} \tag{6}$$

With this setup, each parent Gaussian's opacity serves as an indicator of local detail level. The regularization term $\mathcal{L}_\alpha$ in Eq. (11), which reaches its minimum only when all parent Gaussians are fully opaque, competes with the photometric and geometric loss terms. When the other loss terms exceed a certain threshold, indicating the presence of finer local details, parent Gaussians' opacities tend to zero, allowing child Gaussians to emerge and model these finer details. This mechanism adaptively allocates Gaussian population, ensuring that regions with higher complexity receive higher-resolution representations, while simpler areas remain efficiently represented by fewer and larger parent Gaussians, thereby maintaining an adaptive and efficient Gaussian hierarchy.

3.2.3 Gaussian Constraints

We propose two constraints on the Gaussians to avoid view-overfitting. First, Gaussian scales are constrained to a maximum of one-fourth of the base and height of their respective

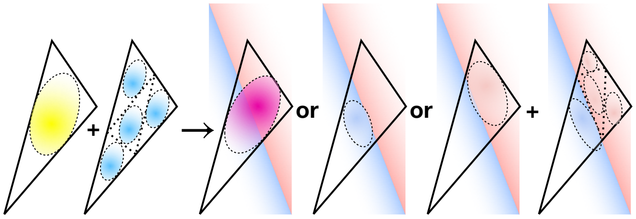

Figure 3. **Learnable subdivision ratio fits boundary better.** Yellow Gaussian is a parent Gaussian, blue Gaussians are child Gaussians. Color boundary denoted by red and blue regions can be better modeled by child Gaussians than parent. Child Gaussians' opacities will naturally become one through optimization.

triangles by applying a sigmoid activation to the decoded $\mathbf{s_{2D}}$ in Eq. (5). Gaussians are also allowed to rotate around normals by the decoded rotation offset $\mathbf{q_{2D}}$. This way, the Gaussians are initialized with just enough scale to span the surface which naturally minimizes overlapping. As the subdivision mechanism progresses, new Gaussians are dynamically introduced to fill regions where the initial Gaussians fail to provide sufficient coverage.

In addition, for Gaussian offset along the face normal direction, rather than employing a soft Gaussian anchoring constraint as in DG-Mesh [25], we introduce a scale-dependent anchoring strategy. Gaussians associated with smaller neural faces are permitted to have larger offsets while initial Gaussians with parent being mesh faces have zero offset, judged by the value:

$$u = \tanh(\frac{e_p}{e_g}) \tag{7}$$

where e_p and e_g are the mean edge length of the root mesh face and the mean edge length of the Gaussian face. However, no offset is allowed to exceed the mean edge length of the root physical mesh face, ensured by

$$\begin{aligned} w_{bar} &= (1-w_1)(1-w_2)(1-w_3) \\ offset &= w_{bar} \cdot u \cdot e_p \cdot \tanh(z_{offset}) \end{aligned} \tag{8}$$

where w_1, w_2, and w_3 are the barycentric coordinates of Gaussian points with respect to their root mesh faces.

This ensures that finer geometric details are only carved after the object's motion has already been well optimized, which prevents premature deformations that could lead to inconsistent geometry. This constraint also ensures that strong gradient will flow to mesh vertices to optimize geometric details through photometric losses. The effectiveness of the contraints are shown in Sec. 4.3.

3.2.4 Adaptive Gaussian Population Control

Thanks to our proposed adaptive mesh subdivision process, our neural GS can model very fine level of details with our mesh-Gaussian quad tree, as illustrated by Fig. 1. Our model subdivides a parent Gaussian by turning its 4 child Gaussians into 4 new parent Gaussians, making each child face a new parent face. During this process, we remove the original parent Gaussian, compute its three edge features, and reassign them as new vertex features. The newly introduced edge features are initialized to zero, minimizing disruption to the optimization process and allowing us to constantly introduce new Gaussians in a stable manner during training. Edge features of each new parent faces are initialized to zero, minimizing the impact on optimization process and incrementally introducing new Gaussians in a stable manner during training. Our subdivision mechanism is implemented as follows:

- **Parent Gaussian Subdivision:** All parent Gaussians with opacities below 0.1 are subdivided every 5000 iterations except for the initial and last 5000 iterations, ensuring denser Gaussians in finer regions.
- **Child Gaussian Deactivation:** Excessive child Gaussians whose parent opacities remain above 0.9 in 90% of the iterations since the last subdivision are turned off, and we also exclude them when calculating opacity regularization $\mathcal{L}_\alpha$.

3.2.5 Loss Terms

Our losses are as follows:

$$\mathcal{L}_{pho} = \mathcal{L}_1 + \mathcal{L}_{ssim} \tag{9}$$

where we used $\mathcal{L}_1$ and SSIM loss between GT image and rendered image to supervise photometric appearance,

$$\mathcal{L}_{edge} = \frac{1}{|\mathcal{E}|} \sum_{(i,j)\in\mathcal{E}} (\|\mathbf{v}_{i,t_0} - \mathbf{v}_{j,t_0}\| - \|\mathbf{v}_{i,t} - \mathbf{v}_{j,t}\|)^2 \tag{10}$$

where we penalize edge length changes with respect to canonical mesh, $\mathbf{v}_{i,t_0}$ is the canonical vertex coordinate and $\mathbf{v}_{i,t}$ is its coordinate at timestep t. $\mathcal{E}$ is the set of neighboring vertices.

$$\mathcal{L}_\alpha = \frac{1}{N} \sum_{i}^{N} \alpha_i \tag{11}$$

The above $\mathcal{L}_\alpha$ is the mean opacity of all Gaussians. Due to our parent-child opacity coupling, its minimum is when all parents are fully opaque. Our final loss $\mathcal{L}_{TGS}$ is:

$$\begin{aligned} \mathcal{L}_{reg} &= \mathcal{L}_{lap} + w_{edge}\mathcal{L}_{edge} + w_\alpha \mathcal{L}_\alpha \\ \mathcal{L}_{prior} &= w_{flow}\mathcal{L}_{flow} + w_{normal}\mathcal{L}_{normal} \\ \mathcal{L}_{TGS} &= \mathcal{L}_{pho} + w_{reg}\mathcal{L}_{reg} + w_{prior}\mathcal{L}_{prior} \end{aligned} \tag{12}$$

where $\mathcal{L}_{flow}$ is the difference between the rendered optical flow and the optical flow predicted by RAFT [41]. $\mathcal{L}_{lap}$ is mesh Laplacian regularization further explained in supplementary material. We use normal supervision $\mathcal{L}_{normal}$ provided by DSINE [2] only for static cameras.

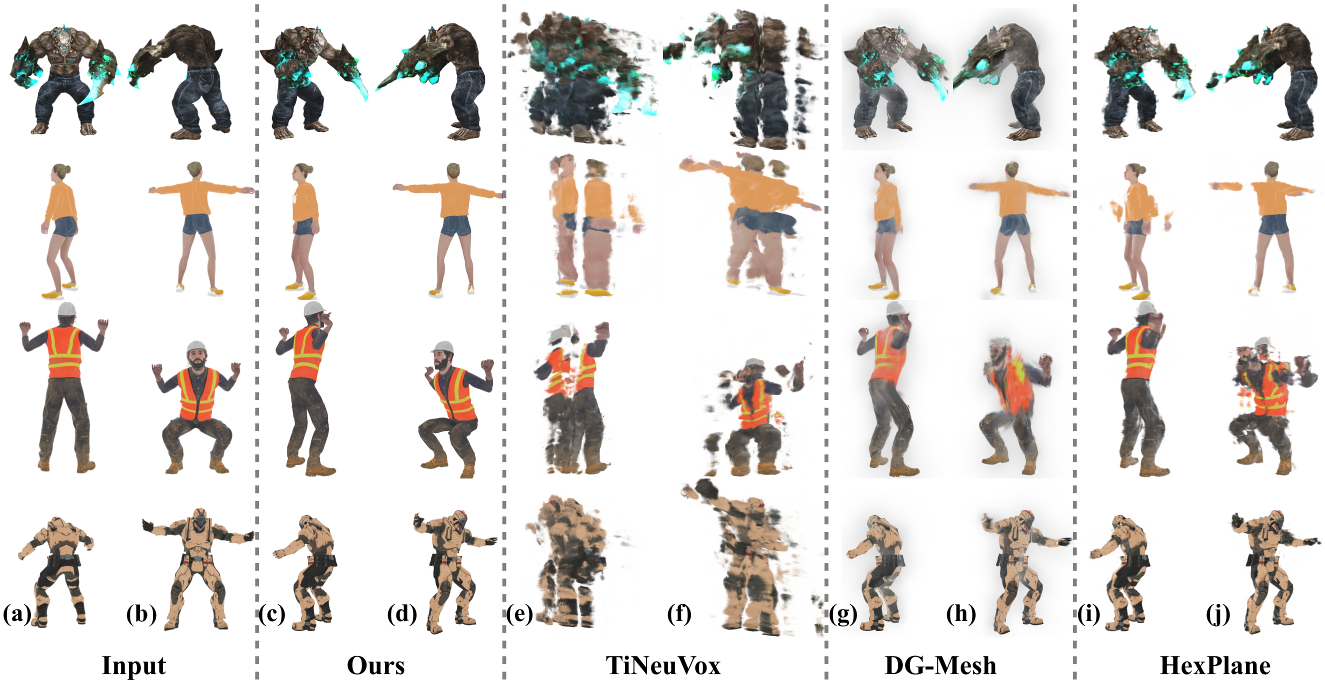

Figure 4. **Test results on Smooth D-NeRF.** (a) and (b): input images at two different timesteps. (c), (e), (g), and (i): rendering results at the first timestep. (d), (f), (h), and (j): rendering results at the second timestep. Our results are visually better than all other methods. The second best results are produced by DG-Mesh [25]. Their 3D Gaussians' unconstrained scales result in foggy appearance caused by large Gaussian floaters.

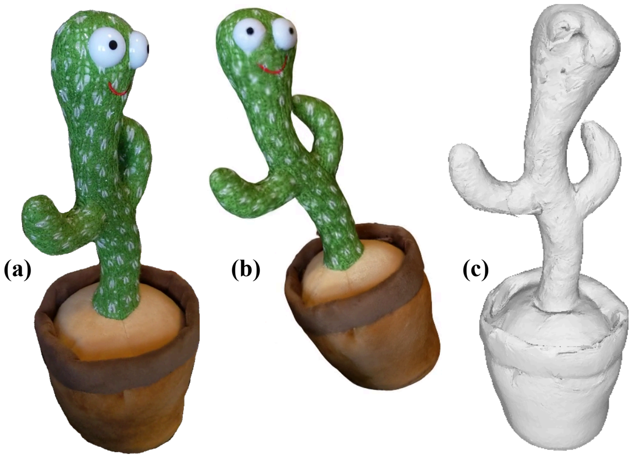

Figure 5. **Unbiased4d [17] results.** (a): input image. (b): rendered novel view. (c): extracted mesh.

4. Experiments

4.1. Datasets

Original D-NeRF dataset, which employs teleporting cameras to simulate multi-view supervision, is impractical for real-world applications. Thus we re-render D-NeRF's humanoid characters[1] with naturally moving cameras. We rotate the camera around the character at constant speed and return camera to the starting position as the animation stops. We place 2 test cameras rotating with the training camera, both looking at the character but are 45 degrees away from the training camera. Further illustration of the setup is available in the supplementary material. In addition to blender rendered dataset, we qualitatively tested our model on Unbiased4D [17], including one video of a deforming cactus toy captured using a hand-held camera. We also tested on People-Snapshot [1] comparing against Gaussian Avatar [13], featuring self-rotating actors captured with a static camera. We train our model on a single 3090 GPU with Adam [19] and PyTorch. The whole training pipeline takes about 90 minutes.

4.2. Main Results

Our main results on Smooth D-NeRF are demonstrated in Fig. 4, Fig. 8, and Sec. 4. We achieved better results than all compared methods. Since TiNeuVox and HexPlane lack an underlying geometric representation, they exhibit severe overfitting to training views. DG-Mesh, while more robust due to mesh-anchored Gaussians, still suffers from mild view-overfitting due to its use of vanilla 3D GS. However, the ”mutant” data features a relatively stationary motion, and our model's flexible motion representation is disadvantaged but still achieved better perceptive quality measured by LPIPS. Shown in Fig. 8, DG-Mesh generates noisy meshes due to its reliance on SfM to initialize meshes,

[1] Available on Adobe's Mixamo.

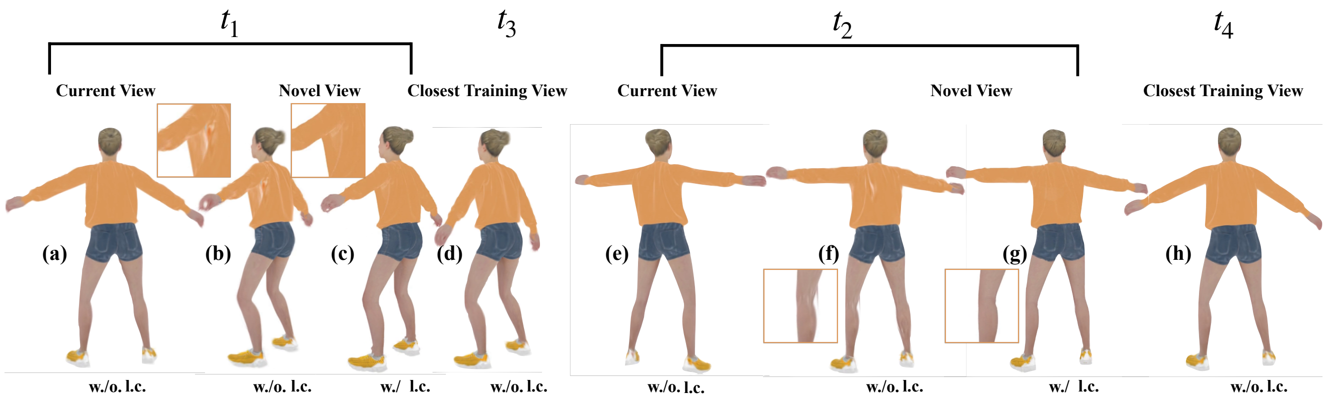

Figure 6. **Ablation comparison of locality constraints (l.c.).** (a) and (e): ablation rendering results from training pose at current timestep. (b) and (f): ablation rendering results from test pose. (c) and (g): rendering results from test pose without ablation. (d) and (h): spatially closest training views to test views at t_1 and t_2. (b) demonstrated view-overfitting caused by occluded Gaussians not being optimized, as shown by the large red Gaussian. Comparison between (f) and (g) shows our pipeline avoids 2D GS scales overfitting to the training view. (a) and (e) shows that these artifacts are not be visible to training views but will appear in novel test views.

Method	Hook				Mutant			
	PSNR ↑	SSIM ↑	LPIPS ↓	CD ↓	PSNR ↑	SSIM ↑	LPIPS ↓	CD ↓
TiNeuVox-B	13.996	0.859	0.1803	-	16.380	**0.914**	0.1801	-
HexPlane	27.836	0.966	0.0297	-	17.632	0.880	0.1126	-
DG-Mesh	26.123	0.968	0.0506	2	**18.506**	0.879	0.1057	**2.8**
Ours	**32.302**	**0.981**	**0.0208**	**0.8**	17.104	0.882	**0.1017**	3.8
Method	**Standup**				**Jumping Jacks**			
	PSNR ↑	SSIM ↑	LPIPS ↓	CD ↓	PSNR ↑	SSIM ↑	LPIPS ↓	CD ↓
TiNeuVox-B	12.583	0.839	0.2446	-	16.417	0.915	0.1799	-
HexPlane	21.256	0.898	0.0936	-	24.977	0.940	0.0770	-
DG-Mesh	22.481	0.937	0.0875	1.1	25.433	0.962	0.0665	7
Ours	**23.464**	**0.942**	**0.0572**	**0.9**	**25.995**	**0.964**	**0.0404**	**1.1**

Table 1. Quantitative comparison against previous works on Smooth D-NeRF. The best method in each metric is bolded. Chamfer distance (CD) is reported in scale $1\mathrm{e}{-3}$ and is unitless, same as training data of LRM.

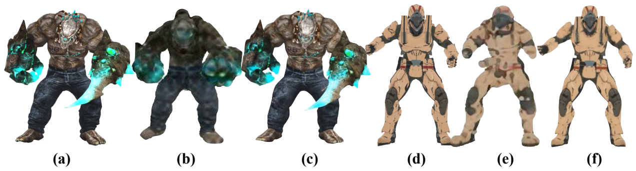

Figure 7. **Comparison with LRM direct output.** (a) and (d): ground-truth. (b) and (e): LRM direct output. (c) and (f): our pipeline's output. Further comparison available in supplementary material.

which performs poorly when object motion dominates over camera motion, and its direct use of time-conditioned MLPs for deformation, which is under-constrained. Our method produces accurate meshes with sharp details both because we initialize meshes directly from 3D priors and our use of merely 30 learnable control points to drive our meshes, providing strong local rigidity guarantee. We also show side-by-side comparison with LRM's direct rendering result in Fig. 7, and our model refines both geometry and appearance significantly. In addition, our results shown in Fig. 5 and Fig. 9 shows our model is robust against in-the-wild camera motion and static camera, achieving similar performance to Gaussian Avatar [13].

4.3. Ablation Studies

4.3.1 Effectiveness of Pruning Strategy

We tested our pipeline without child Gaussian pruning mechanism. As shown in Sec. 4.3.2, the improvement in PSNR is minimal, suggesting excessive Gaussians that don't contribute much to the photometric quality have indeed be pruned. The number of Gaussians, training time, and GPU memory usage also become much worse without child Gaussian pruning.

4.3.2 Effectiveness of Locality Constraints

We allow the 2D Gaussians to freely optimize their attributes, only keeping their normal direction aligned to the interpolated face normal and scale proportional to face size,

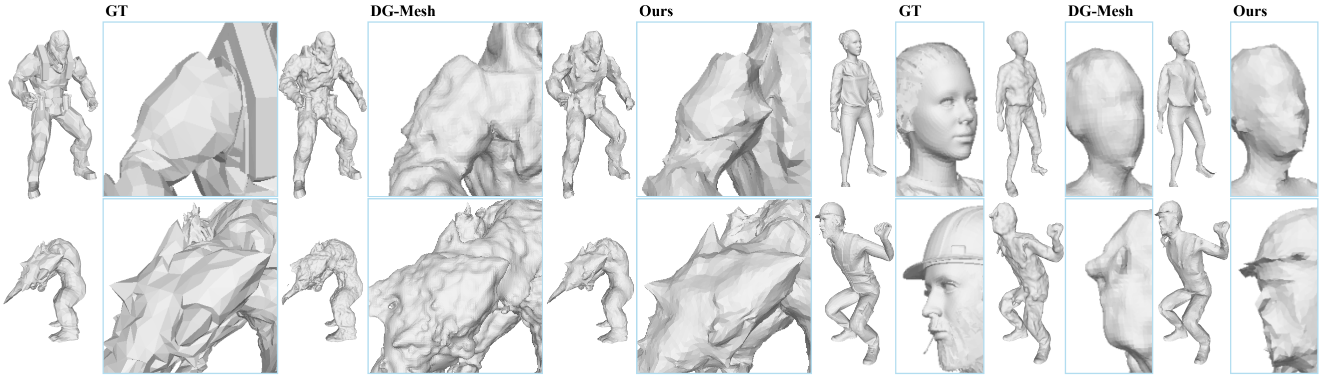

Figure 8. **Qualitative comparison of mesh reconstruction.** Our meshes contain much more geometric details due to the strong correlation between the photometric appearance and geometries brought by the Gaussian offset constraint.

similar to Mesh-GS [8]. Shown in Fig. 6, two types of artifacts occur: occluded Gaussians not being optimized, shown by the large red Gaussian in Fig. 6(b) that's not visible to any training view, and thin long Gaussians shown in Fig. 6(f) being Gaussians freely elongate along viewing direction. Our model is able to suppress both artifacts with our scale locality constraints.

	PSNR	CD	Num GS	Time	Memory
w./o. prune	24.860	-	∼750k	∼240 mins	23GB
w./o. offset	24.537	2.53	-	-	-
Full model	24.716	1.65	∼40k	∼80 mins	6GB

Table 2. Ablation comparison with and without offset constraint and child Gaussian pruning. CD reported in $1e{-}3$ scale.

4.3.3 Effectiveness of Offset Constraints

To test the effectiveness of the offset constraint discussed in Sec. 3.2.3, we remove the constraint and let the Gaussians freely offset along the face normal. We tested the average chamfer distance of the mesh sequences under this setup, and as shown in Sec. 4.3.2, the chamfer distance is higher. As shown in Fig. 10, the generated mesh seems coarser and lacks fine details, which suggests our offset constraint indeed allows for more accurate geometric reconstruction from photometric supervision.

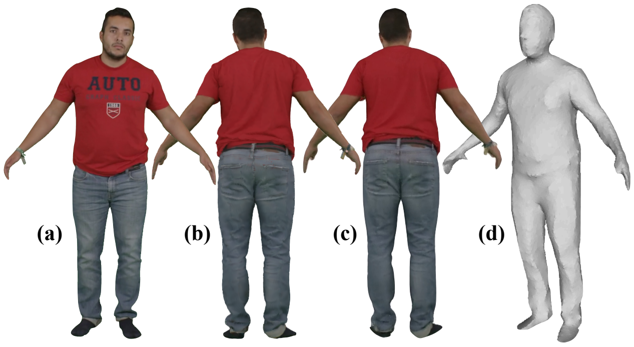

Figure 9. **People-Snapshot results.** (a): input view rendering result. (b): our method's novel view rendering result. (c): Gaussian Avatar's novel view rendering result. (d): our extracted mesh.

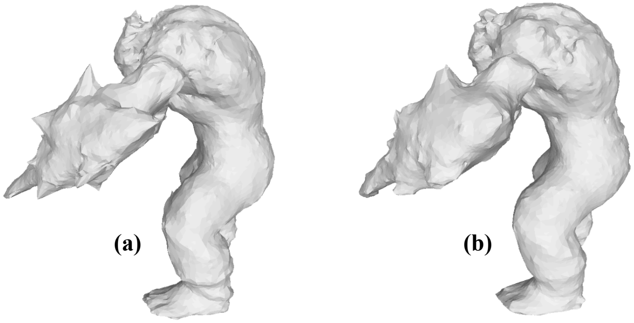

Figure 10. **Comparison of meshes extracted with and without offset constraint.** (a): mesh with Gaussian offset constraint. (b): mesh without Gaussian offset constraint.

5. Conclusion

We present Tessellation GS in this work, a pipeline to reconstruct dynamic objects using mesh and novel structured neural Gaussians with robust view extrapolation performance from natural monocular videos. Tessellation GS distributes 2D Gaussians on mesh faces in a structured way with minimal overlapping and decode their attributes from neural features on the vertices. Their population is adaptively controlled through carefully designed mesh-Gaussian quad trees linked by Gaussian opacities that adaptively add more Gaussians in regions with finer details. With scales strongly linked and constrained by mesh face shapes, Tessellation GS has shown excellent robustness to view-overfitting and achieved SOTA performance on monocular reconstruction task using natural camera motion in terms of both photometric performance and mesh quality.

Acknowledgements

This work was supported by National Science and Technology Major Project No.2022ZD0119404.

References

[1] Thiemo Alldieck, Marcus Magnor, Weipeng Xu, Christian Theobalt, and Gerard Pons-Moll. Video based reconstruction of 3d people models. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8387–8397, 2018. 6

[2] Gwangbin Bae and Andrew J Davison. Rethinking inductive biases for surface normal estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9535–9545, 2024. 5

[3] Ang Cao and Justin Johnson. Hexplane: A fast representation for dynamic scenes. In *CVPR*, pages 130–141, 2023. 1, 2

[4] Shenchang Eric Chen and Lance Williams. View interpolation for image synthesis. In *SIGGRAPH*, pages 279–288, 1993. 1

[5] Devikalyan Das, Christopher Wewer, Raza Yunus, Eddy Ilg, and Jan Eric Lenssen. Neural parametric gaussians for monocular non-rigid object reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10715–10725, 2024. 1

[6] Jiemin Fang, Taoran Yi, Xinggang Wang, Lingxi Xie, Xiaopeng Zhang, Wenyu Liu, Matthias Nießner, and Qi Tian. Fast dynamic radiance fields with time-aware neural voxels. In *SIGGRAPH Asia*, pages 1–9, 2022. 1, 2

[7] Hang Gao, Ruilong Li, Shubham Tulsiani, Bryan Russell, and Angjoo Kanazawa. Monocular dynamic view synthesis: A reality check. *Advances in Neural Information Processing Systems*, 35:33768–33780, 2022. 1, 2

[8] Lin Gao, Jie Yang, Bo-Tao Zhang, Jia-Mu Sun, Yu-Jie Yuan, Hongbo Fu, and Yu-Kun Lai. Mesh-based gaussian splatting for real-time large-scale deformation. *CoRR*, 2024. 2, 4, 8

[9] Antoine Guédon and Vincent Lepetit. Sugar: Surface-aligned gaussian splatting for efficient 3d mesh reconstruction and high-quality mesh rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5354–5363, 2024. 2

[10] Chen Guo, Tianjian Jiang, Xu Chen, Jie Song, and Otmar Hilliges. Vid2avatar: 3d avatar reconstruction from videos in the wild via self-supervised scene decomposition. In *CVPR*, pages 12858–12868, 2023. 1

[11] Zhiyang Guo, Wengang Zhou, Li Li, Min Wang, and Houqiang Li. Motion-aware 3d gaussian splatting for efficient dynamic scene reconstruction. *IEEE Transactions on Circuits and Systems for Video Technology*, 2024. 1

[12] Yicong Hong, Kai Zhang, Jiuxiang Gu, Sai Bi, Yang Zhou, Difan Liu, Feng Liu, Kalyan Sunkavalli, Trung Bui, and Hao Tan. Lrm: Large reconstruction model for single image to 3d. In *The Twelfth International Conference on Learning Representations*. 3

[13] Liangxiao Hu, Hongwen Zhang, Yuxiang Zhang, Boyao Zhou, Boning Liu, Shengping Zhang, and Liqiang Nie. Gaussianavatar: Towards realistic human avatar modeling from a single video via animatable 3d gaussians. In *CVPR*, pages 634–644, 2024. 1, 4, 6, 7

[14] Binbin Huang, Zehao Yu, Anpei Chen, Andreas Geiger, and Shenghua Gao. 2d gaussian splatting for geometrically accurate radiance fields. In *ACM SIGGRAPH 2024 conference papers*, pages 1–11, 2024. 2

[15] Yi-Hua Huang, Yang-Tian Sun, Ziyi Yang, Xiaoyang Lyu, Yan-Pei Cao, and Xiaojuan Qi. Sc-gs: Sparse-controlled gaussian splatting for editable dynamic scenes. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4220–4230, 2024. 1

[16] Wei Jiang, Kwang Moo Yi, Golnoosh Samei, Oncel Tuzel, and Anurag Ranjan. Neuman: Neural human radiance field from a single video. In *ECCV*, pages 402–418, 2022. 2

[17] Erik Johnson, Marc Habermann, Soshi Shimada, Vladislav Golyanik, and Christian Theobalt. Unbiased 4d: Monocular 4d reconstruction with a neural deformation model. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6598–6607, 2023. 6

[18] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM TOG*, 42(4):1–14, 2023. 1, 2

[19] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 6

[20] Youngjoong Kwon, Dahun Kim, Duygu Ceylan, and Henry Fuchs. Neural human performer: Learning generalizable radiance fields for human performance rendering. *NeurIPS*, 34:24741–24752, 2021. 2

[21] Tianye Li, Mira Slavcheva, Michael Zollhoefer, Simon Green, Christoph Lassner, Changil Kim, Tanner Schmidt, Steven Lovegrove, Michael Goesele, Richard Newcombe, and Zhaoyang Lv. Neural 3d video synthesis from multi-view video. In *CVPR*, pages 5521–5531, 2022. 2

[22] Zhe Li, Zerong Zheng, Lizhen Wang, and Yebin Liu. Animatable gaussians: Learning pose-dependent gaussian maps for high-fidelity human avatar modeling. In *CVPR*, pages 19711–19722, 2024. 1, 4

[23] Haotong Lin, Sida Peng, Zhen Xu, Yunzhi Yan, Qing Shuai, Hujun Bao, and Xiaowei Zhou. Efficient neural radiance fields for interactive free-viewpoint video. In *SIGGRAPH Asia*, pages 1–9, 2022. 2

[24] Youtian Lin, Zuozhuo Dai, Siyu Zhu, and Yao Yao. Gaussian-flow: 4d reconstruction with dynamic 3d gaussian particle. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 21136–21145, 2024. 1

[25] Isabella Liu, Hao Su, and Xiaolong Wang. Dynamic gaussians mesh: Consistent mesh reconstruction from monocular videos. *CoRR*, 2024. 1, 5, 6

[26] Matthew Loper, Naureen Mahmood, Javier Romero, Gerard Pons-Moll, and Michael J Black. Smpl: A skinned multi-person linear model. *ACM TOG*, 34(6):1–16, 2015. 1

[27] Tao Lu, Mulin Yu, Linning Xu, Yuanbo Xiangli, Limin Wang, Dahua Lin, and Bo Dai. Scaffold-gs: Structured 3d gaussians for view-adaptive rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 20654–20664, 2024. 2

[28] Jonathon Luiten, Georgios Kopanas, Bastian Leibe, and Deva Ramanan. Dynamic 3d gaussians: Tracking by persistent dynamic view synthesis. In *3DV*, 2024. 1, 2

[29] Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. In *ECCV*, pages 405–421, 2020. 1

[30] Richard A Newcombe, Shahram Izadi, Otmar Hilliges, David Molyneaux, David Kim, Andrew J. Davison, Pushmeet Kohli, Jamie Shotton, Steve Hodges, and Andrew Fitzgibbon. Kinectfusion: Real-time dense surface mapping and tracking. *IEEE international symposium on mixed and augmented reality.*, pages 127–136, 2011. 1

[31] Richard A Newcombe, Dieter Fox, and Steven M Seitz. Dynamicfusion: Reconstruction and tracking of non-rigid scenes in real-time. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 343–352, 2015. 1

[32] Xiao Pan, Zongxin Yang, Jianxin Ma, Chang Zhou, and Yi Yang. Transhuman: A transformer-based human representation for generalizable neural human rendering. In *ICCV*, pages 3544–3555, 2023. 2

[33] Keunhong Park, Utkarsh Sinha, Jonathan T Barron, Sofien Bouaziz, Dan B Goldman, Steven M Seitz, and Ricardo Martin-Brualla. Nerfies: Deformable neural radiance fields. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 5865–5874, 2021. 1, 2

[34] Keunhong Park, Utkarsh Sinha, Peter Hedman, Jonathan T Barron, Sofien Bouaziz, Dan B Goldman, Ricardo Martin-Brualla, and Steven M Seitz. Hypernerf: a higher-dimensional representation for topologically varying neural radiance fields. *ACM Transactions on Graphics (TOG)*, 40 (6):1–12, 2021. 2

[35] Sida Peng, Yuanqing Zhang, Yinghao Xu, Qianqian Wang, Qing Shuai, Hujun Bao, and Xiaowei Zhou. Neural body: Implicit neural representations with structured latent codes for novel view synthesis of dynamic humans. In *CVPR*, pages 9054–9063, 2021. 2

[36] Albert Pumarola, Enric Corona, Gerard Pons-Moll, and Francesc Moreno-Noguer. D-nerf: Neural radiance fields for dynamic scenes. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10318–10327, 2021. 1, 2

[37] Ruizhi Shao, Liliang Chen, Zerong Zheng, Hongwen Zhang, Yuxiang Zhang, Han Huang, Yandong Guo, and Yebin Liu. Floren: Real-time high-quality human performance rendering via appearance flow using sparse rgb cameras. In *SIGGRAPH Asia*, pages 1–10, 2022. 2

[38] Ruizhi Shao, Hongwen Zhang, He Zhang, Mingjia Chen, Yan-Pei Cao, Tao Yu, and Yebin Liu. Doublefield: Bridging the neural surface and radiance fields for high-fidelity human reconstruction and rendering. In *CVPR*, pages 15872–15882, 2022.

[39] Xin Suo, Yuheng Jiang, Pei Lin, Yingliang Zhang, Minye Wu, Kaiwen Guo, and Lan Xu. Neuralhumanfvv: Real-time neural volumetric human performance rendering using rgb cameras. In *CVPR*, pages 6226–6237, 2021. 2

[40] G. Taubin. Curve and surface smoothing without shrinkage. In *Proceedings of IEEE International Conference on Computer Vision*, pages 852–857, 1995. 3

[41] Zachary Teed and Jia Deng. Raft: Recurrent all-pairs field transforms for optical flow. In *ECCV*, pages 402–419, 2020. 5

[42] Dmitry Tochilkin, David Pankratz, ZeXiang Liu, Zixuan Huang, Adam Letts, Yangguang Li, Ding Liang, Christian Laforte, Varun Jampani, and Yan-Pei Cao. Triposr: Fast 3d object reconstruction from a single image. *CoRR*, 2024. 3

[43] Chung-Yi Weng, Brian Curless, Pratul P Srinivasan, Jonathan T Barron, and Ira Kemelmacher-Shlizerman. Humannerf: Free-viewpoint rendering of moving people from monocular video. In *CVPR*, pages 16210–16220, 2022. 2

[44] Guanjun Wu, Taoran Yi, Jiemin Fang, Lingxi Xie, Xiaopeng Zhang, Wei Wei, Wenyu Liu, Qi Tian, and Xinggang Wang. 4d gaussian splatting for real-time dynamic scene rendering. In *CVPR*, pages 20310–20320, 2024. 1

[45] Guanjun Wu, Taoran Yi, Jiemin Fang, Lingxi Xie, Xiaopeng Zhang, Wei Wei, Wenyu Liu, Qi Tian, and Xinggang Wang. 4d gaussian splatting for real-time dynamic scene rendering. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 20310–20320, 2024. 2

[46] Gengshan Yang, Minh Vo, Natalia Neverova, Deva Ramanan, Andrea Vedaldi, and Hanbyul Joo. Banmo: Building animatable 3d neural models from many casual videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2863–2873, 2022. 3

[47] Ziyi Yang, Xinyu Gao, Wen Zhou, Shaohui Jiao, Yuqing Zhang, and Xiaogang Jin. Deformable 3d gaussians for high-fidelity monocular dynamic scene reconstruction. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 20331–20341, 2024. 1, 2

[48] Yufeng Zheng, Wang Yifan, Gordon Wetzstein, Michael J Black, and Otmar Hilliges. Pointavatar: Deformable point-based head avatars from videos. In *CVPR*, pages 21057–21067, 2023. 1