Spurfies: Sparse-View Surface Reconstruction using Local Geometry Priors

Supplementary Material

A. Architectures

Our method employs four Multi-Layer Perceptrons (MLPs) and two sets of learnable latent codes. The MLPs are: A_{LP}, G_{LP}, used for local processing, signed distance regressor G_{REG}, and Radiance regressor A_{REG}. The latent codes are for color f^a and geometry f^g. Here are the details of these components:

Latent codes: The color latent codes $f^a \in \mathbb{R}^{64}$ and the geometry latent codes $f^g \in \mathbb{R}^{32}$ are both initialized from a normal distribution with variance $1\text{e}{-4}$.

Radiance Local Processing A_{LP}: This MLP comprises four linear layers with intermediate dimensions of 128. It takes as input the color latent codes f^a and the relative distance of query points to the neural points, with positional encoding using 6 frequencies.

Raidance Regression A_{REG}: processes the aggregated color latent codes along with the view direction (without positional encoding). It consists of three linear layers with dimensions [259, 128, 3].

Geometry Local Processing G_{LP}: This MLP also has four linear layers with intermediate dimensions of 128. It processes the geometry latent codes f^g and the relative distance of query points to the neural points, without any positional encoding. This MLP is frozen after learning the local geometry prior and remains unchanged during sparse view surface reconstruction.

Signed Distance Regression G_{REG}: This consists of a single frozen linear layer that maps the processed geometry latent codes to an SDF value.

B. Loss Functions

Feature Consistency Loss $\mathcal{L}_{\text{FC}}$ [6]: First, we estimate a set of surface points $\hat{\mathcal{P}} = \{\mathbf{p} | \mathbf{p} = \mathbf{x}_{i,u,v}(t^*)\}$ by finding zero-crossings t^* along each ray $\mathbf{x}_{i,u,v}(t)$ that are computed using linear interpolation between adjacent samples:

$$t^* = \frac{\hat{s}(\mathbf{x}(t_j))t_{j+1} - \hat{s}(\mathbf{x}(t_{j+1}))t_j}{\hat{s}(\mathbf{r}(t_j)) - \hat{s}(\mathbf{x}(t_{j+1}))}. \tag{1}$$

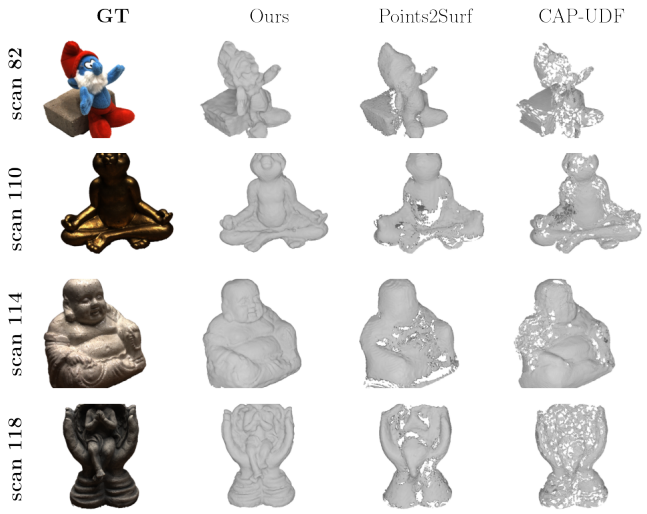

Figure 1. **Qualitative comparison** of mesh reconstruction with the point-based mesh reconstruction methods. In contrast to our approach, point-based mesh reconstruction methods often show missing areas, even when initialized with DUST3R [10] point clouds.

Figure 2. Sampled points from the reconstructed mesh on few scans from DTU dataset.

where t_j is estimated using $\hat{s}((t_j)) \cdot \hat{s}((t_{j+1})) < 0$. We then define the photo-consistency loss as:

$$\mathcal{L}_{\text{FC}} = \frac{1}{|\hat{\mathcal{P}}||\mathcal{I}|} \sum_{p_i \in \hat{\mathcal{P}}} \sum_{\pi_j \in \Pi} \| f_\phi(\pi_j(p_i)) - f_\phi(\pi_0(p_i)) \|_1, \tag{2}$$

where Π is the set of the projection matrices for the images $\mathcal{I}$, with $\mathcal{I}_0$ being the reference view, and f_ϕ computed with VisMVSNet [15].

Pseudo loss $\mathcal{L}_{\text{pseu}}$: We estimate surface points using rendering weights [5]. This approach ensures that the estimated points have a SDF value close to zero, effectively lying on the surface. We compute the estimated surface point location t^* along a ray $\mathbf{x}(t)$ as a weighted average of sample

positions:

$$t^* = \sum_i \frac{w_i \cdot t_i}{\sum w_i}, \tag{3}$$

where w_i are the rendering weights and t_i are the sample depths along the ray. Using these estimated surface points, we introduce the pseudo ground-truth loss:

$$\mathcal{L}_{\text{Pseu}} = \frac{1}{N} \sum ||\hat{s}(\mathbf{x}(t^*))||\,. \tag{4}$$

C. Implementation Details

Training: For sparse view reconstruction, we optimize a composite loss function:

$$L_{total} = \mathcal{L}_{\text{ren}} + \lambda_{\text{fc}} \cdot \mathcal{L}_{\text{fc}} + \lambda_{\text{pseu}} \cdot \mathcal{L}_{\text{pseu}} + \lambda_{\text{TV}} \cdot \mathcal{L}_{\text{TV}}, \tag{5}$$

where $\lambda_{\text{fc}} = 0.5$, $\lambda_{\text{pseu}} = 0.5$, and $\lambda_{\text{TV}} = 0.01$. We train the model for 100,000 iterations using the Adam optimizer [8] on a single A100 GPU.

We implemented efficient querying of K neural neighbors implemented using a GPU-accelerated VoxelGrid approach [11, 13]. The VoxelGrid parameters are configured as follows:

```
voxel_size = (0.025, 0.025, 0.025)  % Voxel size for each dimension
voxel_scale = (2, 2, 2)  % Voxel scale for each dimension
kernel_size = (3, 3, 3)  % Range of voxels searched for neighbors
max_points_per_voxel = 26  % Maximum number of points stored in a voxel
max_occ_voxels_per_example = 20000  % Maximum number of occupied voxels per point cloud
ranges = (-1.0, -1.0, -1.0, 1.0, 1.0, 1.0)  % Maximum ranges the VoxelGrid spans
```

The voxel size is set to match the average distance between neural points, ensuring an appropriate spatial distribution. Each voxel is limited to containing a maximum of 26 points, balancing between spatial resolution and computational efficiency.

We set K = 8 for neighbor queries, aiming to have, on average, 8 queried neural points for every ray-marched query point. This configuration strikes a balance between capturing sufficient local information and maintaining computational efficiency.

Datasets: For evaluation on the DTU [7] dataset, we adhere to the split established by S-VolSDF [12]. This protocol excludes scans from the training set of multi-view stereo methods, utilizing only those in the test/validation splits. Additionally, we follow the standard protocol employed by [1, 9, 14] for the quantitative evaluation and use masks for training.

Since, there are no previous sparse view method tested on Mip-NeRF 360 [2], we randomly select three input views for all qualitative evaluations. The lack of ground-truth point clouds precludes Chamfer Distance (CD) evaluation, limiting our analysis to qualitative results. Our evaluation encompasses four scenes from Mip-NeRF 360, with corresponding view IDs as follows: Garden: (DSC08116, DSC08121, DSC08140) Kitchen: (DSCF0683, DSCF0700, DSCF0716) Treehill: (_DSC9004, _DSC9005, _DSC9006) Stump: (_DSC9307, _DSC9313, _DSC9328).

Results: Our method demonstrates superior performance in mesh reconstruction compared to point-based techniques such as Points2Surf [4] and CAP-UDF [16]. As shown in Figure 1, even when using points obtained from Dust3R [10], these alternative methods often produce reconstructions with holes and fail to capture fine details. In contrast, our approach achieves more complete and detailed reconstructions, highlighting the effectiveness of our neural point-based representation and learned local geometry prior.

Additional reconstruction and Novel View Synthesis (NVS) results on the Mip-NeRF 360 dataset are presented in Figure 3 in comparison with NeuSurf [6] and S-VolSDF [12]. We also show points sampled from the reconstructed mesh Figure 2.

D. Local Prior

Data: To train our local prior, we design a setup that emulates volume rendering conditions. We sample two distinct sets of points:

1) **Query points** $\mathcal{X} = \{(\mathrm{x}_i, \rho_i)\}_{i=1}^{\mathrm{N}}$, sampled in close proximity to the mesh surface. These points are generated using two different variances (0.05 and 0.001) to simulate the ray-marching process in volume rendering. On average, we sample $\mathrm{N} = 500k$ query points for every mesh.

2) **Neural points** $\mathcal{N} = \{(\mathrm{p}_j, \mathrm{f}_j^g)\}_{j=1}^{\mathrm{M}}$, which represent the underlying structure of our reconstruction. To ensure density-agnostic learning during local prior training, we employ farthest point sampling on the mesh surface, maintaining an average inter-point distance of 0.025. The number of neural points, M, varies based on the mesh size, which is normalized to fit within a unit cube. During inference for sparse view reconstruction, we subsample the neural points to match the same density as training.

Our training data is from five classes in the ShapeNet [3] dataset: sofas, chairs, planes, tables, and lamps. To enhance robustness against noise, we add Gaussian noise with a variance of 0.01 to the neural points.

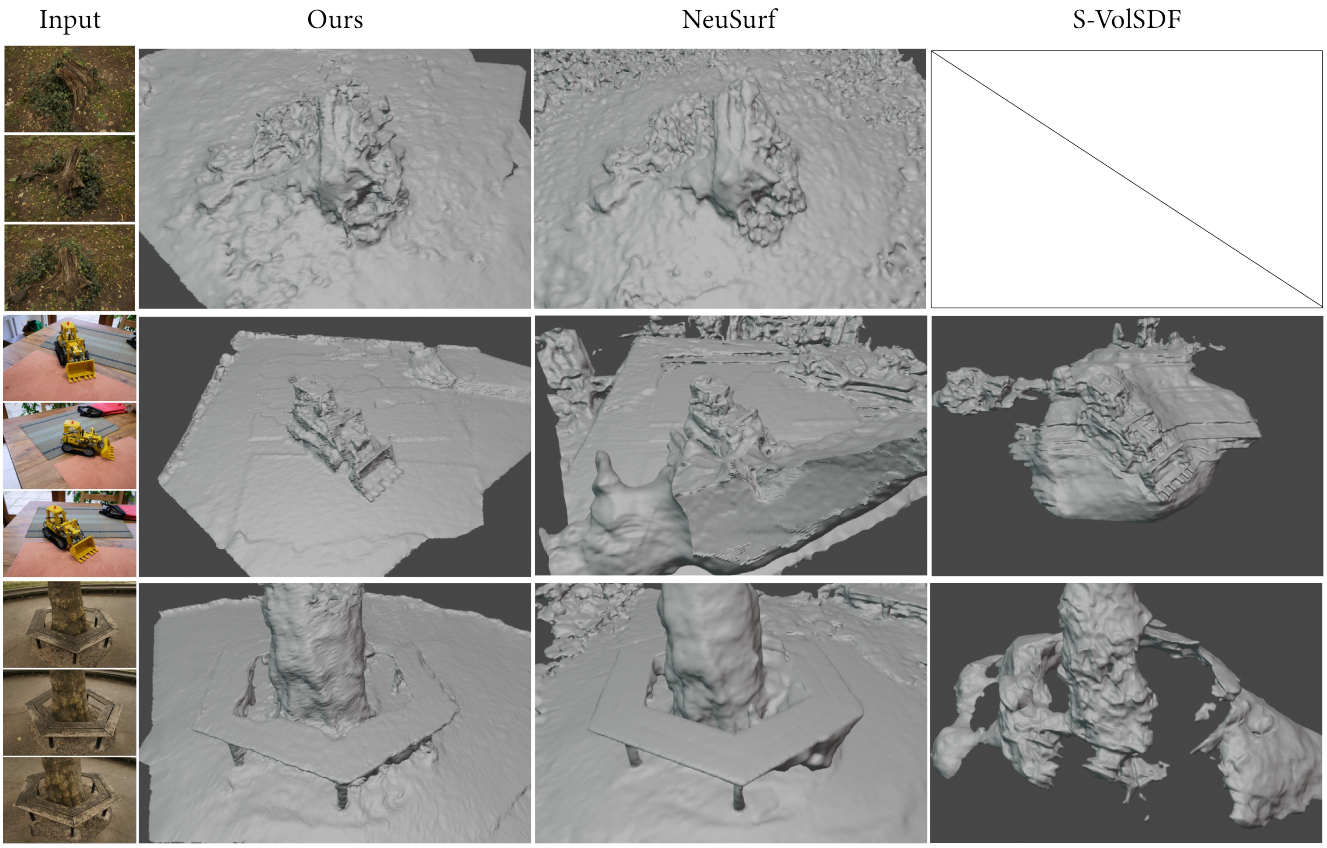

Figure 3. **Qualitative mesh reconstruction on Mip-NeRF360 [2]**. Compared to previous sparse view methods, we can achieve much better reconstruction on larger, unbounded scenes. S-VolSDF completely failed on the stump scene.

Training: To train the local prior, we employ a combination of loss functions:

$$\mathcal{L}_{\text{prior}} = \mathcal{L}_{\text{SDF}} + \lambda_{\text{TV}} \cdot \mathcal{L}_{\text{TV}} + \lambda_{\text{eik}} \cdot \mathcal{L}_{\text{eik}} \tag{6}$$

where $\lambda_{\text{TV}} = 1\text{e}{-2}$ and $\lambda_{\text{eik}} = 1\text{e}{-3}$.

Our training process utilizes a batch size of 5. Each batch instance comprises 40,000 randomly sampled query points, equally distributed between positive and negative SDF samples, along with 2,000 neural points. These neural points are padded with points outside the unit cube to ensure consistent batch size.

We train the geometry MLP and the latent codes for 5,000 epochs. For the latent codes, we implement a cosine annealing learning rate schedule, starting at $1\text{e}{-2}$ and gradually decreasing to $3\text{e}{-4}$. The MLP is trained with a constant learning rate of $3\text{e}{-4}$. We use the Adam [8] optimizer throughout the training process. Instead of $K = 8$ during sparse view reconstruction, we set $K = 4$ neighbors during the local prior training.

The total training time for the local prior is approximately 8 hours, utilizing a single A100 GPU. This comprehensive training approach ensures that our local prior effectively captures the geometric properties of diverse shapes, enabling robust sparse-view reconstruction.

Results: We show some quantitative results in Figure 5 of surface reconstruction on ShapeNet [3] dataset and the Stanford bunny. These results are shown for unseen objects after training the prior. The geometry MLP is frozen and only the geometry latent codes are optimized. We achieve quality mesh reconstruction with high surface details.

We extend our analysis to demonstrate the potential of our optimized geometry latent codes for point cloud clustering. Figure 6 illustrates this capability using the Stanford bunny model, where we present six distinct clusters derived from these codes.

The clustering results reveal an property of our optimized geometry latent codes. These codes appear to capture and encode local surface orientations effectively. As a result, the clustering process groups points with similar local geometric characteristics together. This suggests that our method not only reconstructs the surface accurately but also learns a meaningful representation of local surface properties. Specifically, we observe that: 1) Points belong-

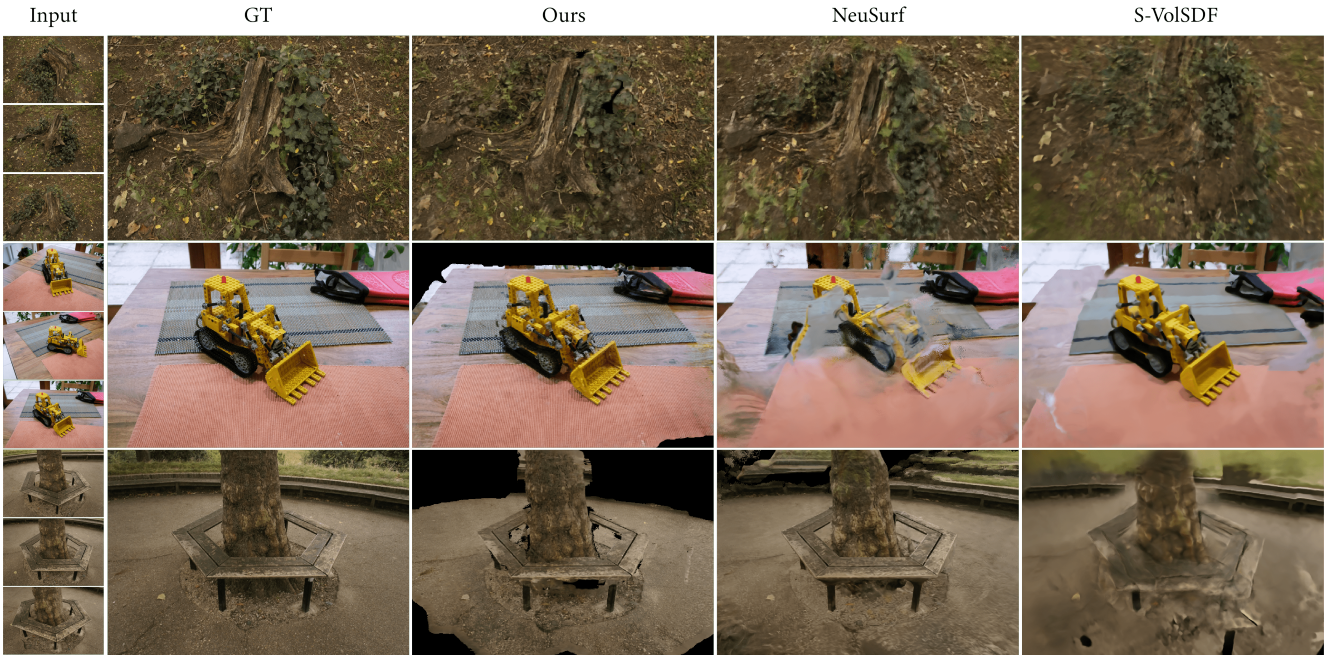

Figure 4. **Qualitative NVS on Mip-NeRF360 [2].** Spurfies can synthesize novel views in higher quality than previous sparse-view methods.

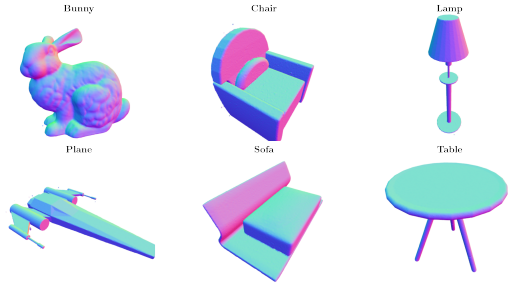

Figure 5. **Mesh reconstruction** results on a few unseen objects from ShapeNet [3] and the Stanford bunny.

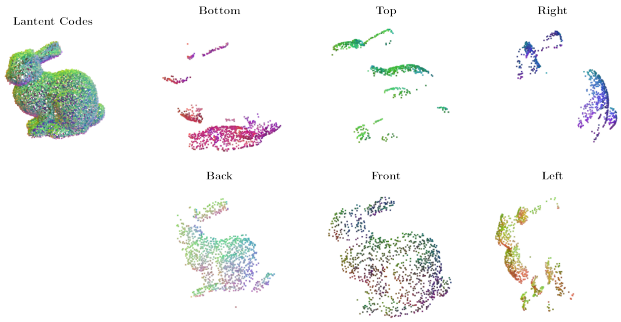

Figure 6. Clustering of optimized geometry latent codes based on six orientations. The geometry latent codes add local descriptive information to point clouds.

ing to the same cluster tend to have similar surface normals or curvature properties. 2) Transitions between clusters are generally smooth, indicating a continuous representation of geometric features which is achieved using $\mathcal{L}_{\text{TV}}$.

This clustering capability demonstrates an additional utility of our approach beyond surface reconstruction. It suggests potential applications in shape analysis, feature detection, and semantic segmentation of 3D models.

References

[1] Henrik Aanæs, Rasmus Ramsbøl Jensen, George Vogiatzis, Engin Tola, and Anders Bjorholm Dahl. Large-scale data for multiple-view stereopsis. *International Journal of Computer Vision*, 120:153–168, 2016. 2

[2] Jonathan T Barron, Ben Mildenhall, Dor Verbin, Pratul P Srinivasan, and Peter Hedman. Mip-nerf 360: Unbounded anti-aliased neural radiance fields. In *CVPR*, 2022. 2, 3, 4

[3] Angel X Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, et al. Shapenet: An information-rich 3d model repository. *arXiv*, 2015. 2, 3, 4

[4] Philipp Erler, Paul Guerrero, Stefan Ohrhallinger, Niloy J Mitra, and Michael Wimmer. Points2surf learning implicit surfaces from point clouds. In *ECCV*, 2020. 2

[5] Qiancheng Fu, Qingshan Xu, Yew-Soon Ong, and Wenbing Tao. Geo-neus: Geometry-consistent neural implicit surfaces learning for multi-view reconstruction. In *NeurIPS*, 2022. 1

[6] Han Huang, Yulun Wu, Junsheng Zhou, Ge Gao, Ming Gu, and Yu-Shen Liu. Neusurf: On-surface priors for neural surface reconstruction from sparse input views. In *AAAI*, 2024. 1, 2

[7] Rasmus Jensen, Anders Dahl, George Vogiatzis, Engil Tola, and Henrik Aanæs. Large scale multi-view stereopsis evaluation. In *CVPR*, 2014. 2

[8] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv*, 2014. 2, 3

[9] Michael Niemeyer, Jonathan T Barron, Ben Mildenhall, Mehdi SM Sajjadi, Andreas Geiger, and Noha Radwan. Regnerf: Regularizing neural radiance fields for view synthesis from sparse inputs. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5480–5490, 2022. 2

[10] Shuzhe Wang, Vincent Leroy, Yohann Cabon, Boris Chidlovskii, and Jerome Revaud. Dust3r: Geometric 3d vision made easy. In *arXiv*, 2023. 1, 2

[11] Christopher Wewer, Eddy Ilg, Bernt Schiele, and Jan Eric Lenssen. SimNP: Learning self-similarity priors between neural points. In *ICCV*, 2023. 2

[12] Haoyu Wu, Alexandros Graikos, and Dimitris Samaras. S-volsdf: Sparse multi-view stereo regularization of neural implicit surfaces. In *ICCV*, 2023. 2

[13] Qiangeng Xu, Zexiang Xu, Julien Philip, Sai Bi, Zhixin Shu, Kalyan Sunkavalli, and Ulrich Neumann. Point-NeRF: Point-based neural radiance fields. In *CVPR*, 2022. 2

[14] Lior Yariv, Jiatao Gu, Yoni Kasten, and Yaron Lipman. Volume rendering of neural implicit surfaces. In *NeurIPS*, 2021. 2

[15] Jingyang Zhang, Shiwei Li, Zixin Luo, Tian Fang, and Yao Yao. Vis-mvsnet: Visibility-aware multi-view stereo network. In *ICCV*, 2023. 1

[16] Junsheng Zhou, Baorui Ma, Shujuan Li, Yu-Shen Liu, Yi Fang, and Zhizhong Han. Cap-udf: Learning unsigned distance functions progressively from raw point clouds with consistency-aware field optimization. In *IEEE TPAMI*, 2024. 2