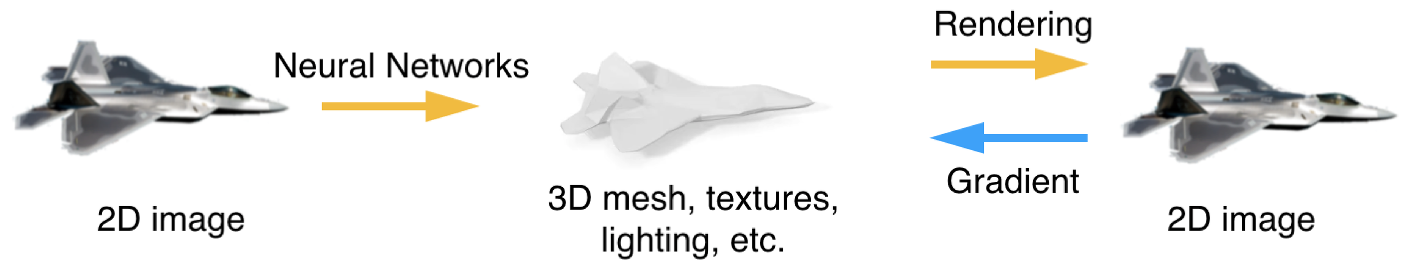
Soft Rasterizer: A Differentiable Renderer for Image-based 3D Reasoning

Shichen Liu[1,2], Tianye Li[1,2], Weikai Chen[1], and Hao Li[1,2,3]

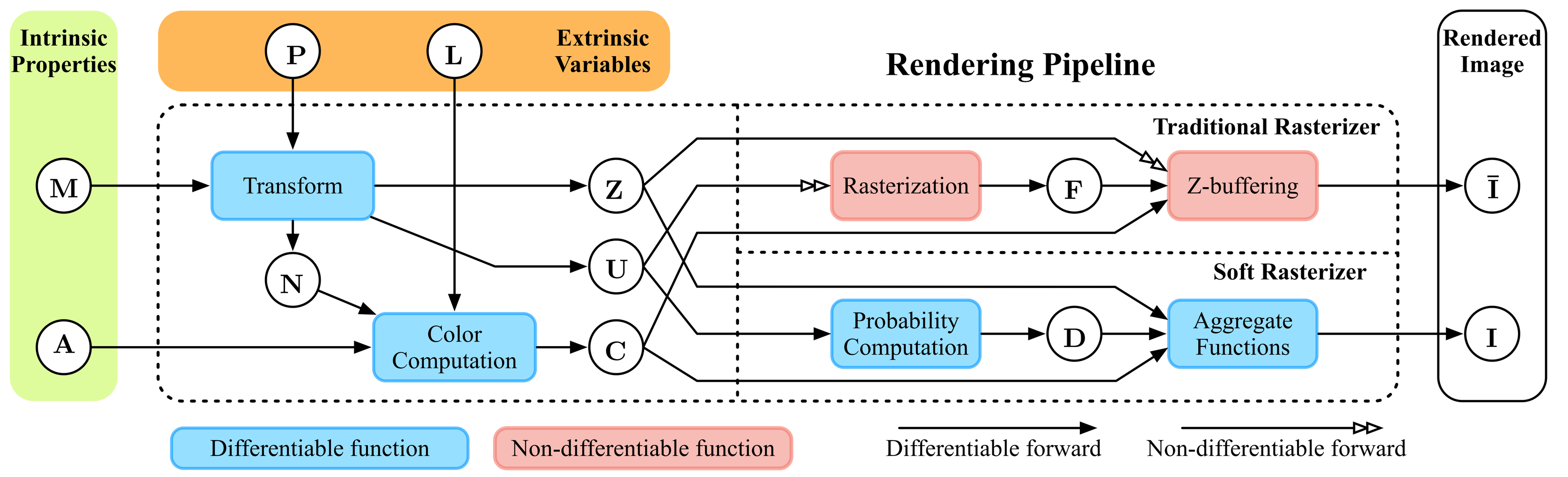
[1]USC Institute for Creative Technologies [2]University of Southern California [3]Pinscreen

Overview



- For image-based 3D reasoning tasks, supervised methods rely on a large amount of labelled 3D data that are hard to acquire.
- We propose SoftRas, a truly differentiable mesh renderer, which enables 3D unsupervised learning of 3D properties, including geometry, texture, pose, etc., only from 2D images.
- SoftRas can 1) directly render color mesh using differentiable functions and 2) flow gradients from pixels to all mesh vertices, including those occluded and far-range ones.

Rendering Pipeline

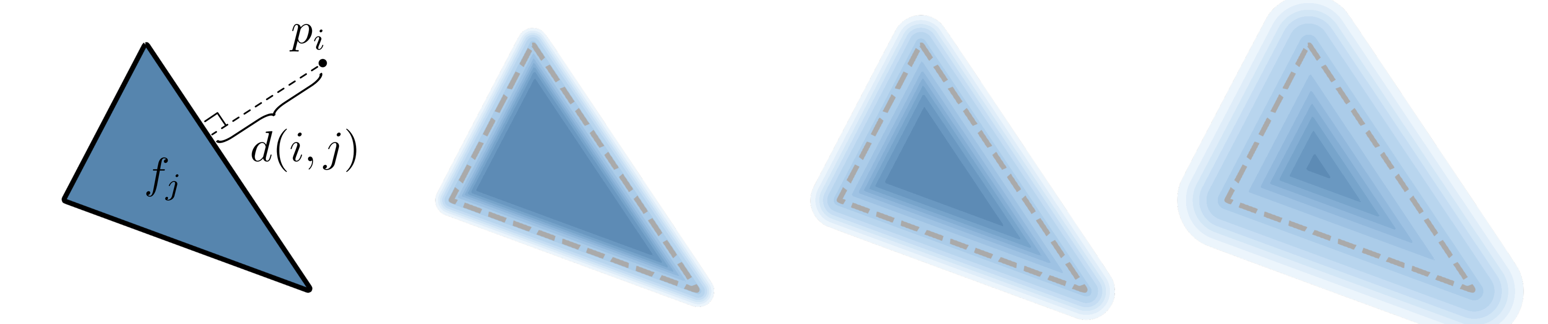


- We compare standard rendering pipeline (upper branch) and our rendering framework (lower branch).
- Standard graphics rendering: rasterization and z-buffering are not differentiable due to the discrete sampling operations.
- Ours: we propose probability maps and aggregate functions as their differentiable substitutes.

Probability Maps

- We compute the influence of each triangle to each pixel as

$$\mathcal{D}_j^i = sigmoid(\delta_j^i \cdot \frac{d^2(i,j)}{\sigma}) \quad (1)$$



(a) ground truth (b) $\sigma = 0.003$ (c) $\sigma = 0.01$ (d) $\sigma = 0.03$

Aggregate Functions

- We compute soft color in a softmax mannar:

$$I^i = \mathcal{A}_S(\{C_j\}) = \sum_j w_j^i C_j^i + w_b^i C_b, \quad (2)$$

where w_j^i is weighted by the relative depth and distance to pixel P_i of the triangles.

$$w_j^i = \frac{\mathcal{D}_j^i \exp(z_j^i/\gamma)}{\sum_k \mathcal{D}_k^i \exp(z_k^i/\gamma) + \exp(\epsilon/\gamma)} \quad (3)$$

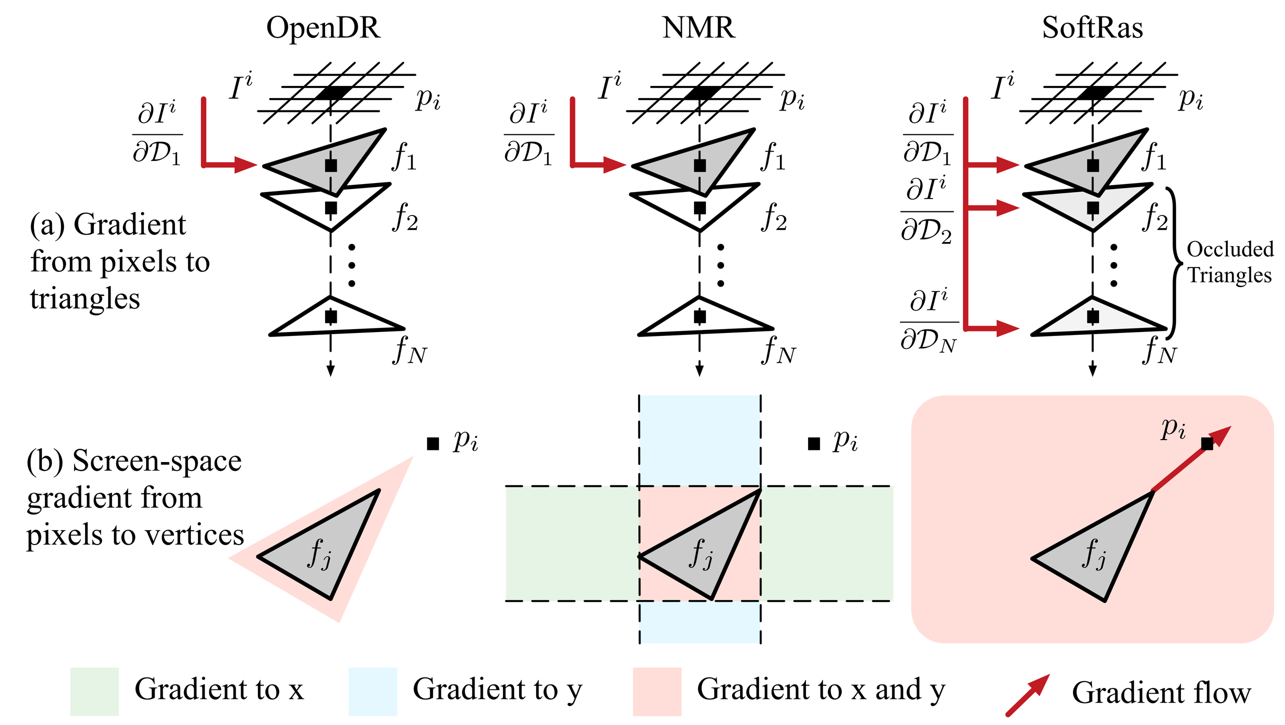
- This formulation leads to reasonable gradients:

$$\begin{cases} \frac{\partial I^i}{\partial \mathcal{D}_j^i} = \sum_k \frac{\partial I^i}{\partial w_k^i}\frac{\partial w_k^i}{\partial \mathcal{D}_j^i} + \frac{\partial I^i}{w_b^i}\frac{\partial w_b^i}{\partial \mathcal{D}_j^i} = \frac{w_j^i}{\mathcal{D}_j^i}(C_j^i - I^i) \\ \frac{\partial I^i}{\partial z_j^i} = \sum_k \frac{\partial I^i}{\partial w_k^i}\frac{\partial w_k^i}{\partial z_j^i} + \frac{\partial I^i}{w_b^i}\frac{\partial w_b^i}{\partial z_j^i} = \frac{w_j^i}{\gamma}(C_j^i - I^i) \end{cases} \quad (4)$$

- Our silhouette aggregation function is defined as:

$$I_s^i = \mathcal{A}_O(\{\mathcal{D}_j\}) = 1 - \prod_j (1 - \mathcal{D}_j^i) \quad (5)$$

Comparison to Previous Methods



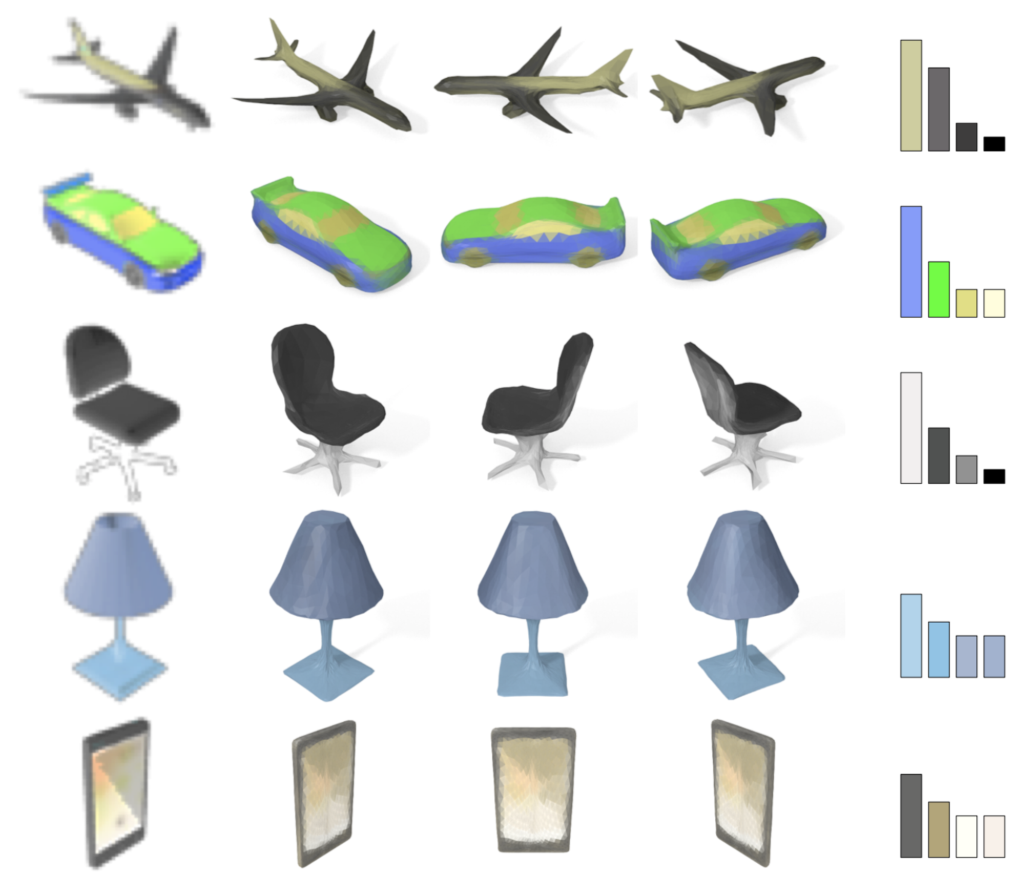
Experimental Results (3D IoU on ShapeNet)

Category	Airplane	Bench	Dresser	Car	Chair	Display	Lamp
retrieval	0.5564	0.4875	0.5713	0.6519	0.3512	0.3958	0.2905
voxel	0.5556	0.4924	0.6823	0.7123	0.4494	0.5395	0.4223
NMR	0.6172	0.4998	0.7143	0.7095	0.4990	0.5831	0.4126
Ours (sil.)	0.6419	0.5080	0.7116	0.7697	0.5270	0.6156	**0.4628**
Ours (full)	**0.6670**	**0.5429**	**0.7382**	**0.7876**	**0.5470**	**0.6298**	0.4580

Category	Speaker	Rifle	Sofa	Table	Phone	Vessel	Mean
retrieval	0.4600	0.5133	0.5314	0.3097	0.6696	0.4078	0.4766
voxel	0.5868	0.5987	0.6221	0.4938	0.7504	0.5507	0.5736
NMR	0.6536	0.6322	0.6735	0.4829	0.7777	0.5645	0.6015
Ours (sil.)	0.6654	**0.6811**	0.6878	0.4487	0.7895	0.5953	0.6234
Ours (full)	**0.6807**	0.6702	**0.7220**	**0.5325**	**0.8127**	**0.6145**	**0.6464**

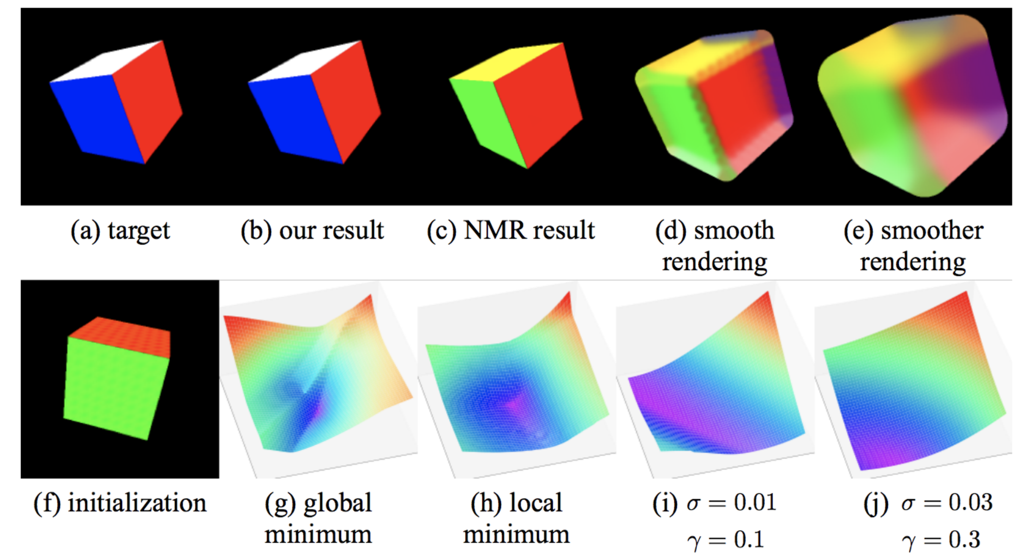
Experimental Results (cont.)

- Visual results of 3D unsupervised single-view reconstruction



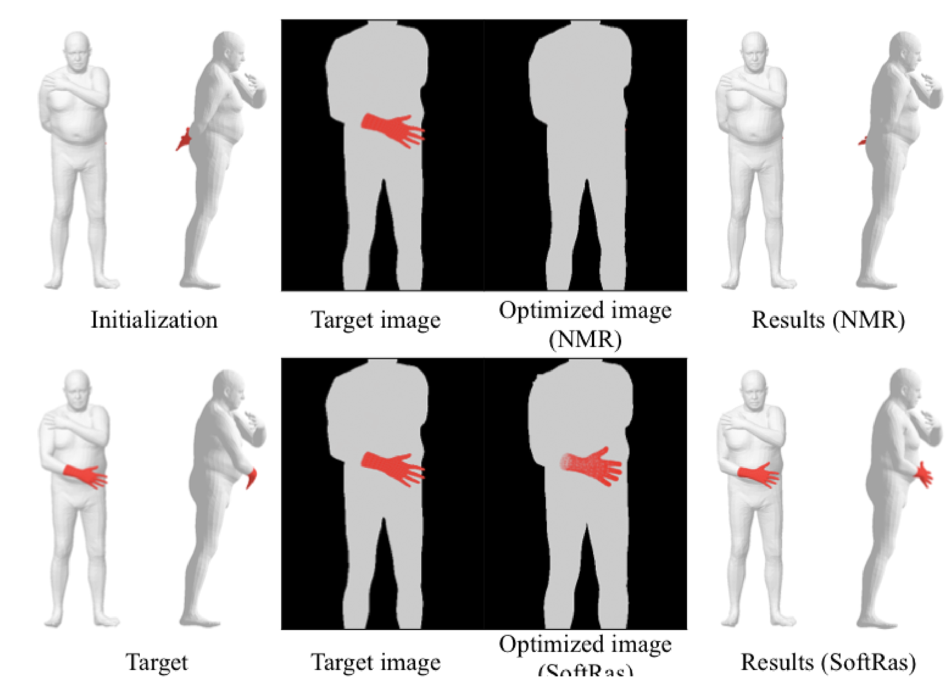
Input Reconstructed Results Learned Color Palettes

- Rigid pose estimation. We can smooth the loss landscape to avoid local minima by tuning the sharpness of rendering.



(a) target (b) our result (c) NMR result (d) smooth rendering (e) smoother rendering
(f) initialization (g) global minimum (h) local minimum (i) $\sigma = 0.01$ $\gamma = 0.1$ (j) $\sigma = 0.03$ $\gamma = 0.3$

- Non-rigid pose fitting. SoftRas can flow gradients to occluded vertices.



Initialization Target image Optimized image (NMR) Results (NMR)
Target Target image Optimized image (SoftRas) Results (SoftRas)

Discussion

Our code on PyTorch is released at:
https://github.com/ShichenLiu/SoftRas