Generalized Patch-Based Neural Rendering

ECCV 2022 (Oral)

Mohammed Suhail
University of British Columbia

Carlos Esteves
Google Research

Leonid Sigal
University of British Columbia

Ameesh Makadia
Google Research

Jiho Park
2019142056
2023.10.22

If you’re interested in 3D Generation…

ProlificDreamer (NeurIPS 2023 Spotlight)

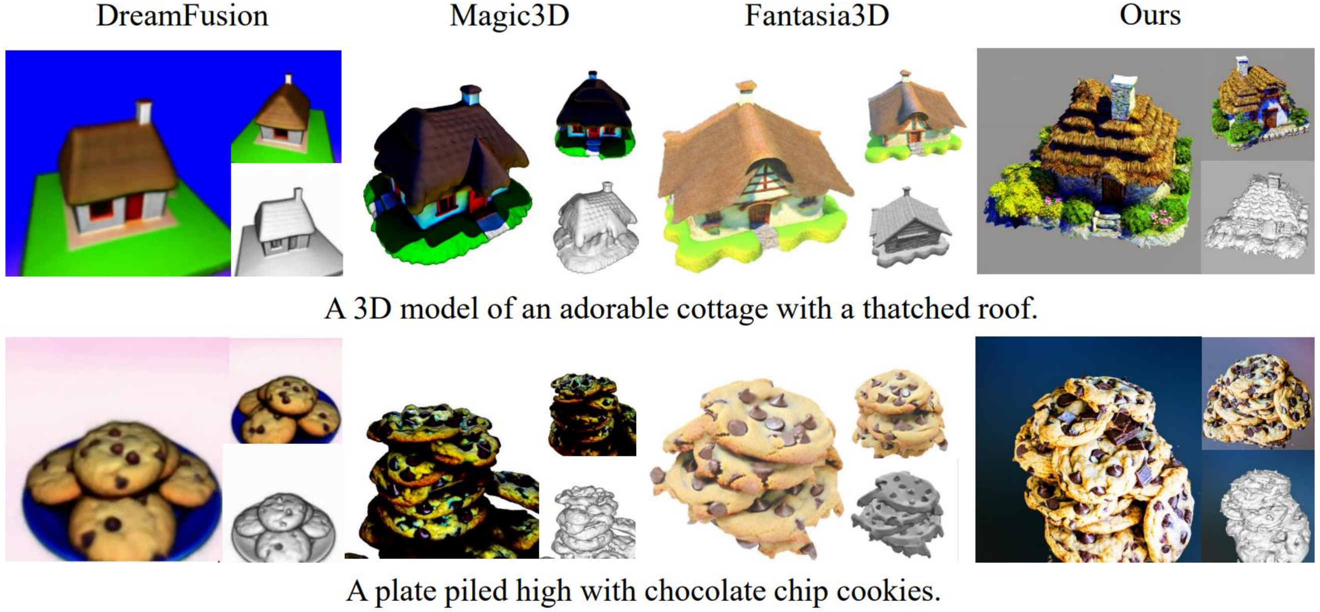
Score Distillation

$$\nabla_\theta \mathcal{L}_{\text{SDS}}(\theta) \approx \mathbb{E}_{t,\boldsymbol{\epsilon},c}\left[\omega(t)(\boldsymbol{\epsilon}_{\text{pretrain}}(\boldsymbol{x}_t,t,y)-\boldsymbol{\epsilon})\frac{\partial \boldsymbol{g}(\theta,c)}{\partial\theta}\right]$$

↓

Variational Score Distillation
(w/ LoRA)

$$\nabla_\theta \mathcal{L}_{\text{VSD}}(\theta) \triangleq \mathbb{E}_{t,\boldsymbol{\epsilon},c}\left[\omega(t)\left(\boldsymbol{\epsilon}_{\text{pretrain}}(\boldsymbol{x}_t,t,y)-\boldsymbol{\epsilon}_\phi(\boldsymbol{x}_t,t,c,y)\right)\frac{\partial \boldsymbol{g}(\theta,c)}{\partial\theta}\right]$$

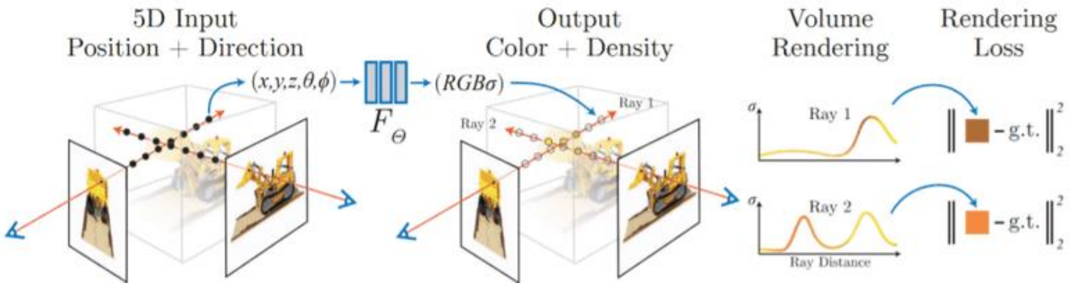


Contents

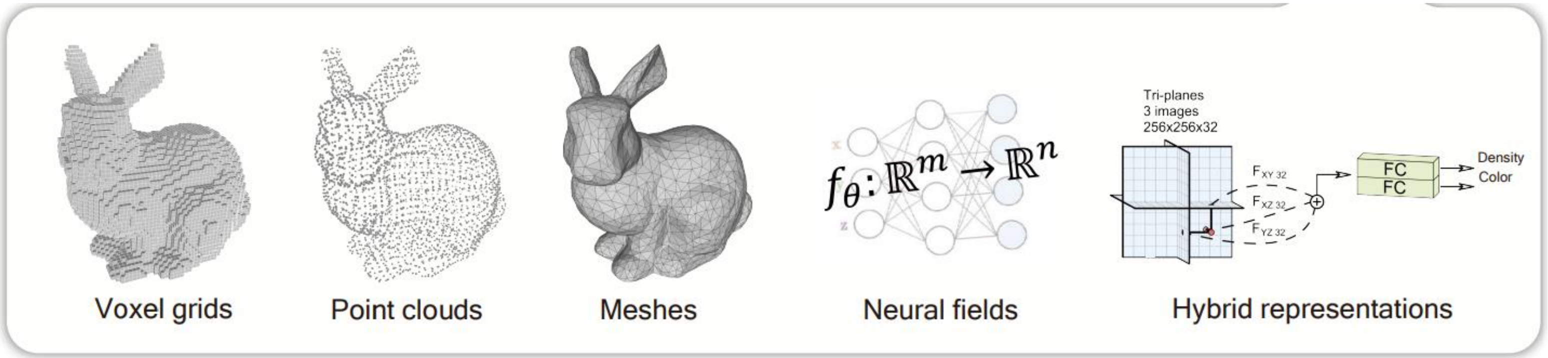
[1] Preliminaries: Novel View Synthesis

1) NeRF-like methods

- Representing 3D scene with Neural Network (Implicit)



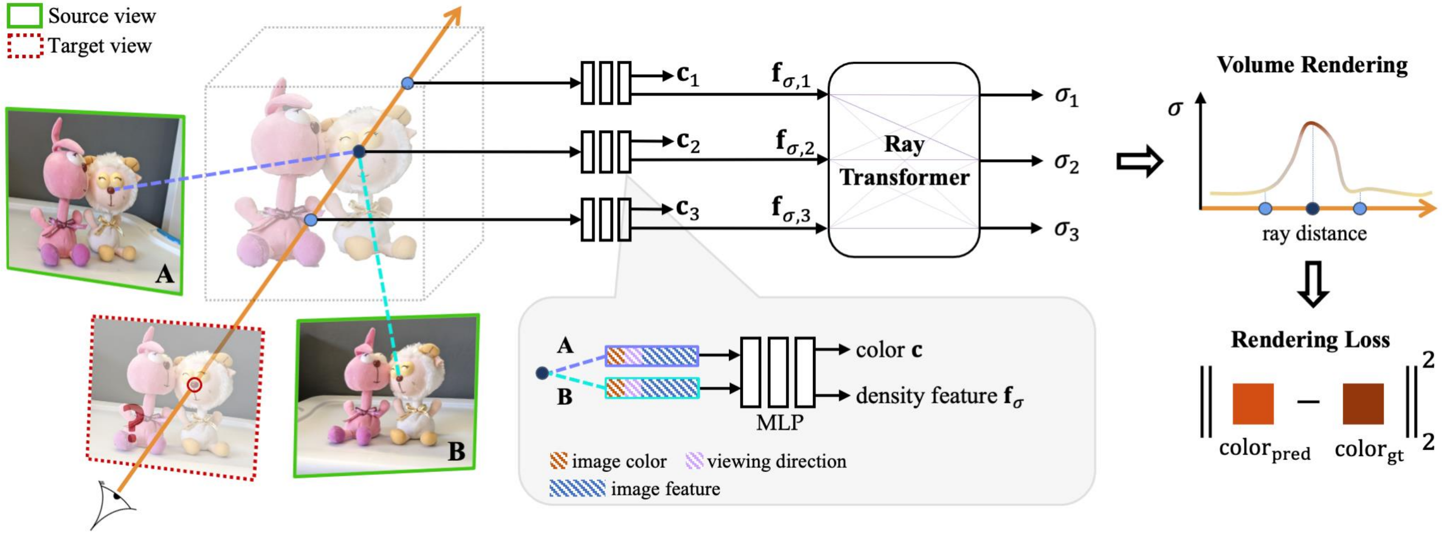
- *Now SOTA 3D Representations:* Hybrid (e.g. InstantNGP, Tri-mipRF, Pointcloud + MLP ..)



[1] Preliminaries: Novel View Synthesis

2) Image Based Rendering Methods

- Learning the Interpolation between reference views
- Generalization ↑, Performance ↓
- IBRNet (CVPR 2021)
  - ✓ MLP: Features from near views → color, density feature
  - ✓ Ray Transformer: Blending the density among single ray



[2] Concept

Image Based Rendering

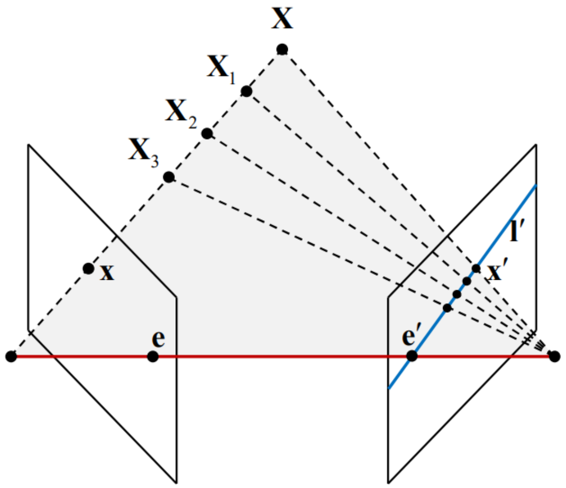
1) Generalization

- Previous NeRF: single network ↔ single scene
- Generalization in NeRF: MVSNeRF, PixelNeRF …
- Every Settings from GPNR are focused on “Generalization”.

2) Leveraging Epipolar Constraint

- pixel ‘x’ is from $X_1, X_2, \ldots$

→ Epipolar line contains possible solutions

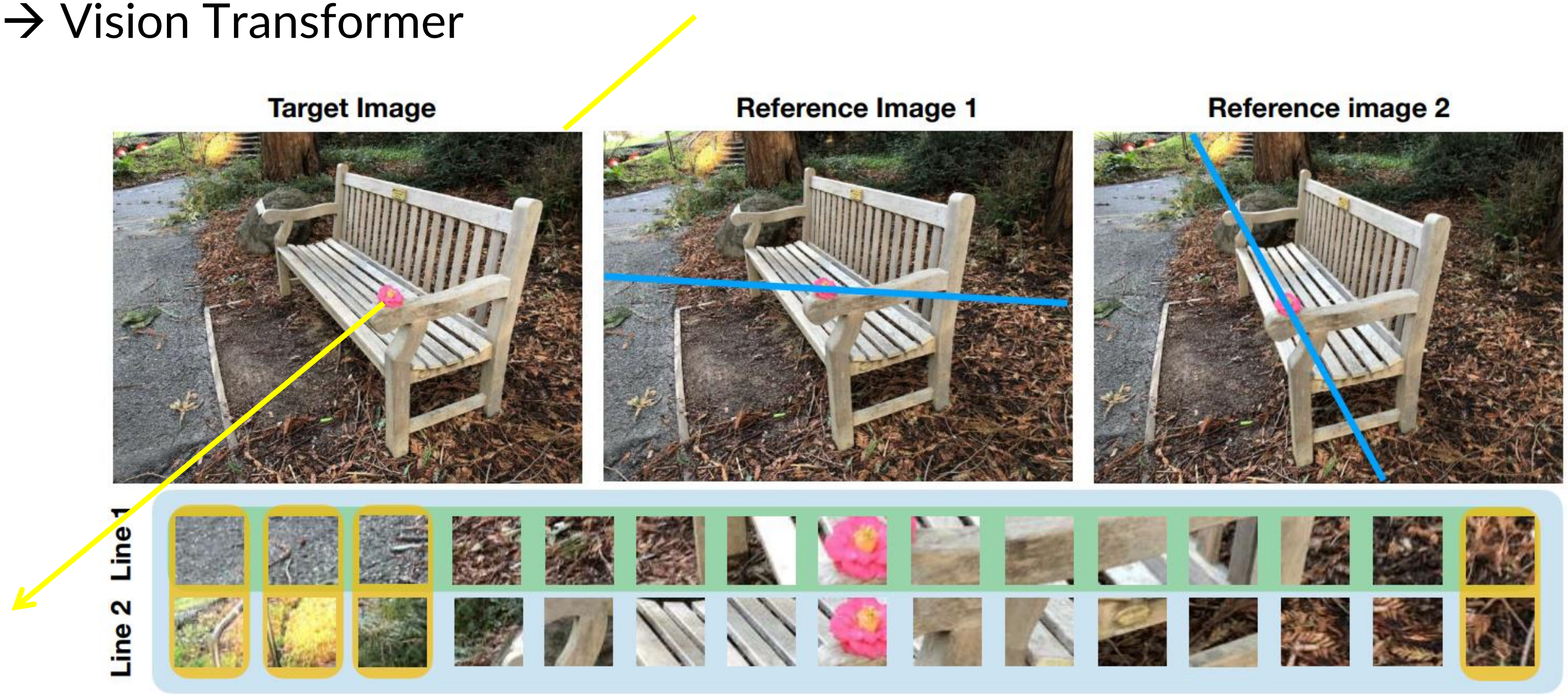


Epipolar constraint
: For each point observed in one image, the same point must be observed in the other image must be on a corresponding epipolar line

[2] Concept

2) Leveraging Epipolar Constraint

- Target Ray is hitting the flower pixel
- Inference the target pixel with the patches from epipolar line
- How? → Vision Transformer



[3] Generalized Patch Based Neural Rendering

1. Ray Representation

2. Three Positional Encodings

3. Three Transformers

4. Experiments

[3] Generalized Patch Based Neural Rendering

1. Ray Representation (Target Ray & Reference Ray to model)

Focusing on Generalization

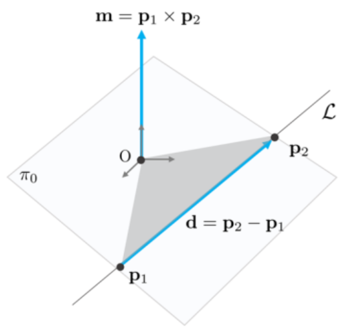
[Step 1] **Plücker Coordinates**

✓ Generalized Representation

w/o Constraint

Plücker Coordinates

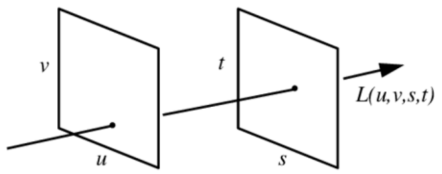
$L = L(d, m)$



w/ Constraint

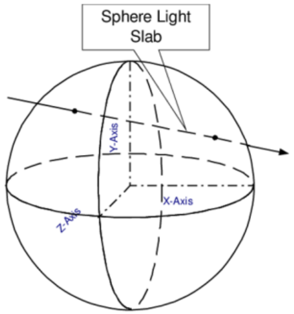
Light Slab: Forward Facing Scene

$L = L((u, v), (s, t))$



Sphere: Bounded Scene

$L = L((\theta_1, \phi_1), (\theta_2, \phi_2))$



[Step 2] **Canonicalization**

✓ Target Ray: origin = (0,0,0), direction = (0,0,1)

[3] Generalized Patch Based Neural Rendering

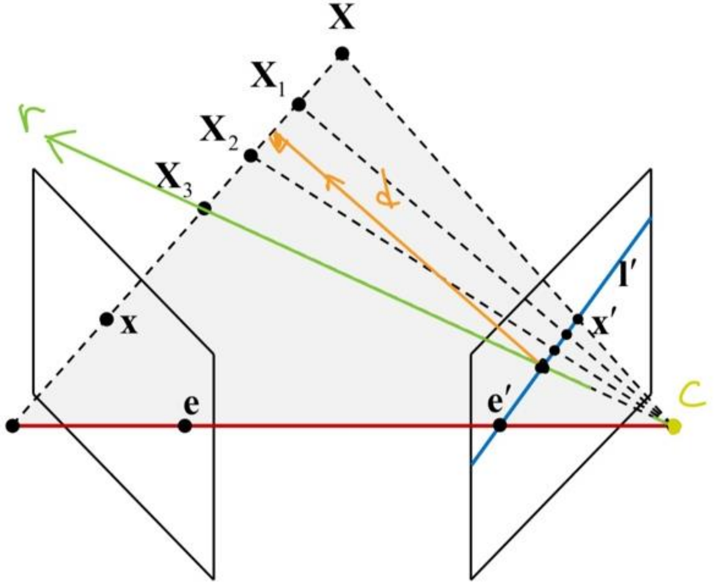
2. Three Positional Encodings

Reference View/Patches Information → Positional Encoding

1) Reference Ray: *r*

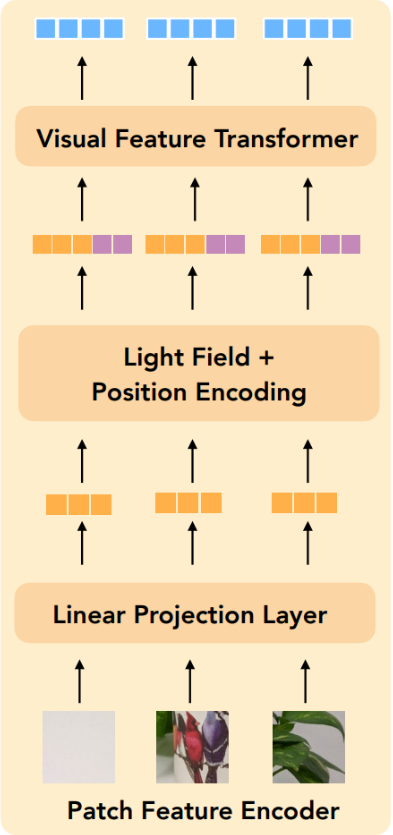
2) Distance to Target Ray: *d*

3) Relative Camera Pose: *c*



Additional Notations
- ***k*** : view number
- ***m*** : patch number(depth)
- ***p*** :patch

$$f_0^{k,m} = [\, p_k^m \,||\, r_k^m \,||\, d^m \,||\, c_k \,] \rightarrow$$



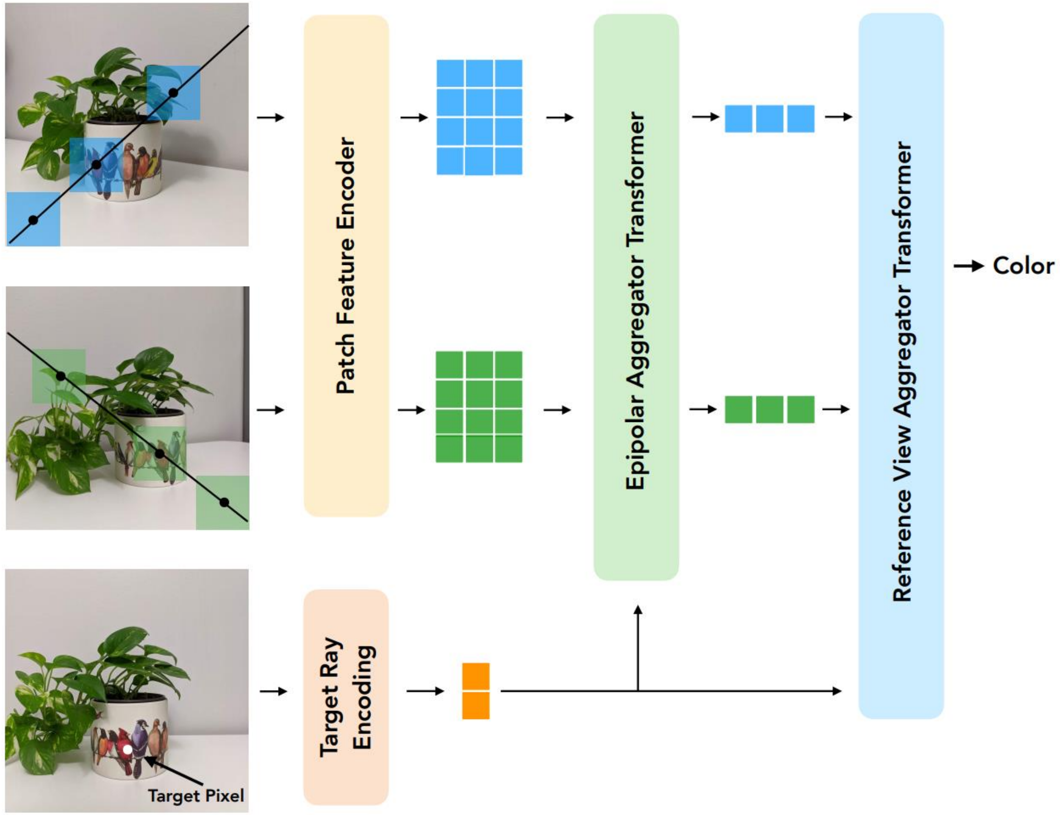
[3] Generalized Patch Based Neural Rendering

3. Three Transformers

1) Patch Feature Encoder

2) Epipolar Aggregator Transformer

3) Reference View Aggregator Transformer



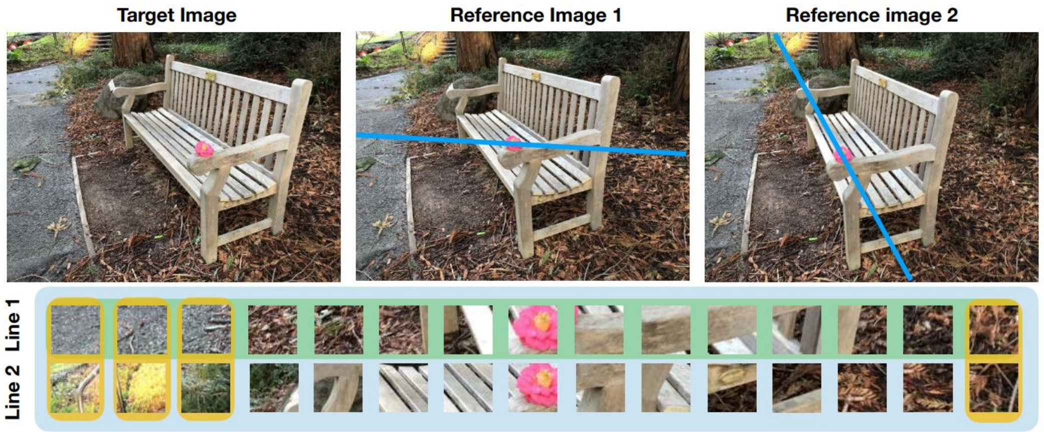
[3] Generalized Patch Based Neural Rendering

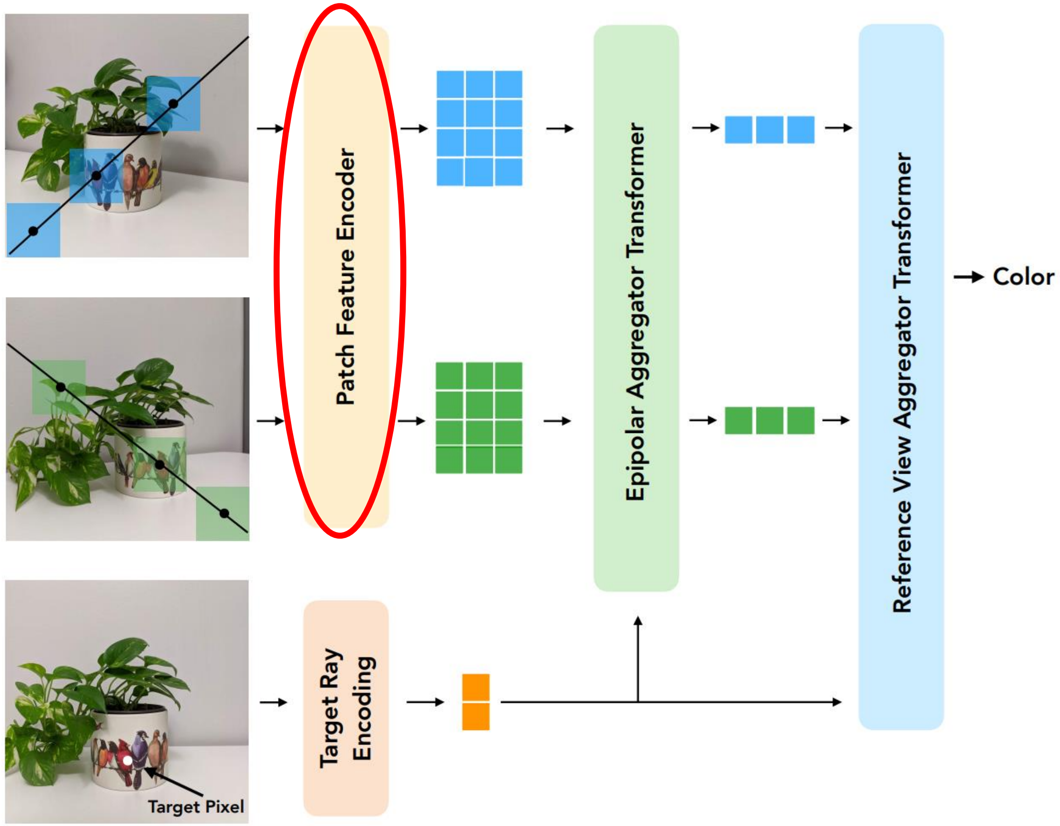
3. Three Transformers

1) Patch Feature Encoder

- Attention among different views

  (Yellow Direction of the below)





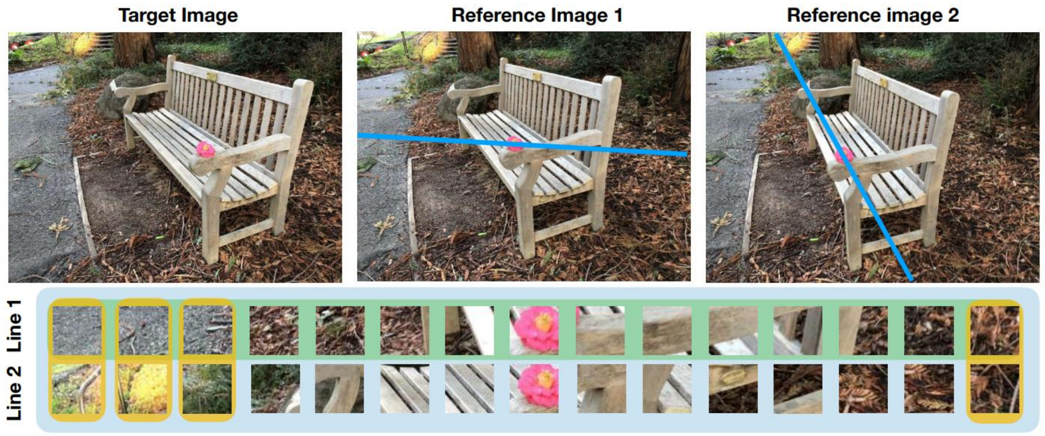
[3] Generalized Patch Based Neural Rendering

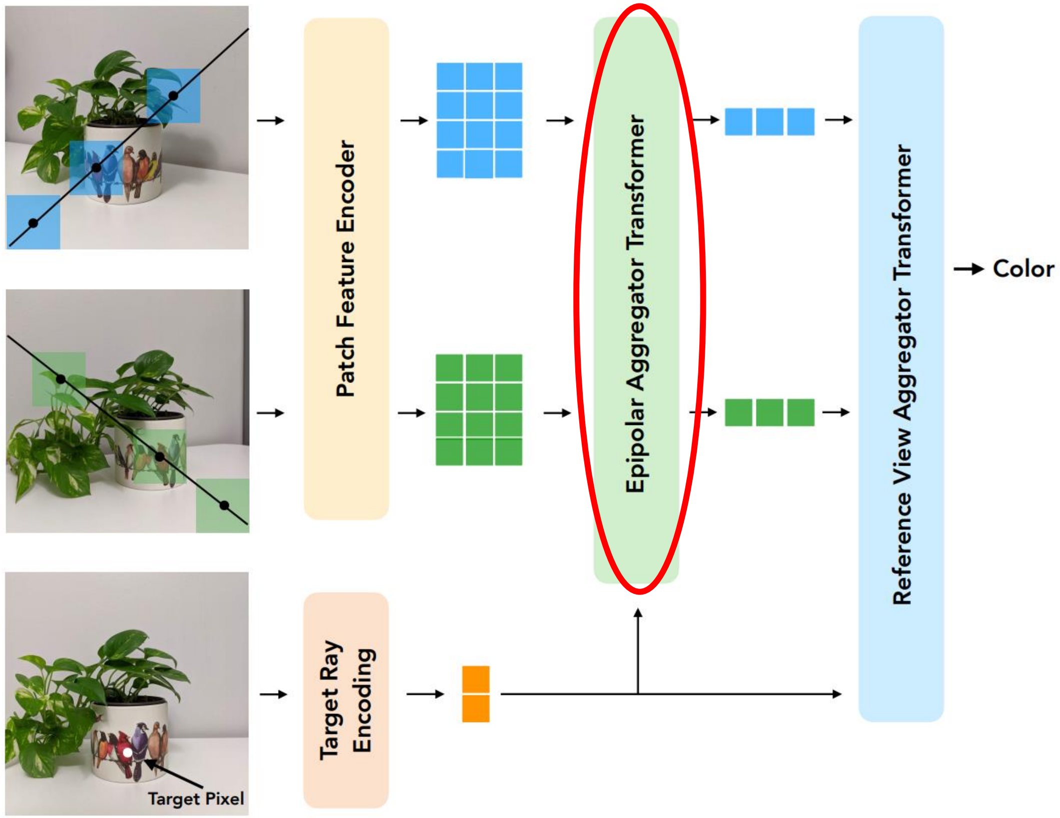
3. Three Transformers

2) Epipolar **Aggregator** Transformer

- Attention among epipolar line (among same view)
- **Aggregate** the patches (softmax w/ learnable α_k^m)

  (Green Direction of the below)



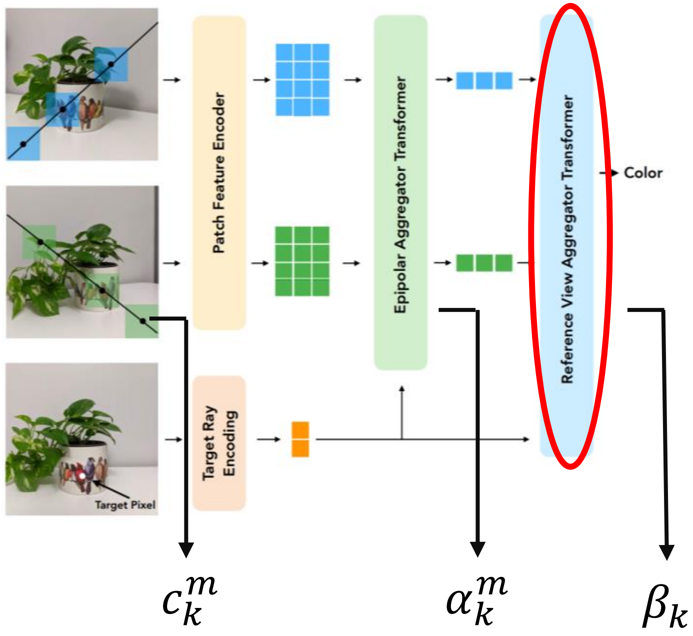


[3] Generalized Patch Based Neural Rendering

3. Three Transformers

3) Reference View **Aggregator** Transformer

- Attention among different views
- **Aggregate** the patches (softmax w/ learnable β_k)



Using Only Aggregation weights! (α_k^m, β_k)

Actual pixel values(c_k^m) from epipolar line are used for final inference!

(Patch Feature are not used directly)

Authors' Argument

Using the input pixel value from reference views

helps the generalization! (shown experimentally)

$$\mathbf{c} = \sum_{k=1}^{K} \beta_k \left(\sum_{m=1}^{M} \alpha_k^m \mathbf{c}_k^m \right)$$

[3] Generalized Patch Based Neural Rendering

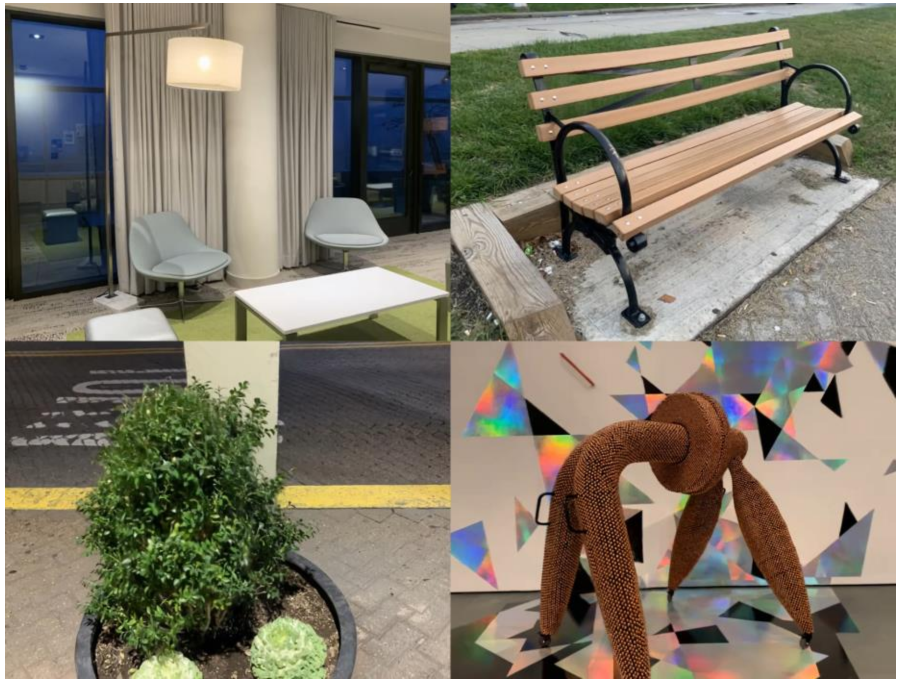
4. Experiment

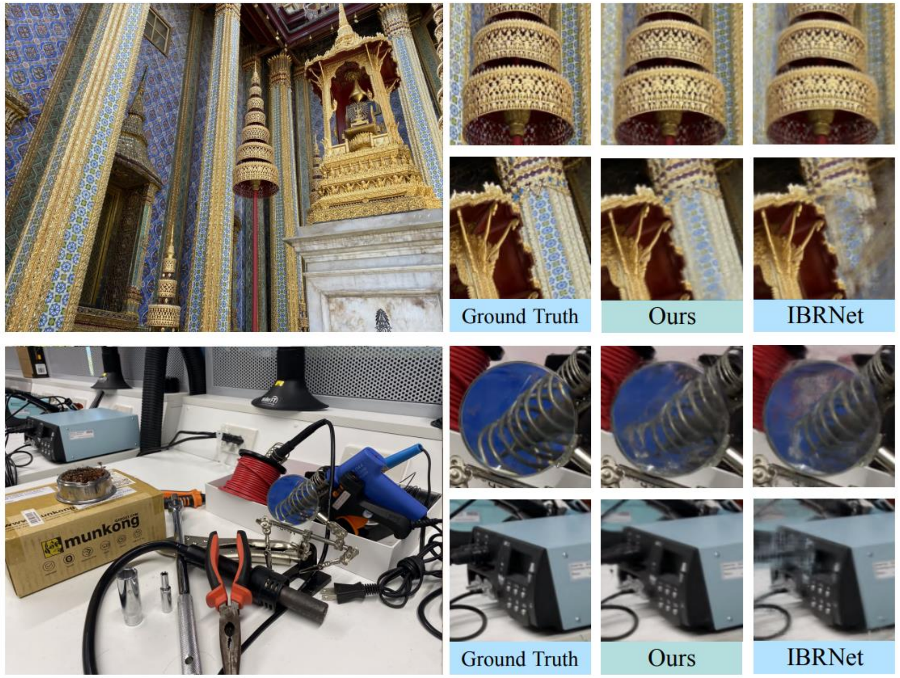
Model Setting

- ✓ Transformer: 8 Blocks , 256 feature dimension
- ✓ Reference View = 10

Train Setting

- ✓ Batch size = 4096 rays
- ✓ Optimizer: Adam, lr = 3*1e-4
- ✓ Training: iter = 250k , 32 TPUs, 24hrs





[3] Generalized Patch Based Neural Rendering

4. Experiment

<Experiment 1>

Baseline Setting: IBRNet

Train Dataset
- ✓ LLFF 37
- ✓ IBRNet 131

Eval Dataset
- ✓ Real Forward-facing
- ✓ Shiny
- ✓ Blender

<Experiment 2>

Baseline Setting: MVSNeRF

Train Dataset
- ✓ DTU 88

Eval Dataset
- ✓ DTU 16
- ✓ Blender

Method	Real Forward-Facing			Shiny-6			Blender		
	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS
LLFF [36]	24.13	0.798	0.212	-	-	-	24.88	0.911	0.114
IBRNet [69]	25.13	0.817	0.205	23.60	0.785	0.180	25.49	0.916	0.100
GeoNeRF [24]	25.44	0.839	0.180	-	-	-	28.33	0.938	0.087
IBRNet*	24.33	0.801	0.213	23.37	0.784	0.181	21.32	0.888	0.131
Ours	25.72	0.880	0.175	24.12	0.860	0.170	26.48	0.944	0.091

Table 1. Results for setting 1. Our model outperforms the baselines even when training with strictly less data. IBRNet uses three datasets that are not part of our training set, while GeoNeRF uses one extra dataset and also leverages input depth maps during training. IBRNet* was trained using the same training set as our method; in this fair comparison, our advantage in accuracy widens.

Method	DTU			Blender		
	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS
PixelNeRF [78]	19.31	0.789	0.671	7.39	0.658	0.411
IBRNet [69]	26.04	0.917	0.190	22.44	0.874	0.195
MVSNeRF [9]	26.63	0.931	0.168	23.62	0.897	0.176
Ours	28.50	0.932	0.167	24.10	0.933	0.097

Table 2. Results for setting 2. All models are trained on DTU and evaluated on either the DTU held-out set or Blender. Our approach outperforms the baselines.

[3] Generalized Patch Based Neural Rendering

4. Experiment

Ablation

Visual Transformer	Ray Canonicalization	Coordinate Canonicalization	PSNR	SSIM	LPIPS
✗	✗	✗	22.62	0.763	0.313
✓	✗	✗	25.42	0.879	0.154
✓	✓	✗	25.86	0.885	0.142
✓	✓	✓	26.42	0.896	0.129

Table 3. Ablations. Ablation study for model trained on LLFF and IBRNet scenes and tested on RFF with a resolution of 504 × 378. Results show that our main contributions – the visual feature transformer and the canonicalized positional encoding – lead to superior generalization performance.

Using the pixel value vs Using transformer feature

$$\mathfrak{c} = \mathtt{MLP}\left(\sum_{k=1}^{K} \beta_k f_3^k\right)$$

Interpolation Method	Real-Forward-Facing		
	PSNR	SSIM	LPIPS
Features	25.08	0.86	0.199
Colors (ours)	25.72	0.88	0.175

[4] Conclusion

Limitation

- Requires many adjacent reference views(10)

Follow-up Work

- IS ATTENTION ALL THAT NERF NEEDS? (ICLR 2023)
  (View Transformer, Ray Transformer)

Discussion

- Task Driven Thinking
- **Leveraging 3D Geometry Constraints**
  - ✓ Using Epipolar Constraints
  - ✓ Using the pixel value, Not ViT patch!

Reference

- Generalizable Patch-Based Neural Rendering (https://arxiv.org/abs/2207.10662)
- IBRNet: Learning Multi-View Image-Based Rendering (https://ibrnet.github.io/)
- GeoNeRF: Generalizing NeRF with Geometry Priors (https://arxiv.org/abs/2111.13539)
- MVSNeRF: Fast Generalizable Radiance Field Reconstruction from Multi-View Stereo (https://apchenstu.github.io/mvsnerf/)
- pixelNeRF: Neural Radiance Fields from One or Few Images (https://alexyu.net/pixelnerf/)

Appendix

$$f_1^m = T_1\left(\left\{f_0^{k,m} \mid 1 \leq k \leq K\right\}\right)$$

$$f_2^k = T_2\left(\left\{r^0\right\} \bigcup \left\{\left[f_1^{k,m} \,\middle\|\, r_k^m \,\middle\|\, d^m \,\middle\|\, c_k\right] \,\middle|\, 1 \leq m \leq M\right\}\right)$$

$$\alpha_k^m = \frac{\exp\left(W_1\left[f_2^{k,0} \,\middle\|\, f_2^{k,m}\right]\right)}{\sum_{m'=1}^{M} \exp\left(W_1\left[f_2^{k,0} \,\middle\|\, f_2^{k,m'}\right]\right)},$$

$$f_{2'}^k = \sum_{m=1}^{M} \alpha_k^m f_2^{k,m},$$

$$f_3 = T_3\left(\left\{r^0\right\} \bigcup \left\{\left[f_{2'}^k \,\middle\|\, c_k\right] \,\middle|\, 1 \leq k \leq K\right\}\right)$$

$$\beta_k = \frac{\exp\left(W_2\left[f_3^0 \,\middle\|\, f_3^k\right]\right)}{\sum_{k'=1}^{K} \exp\left(W_2\left[f_3^0 \,\middle\|\, f_3^k\right]\right)},$$

$$\mathfrak{c} = \sum_{k=1}^{K} \beta_k \left(\sum_{m=1}^{M} \alpha_k^m \mathfrak{c}_k^m\right)$$