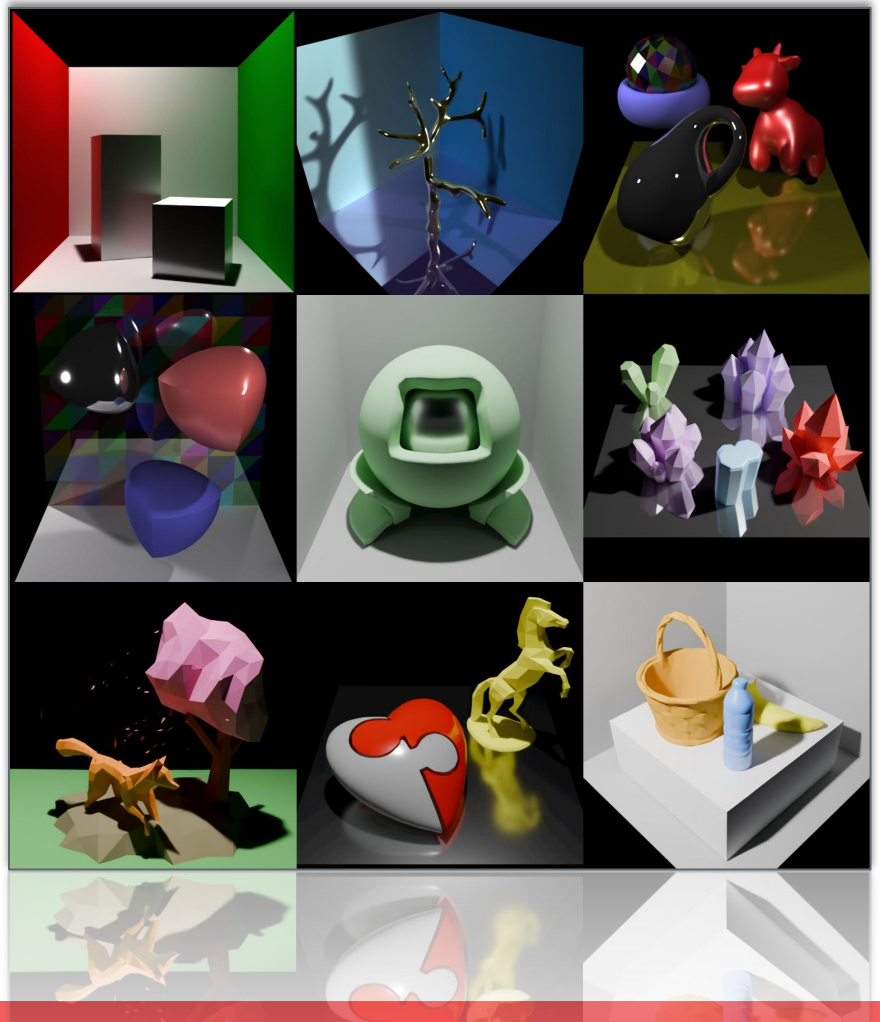
RenderFormer

Lecture Presentation

27th Oct, 2025

Team 4
Kyaw Ye Thu, Janghyun



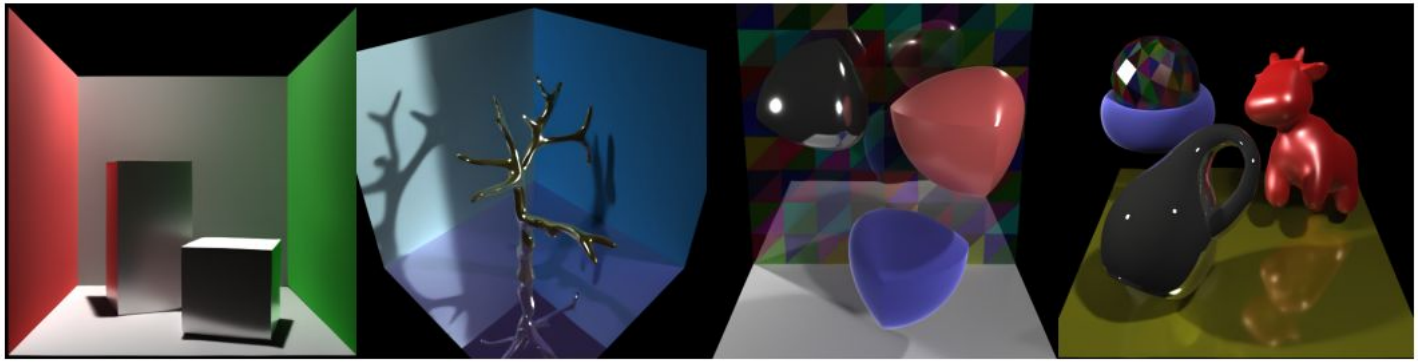
CS580: Computer Graphics

What is RenderFormer?

SIGGRAPH 2025

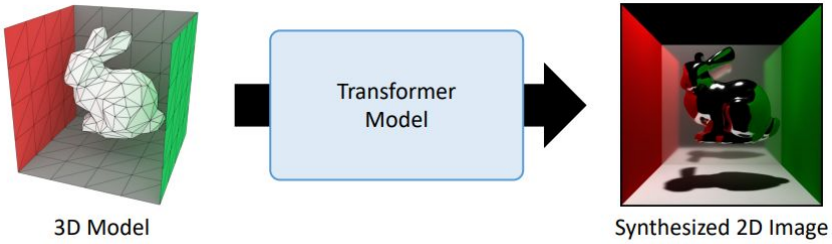
RenderFormer: Transformer-based Neural Rendering of Triangle Meshes with Global Illumination

CHONG ZENG, State Key Lab of CAD & CG, Zhejiang University, China and Microsoft Research Asia, China
YUE DONG, Microsoft Research Asia, China
PIETER PEERS, College of William & Mary, USA
HONGZHI WU, State Key Lab of CAD & CG, Zhejiang University, China
XIN TONG, Microsoft Research Asia, China



https://microsoft.github.io/renderformer/

What is RenderFormer?



Explicitly Solving the Rendering Equation

$$L_r(x \rightarrow \Theta) = \int_{\Psi} L(x \leftarrow \Psi) f_r(x, \Psi \rightarrow \Theta) \cos\theta_x dw_{\Psi}$$

Monte Carlo Estimation of Path Integral

- Path Tracing
- Metropolis Light Transport
- Path Guiding
- Multiple Importance Sampling
- ...

Radiosity Techniques (Finite Elements Methods)

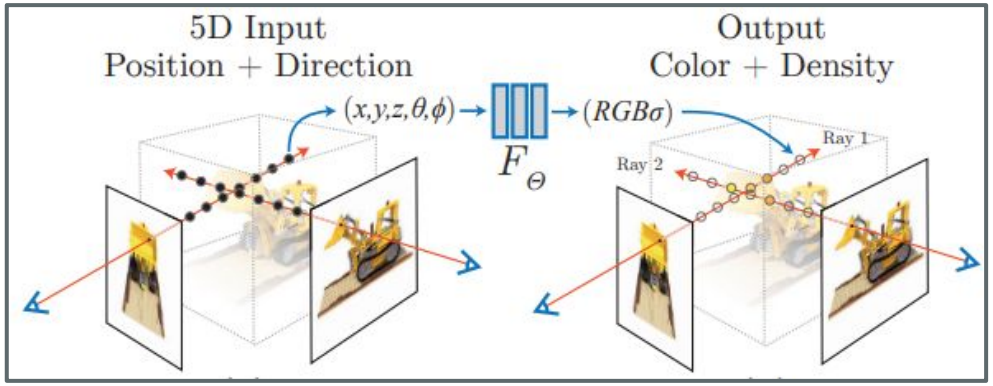
- Polynomial Basis Functions
- Wavelets
- Meshless Basis Functions
- Neural Radiosity
- ...

Slide Source: https://saeedhd96.github.io/neural-radiosity/

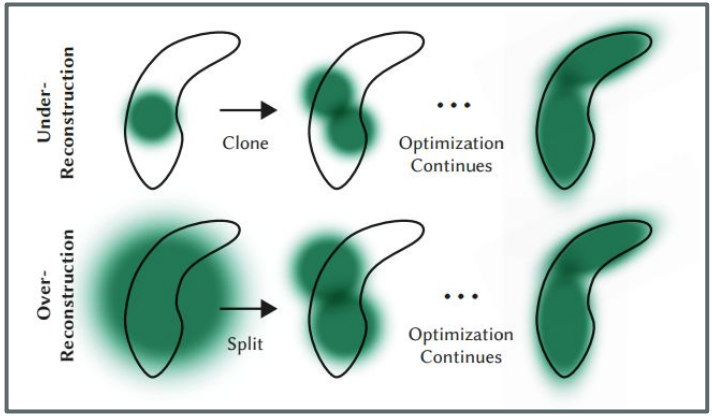
Implicit Representation of Light Transport

$$L_r(x \to \Theta) = \int_{\Psi} \text{not exposed} \to \Theta) \cos\theta_x dw_\Psi$$

NeRF



3DGS



Slide Source: https://saeedhd96.github.io/neural-radiosity/

NeRF

Limitations of NeRF

1. **Slow training & inference speed**

   Do per-scene optimization which requires tens of minutes to hours of training, leads to high latency.
2. **Poor generalization / Requires per-scene retraining**

   hard to infer in disoccluded or unseen regions when there are limited viewpoints
3. **Sensitive to calibration & exposure**

   Performance down due to internal/external camera parameter errors and exposure differences

Research directions for overcoming

1. **Slow training & inference speed**

   Replace pure MLPs with fast encodings/structures.

   Ex) Instant-NGP (multi-resolution hash encoding)

2. **Poor generalization / Requires per-scene retraining**

   Condition in NeRF on 2D features projected to 3D and pretrain across scenes.

   Ex) **PixelNeRF** (few/zero-shot), IBRNet

3. **Sensitive to calibration & exposure**

   Co-optimize poses and fields, absorb exposure/style shifts via latent codes.

   Ex) BARF (bundle-adjusting NeRF), NeRF-W (appearance embeddings)

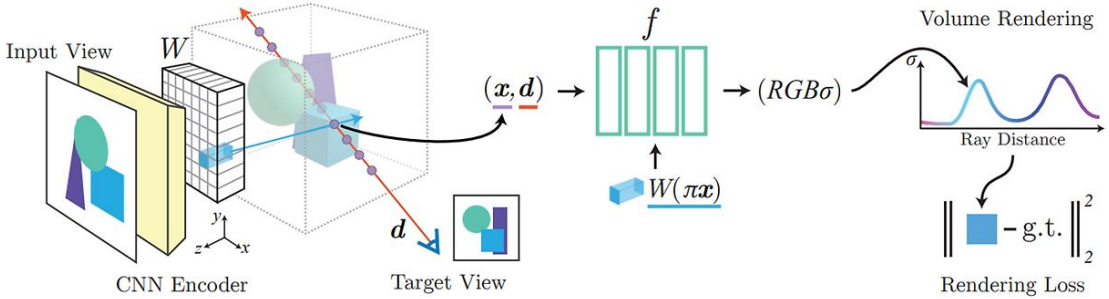
PixelNeRF

What is PixelNeRF?

Goal: synthesize novel views of a new scene with **only 1-3 input images(few-/zero-shot)**.

Early NeRF research required 50 to 100 dense input images to represent a single scene(per-scene optimization).

Instead of only using 3D coordinates (x, y, z) as input, PixelNeRF conditions the network on 2D image features extracted from the input views.

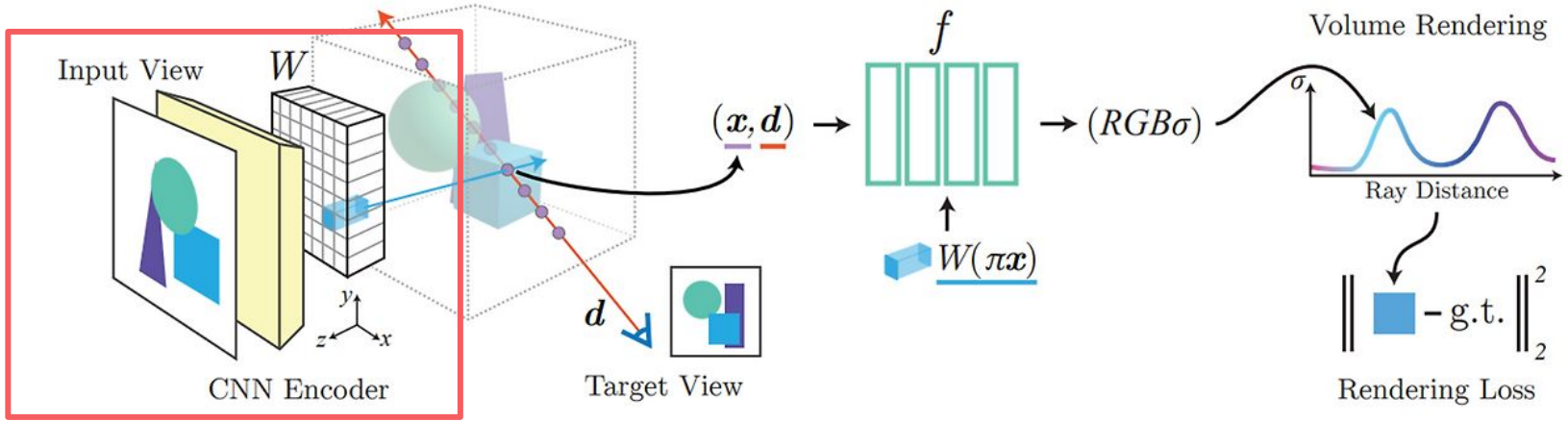


How it works? By Pixel-aligned Conditioning

1. Encode Images

   Use a CNN to extract a multi-scale feature map (W) from each 2D input image.

   This map (W) now holds rich visual information for every pixel.

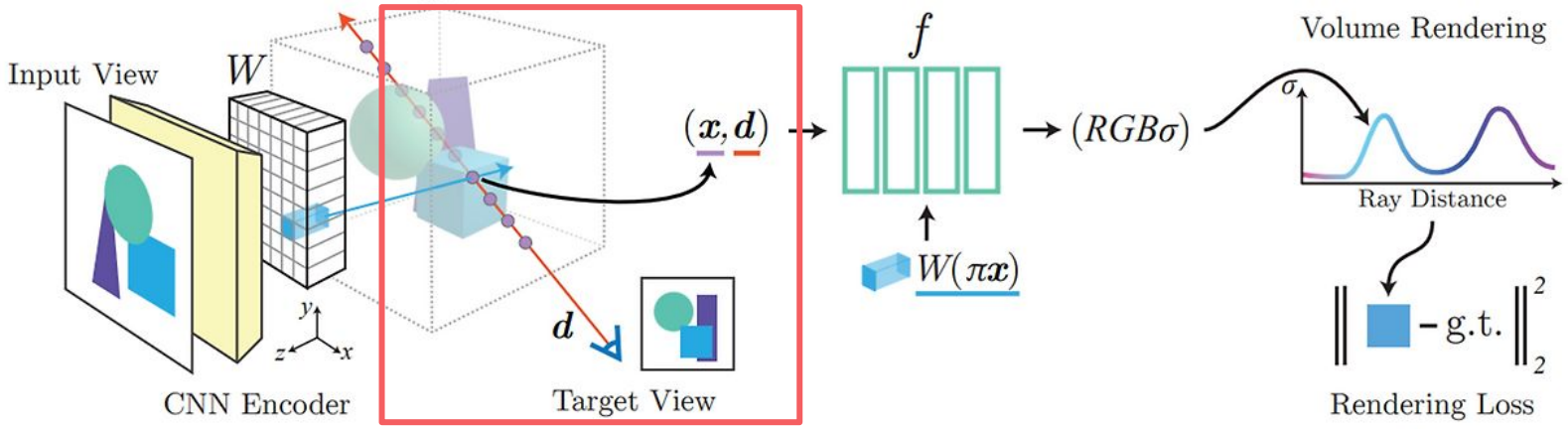


How it works? By Pixel-aligned Conditioning

2. Project & Sample Features

For a 3D sample point (x) along a ray, project it onto each 2D image plane using camera poses.

Look up and sample the corresponding pixel-aligned feature W(πx) from each feature map

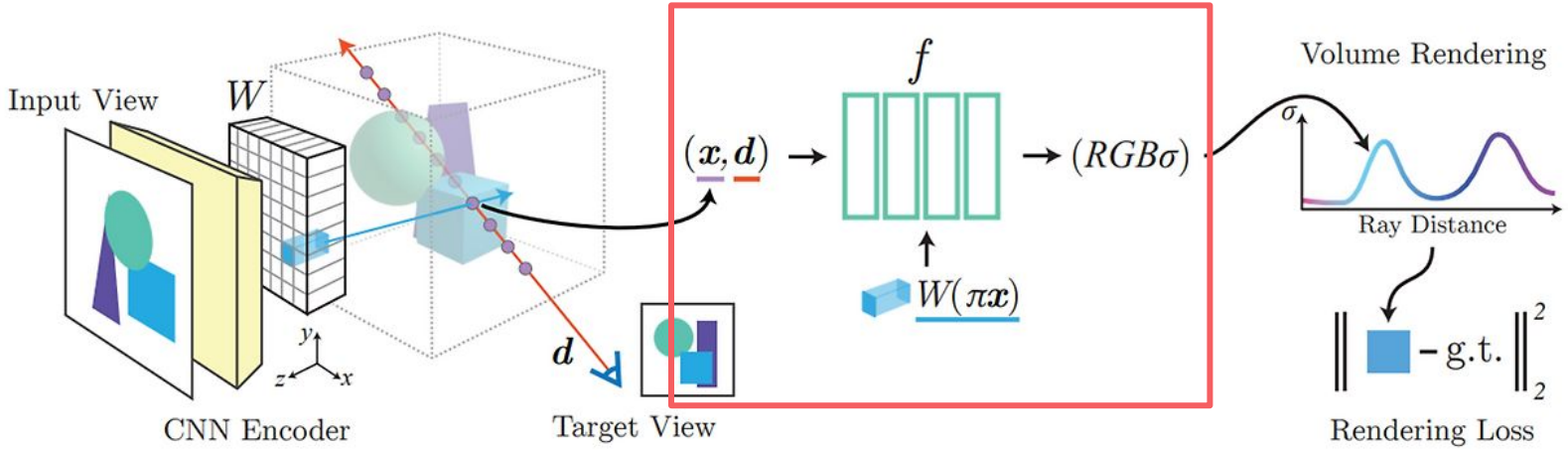


How it works? By Pixel-aligned Conditioning

3. Aggregate & Predict

Combine (e.g., average) the sampled features from all views into a single feature vector.

Feed the aggregated feature, positional encoding (of x), and view direction (d) into an MLP to get density (σ) and color (RGB).

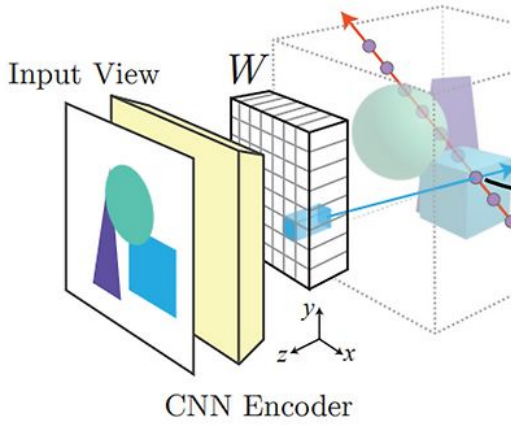


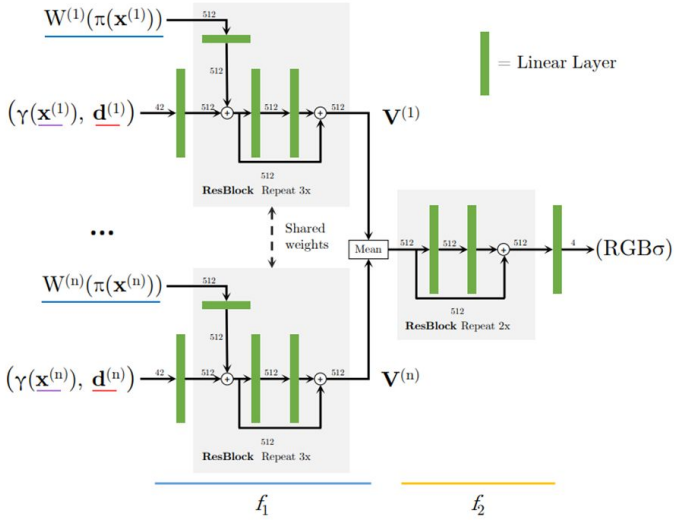
How it works? By Pixel-aligned Conditioning

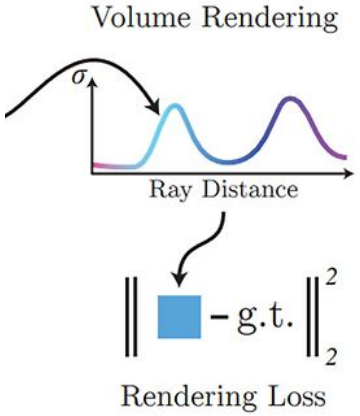
3. Aggregate & Predict

Combine (e.g., average) the sampled features from all views into a single feature vector.

Feed the aggregated feature, positional encoding (of x), and view direction (d) into an MLP to get density (σ) and color (RGB).





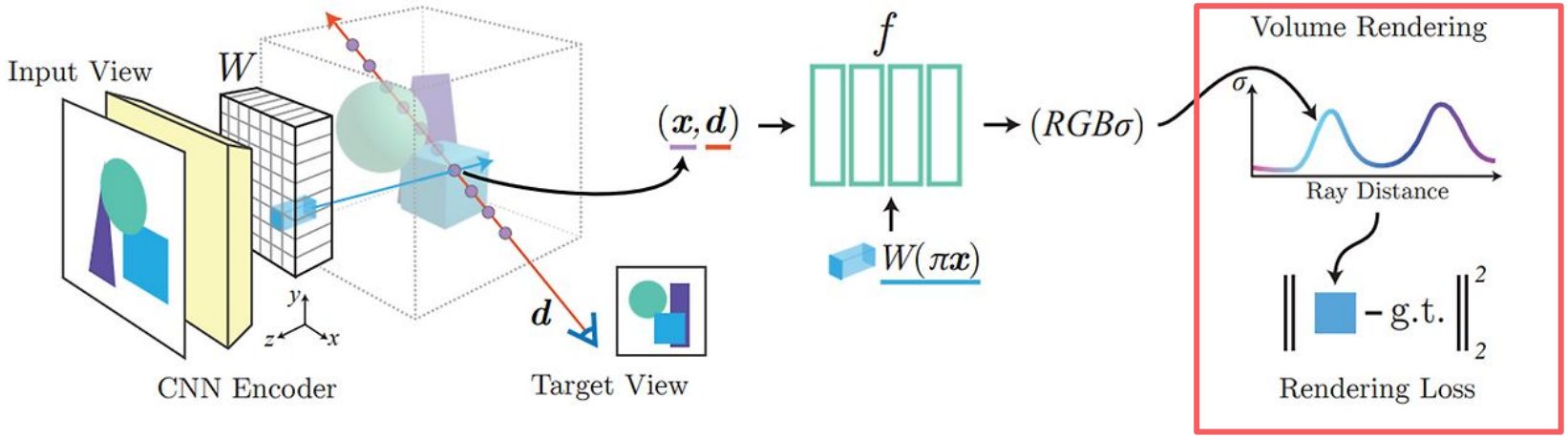


How it works? By Pixel-aligned Conditioning

4. Render Pixel

Accumulate these (σ, c) values along the ray using standard NeRF volumetric rendering to compute the final pixel color.

Loss & learning: minimize photometric MSE over target rays, backprop through the rendering & projections.



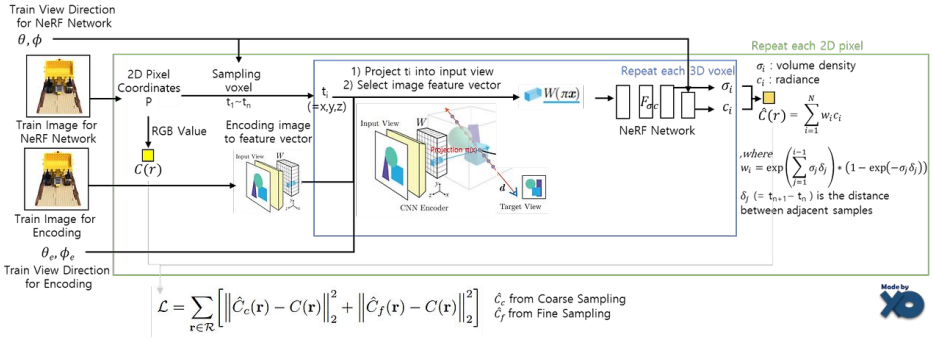
Training

Training Loop: Randomly sample one scene, 1-3 source views, and one target view per step.

End-to-End Training: Rendering loss from the target view back-propagates to train both the NeRF Network (F) and CNN Encoder (W) simultaneously.

Sampling Strategy (Coarse-to-Fine):

- First 2/3 steps: Sample points within the object's bounding box.
- Final 1/3 steps: Sample points across the entire image to refine the background and reduce artifacts.

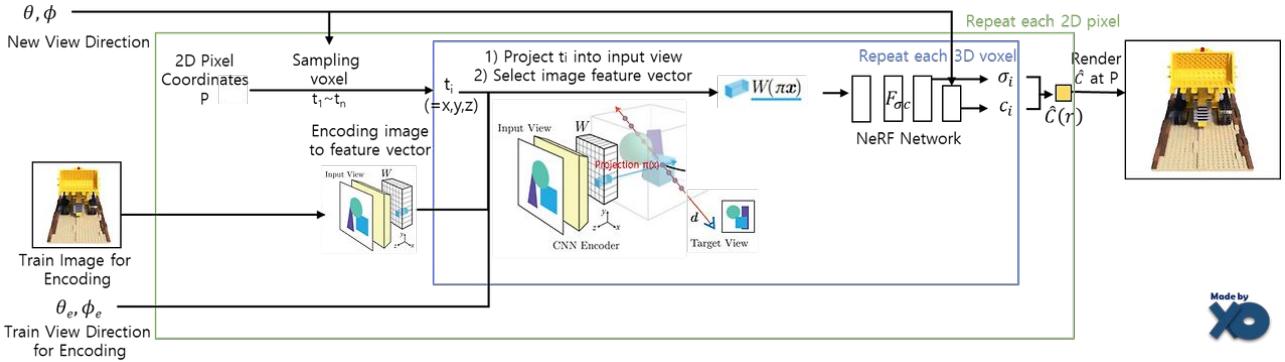


Testing

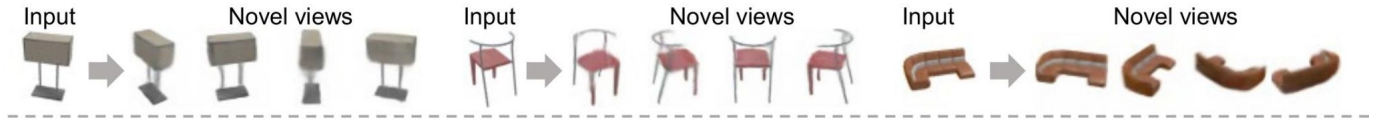
Given: 1-3 source images from a new (test) scene.

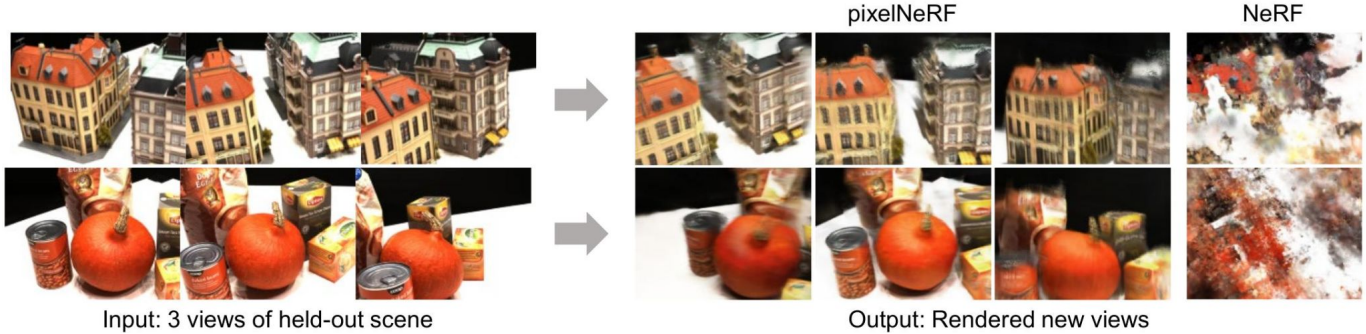
Generate feature maps for the source images using the pre-trained CNN Encoder (W).

Synthesize novel views using the pre-trained NeRF Network (F), which takes those features as input.



Result



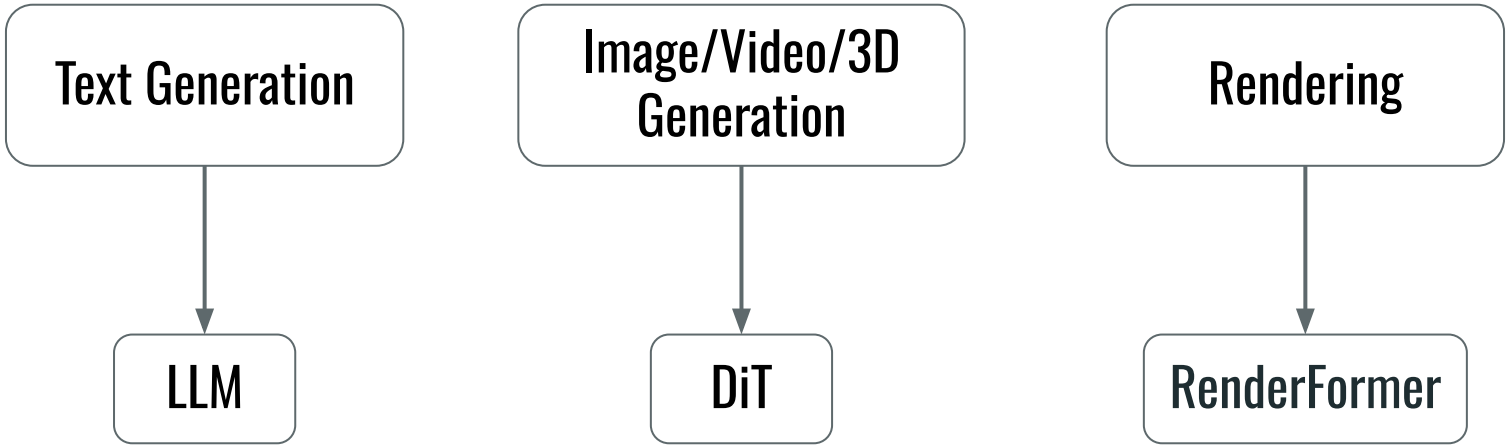


Input: 3 views of held-out scene

Output: Rendered new views

Attention

Is Attention All You Need for Rendering?



Slide Source: https://renderformer.github.io/pdfs/renderformer-slides.pdf

Self-Attention in Text Generation

$$\text{Attention}(Q, K, V) = \text{softmax}(\frac{QK^T}{\sqrt{d_k}})V$$

"A fluffy blue creature roamed the verdant forest."

Image Source: 3blue1brown

Self-Attention in Text Generation

$$\text{Attention}(Q, K, V) = \text{softmax}(\frac{QK^T}{\sqrt{d_k}})V$$

"A fluffy blue creature roamed the verdant forest."

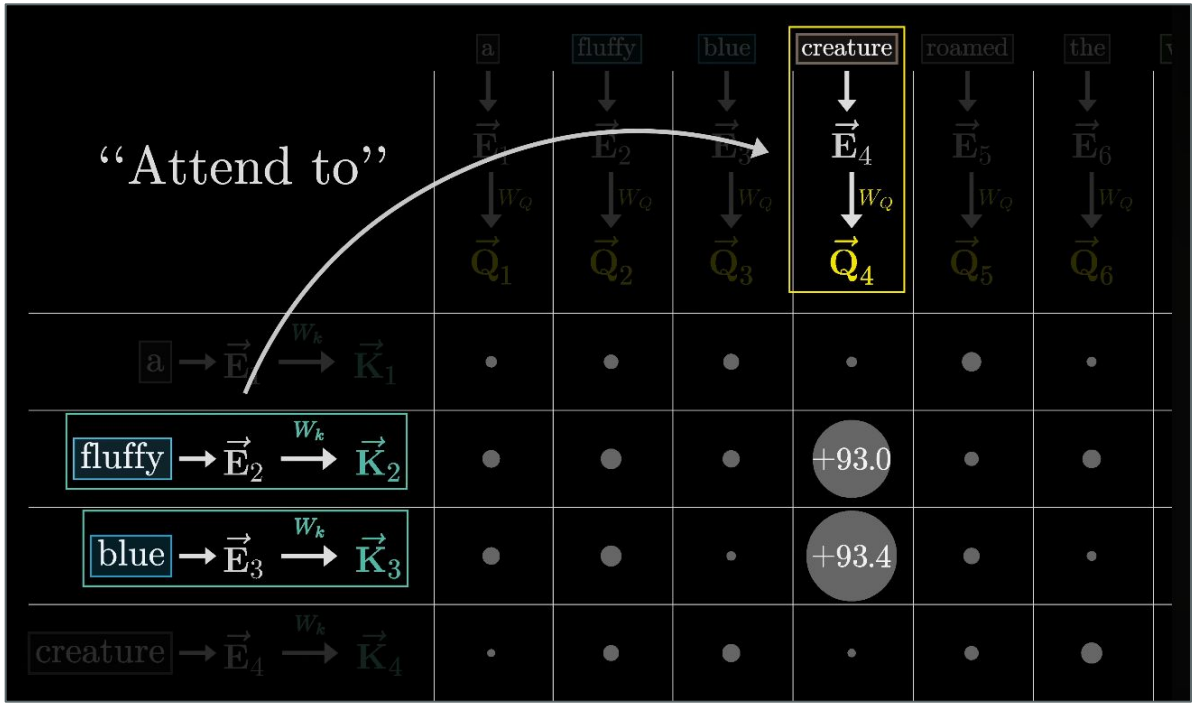
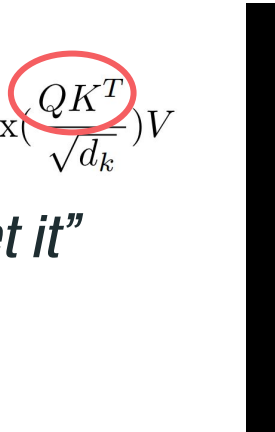


Image Source: 3blue1brown

Cross-Attention in Text Generation

$$\text{Attention}(Q, K, V) = \text{softmax}(\frac{QK^T}{\sqrt{d_k}})V$$

"I do not want to pet it"



"Je ne veux pas le caresser"

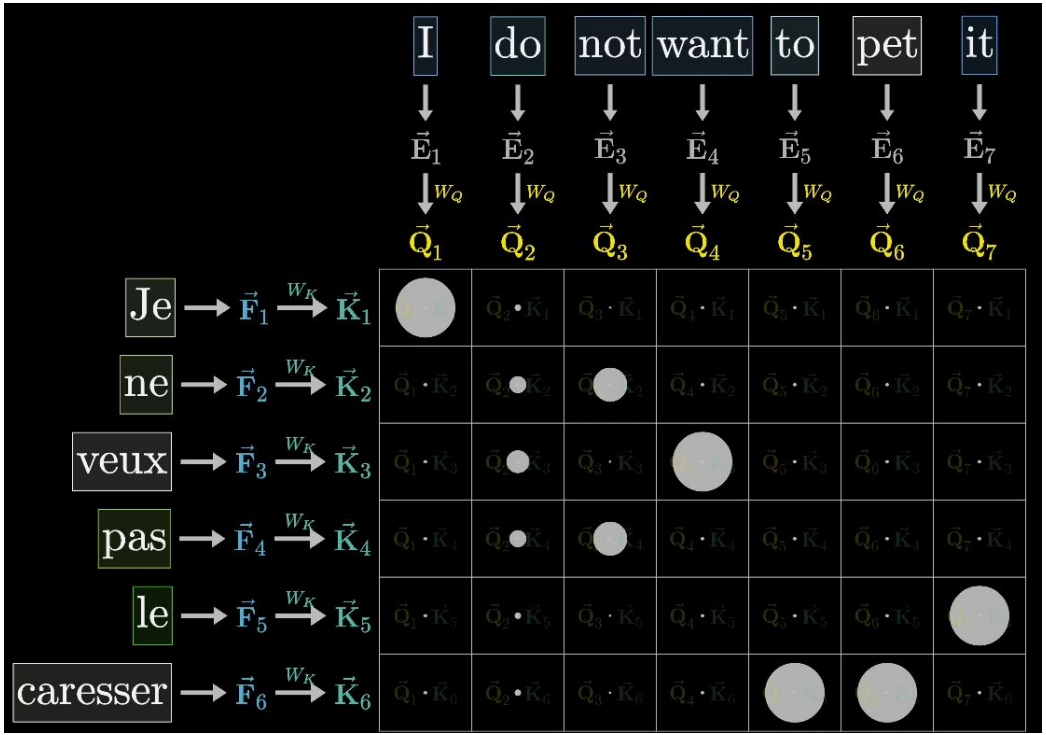


Image Source: 3blue1brown

Multi-headed Attention

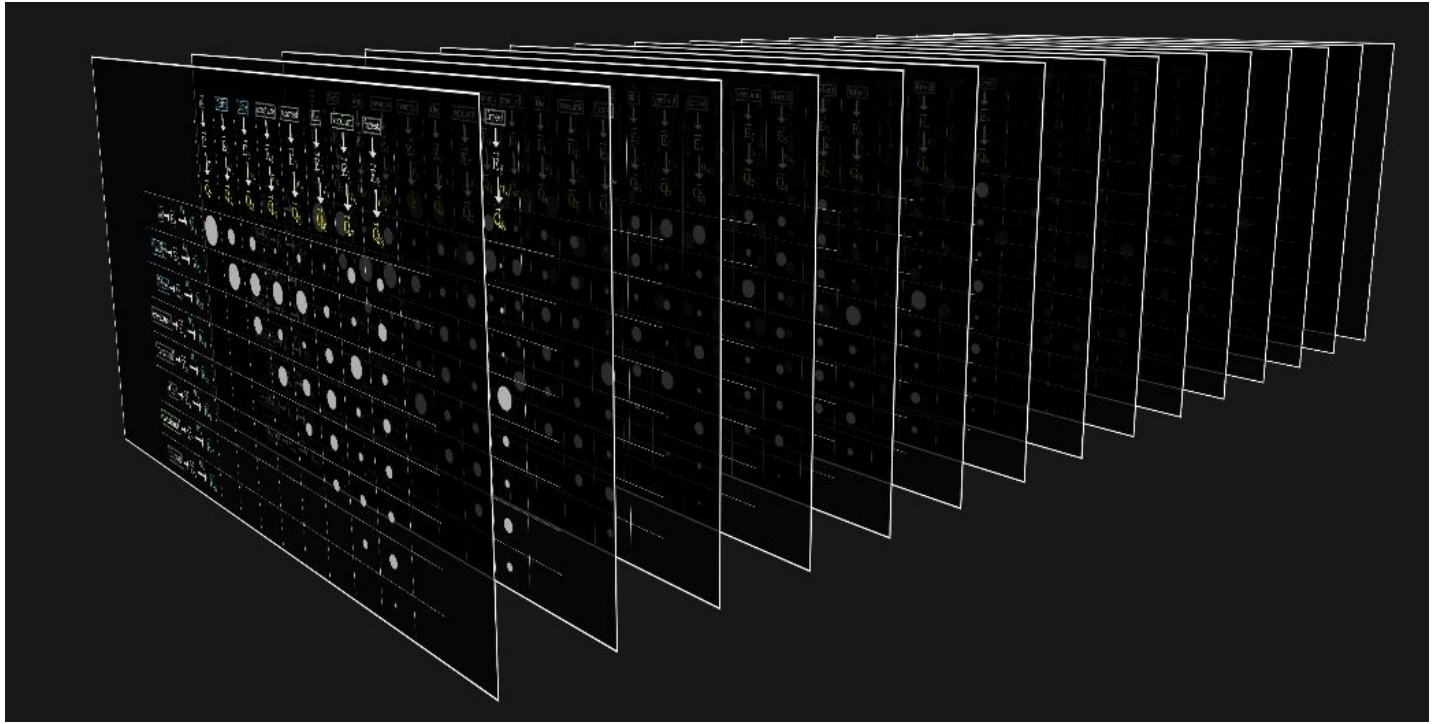
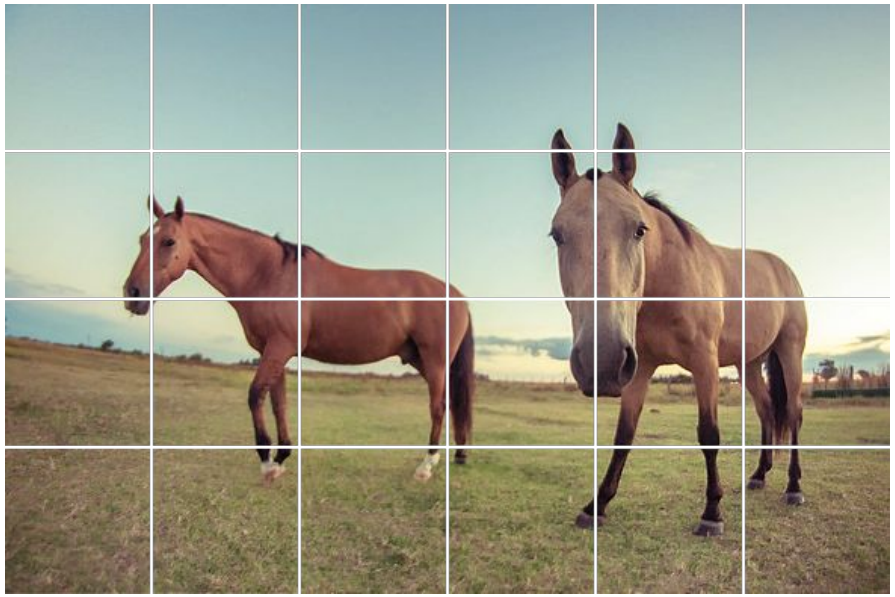


Image Source: 3blue1brown

Self-Attention for Vision Transformer



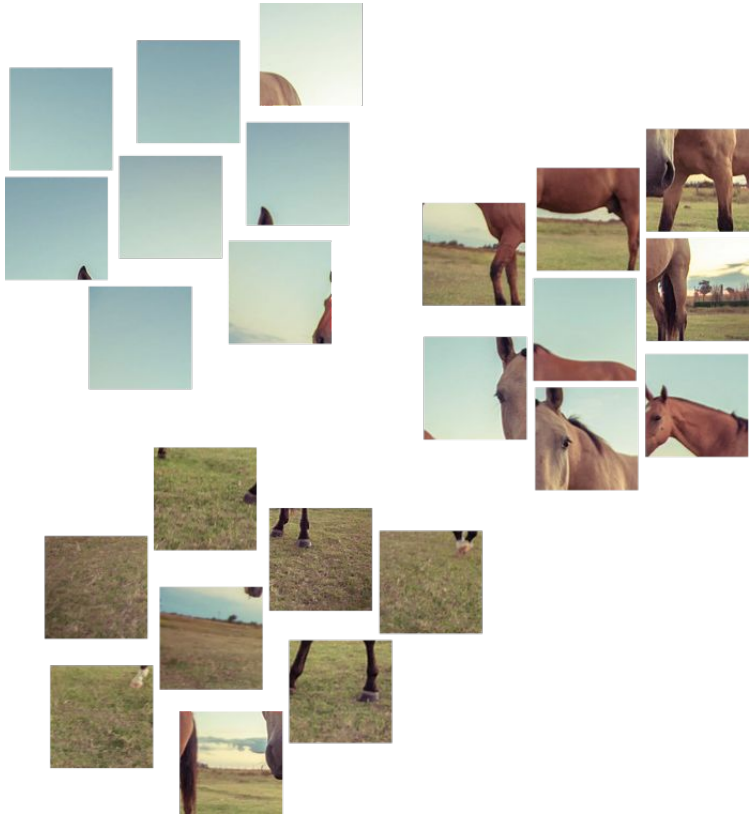


Image Source: Thalles' blog

Self-Attention for Vision Transformer

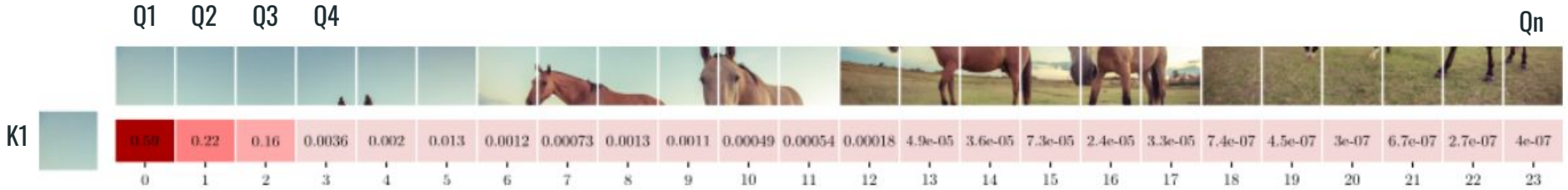


Fig. The first row of the normalized scores tensor shows the similarities between the first patch and all the others.

Image Source: Thalles' blog

What about Attention for Rendering?

3D Inductive Bias

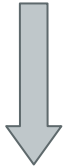
Related Work

LVSM: A Large View Synthesis Model with Minimal 3D Inductive Bias

ICLR 2025

Haian Jin, Hanwen Jiang, et. al

Sparse input views with camera poses



High-quality novel view synthesis results

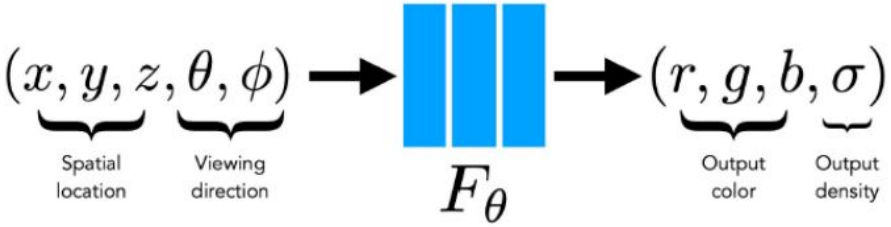
3D Inductive Bias

Built-in assumptions, structures, or priors about the 3D world that are hard-coded into its model or pipeline

3D Inductive Bias (Representation Level)

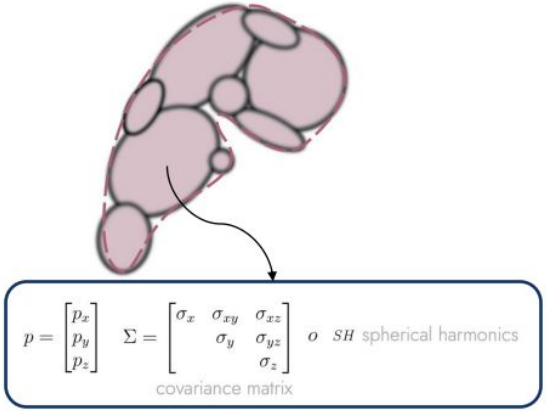
NeRF

Continuous Volumetric Field: Assumes the scene can be represented as a continuous 5D function.



3DGS

Gaussian Primitives: Scenes are composed of fuzzy blobs (Gaussians) that can be combined

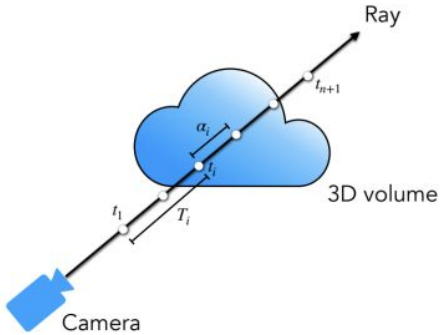


3D Inductive Bias (Rendering Level)

NeRF

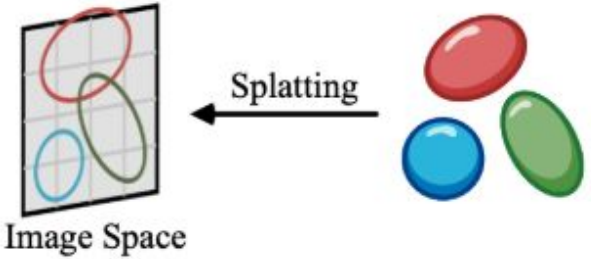
Ray marching with volumetric rendering:
Assumes light travels in straight lines, accumulates multiplicatively, and objects occlude each other in a specific mathematical way

$$C(\mathbf{r}) = \int_{t_n}^{t_f} T(t)\sigma(\mathbf{r}(t))\mathbf{c}(\mathbf{r}(t), \mathbf{d})dt$$



3DGS

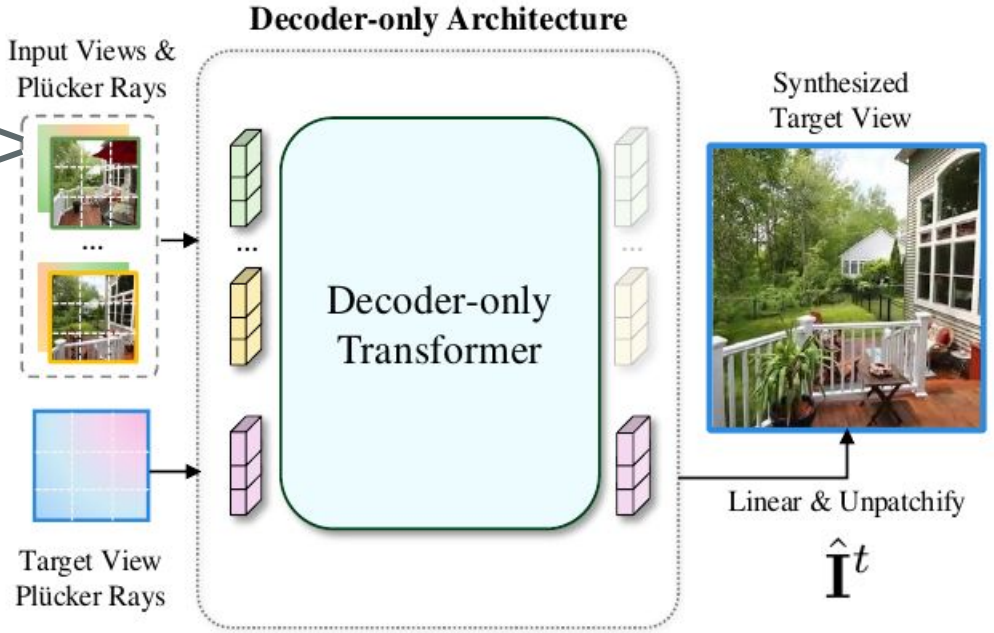
Splatting: Assumes that Rendering is a weighted sum of projected Gaussians



Related Work (LVSM)

1 input Image
...
...
Nth input image

{ Patchified Image Tokens
{ Plücker Ray Embedding

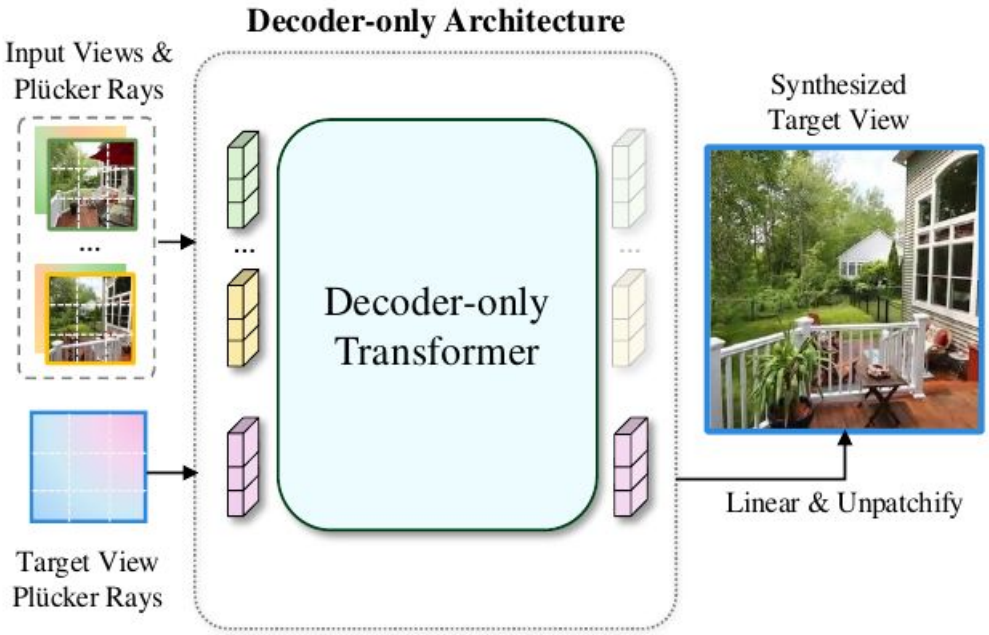


$$\mathcal{L} = \text{MSE}(\hat{\mathbf{I}}^t, \mathbf{I}^t) + \lambda \cdot \text{Perceptual}(\hat{\mathbf{I}}^t, \mathbf{I}^t)$$

Related Work (LVSM)

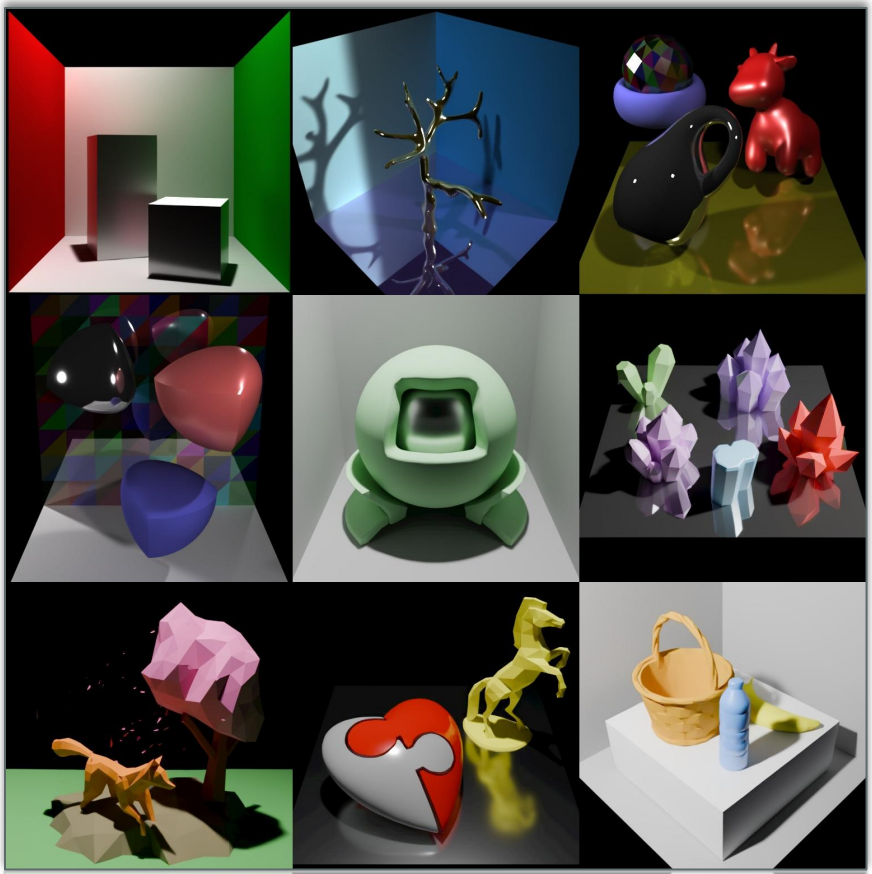
Fully Data-Driven

Target view token → *"attends to"* → Patch of an input view

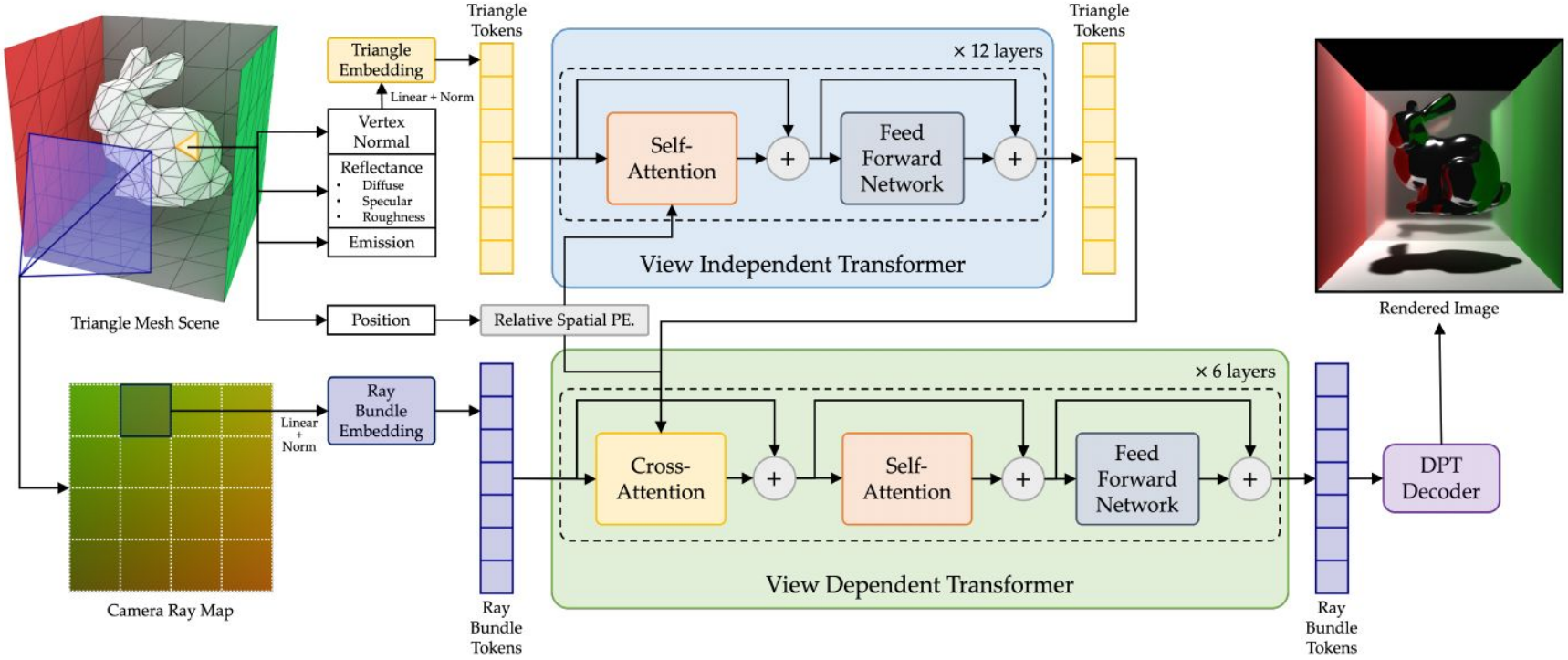


Summary

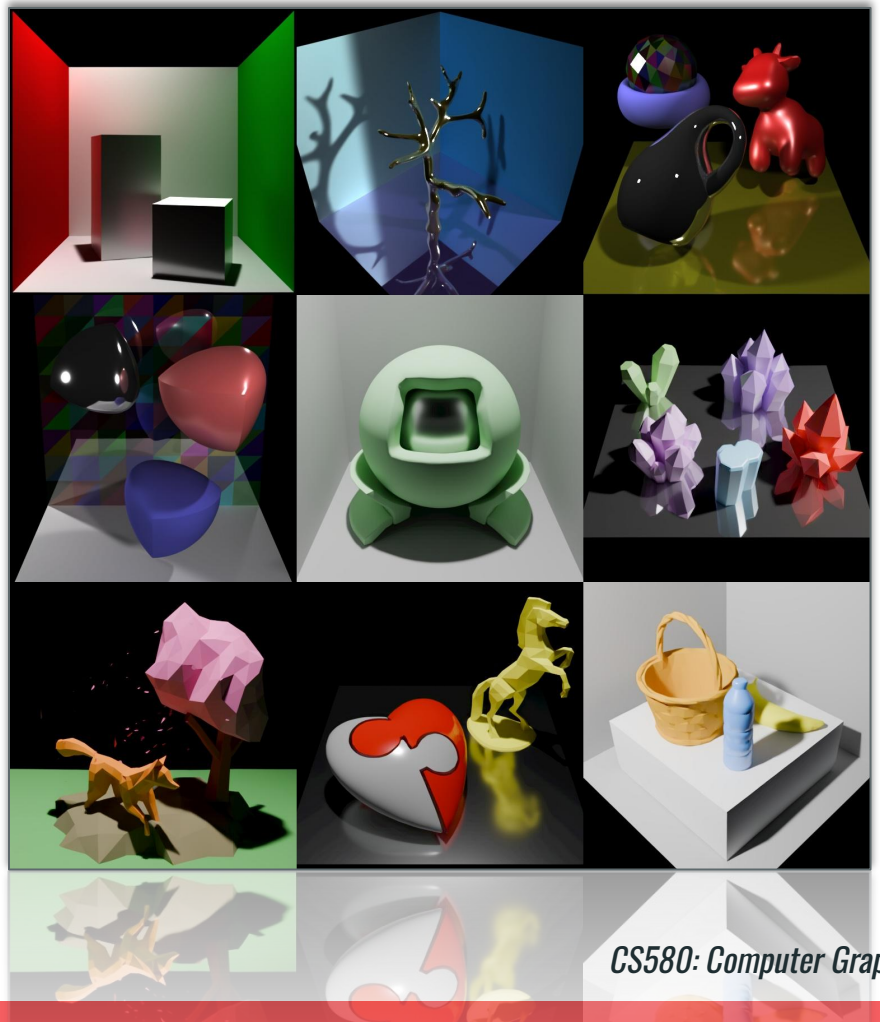
- PixelNeRF
- Instant-NGP
- Attention in Rendering
- Transformers' minimal 3D Inductive Bias



What's next?


Triangle Mesh Scene
Triangle Embedding
Linear + Norm
Vertex Normal
Reflectance
Diffuse
Specular
Roughness
Emission
Position
Triangle Tokens
× 12 layers
Self-Attention
Feed Forward Network
View Independent Transformer
Triangle Tokens
Rendered Image
Relative Spatial PE.
Camera Ray Map
Linear + Norm
Ray Bundle Embedding
× 6 layers
Cross-Attention
Self-Attention
Feed Forward Network
View Dependent Transformer
DPT Decoder
Ray Bundle Tokens
Ray Bundle Tokens

Q&A Time



Quiz Time