Deep Learning for Computer Vision

Fall 2023

https://cool.ntu.edu.tw/courses/31857(NTU COOL)
http://vllab.ee.ntu.edu.tw/dlcv.html (Public website)

Yu-Chiang Frank Wang 王鈺強, Professor

Dept. Electrical Engineering, National Taiwan University

2023/11/28

What to Cover Today?

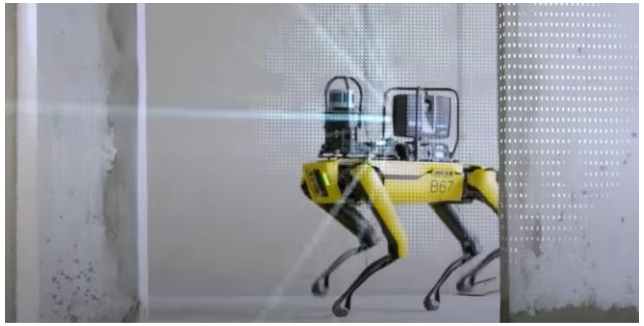
- Introduction to 3D Vision
- Part I: 3D Perception
- Part II: 3D Reconstruction
- Neural Radiance Fields
  - Extensions of NeRF
  - Advanced Topics of NeRF

What is 3D Vision?

- Enable machine to perceive and reconstruct the 3D world which we live in.

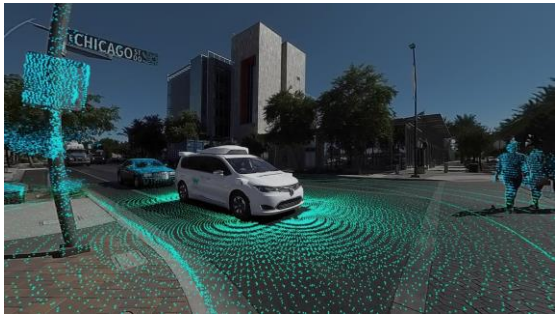
Applications of 3D Vision

- Robotics



- Augmented Reality

- Autonomous driving



References:
Boston Dynamics: https://www.youtube.com/watch?v=gvzljfK-PiU&ab_channel=BostonDynamics
Ikea: https://www.youtube.com/watch?v=UudV1VdFtuQ&ab_channel=IKEA
Waymo: https://www.youtube.com/watch?v=IzZcqCfA8_k&ab_channel=Waymo

How to Represent the 3D World?

- Recap: 2D representations
  - RGB pixels
  - Images/videos
  - Why 2D vision not good enough?
    - Lack of depth, scene geometry, etc. information

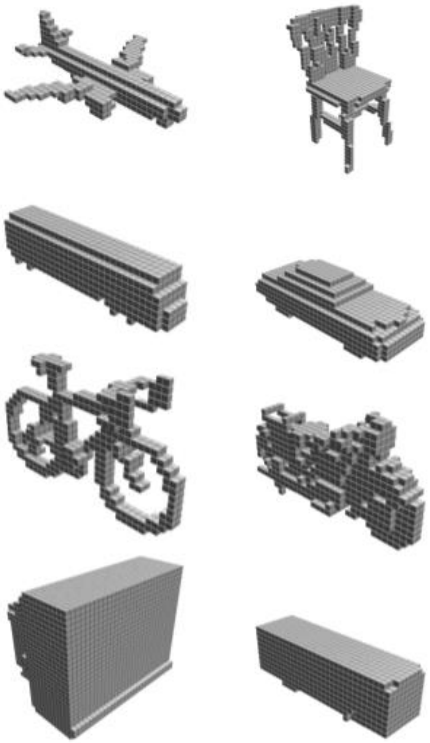
- What about 3D representations?

How to Represent the 3D World? (cont'd)

- Multi-view RGB-D images

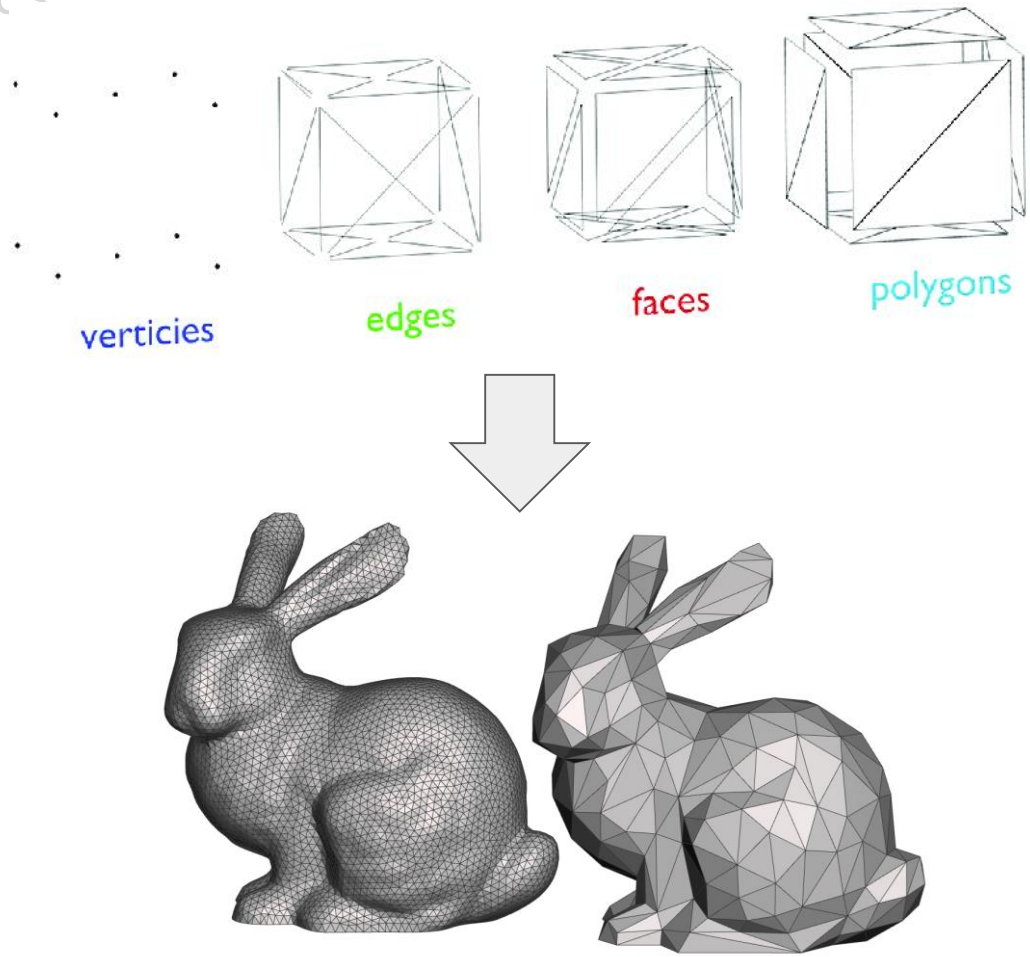
How to Represent the 3D World? (cont'd)

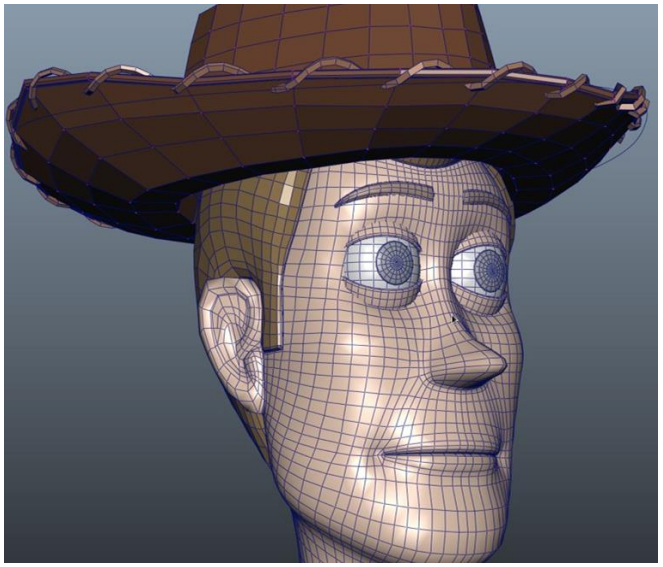
- Multi-view RGB-D images
- Voxels



How to Represent the 3D World? (cont'd)

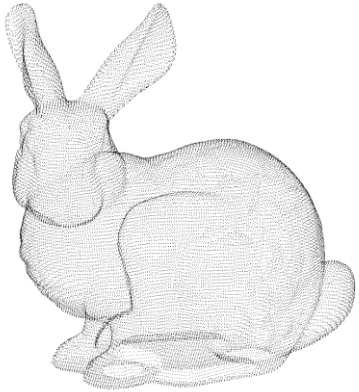
- Multi-view RGB-D images
- Voxels
- Polygon Mesh

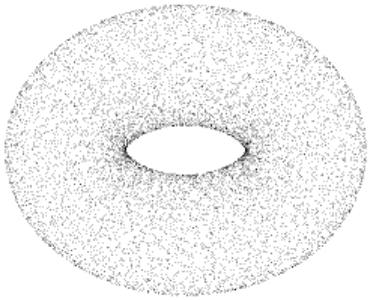




How to Represent the 3D World? (cont'd)

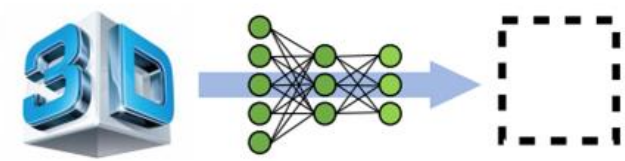
- Multi-view RGB-D images
- Voxels
- Polygon Mesh
- Point Cloud
- …



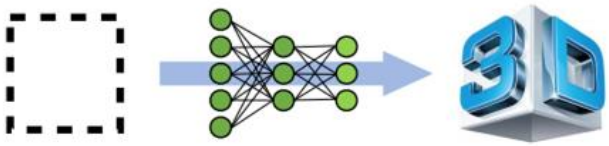


Deep Learning for 3D Vision

- **Perception**: extract information from 3D shapes (Part 1)



- **Reconstruction**: synthesis 3D shapes (Part 2)



What to Cover Today?

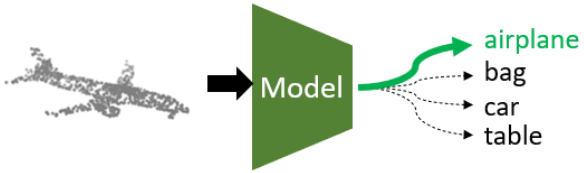
- Introduction to 3D Vision
- Part I: 3D Perception
- Part II: 3D Reconstruction
- Neural Radiance Fields
  - Extension of NeRF
  - Advanced Topics of NeRF

3D Perception

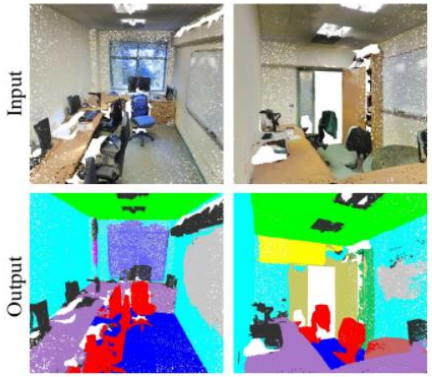
- Extract information from 3D shapes for downstream tasks
  - Classification
  - Object/scene segmentation
  - Pose estimation
  - Object detection
  - Grounding

"bottle on top of the bathroom vanity"

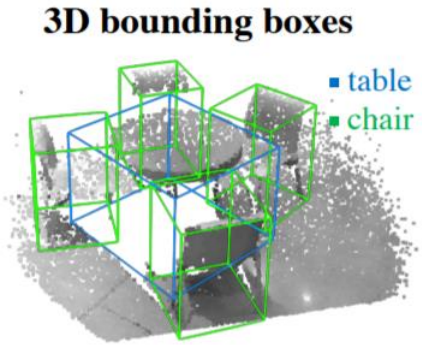
Grounding



Classification



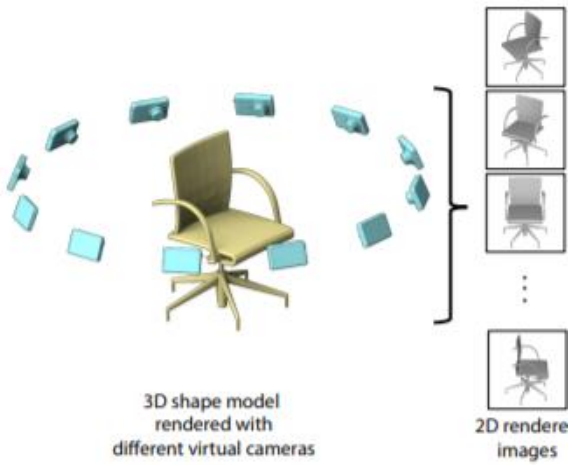
Segmentation

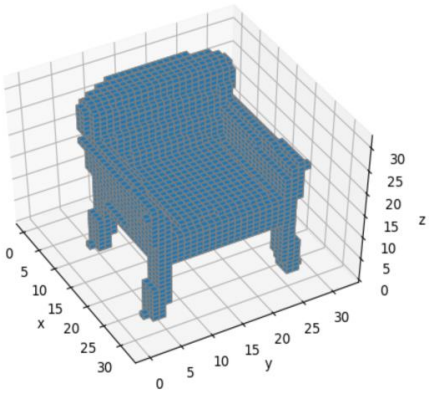


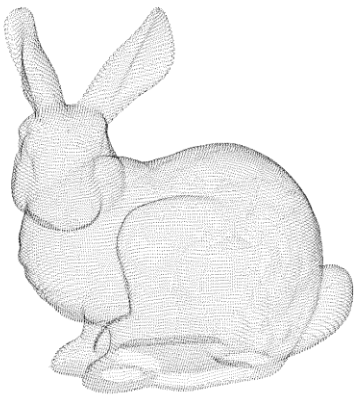
Detection

3D Perception

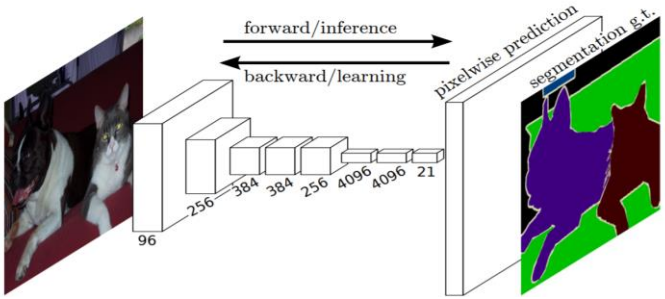
- In this part, we will talk about feature extraction from:
  - Multi-view images
  - Voxel
  - Point cloud





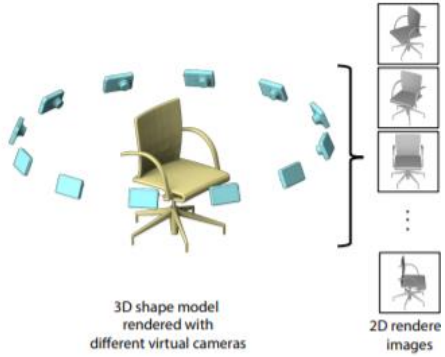


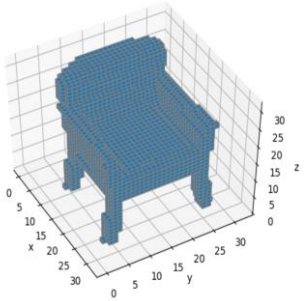
Limitation of CNN

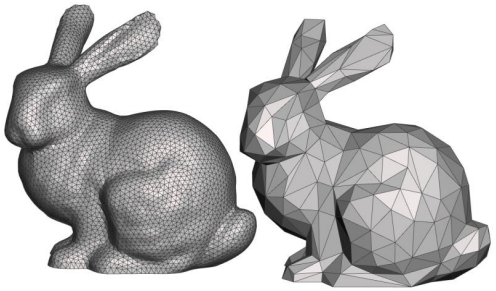


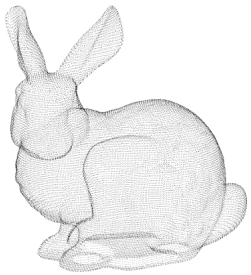
- Can we directly apply CNN on 3D data?
  - Well, it depends…

3D Representation	CNN applicable?
Multi-view images	O
Voxel	O
Mesh	x
Point Cloud	x



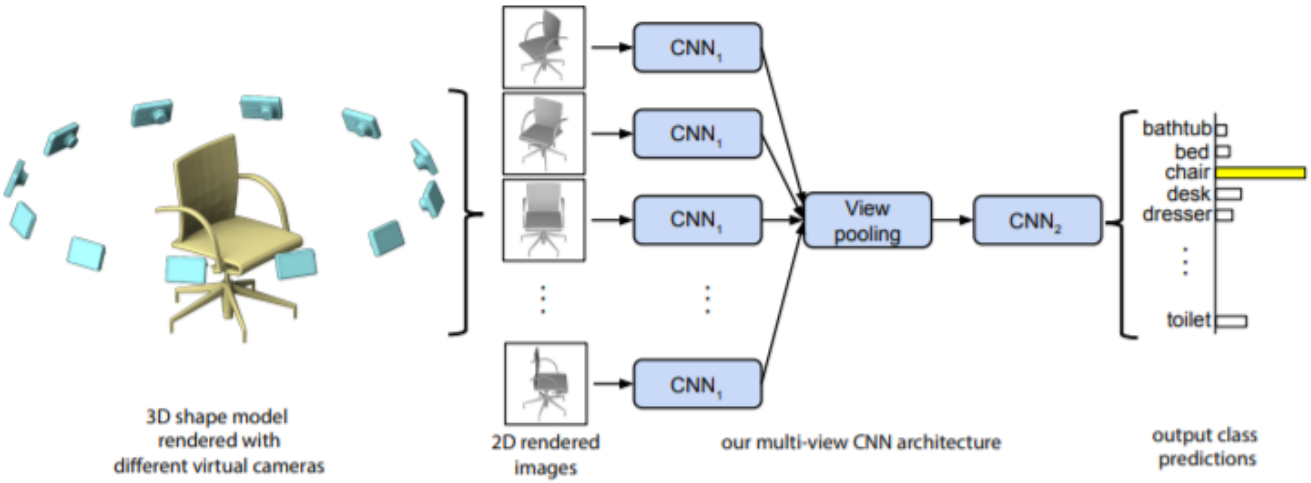






Multi-View Images

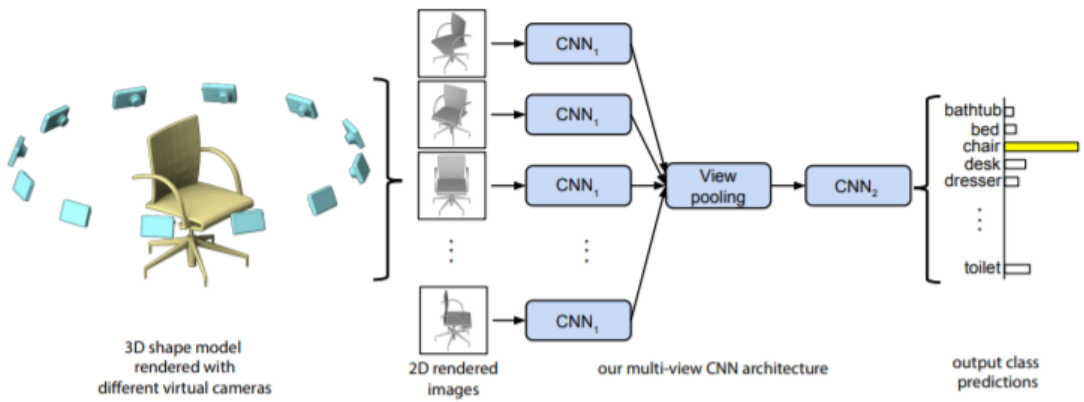
- Represent a 3D object with images captured from multiple views
- MVCNN for object recognition
  - Extract image features with shared CNN
  - Aggregate features from all views with view pooling



Multi-view convolutional neural networks for 3d shape recognition, ICCV 2015

MVCNN (cont'd)

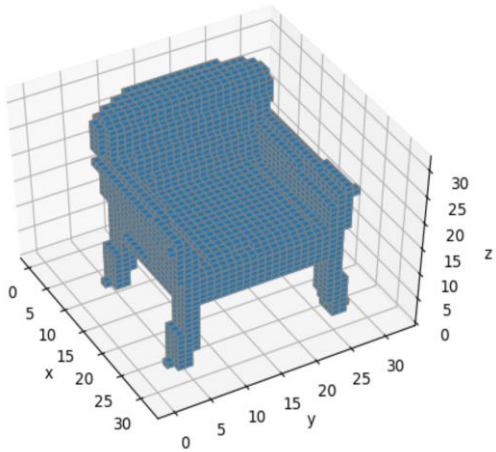
- Pros
  - Can leverage SOTA or pre-trained CNNs for excellent performance
- Cons
  - Setting not necessarily practical (assuming **multi-view** at the same time)
  - Sensitive to (1) viewpoint selection, (2) invisible viewpoint
  - No information on 3D geometry

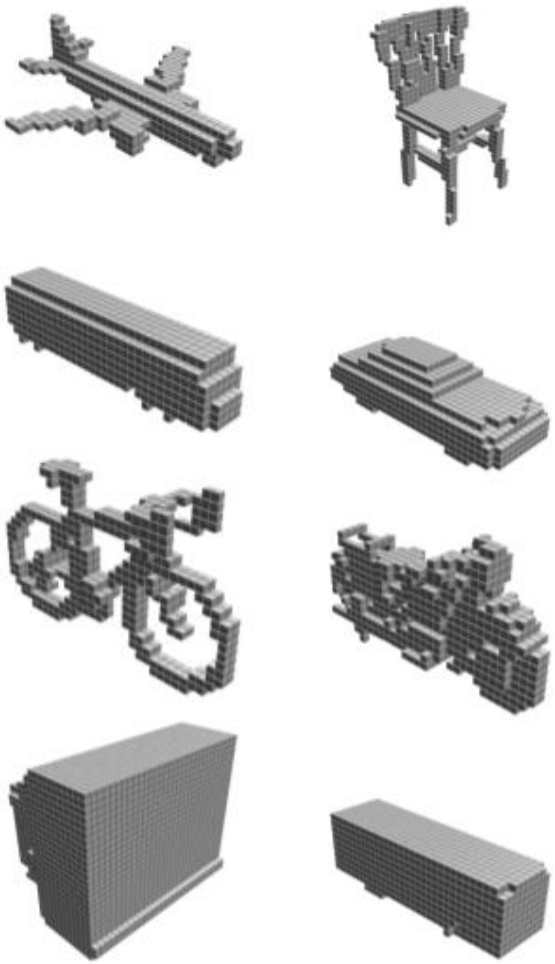


Multi-view convolutional neural networks for 3d shape recognition, ICCV 2015

Voxels

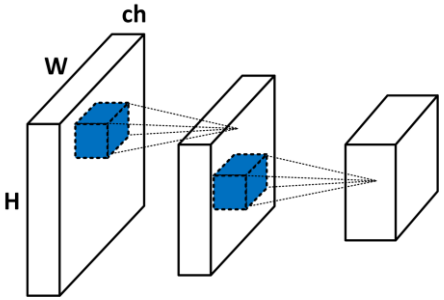
- Grids in fixed resolution $x \times y \times z$
- Each grid contains 0/1: occupancy



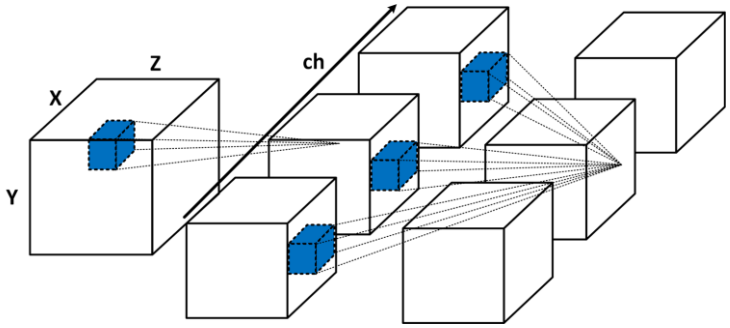


3D Convolution for Voxels

- Convolution for 2D images $$y(i,j) = \sum_m \sum_n x(m,n) \cdot h(i-m, j-n)$$
- Convolution for 3D voxels $$y(i,j,k) = \sum_m \sum_n \sum_p x(m,n,p) \cdot h(i-m, j-n, k-p)$$



2D CNN



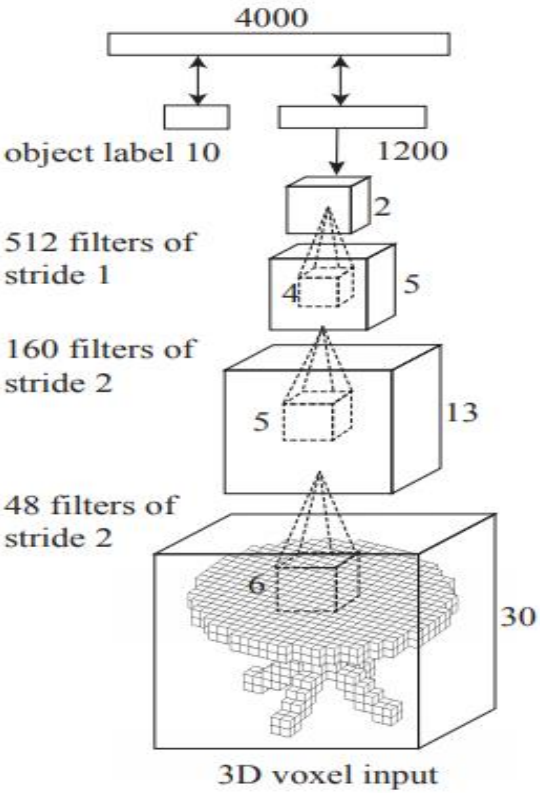
3D CNN

3D ShapeNets

- 3D object classification with voxels via 3D CNNs
- Accuracy only ~77%, not comparable to MVCNN
  - Any explanation?
- Remarks
  - Pros:
    - Represent shape geometry
    - Easy to operate with 3D CNN
  - Cons:
    - Memory consuming...why?

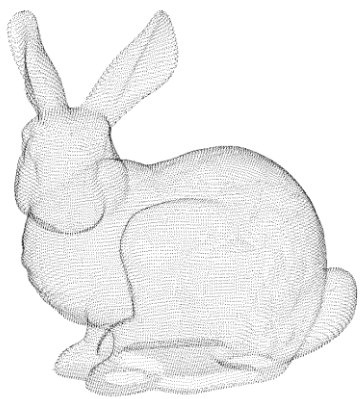
10 x 10 x 10 -> 1000

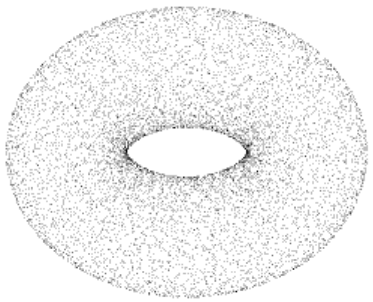
100 x 100 x 100 -> 1000,000



Point Cloud

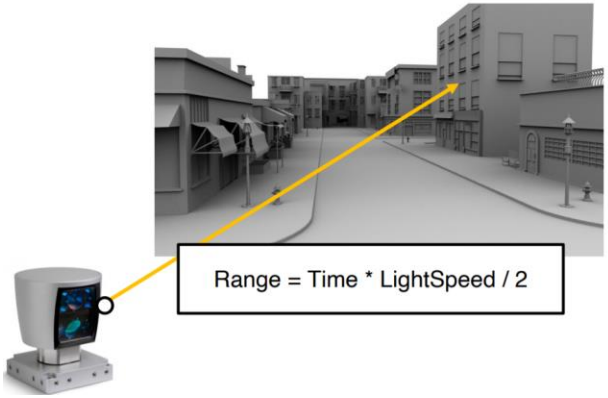
- Point cloud is a point set, representing 3D shapes
- Each point is represented by coordinates (x, y, z)
- Point cloud is stored as a N×3 matrix (N: point number, 3: coordinates)





$$\begin{array}{cc} & 3 \\ N & \begin{bmatrix} (x_1, y_1, z_1) \\ (x_2, y_2, z_2) \\ \\ \\ (x_N, y_N, z_N \end{bmatrix} \end{array}$$

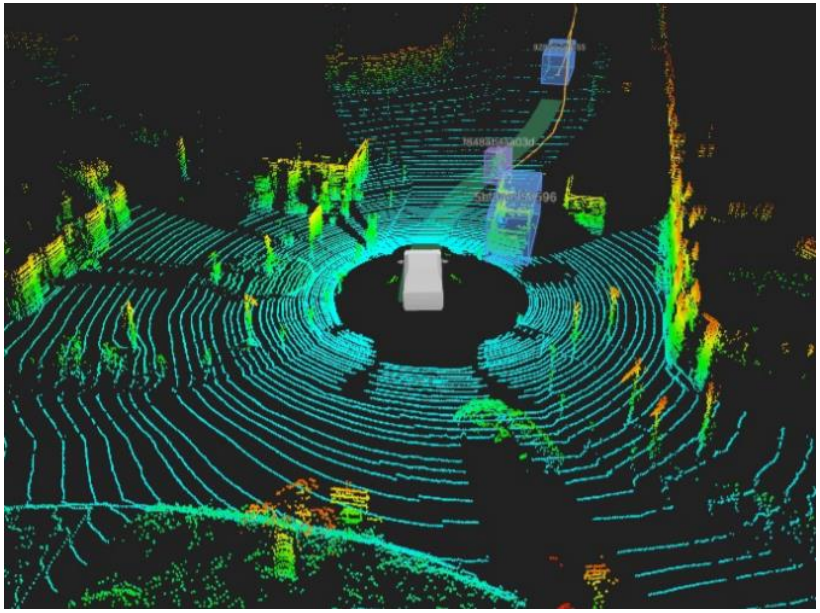
Point Cloud (cont'd)



- Point cloud can be obtained from LiDAR sensors
- Can capture scene geometry

Autonomous driving

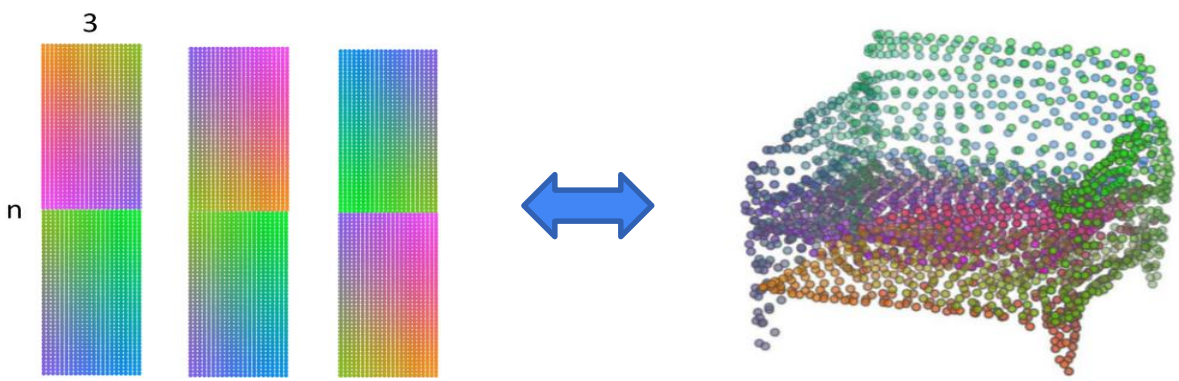
Augmented Reality (AR)



Reference: Robot Perception, taught by Prof. Shenlong Wang, UIUC
https://shenlong.web.illinois.edu/teaching/cs598fall21/assets/slides/lecture3_sensors.pdf

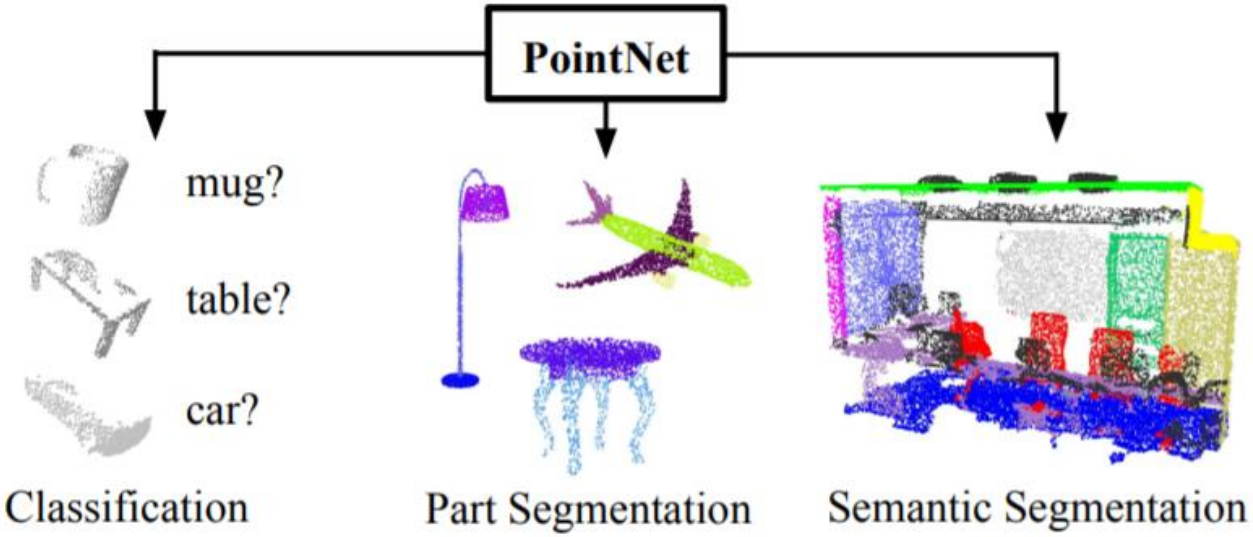
Challenges in Point Cloud

- Can we directly apply CNN on point cloud?
  - No, because point cloud is not grid-structured.
- The shape object can be represented in different orders
- Unknown shape transformation (e.g., translation, rotation…)



PointNet

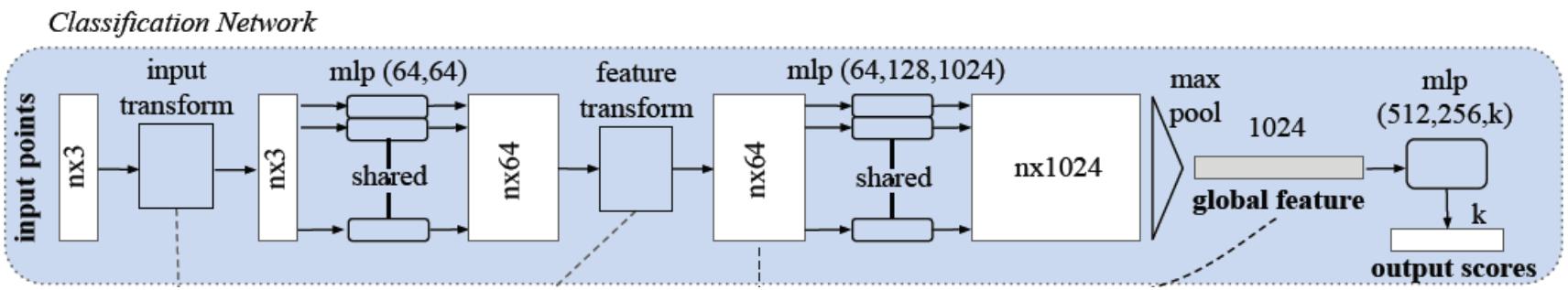
- Goal: Point cloud classification & segmentation

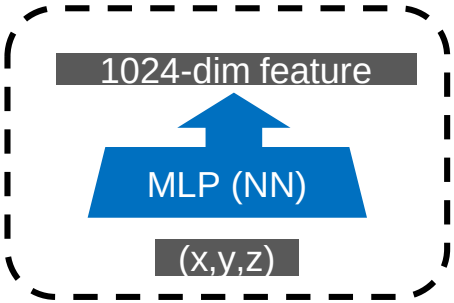


PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation, CVPR 2017

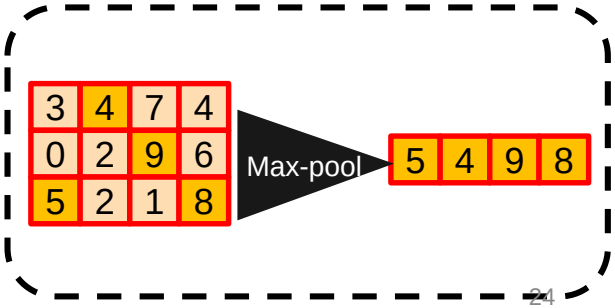
PointNet

- Goal: Point cloud classification & segmentation
- Classification





Multi-Layer Perceptron

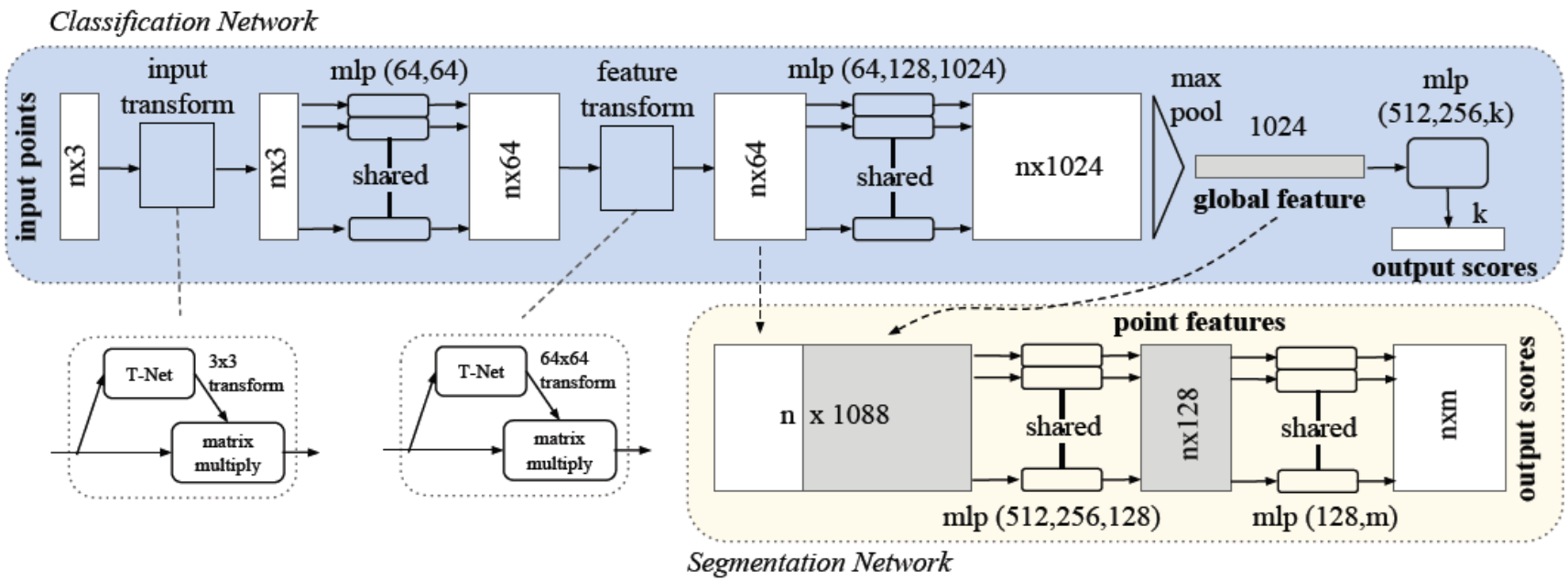


Channel-wise max-pooling

PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation, CVPR 2017

PointNet

- Goal: Point cloud classification & segmentation
- Classification
- Segmentation



PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation, CVPR 2017

PointNet

- Goal: Point cloud classification & segmentation
- Classification & segmentation
- Qualitative results

Part segmentation

Point: (xyz, rgb)

Scene segmentation

PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation, CVPR 2017

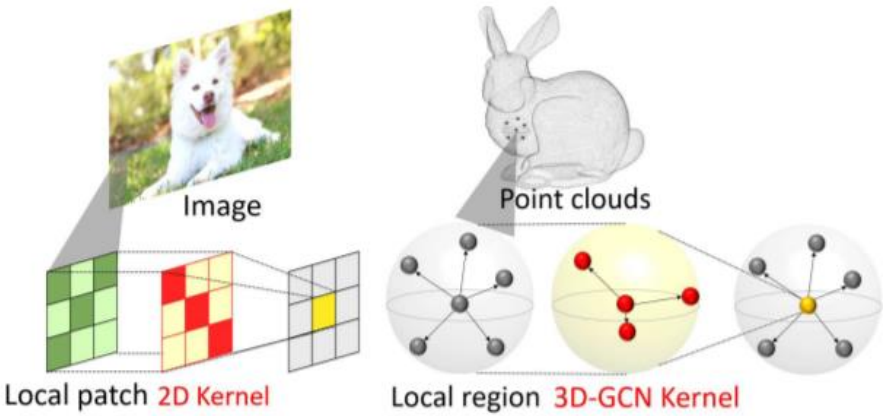
PointNet

- Goal: Point cloud classification & segmentation
- Classification & segmentation
- Qualitative results
- Remarks
  - Pros: extract features from unordered points
  - Cons:
    - Outlier/noisy point cloud data
    - Cannot capture **detailed geometry** (global pooling)
    - Might not robust to transformation like **translation, scaling, rotation**

PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation, CVPR 2017

Extensions of PointNet

- **PointNet++**: Deep Hierarchical Feature Learning on Point Sets in a Metric Space, NIPS 2017
- **Dynamic Graph CNN** for Learning on Point Clouds, TOG 2019
- **KPconv**: Flexible and deformable convolution for point clouds, ICCV 2019
- **Convolution in the cloud**: Learning deformable kernels in 3D graph convolution networks for point cloud analysis, CVPR 2020 (VLLab @ NTU)
- **3D-SelfCutMix:** Self-Supervised Learning for 3D Point Cloud Analysis, ICIP 2022 (VLLab @ NTU)

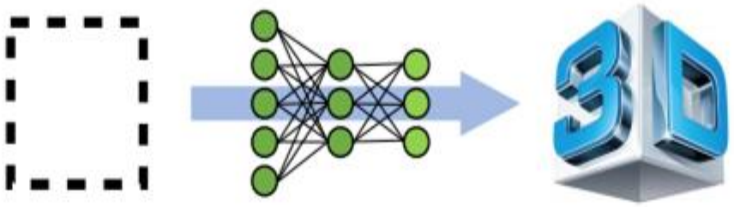


What to Cover Today?

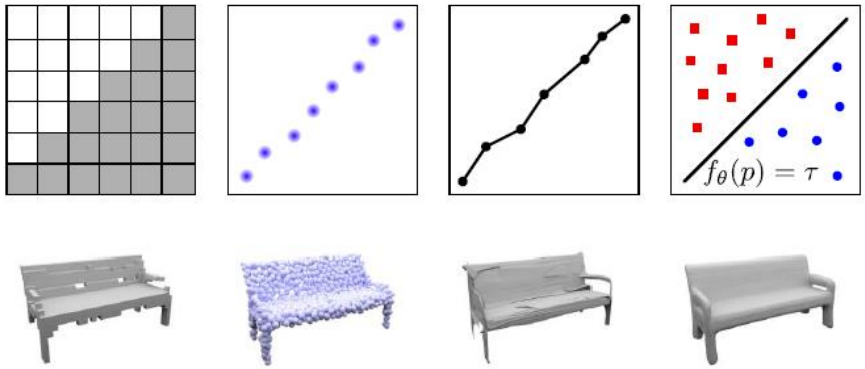
- Introduction to 3D Vision
- Part I: 3D Perception
- Part II: 3D Reconstruction
- Neural Radiance Fields
  - Extension of NeRF
  - Advanced Topics of NeRF

3D Reconstruction

- Reconstruct 3D shapes/scenes from partial observations
  - Single/multi-view images
  - Videos
  - Incomplete point cloud
  - text

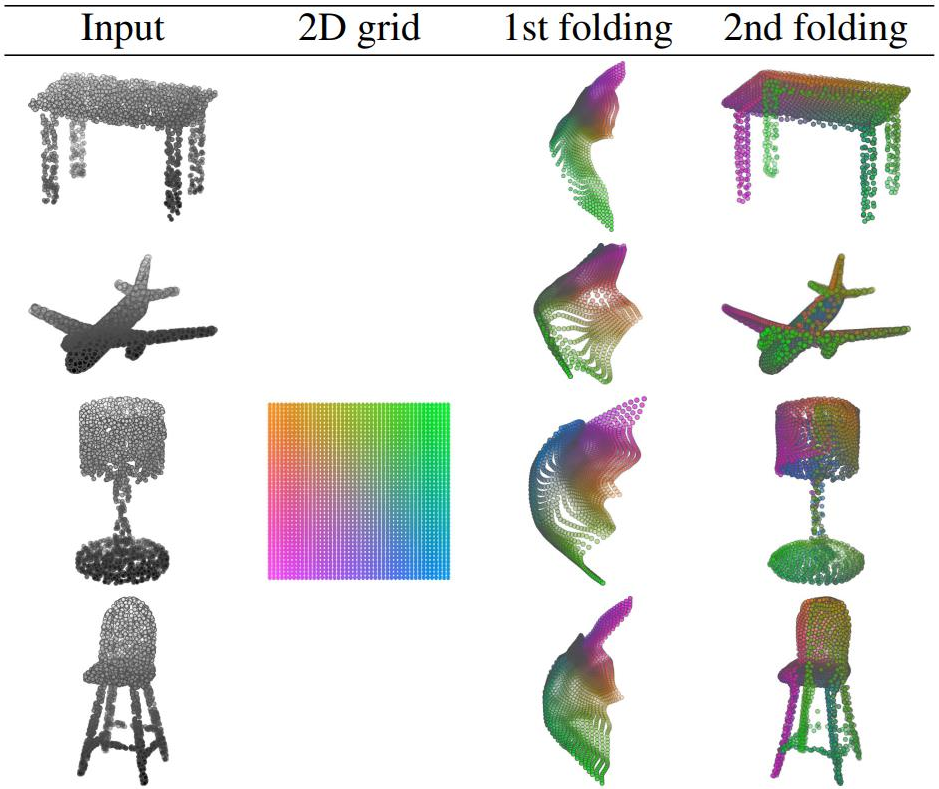


- In this part, we will talk about how to reconstruct
  - ~~Depth~~
  - ~~Voxels~~
  - Point cloud
  - ~~Mesh~~
  - Function (??)



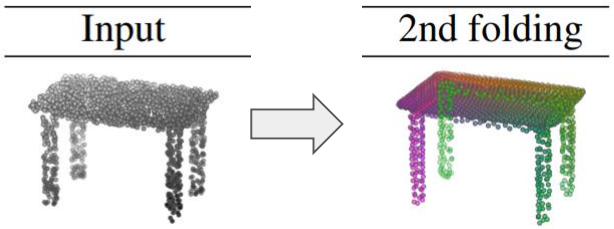
FoldingNet

- 3D reconstruction via point cloud (Auto Encoder)
- Input: point cloud object
- Output: reconstructed point cloud

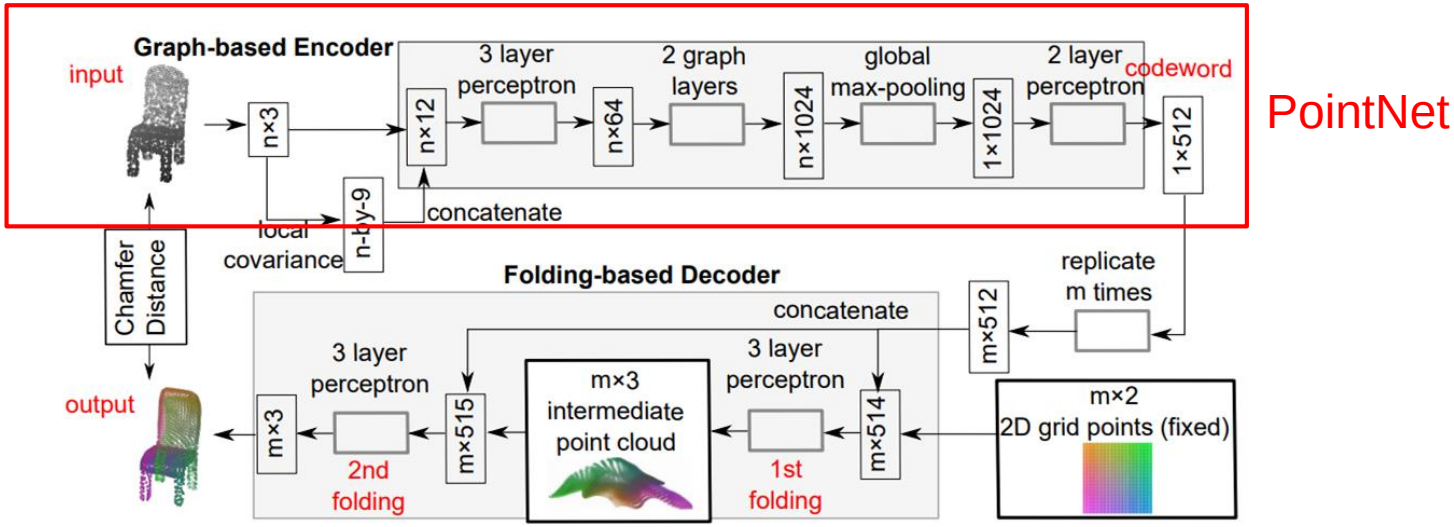


FoldingNet: Point Cloud Auto-encoder via Deep Grid Deformation, CVPR 2018

FoldingNet

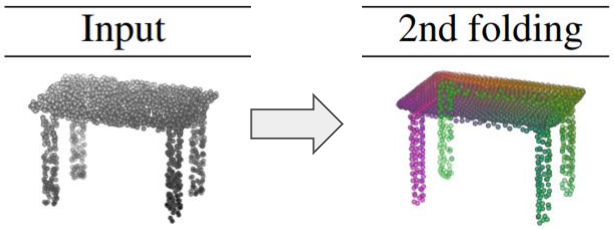


- 3D reconstruction via point cloud (Auto Encoder)
- Input: point cloud object
- Output: reconstructed point cloud
- Decoder: concat 2D coordinates with object feature and pass to MLPs



FoldingNet: Point Cloud Auto-encoder via Deep Grid Deformation, CVPR 2018

FoldingNet

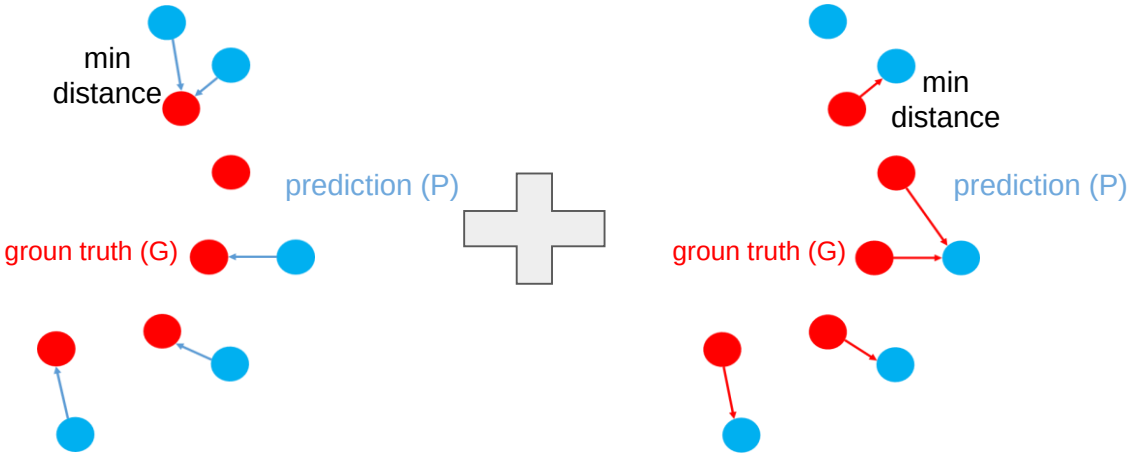


- 3D reconstruction via point cloud
- Input: point cloud object
- Output: reconstructed point cloud
- Loss function: **Chamfer distance**

What if only one-sided?

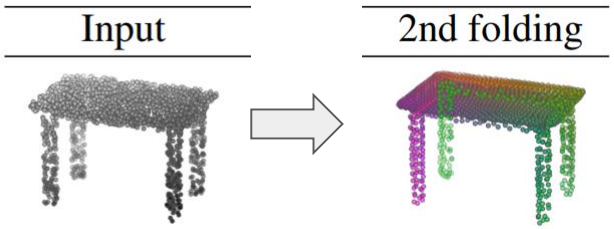
$$d_{CD}(\mathcal{P}, \mathcal{G}) = \frac{1}{|\mathcal{P}|} \sum_{p \in \mathcal{P}} \min_{g \in \mathcal{G}} \|p - g\| + \frac{1}{|\mathcal{G}|} \sum_{g \in \mathcal{G}} \min_{p \in \mathcal{P}} \|g - p\|$$

prediction ground truth



FoldingNet: Point Cloud Auto-encoder via Deep Grid Deformation, CVPR 2018

FoldingNet



- 3D reconstruction via point cloud
- Input: point cloud object
- Output: reconstructed point cloud
- Example results

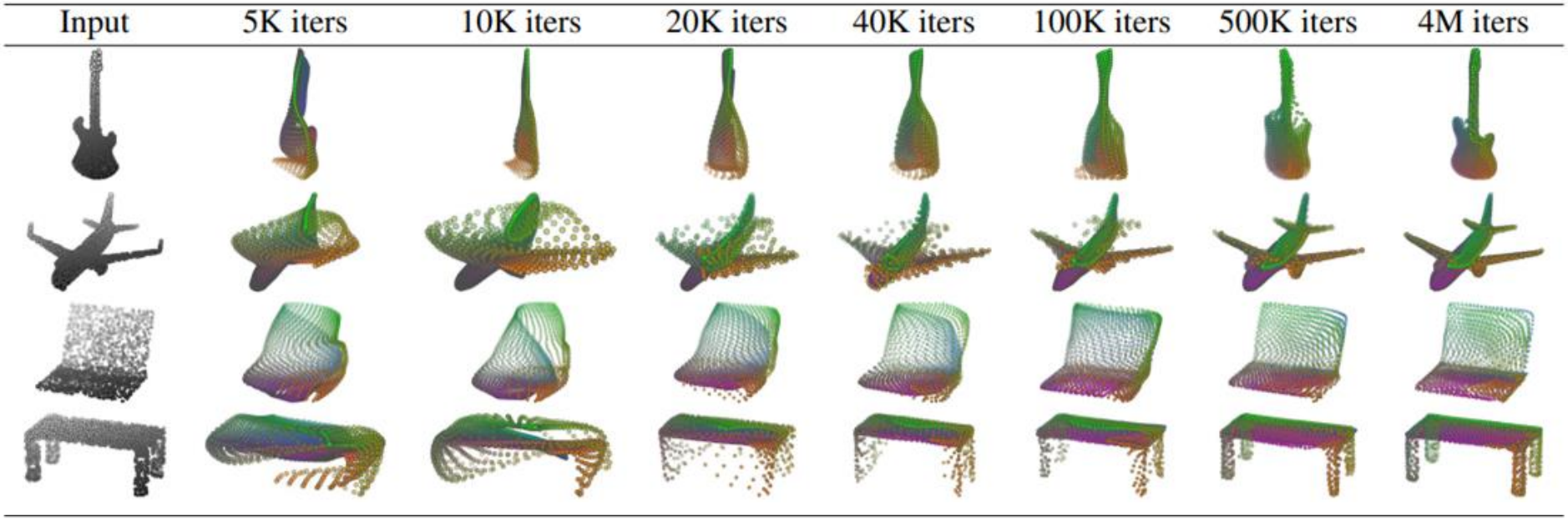


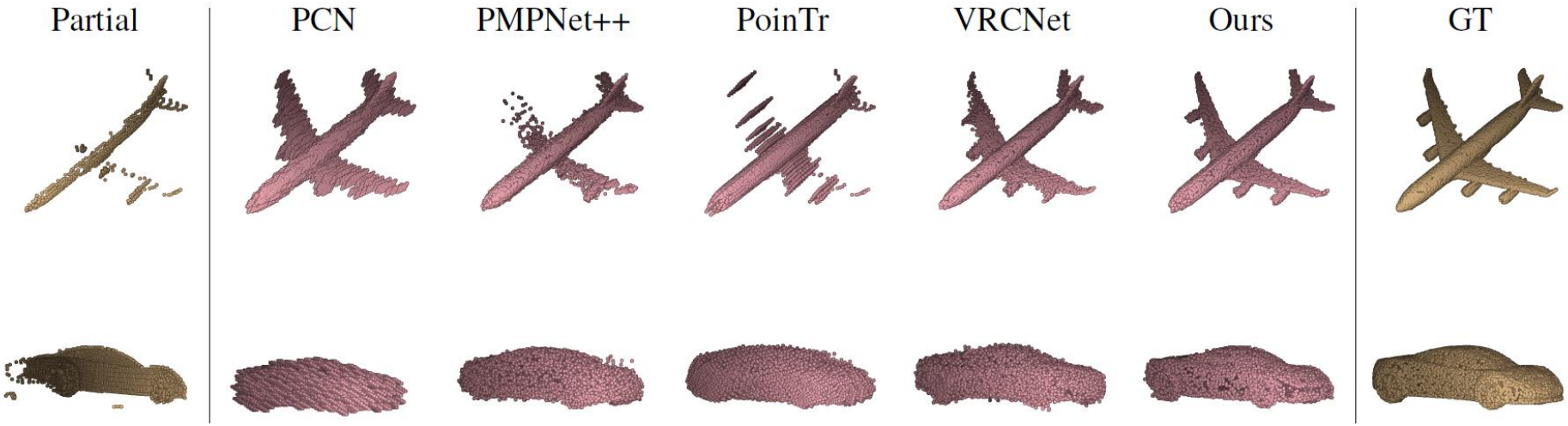
Table 2. Illustration of the training process. Random 2D manifolds gradually transform into the surfaces of point clouds.

FoldingNet: Point Cloud Auto-encoder via Deep Grid Deformation, CVPR 2018

Extensions of FoldingNet

Point Cloud completion: complete partial point clouds

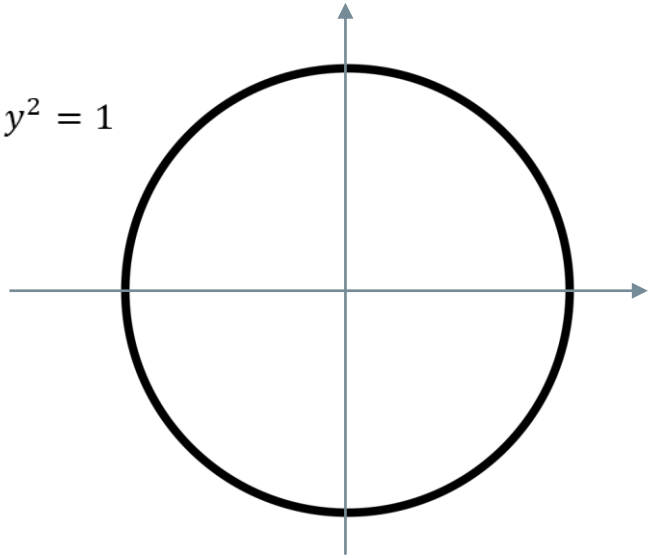
- **PCN**: Point Completion Network, 3DV 2018
- **VRCNet**: Variational Relational Point Completion Network, CVPR 2021
- **PoinTr**: Diverse Point Cloud Completion with Geometry-Aware Transformers, ICCV 2021
- **Variational Transformer** for Dense Point Cloud Semantic Completion, NeurIPS 2022 (VLLab @ NTU)



Implicit Representation

- Represent shapes as “function”
- Tell us whether a point is on the surface

$x^2 + y^2 = 1$



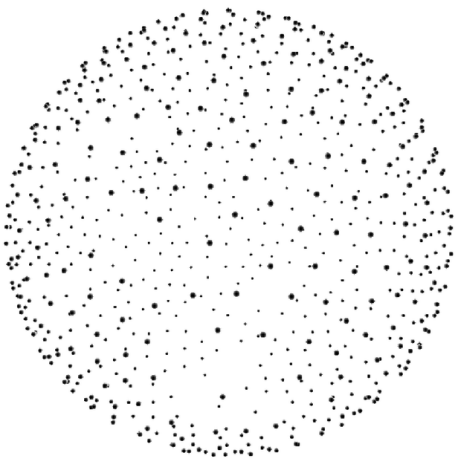
2D circle

Q: Are these points on the circle?
(0, 1)
(1, 0)
(1, 1)
(0, 0)

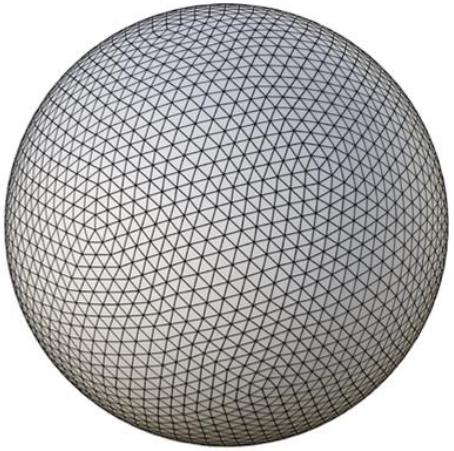
Implicit Representation

- Represent shapes as “function”
- Unit sphere: $f(x, y, z) = x^2 + y^2 + z^2 - 1$
  - Surface is the solution set of $f(.) = 0$

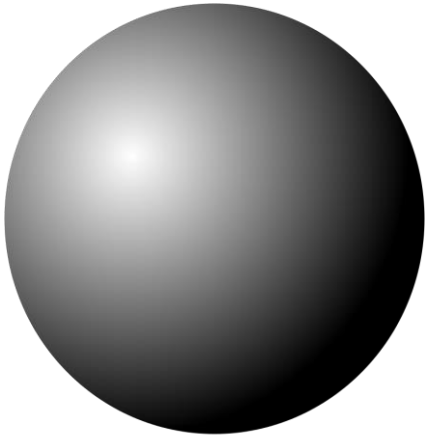
$x^2 + y^2 + z^2 = 1$



Point cloud



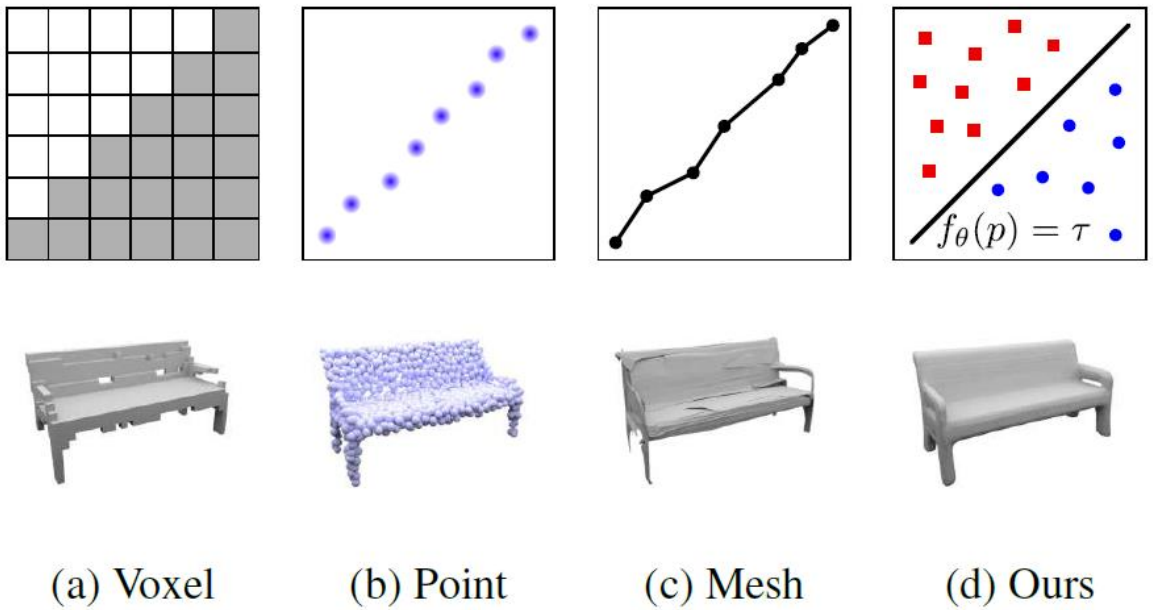
Mesh



Implicit function

Occupancy Network

- Shape is a function that determines a point is inside/outside of it

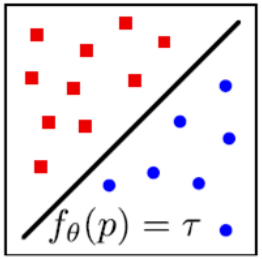


Occupancy Networks: Learning 3D Reconstruction in Function Space, CVPR 2019

Occupancy Network

- Make model learn to predict occupancy at every possible 3D point $p \in R^3$
- Think of occupancy function as a "classifier"
- Condition on object feature X

$$f_\theta : \mathbb{R}^3 \times \mathcal{X} \rightarrow [0, 1]$$

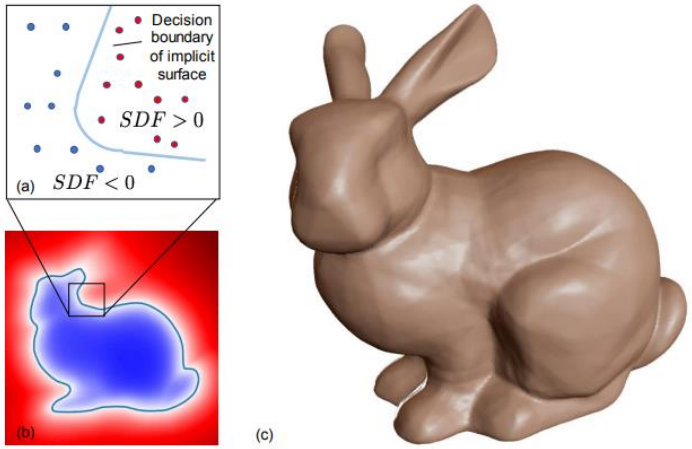


Occupancy Networks: Learning 3D Reconstruction in Function Space, CVPR 2019

Signed Distance Function

- Make model learn to predict **distance to surface** at every possible 3D point $p \in R^3$
- Think of signed distance function as a "regressor"
- Condition on object feature X

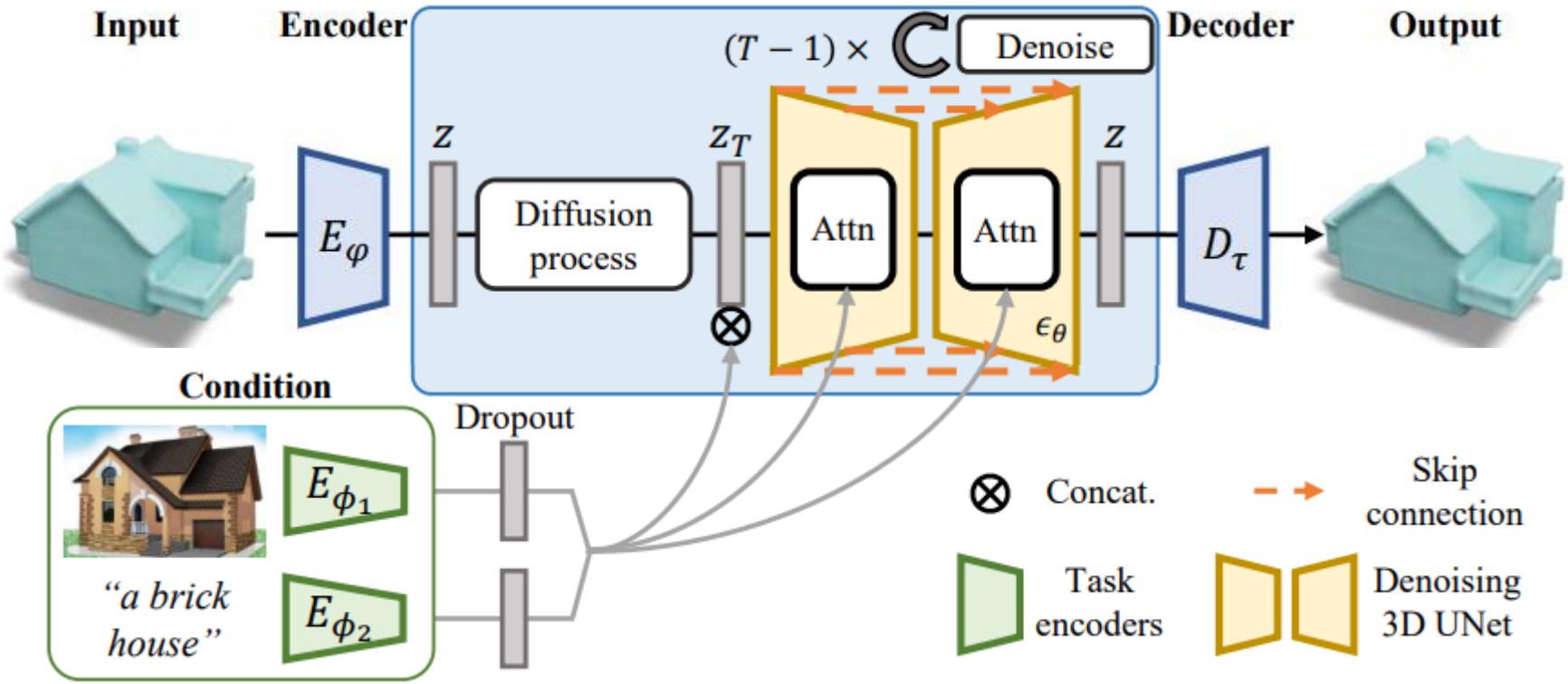
$$f_\theta : \mathbb{R}^3 \times \mathcal{X} \rightarrow [\mathbb{R}]$$



DeepSDF: Learning Continuous Signed Distance Functions for Shape Representation, CVPR 2019

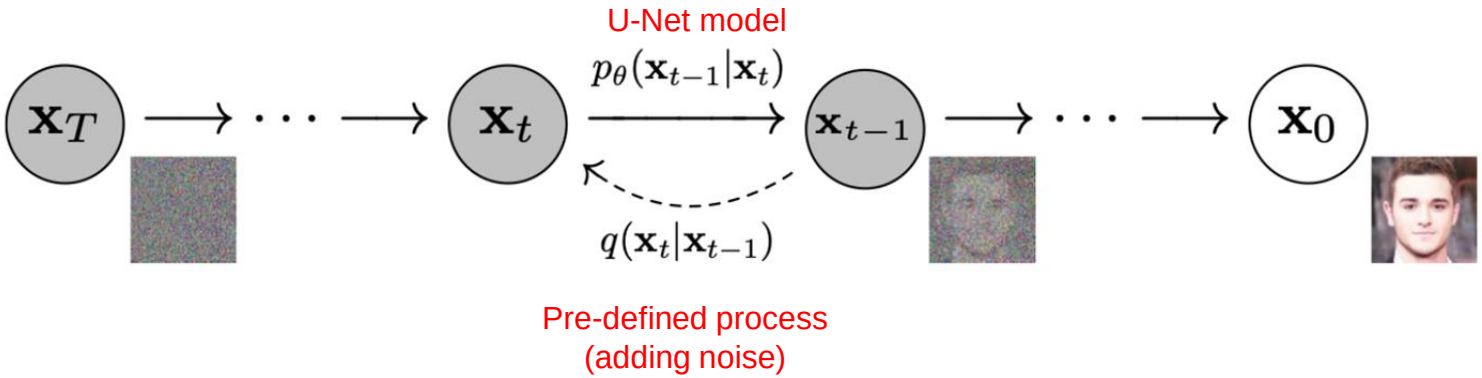
Extension on Signed Distance Function

- Train an Auto Encoder for SDF (input voxel)
- Train a conditional LDM for latent vector z (text or image)



SDFusion: Multimodal 3D Shape Completion, Reconstruction, and Generation, CVPR 2023

Recap: Diffusion model (intuitively)



- Can be viewed as denoising from a Gaussian noise image
- Each step makes little progress of denoising (total about 1000 steps)
- Output image of each step can be seen as the **original image** combining with a **noise** using specific ratio
- The process can also be seen as predicting the **added noise**

Implicit Representation (Occupancy, SDF)

Strength

- Flexible shape topology
- Arbitrary resolution
- Few model parameters

Weakness

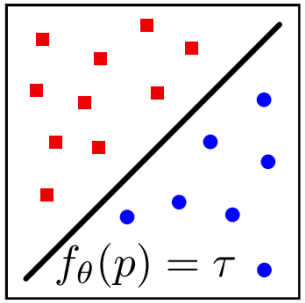
- Require post-processing to get mesh
- Cannot handle complex scene

What to Cover Today?

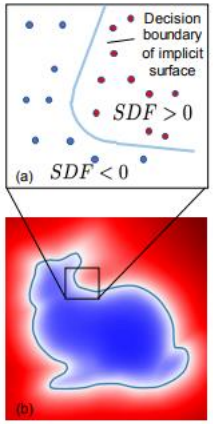
- Introduction to 3D Vision
- Part I: 3D Perception
- Part II: 3D Reconstruction
- Neural Radiance Fields
  - Extension of NeRF
  - Advanced Topics of NeRF

Recap:
Neural Networks as a Continuous Shape Representation

Occupancy Networks
(Mescheder et al. 2019)
(x,y,z) -> occupancy



Deep SDF
(Park et al. 2019)
(x,y,z) -> distance



Pros: Compact and expressive parameterization

Cons: Limited rendering, difficult to optimize

Slide credit: Jon Barron, cs598dwh

NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis

Many slides from Jon Barron and cs598dwh (UIUC)

Ben Mildenhall*

UC Berkeley

Google

Pratul Srinivasan*

UC Berkeley

Google

Matt Tancik*

UC Berkeley

Jon Barron

Google Research

Google

Ravi Ramamoorthi

UC San Diego

UC San Diego

Ren Ng

UC Berkeley

Slide credit: cs598dwh

Problem: Novel view synthesis (NVS)

Inputs: sparsely sampled images of scene

Outputs: *new* views of same scene

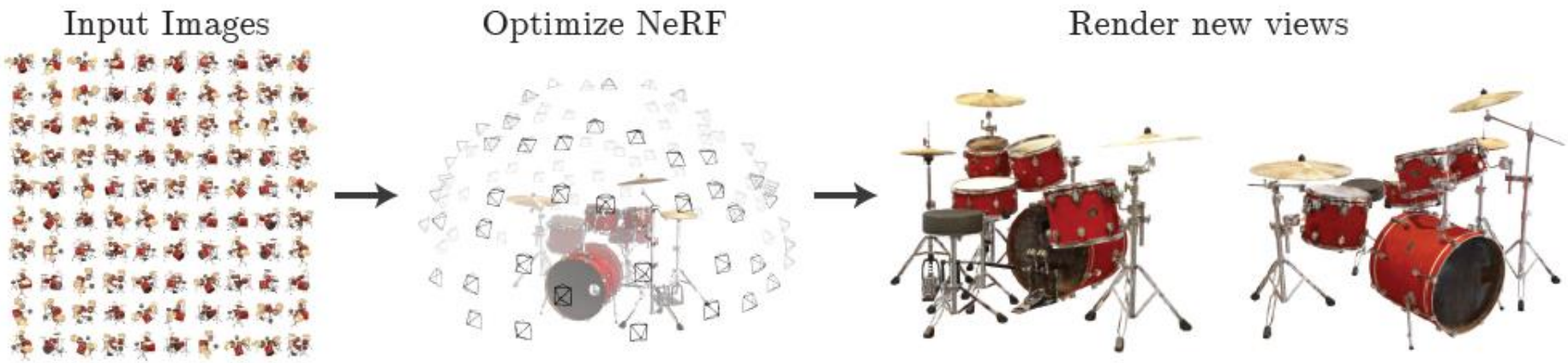
tancik.com/nerf

Slide credit: Jon Barron

NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis, ECCV 2020

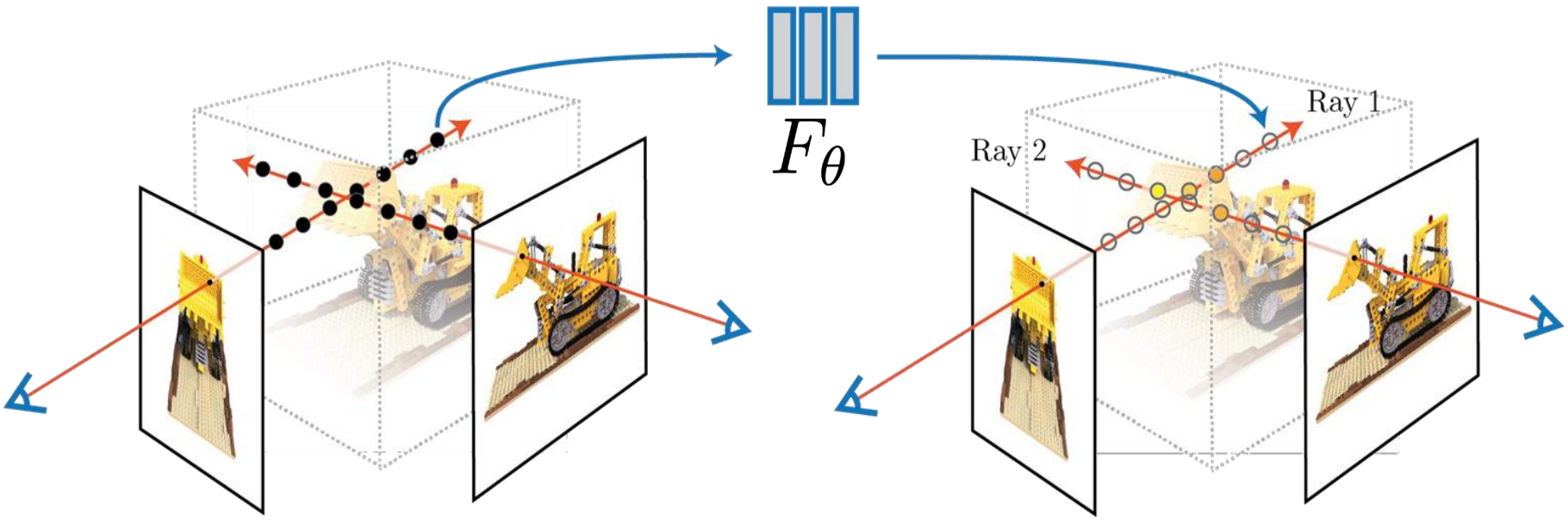
NeRF (Neural randiance field)

- Goal: learn 3D representation, and perform **novel view synthesis**
- Input: multi-view images + camera poses
- Output: 3D representation (neural radiance field)



NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis, ECCV 2020

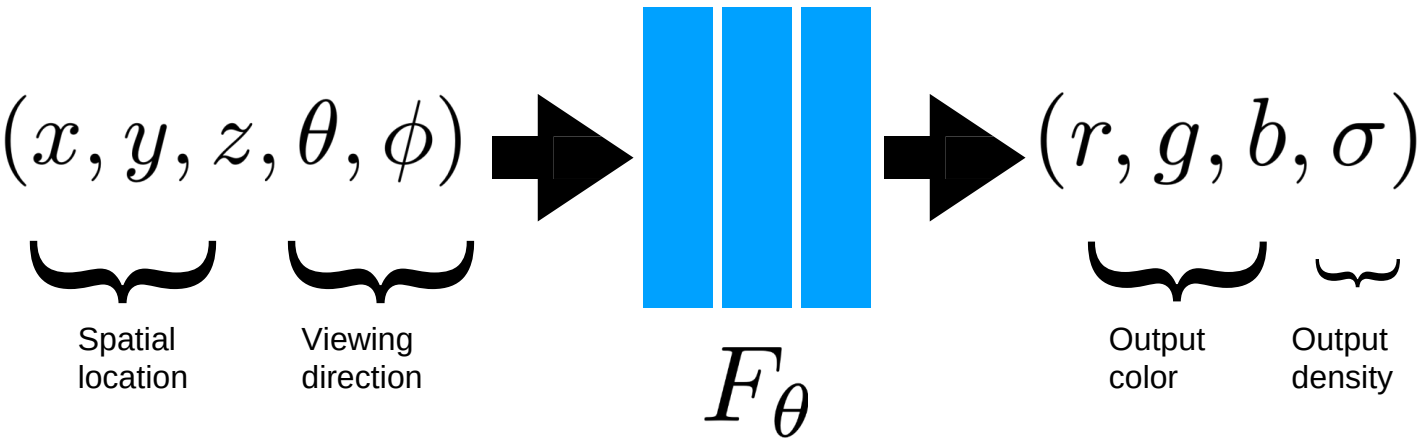
Generate views with traditional volume rendering



Slide credit: Jon Barron

NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis, ECCV 2020

NeRF (Neural randiance field)



Fully-connected neural network
9 layers,
256 channels

Slide credit: Jon Barron

NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis, ECCV 2020

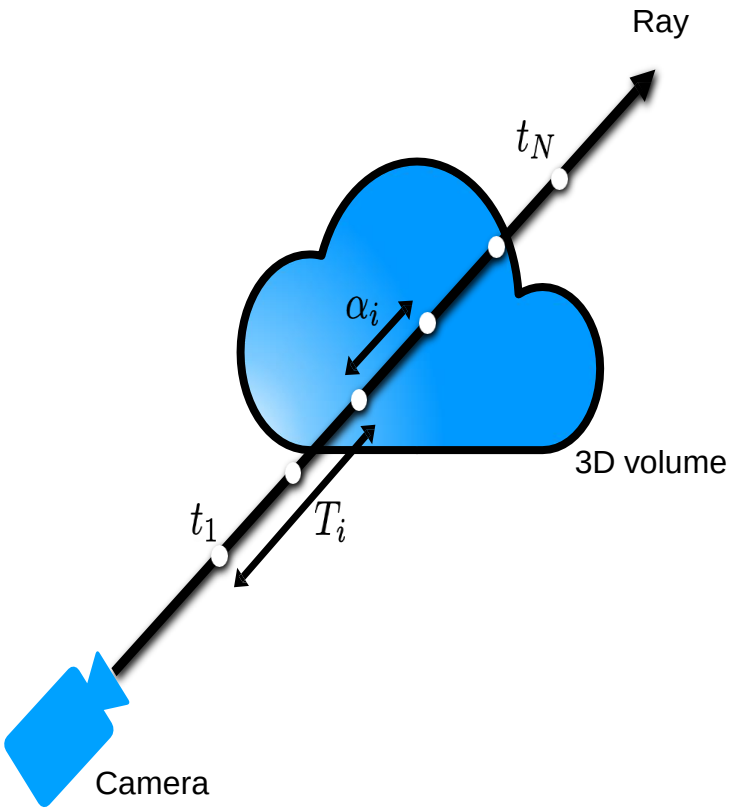
Generate views with traditional volume rendering

Rendering model for ray r(t) = o + td:

$$C \approx \sum_{i=1}^{N} T_i \alpha_i c_i$$

weights

colors



- How much light is blocked earlier along ray:

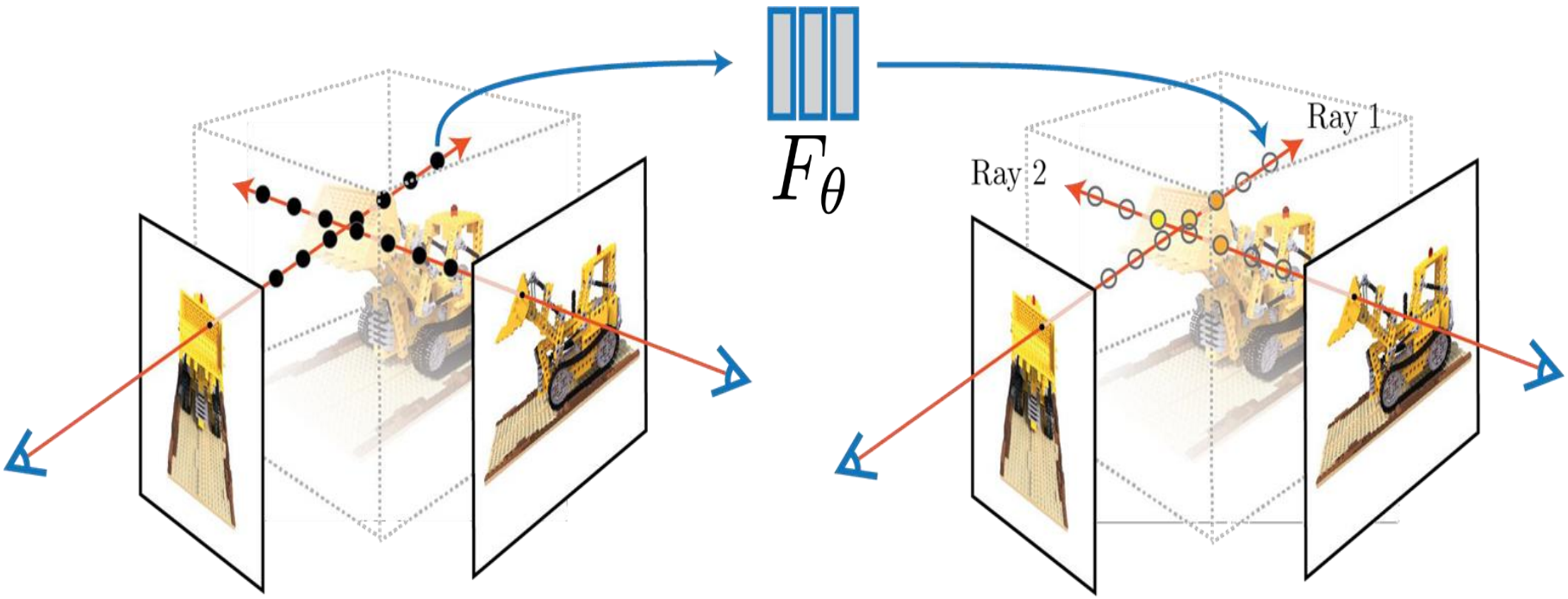
$$T_i = \prod_{j=1}^{i-1} (1 - \alpha_j)$$

- How much light is contributed by ray segment *i*:

$$\alpha_i = 1 - e^{-\sigma_i \delta t_i}$$ ← -Density * Distance Between Points

Slide credit: Jon Barron

Optimize with gradient descent on rendering loss

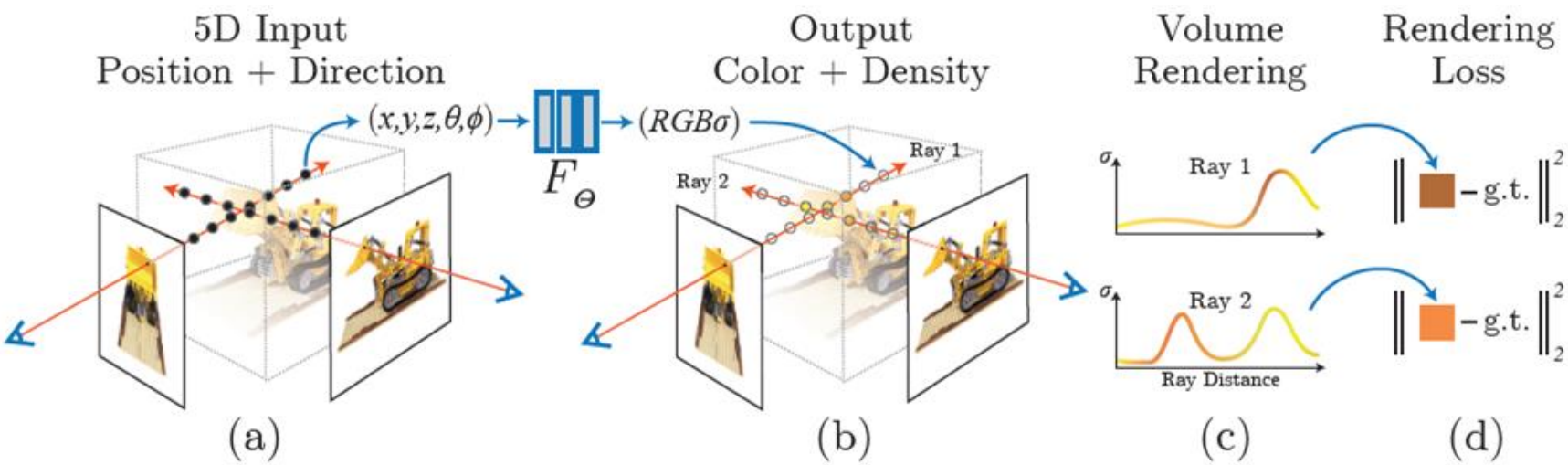


$$\min_\theta \sum_i ||\,\mathrm{render}_i(F_\theta) - I_i||^2$$

Slide credit: Jon Barron

NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis, ECCV 2020

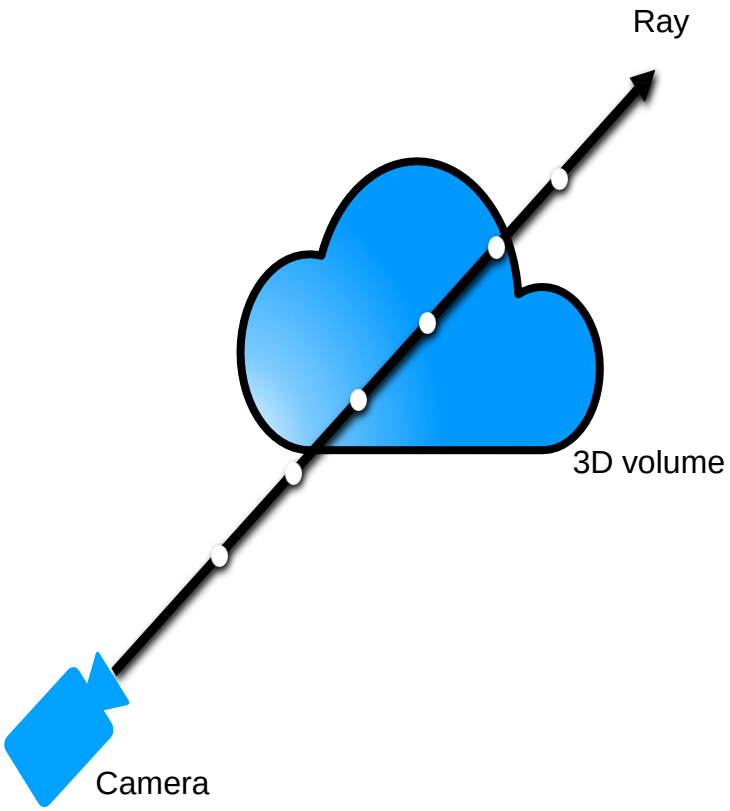
Training network to reproduce all input views of the scene



Slide credit: Jon Barron

NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis, ECCV 2020

Can we allocate samples more efficiently?
--Two pass rendering



NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis, ECCV 2020

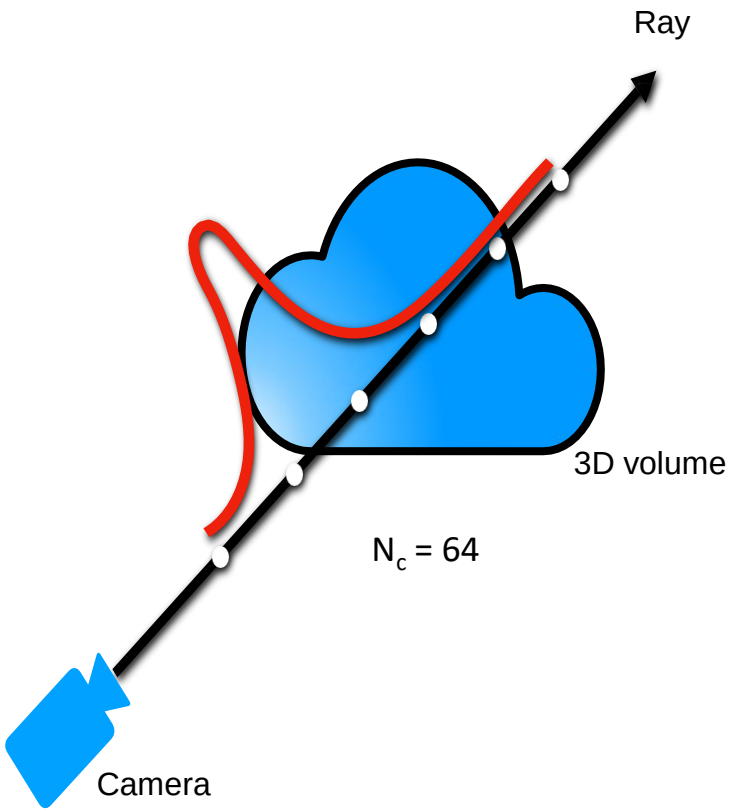
Slide credit: Jon Barron

Two pass rendering:coarse network

- Sparsely sample points along ray
- Serve as a coarse guidance

$$C \approx \sum_{i=1}^{N} \boxed{T_i \alpha_i} c_i$$

treat weights as probability distribution for new samples



Slide credit: Jon Barron

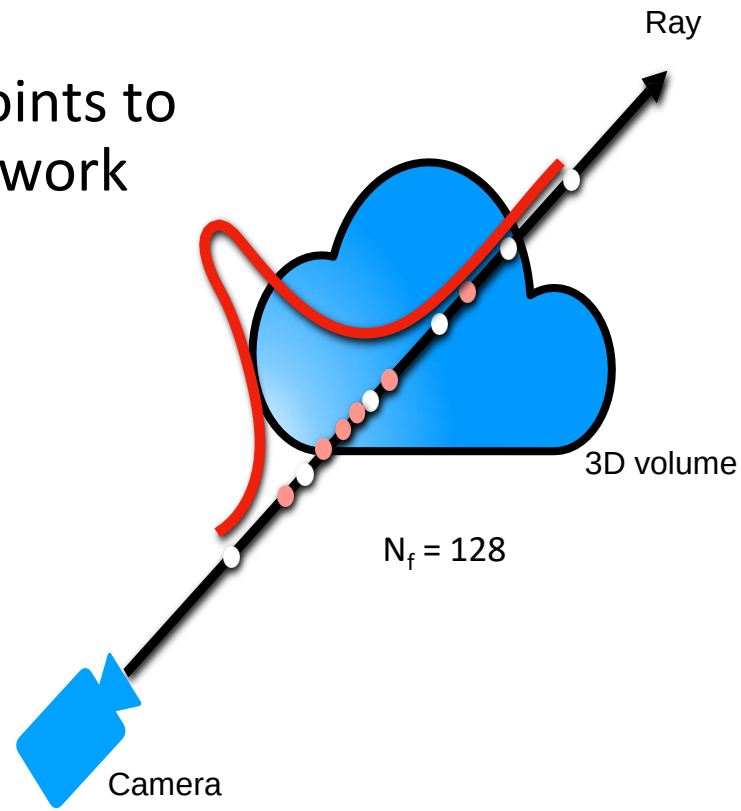
NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis, ECCV 2020

Two pass rendering:fine network

- Use the coarse predicted density to resample new points along ray
- Together compute all $N_c + N_f$ points to calculate final color for fine network

$$C \approx \sum_{i=1}^{N} \boxed{T_i \alpha_i} c_i$$

treat weights as probability distribution for new samples



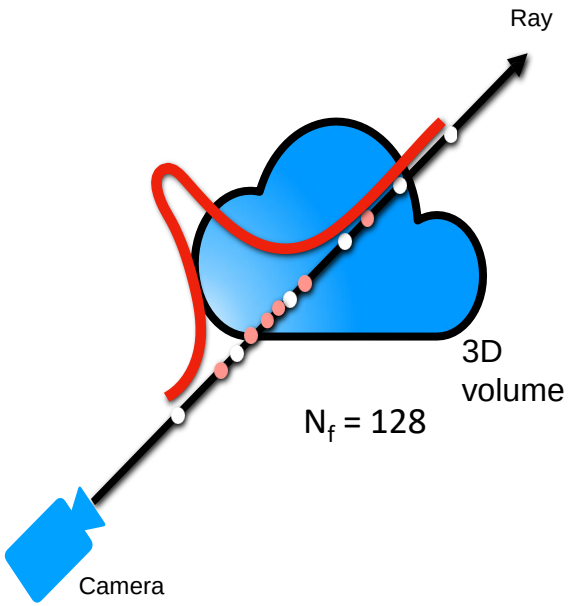
$$\mathcal{L} = \sum_{\mathbf{r} \in \mathcal{R}} \left[\left\| \hat{C}_c(\mathbf{r}) - C(\mathbf{r}) \right\|_2^2 + \left\| \hat{C}_f(\mathbf{r}) - C(\mathbf{r}) \right\|_2^2 \right]$$ (coarse + fine)

Slide credit: Jon Barron

NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis, ECCV 2020

Two pass rendering: optimization

- Optimize coarse network and fine network together
- Only use the prediction of fine network when rendering a new scene



$$\mathcal{L} = \sum_{\mathbf{r}\in\mathcal{R}} \left[\left\| \boxed{\hat{C}_c(\mathbf{r})} - C(\mathbf{r}) \right\|_2^2 + \left\| \boxed{\hat{C}_f(\mathbf{r})} - C(\mathbf{r}) \right\|_2^2 \right] \quad \text{(coarse + fine)}$$

predicted color from coarse network

predicted color from fine network

Slide credit: Jon Barron

NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis, ECCV 2020

Positional encoding

NeRF (Naive)

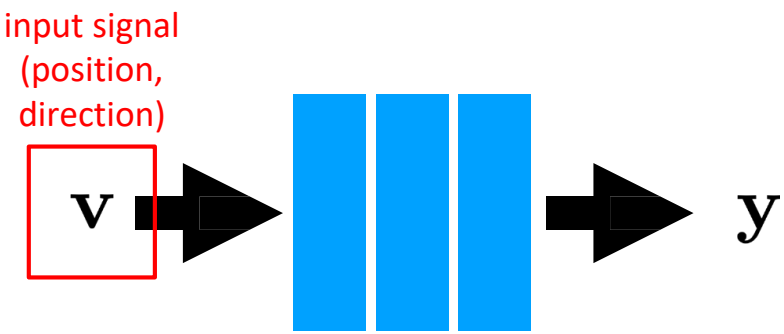
NeRF (with positional encoding)

Slide credit: Jon Barron

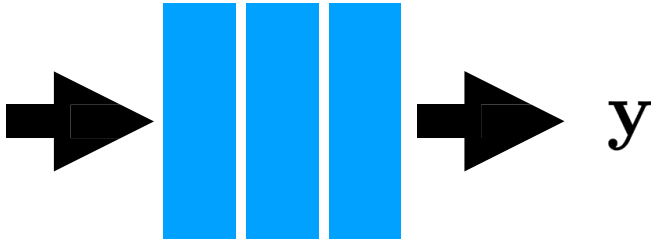
NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis, ECCV 2020

Positional encoding

Naive



$$\begin{pmatrix} \sin(\mathbf{v}), \cos(\mathbf{v}) \\ \sin(2\mathbf{v}), \cos(2\mathbf{v}) \\ \sin(4\mathbf{v}), \cos(4\mathbf{v}) \\ \dots \\ \sin(2^{L-1}\mathbf{v}), \cos(2^{L-1}\mathbf{v}) \end{pmatrix}$$

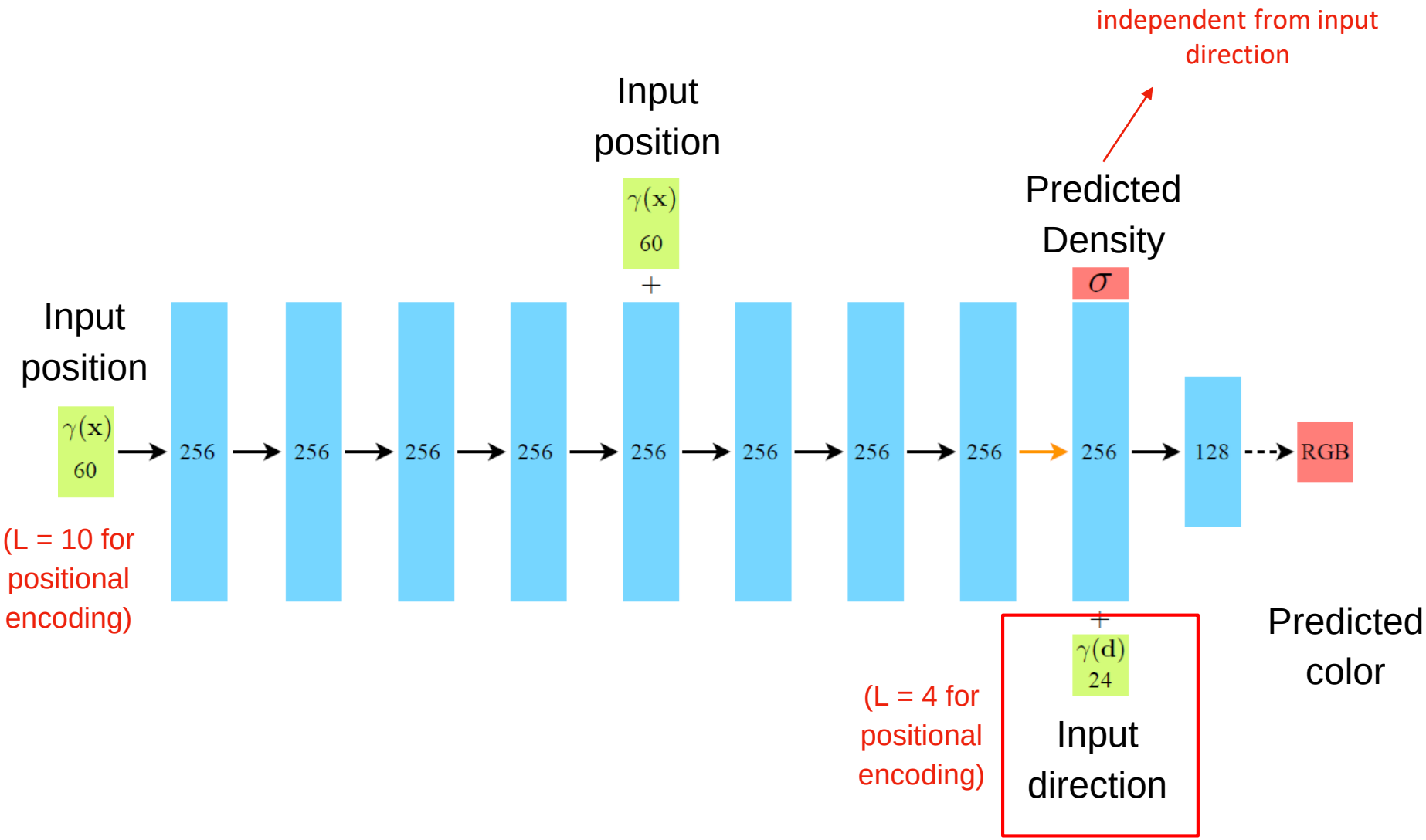


Positional encoding

Slide credit: Jon Barron

NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis, ECCV 2020

Network Structure

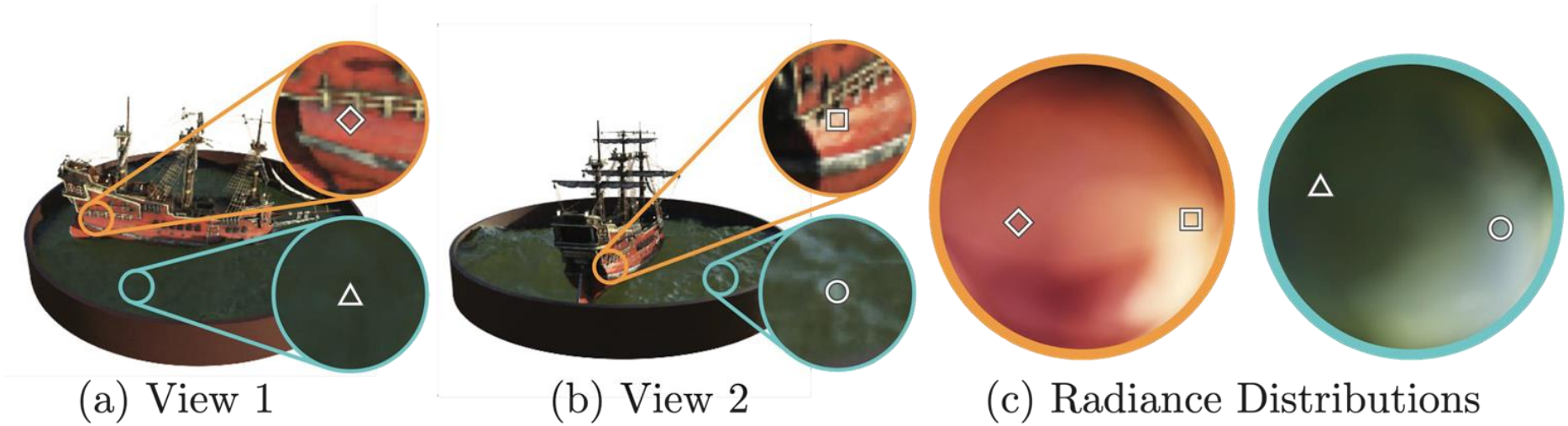


Slide credit: Jon Barron

NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis, ECCV 2020

Viewing directions as input

- The specular reflection (or other changes influenced by lighting) varies across different views



Slide credit: Jon Barron

NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis, ECCV 2020

Viewing directions as input

- The rendered color changes as the viewing direction
- L: image plane change with viewing direction
- R: fixing image plane while the viewing direction feeded to NeRF changes

Viewing directions as input

- Another example

Depth (geometry) Estimation

$$C \approx \sum_{i=1}^{N} T_i \alpha_i c_i$$

$$\downarrow$$

$$d_i$$

Distance from the points to camera

- The predicted density indicates the object surface
- The estimated depth perfectly shows the geometry of foreground object

NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis, ECCV 2020

Depth (geometry) Estimation

- Another example

NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis, ECCV 2020

Depth (geometry) Estimation

- By correctly estimate the depth of the scene, virtual objects are possible to interact with the real scene

NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis, ECCV 2020

NeRF: strength & weakness

Strength

- Photo-realistic texture
- Do not require 3D ground truth
- View-dependent effect

Weakness

- Only fit single scene
- Require much posed images
- Time-consuming rendering (30s per frame) <- Fatal for real-time applications !!

What to Cover Today?

- Introduction to 3D Vision
- Part I: 3D Perception
- Part II: 3D Reconstruction
- Neural Radiance Fields
  - Extension of NeRF: Can We Do Faster??
    - Sample efficiency: NeurMiPs
    - Feature acquired efficiency: DVGO
  - Advanced Topics of NeRF
    - NeRF + generalized
    - NeRF + semantic
    - NeRF + text guided generation
    - When NeRF + splatting

NeurMiPs: Neural Mixture of Planar Experts for View Synthesis





Zhi-Hao Lin Wei-Chiu Ma Hao-Yu Hsu Yu-Chiang Frank Wang Shenlong Wang

UNIVERSITY OF ILLINOIS URBANA-CHAMPAIGN

National Taiwan University 國立臺灣大學

Massachusetts Institute of Technology

https://zhihao-lin.github.io/neurmips/

Slide credit: Zhi-Hao Lin

Method:overview →Sample Efficiency

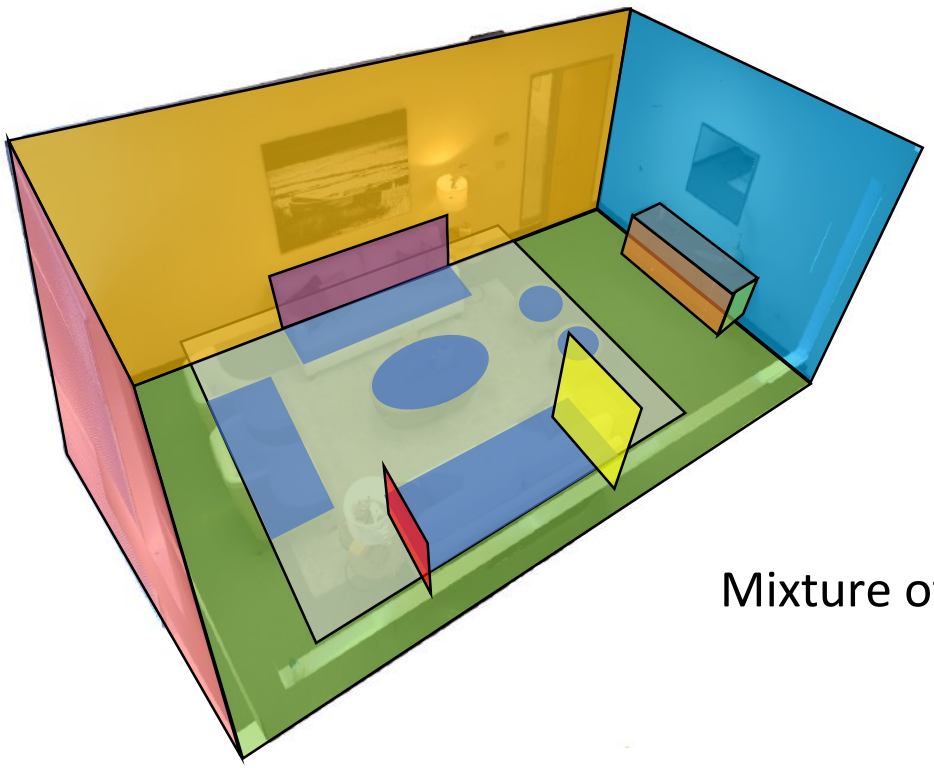
- Represent scene with mixture of local planar surfaces. (non-parallel)

Slide credit: Zhi-Hao Lin

NeurMiPs: Neural Mixture of Planar Experts for View Synthesis (CVPR 2022)

Method:overview →Sample Efficiency

- Represent scene with mixture of local planar surfaces. (non-parallel)



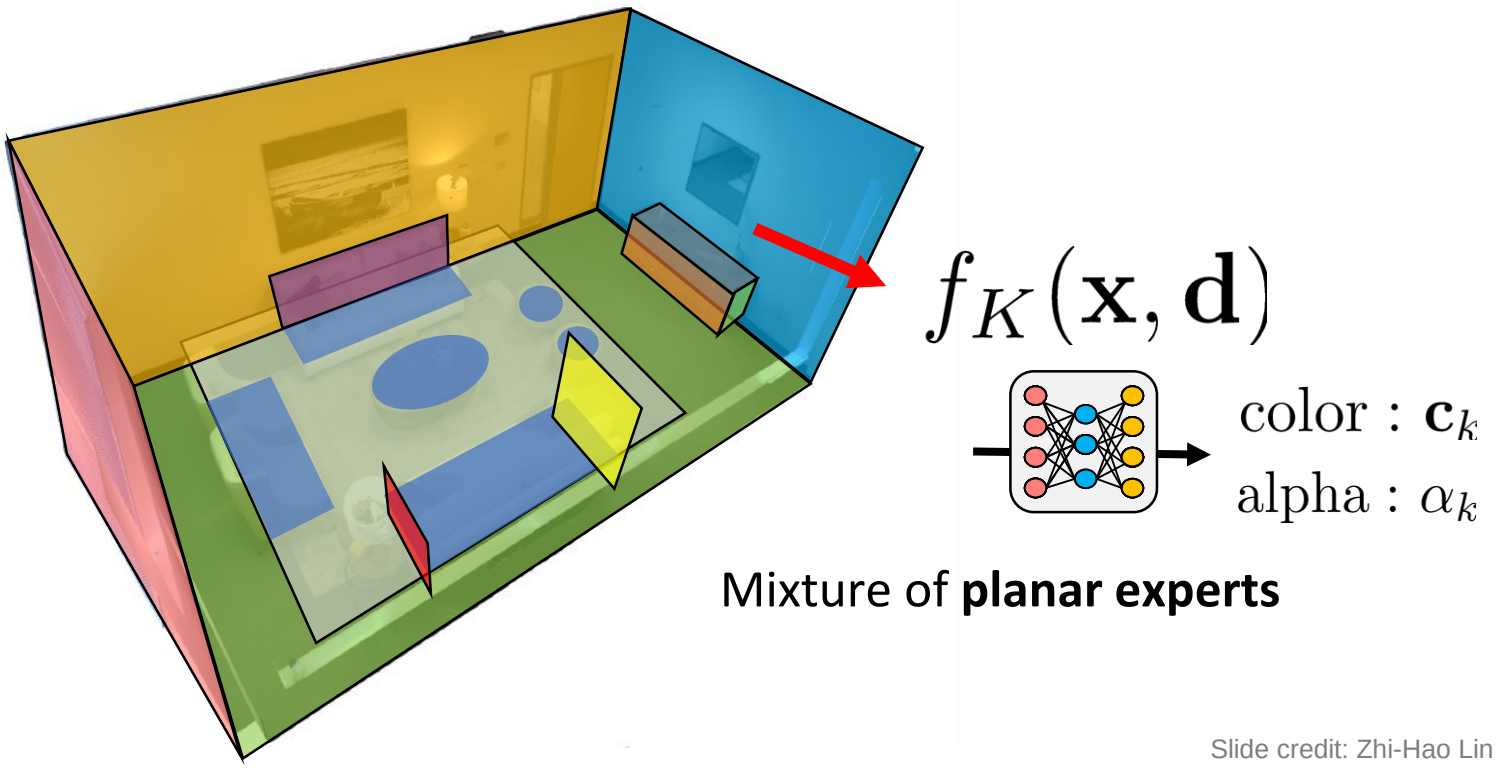
Mixture of **planar experts**

Slide credit: Zhi-Hao Lin

NeurMiPs: Neural Mixture of Planar Experts for View Synthesis (CVPR 2022)

Method:overview →Sample Efficiency

- Represent scene with mixture of local planar surfaces. (non-parallel)

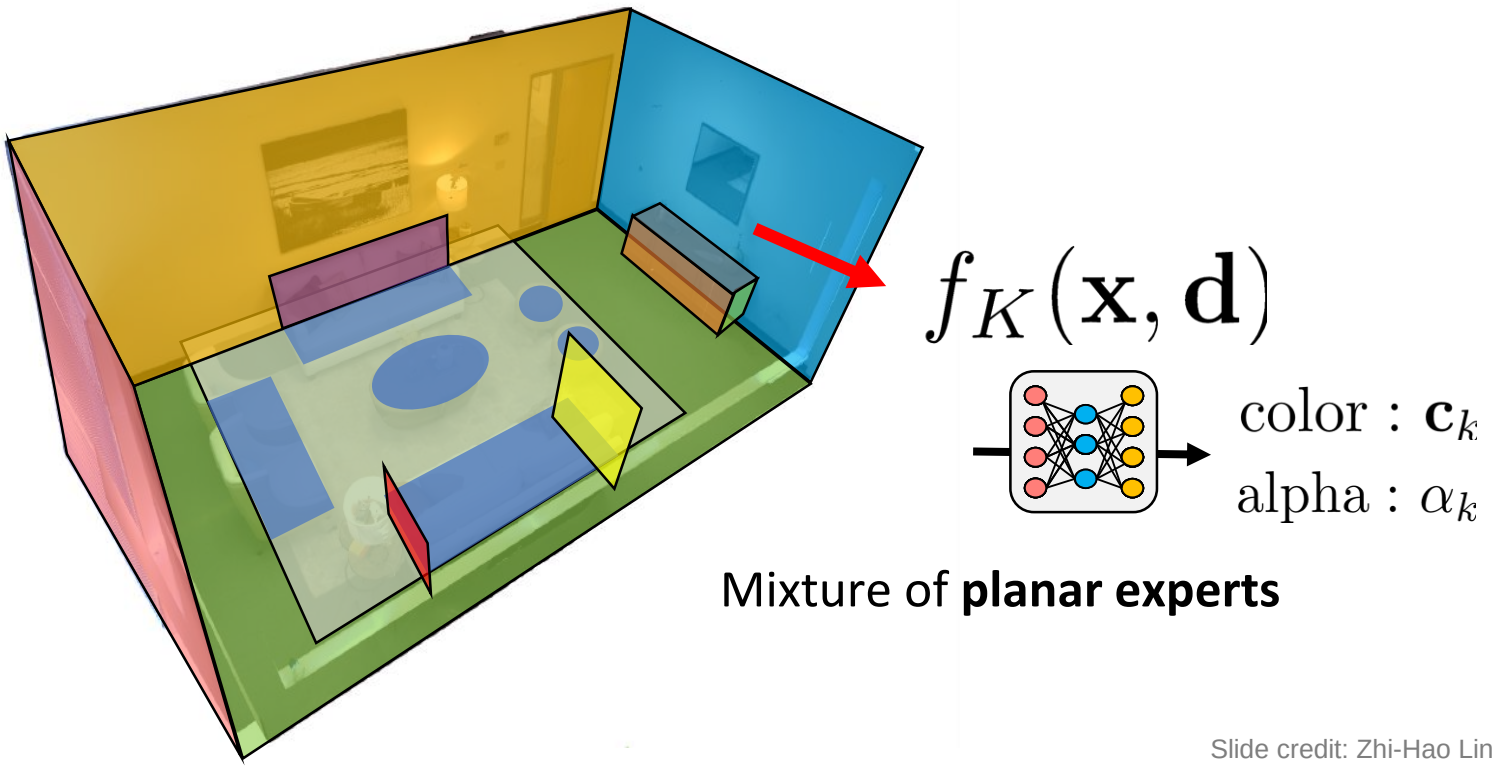


Slide credit: Zhi-Hao Lin

NeurMiPs: Neural Mixture of Planar Experts for View Synthesis (CVPR 2022)

Method:overview →Sample Efficiency

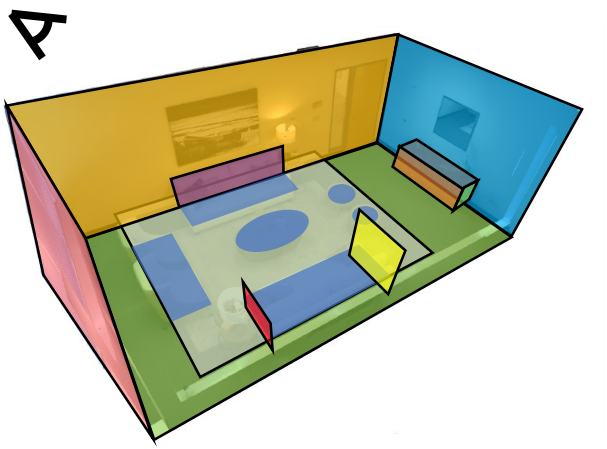
- Represent scene surface: avoid sampling in free space -> speed up
- Flexible plane geometry: allow NVS from wide-range view points



Slide credit: Zhi-Hao Lin

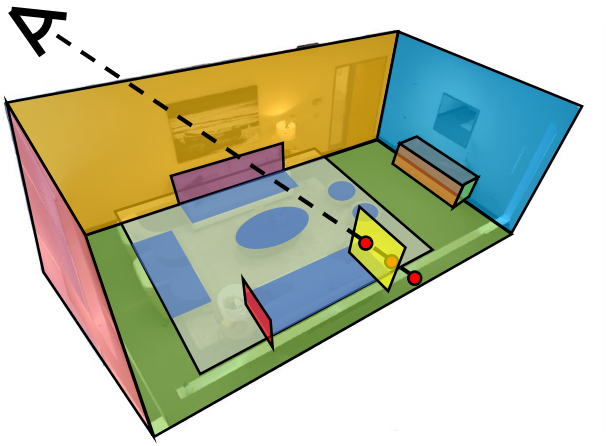
NeurMiPs: Neural Mixture of Planar Experts for View Synthesis (CVPR 2022)

Input Ray and
Mixture of Planes

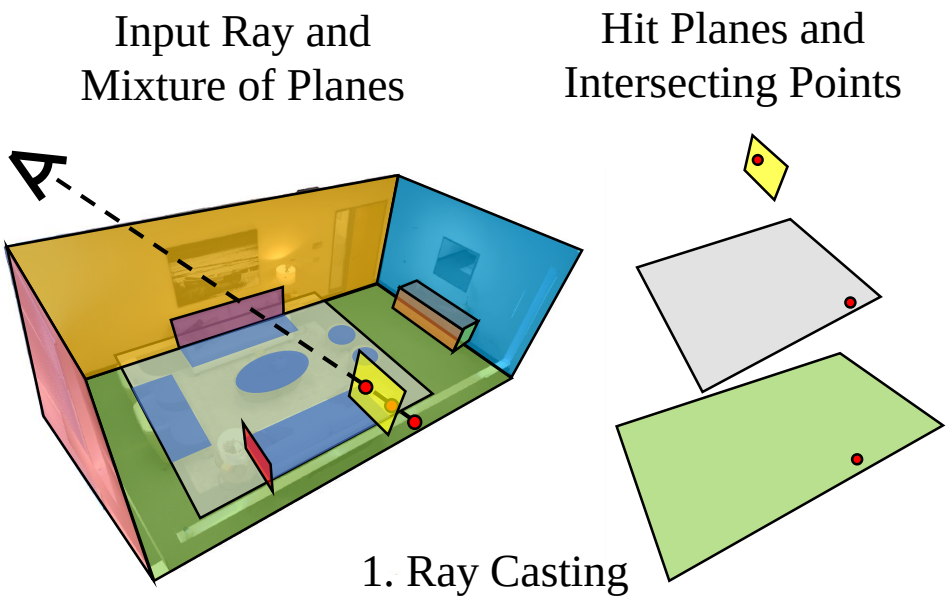


Slide credit: Zhi-Hao Lin

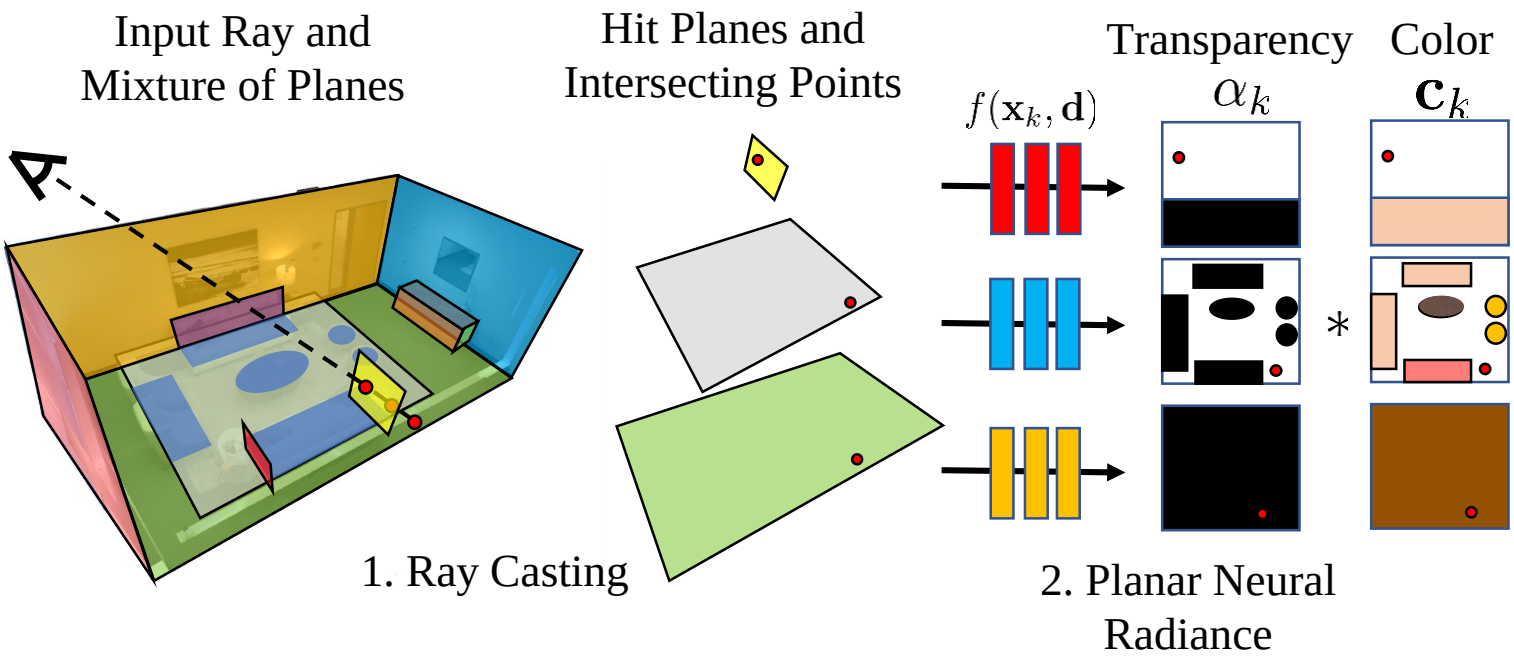
Input Ray and
Mixture of Planes



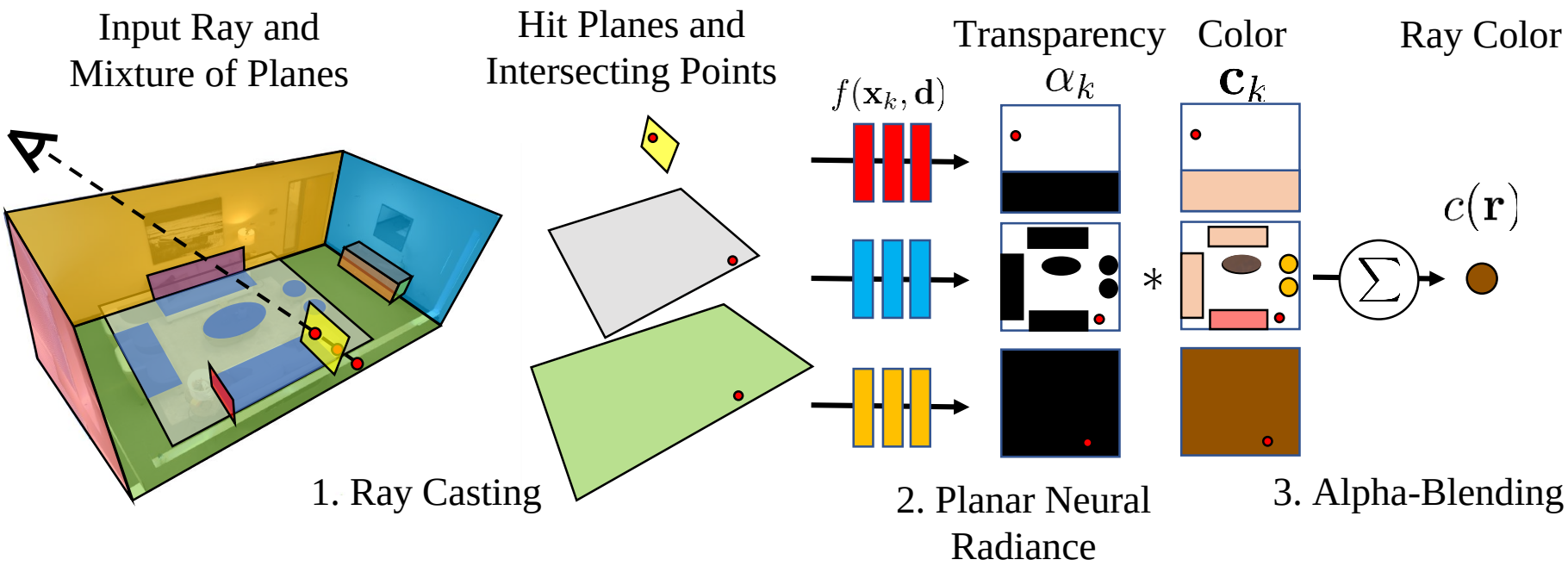
Slide credit: Zhi-Hao Lin



Slide credit: Zhi-Hao Lin

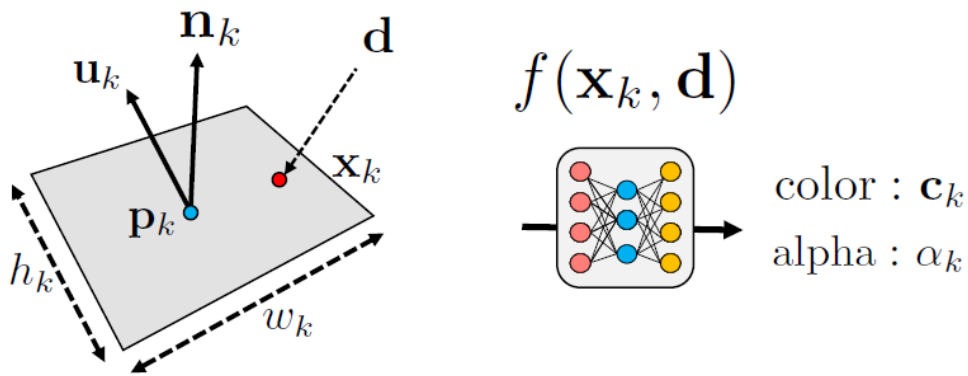


Slide credit: Zhi-Hao Lin



Slide credit: Zhi-Hao Lin

Method: model



- Position $\boldsymbol{p}_k \in R^3$
- orientation $\boldsymbol{n}_k, \boldsymbol{u}_k \in R^3$
- Size (w_k, h_k)

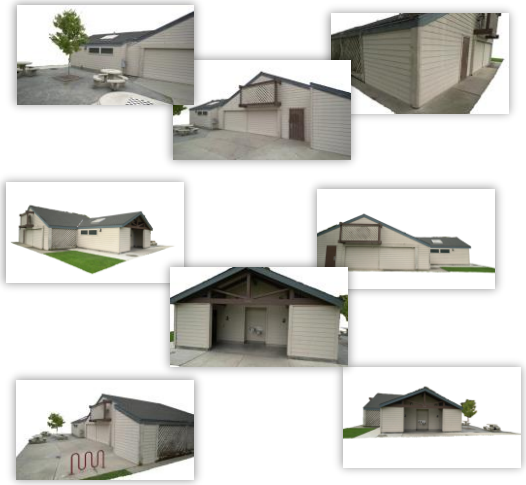
Neural radiance field network

- Input: position, direction
- Output: color, transparency

Slide credit: Zhi-Hao Lin

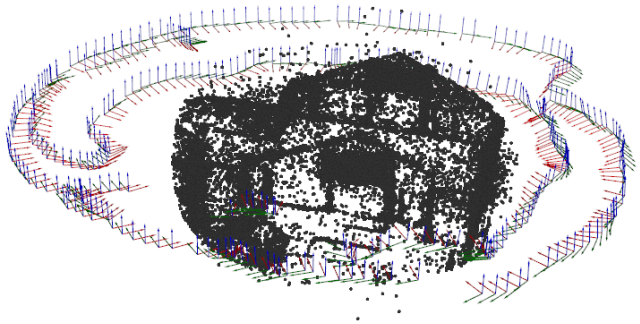
Method: initialization

- extract 3D point cloud from multiview images with COLMAP
- Initialize plane position, orientation on points



Multiview images

COLMAP



3D point cloud

Slide credit: Zhi-Hao Lin

Method: training

For training views, compute and optimize

- Geometry loss:

$$\mathcal{L}_g = \underbrace{\sum_i \min_k d(\mathbf{x}_i, \mathbf{s}_k)}_{\text{Point-rectangle distance}} + \lambda \underbrace{\sum_k (w_k h_k)^2}_{\text{Rectangle area}}$$

- Color loss:

$$\mathcal{L}_c = \frac{1}{B} \sum_{\mathbf{r}} \|c(\mathbf{r}) - c_{\text{gt}}(\mathbf{r})\|_2^2.$$

- Total:

$$\mathcal{L}_{total} = \mathcal{L}_g + \mathcal{L}_c$$

	NeX	NeRF	PlenOctree*	KiloNeRF*	Ours
# Params (M)	21.28	1.19	1457.2	6.21	3.11
FPS	0.142	0.106	78.04	4.19	19.16

Table 7. **Model size and inference speed on Replica.**

Slide credit: Zhi-Hao Lin

NeurMiPs: Neural Mixture of Planar Experts for View Synthesis (CVPR 2022)

Result

Slide credit: Zhi-Hao Lin

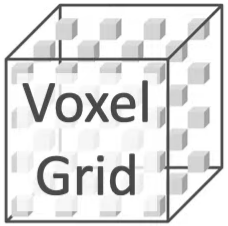
Direct Voxel Grid Optimization: Super-fast Convergence for Radiance Fields Reconstruction

Cheng Sun, Min Sun, Hwann-Tzong Chen

CVPR 2022

Speed up

- **Replacing the** 

with 

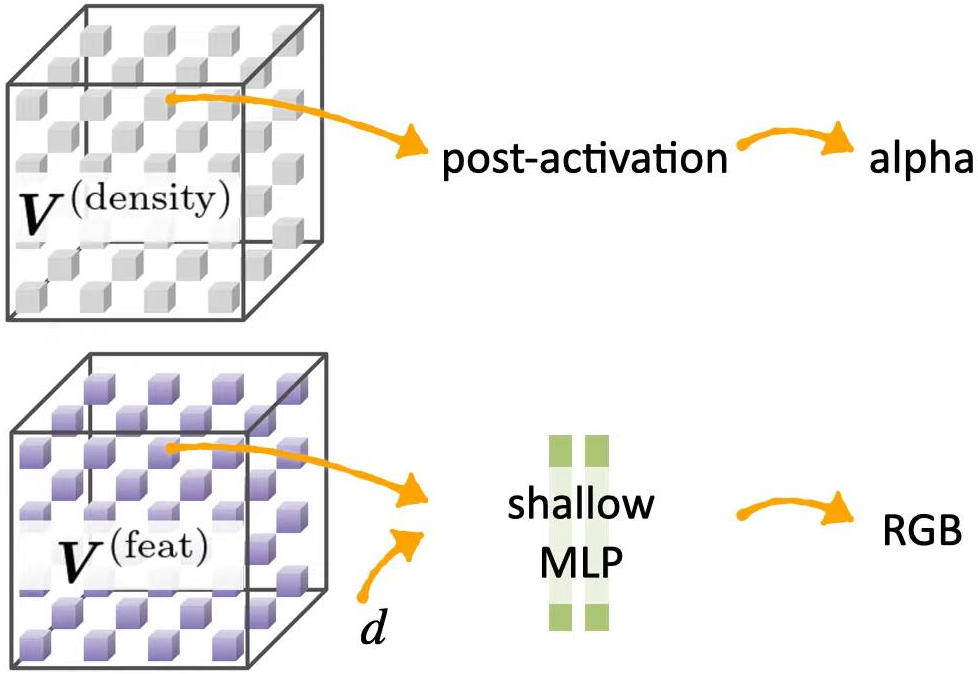
.

- 😀 Previous works[1,2,3] have shown good quality with a large inference time speed up.
- 😕 Limited to inference time. Pre-trained MLP is required.
- 🤔 How to train voxel grid directly from scratch?

[1] PlenOctrees for Real-time Rendering of Neural Radiance Fields, Yu et al.
[2] FastNeRF: High-Fidelity Neural Rendering at 200FPS, Garbin et al.
[3] Baking Neural Radiance Fields for Real-Time View Synthesis, Hedman et al

Slide credit: Cheng Sun

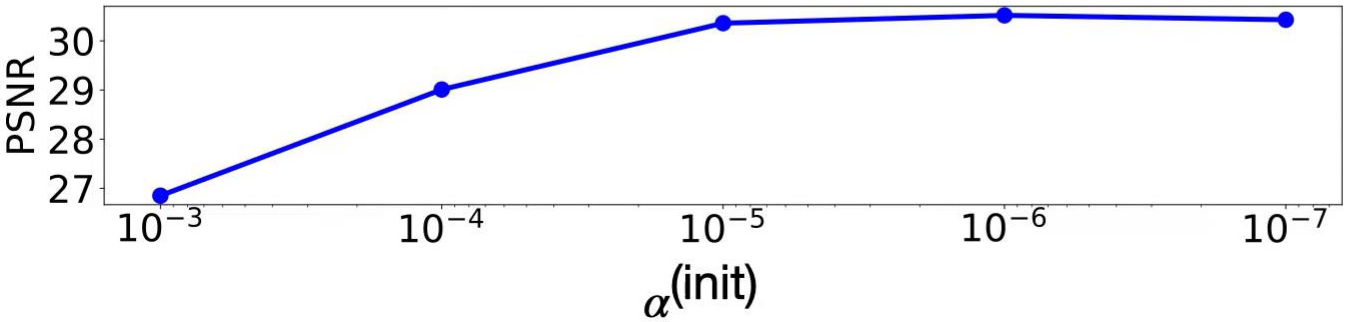
Our volumetric scene representation

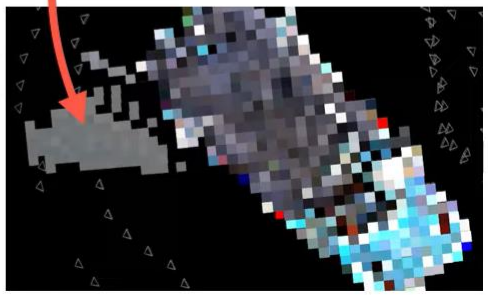


Slide credit: Cheng Sun

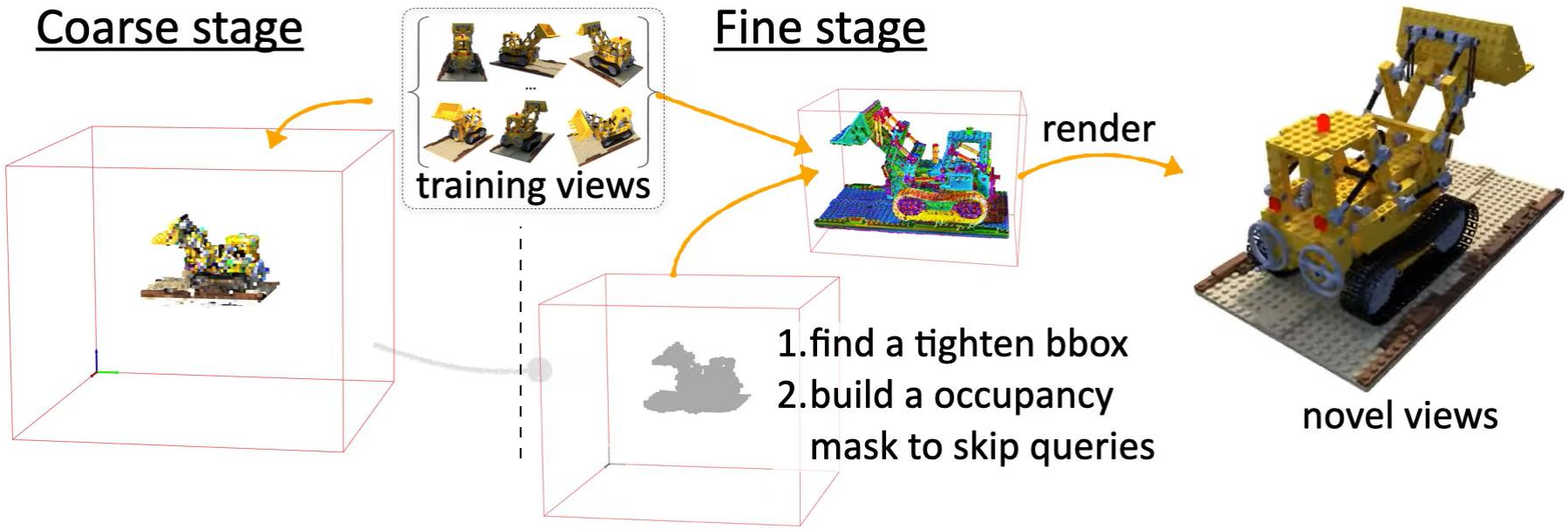
Low-density initialization

- **The initial alpha values (activated from volume density) should close to 0.** We introduce a hyperparameter $\alpha^{(\text{init})}$ to control it.
- $\alpha^{(\text{init})}$ should be small enough to achieve good results and avoid **floater**.





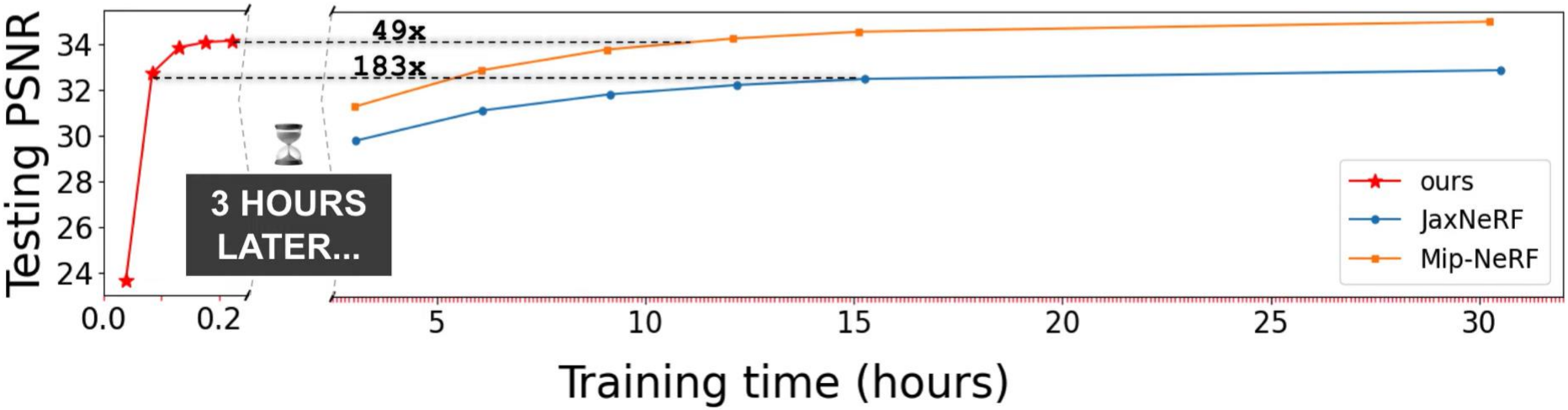
Coarse-to-fine reconstruction



Slide credit: Cheng Sun

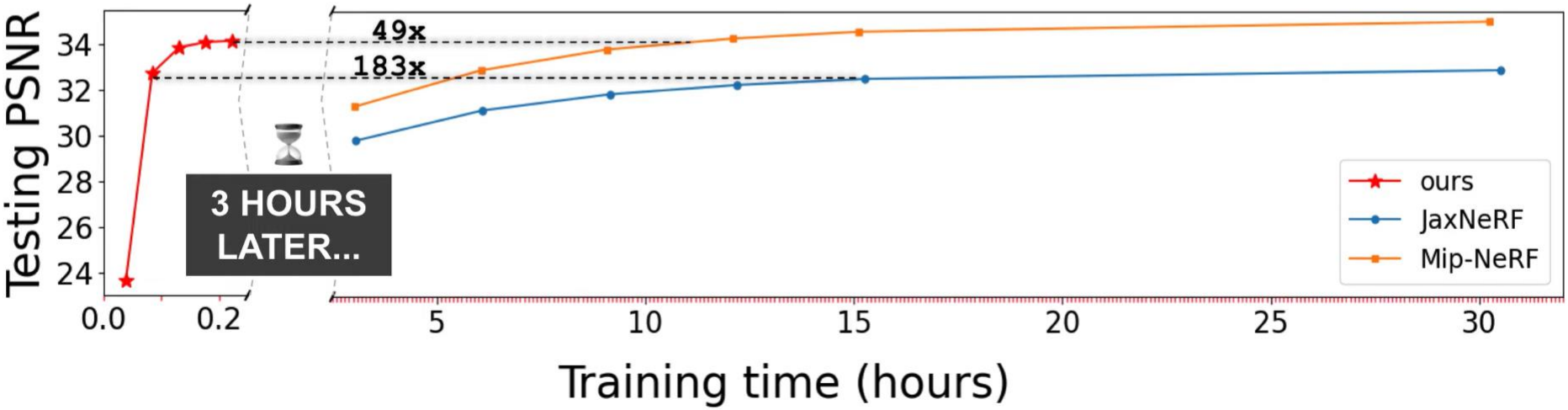
Results

Our code



Slide credit: Cheng Sun

Results



Slide credit: Cheng Sun

Concurrent related work

- **Advanced data structure**
  Octree—Plenoxels: Radiance Fields without Neural Networks
  Hash—Instant Neural Graphics Primitives with a Multiresolution Hash Encoding
  Factorized components—TensoRF: Tensorial Radiance Fields

Slide credit: Cheng Sun

More references about NeRF improvements

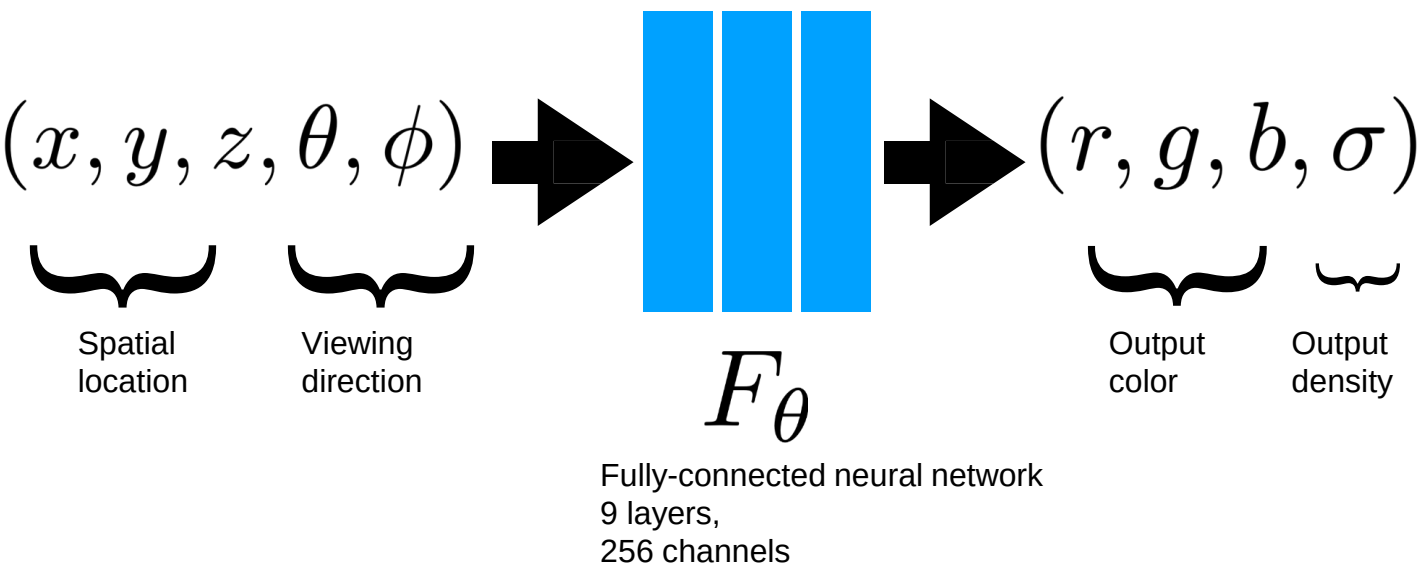
- Fourier Features Let Networks Learn High Frequency Functions in Low Dimensional Domains (NeurIPS 2020) -> explain why positional encoding works
- PlenOctrees for Real-time Rendering of Neural Radiance Fields (ICCV 2021)
- Mip-NeRF: A Multiscale Representation for Anti-Aliasing Neural Radiance Fields (ICCV 2021)
- KiloNeRF: Speeding up Neural Radiance Fields with Thousands of Tiny MLPs (ICCV 2021)
- Plenoxels : Radiance Fields without Neural Networks (CVPR 2022)
- Instant Neural Graphics Primitives with a Multiresolution Hash Encoding (SIGGRAPH 2022)
- TensoRF: Tensorial Radiance Fields (ECCV 2022)

What to Cover Today?

- Introduction to 3D Vision
- Part I: 3D Perception
- Part II: 3D Reconstruction
- Neural Radiance Fields
  - Extension of NeRF
    - Sample efficiency: NeurMiPs
    - Feature acquired efficiency: DVGO
  - Advanced Topics of NeRF
    - NeRF + generalized
    - NeRF + semantic
    - NeRF + text guided generation
    - When NeRF + splatting

Generalized:
Why NeRF has to overfit on one scene

- It works as a **<u>representation</u>**
- No scene specific information as input, model will not know what's the difference.



pixelNeRF: Neural Radiance Fields from One or Few Images

CVPR 2021

Alex Yu Vickie Ye Matthew Tancik Angjoo Kanazawa

UC Berkeley

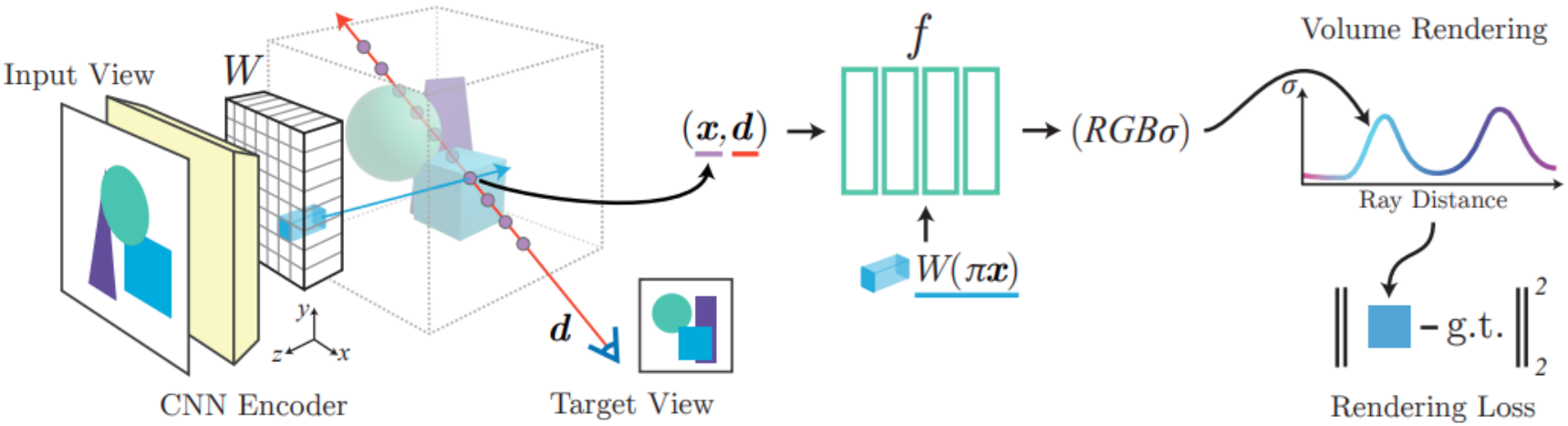
Three view image

pixelNeRF NeRF

https://alexyu.net/pixelnerf/

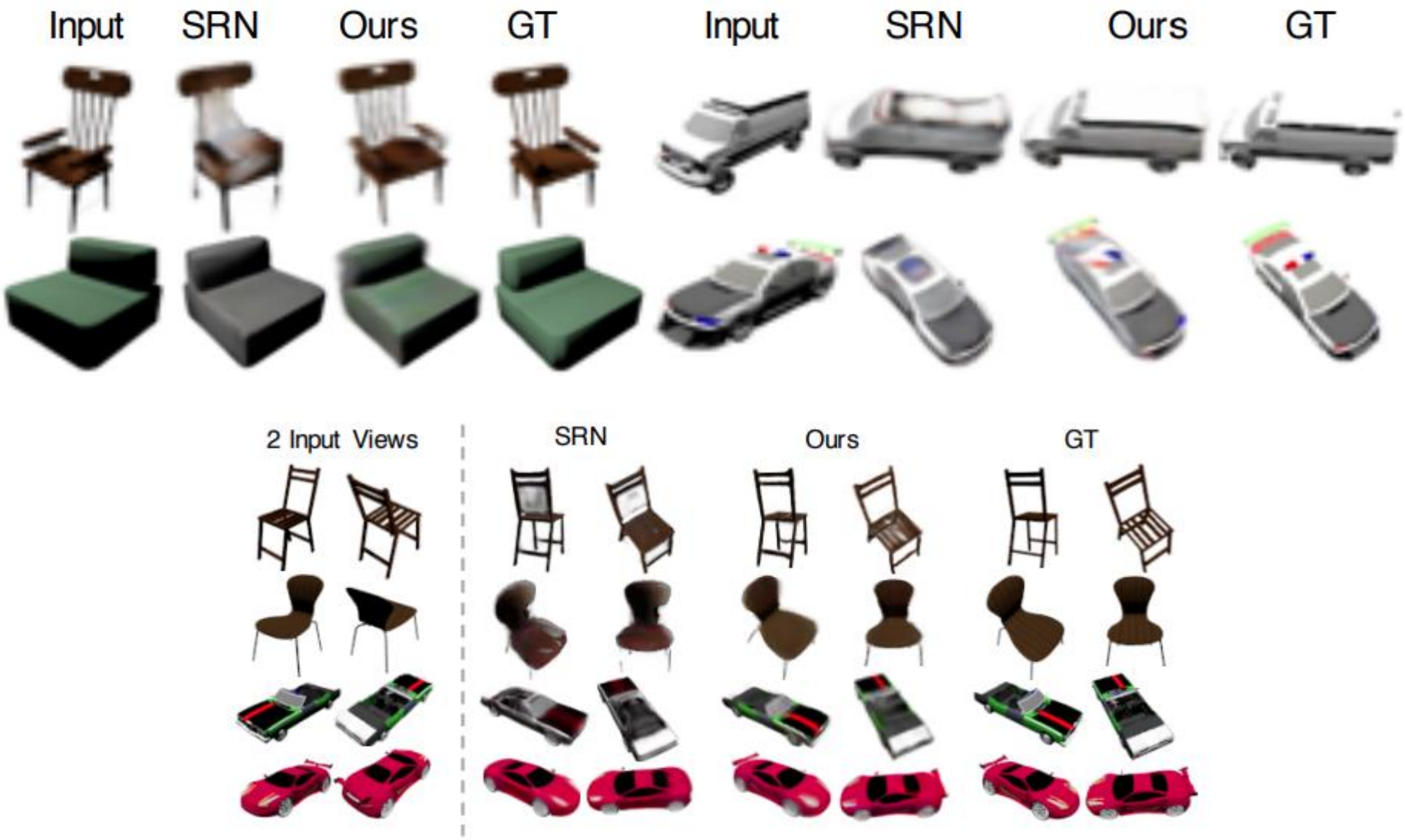
Method

- Image feature as condition of NeRF
- The NeRF itself learns a object prior (e.g., what a general car/chair look like)
- Able to fit different object/scene with only **one** NeRF model
- Only need one image (for encoding image feature) of the scene during testing on a new scene
- Idea: **Instead of storing scene information in the neural network, the neural network synthesize novel view image based on the features of a scene.**



pixelNeRF: Neural Radiance Fields from One or Few Images (CVPR 2021)

Results--single category



pixelNeRF: Neural Radiance Fields from One or Few Images (CVPR 2021)

Results--multi-category

In-Place Scene Labelling and Understanding with Implicit Scene Representation

Shuaifeng Zhi, Tristan Laidlow, Stefan Leutenegger, Andrew Davison

Dyson Robotics Laboratory, Imperial College London

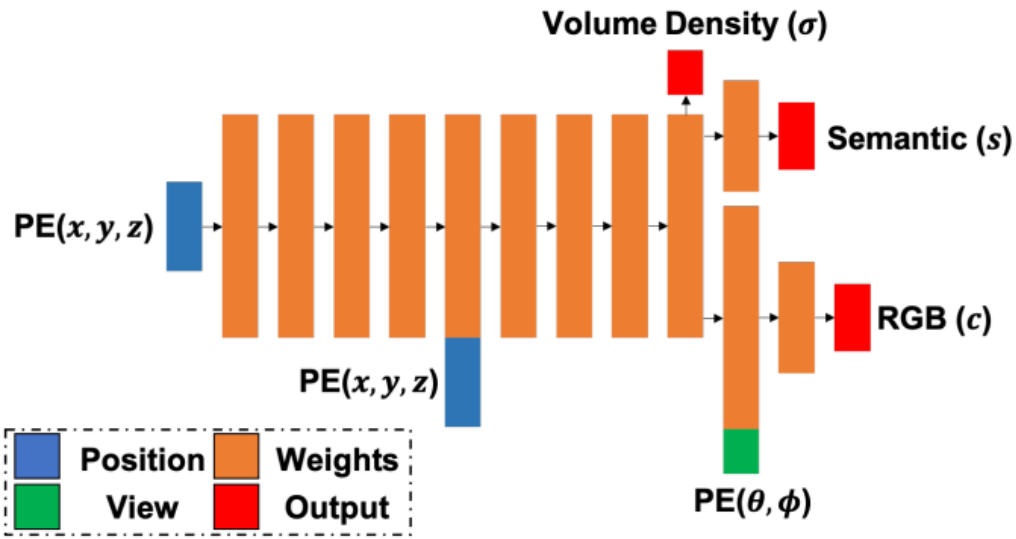
2021 ICCV

Ground Truth

Rendering

Entropy

Method



- Add semantic head after density prediction / before view direction input
- Cross entropy loss

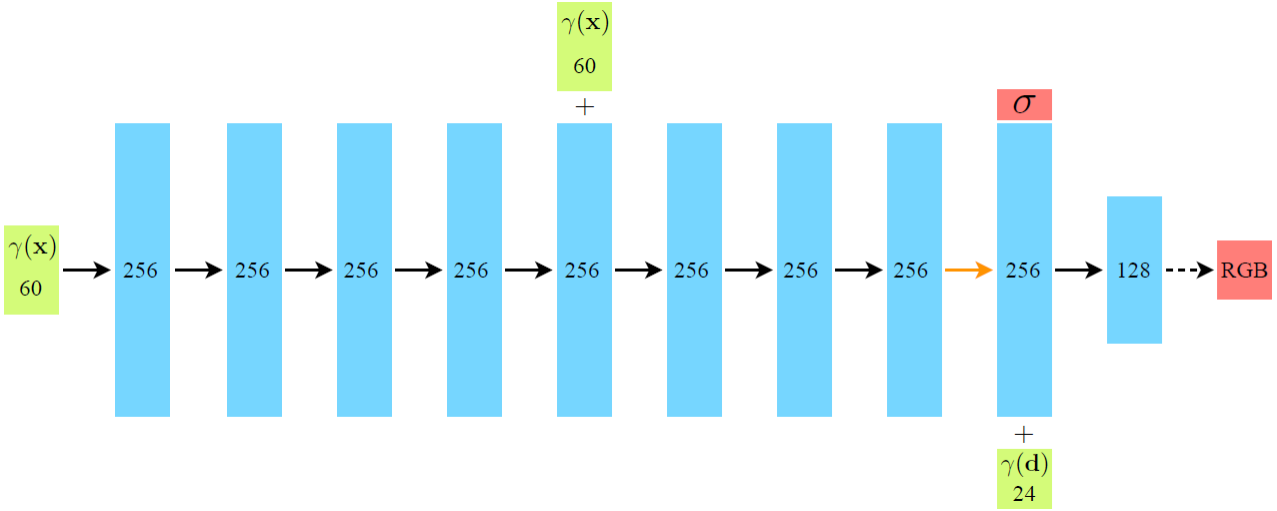
$$\hat{\mathbf{C}}(\mathbf{r}) = \sum_{k=1}^{K} \hat{T}(t_k)\, \alpha\left(\sigma(t_k)\delta_k\right) \mathbf{c}(t_k)$$

$$\hat{\mathbf{S}}(\mathbf{r}) = \sum_{k=1}^{K} \hat{T}(t_k)\, \alpha\left(\sigma(t_k)\delta_k\right) \mathbf{s}(t_k)$$

$$L_p = \sum_{\mathbf{r}\in\mathcal{R}} \left[\left\| \hat{\mathbf{C}}_c(\mathbf{r}) - \mathbf{C}(\mathbf{r}) \right\|_2^2 + \left\| \hat{\mathbf{C}}_f(\mathbf{r}) - \mathbf{C}(\mathbf{r}) \right\|_2^2 \right],$$

$$L_s = -\sum_{\mathbf{r}\in\mathcal{R}} \left[\sum_{l=1}^{L} p^l(\mathbf{r}) \log \hat{p}_c^l(\mathbf{r}) + \sum_{l=1}^{L} p^l(\mathbf{r}) \log \hat{p}_f^l(\mathbf{r}) \right]$$

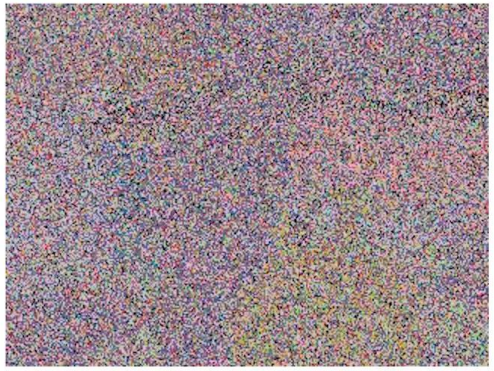
Method

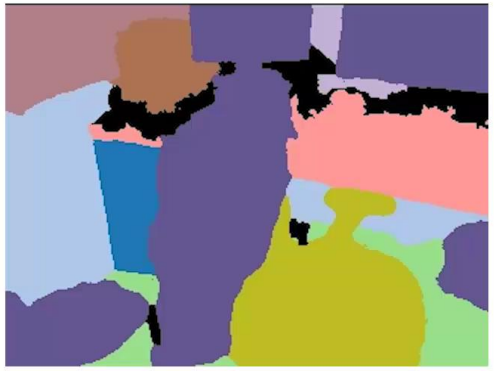


- Original NeRF

Results

- Label denoising
  - Pixel-wise noise (for fun)





- Label propagation
  - Label super resolution

DREAMFUSION: TEXT-TO-3D USING 2D DIFFUSION

Ben Poole
Google Research

Ajay Jain
UC Berkeley

Jonathan T. Barron
Google Research

Ben Mildenhall
Google Research

2023 ICLR

https://dreamfusion3d.github.io/

Goal

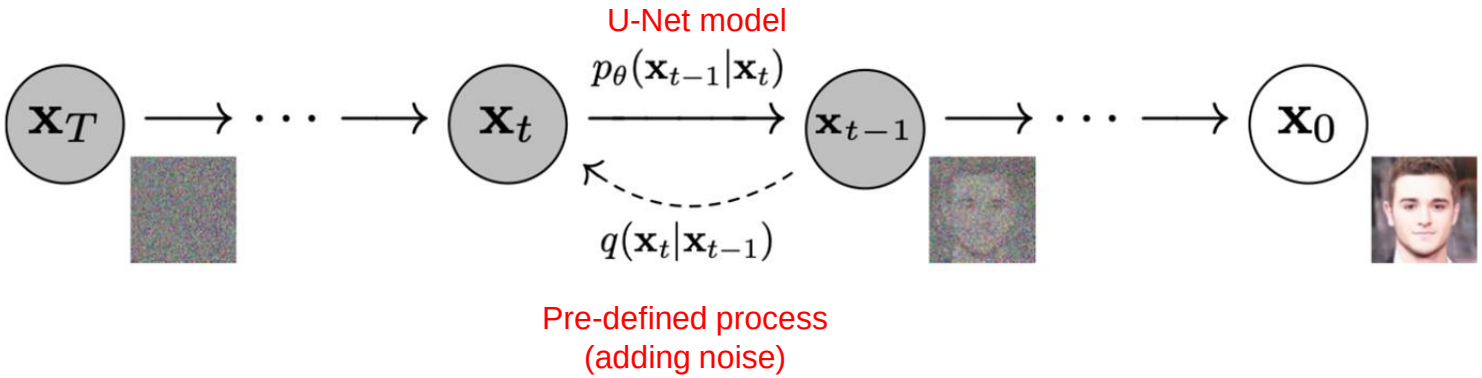
an orangutan making a clay bowl on a throwing wheel*

a raccoon astronaut holding his helmet†

a blue jay standing on a large basket of rainbow macarons*

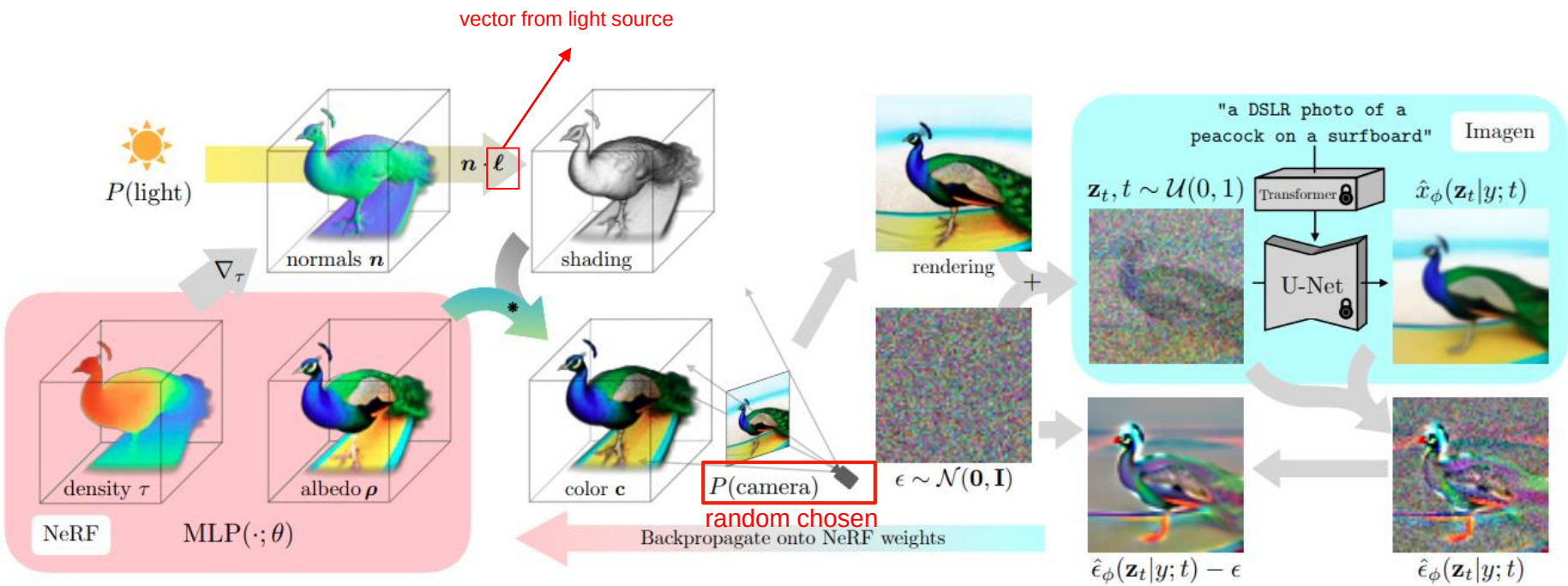
- Take description as input and generate corresponding 3D results (via 2D rendering)
- Without paired “text and 3D object”
- Combining NeRF and 2D text-to-image diffusion model

Recap: Diffusion model (intuitively)



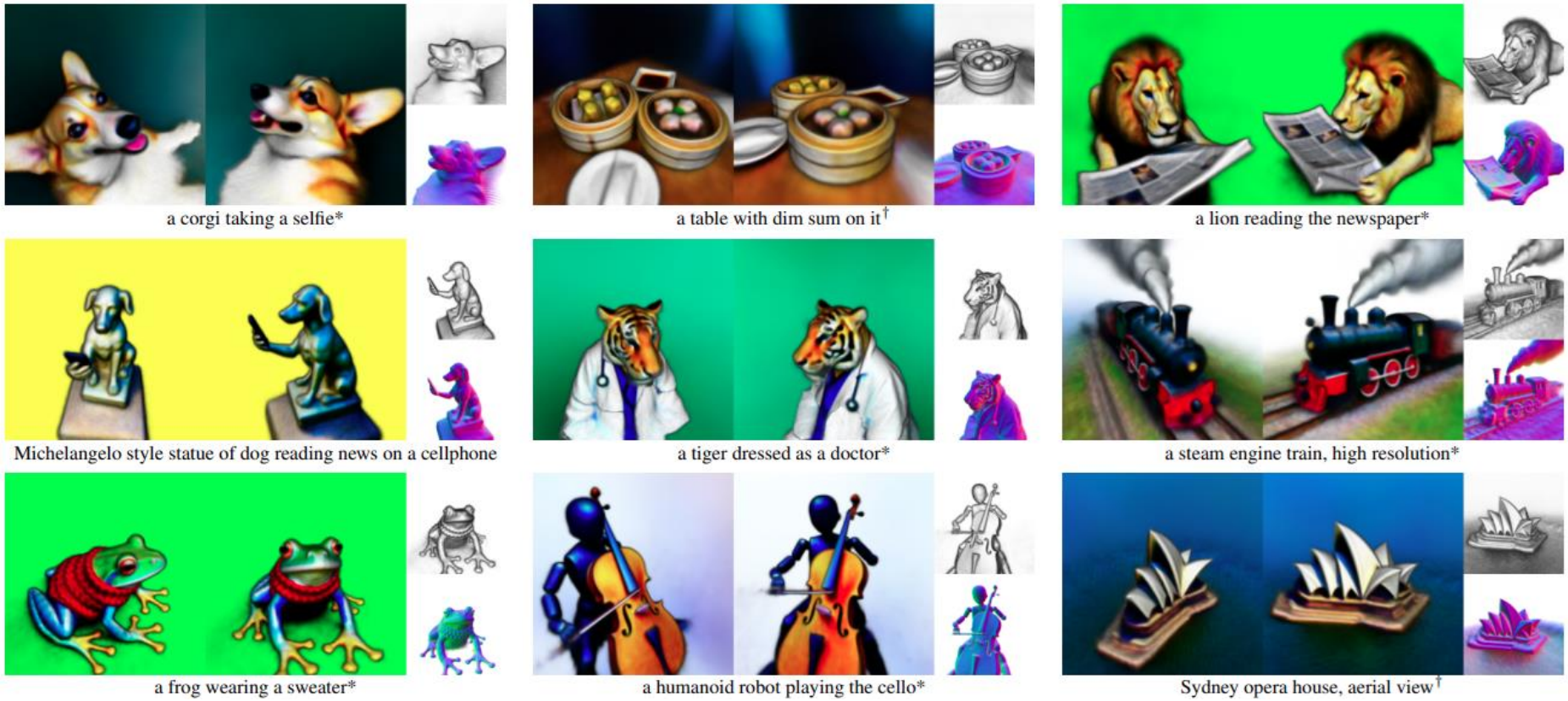
- Can be viewed as denoising from a Gaussian noise image
- Each step makes little progress of denoising (total about 1000 steps)
- Output image of each step can be seen as the **original image** combining with a **noise** using specific ratio
- The process can also be seen as predicting the **added noise**

Method



- The left part is a standard NeRF with shading condition
- Combine the rendered NeRF image with random noise to simulate a state of the text-to-image diffusion model
- The difference between the predicted noise and the inserted noise is treated as the rendering loss to guide NeRF

Result



DREAMFUSION: TEXT-TO-3D USING 2D DIFFUSION

Result

an all-utility vehicle driving across a stream†

a chimpanzee dressed like Henry VIII king of England*

a baby bunny sitting on top of a stack of pancakes†

a sliced loaf of fresh bread

a bulldozer clearing away a pile of snow*

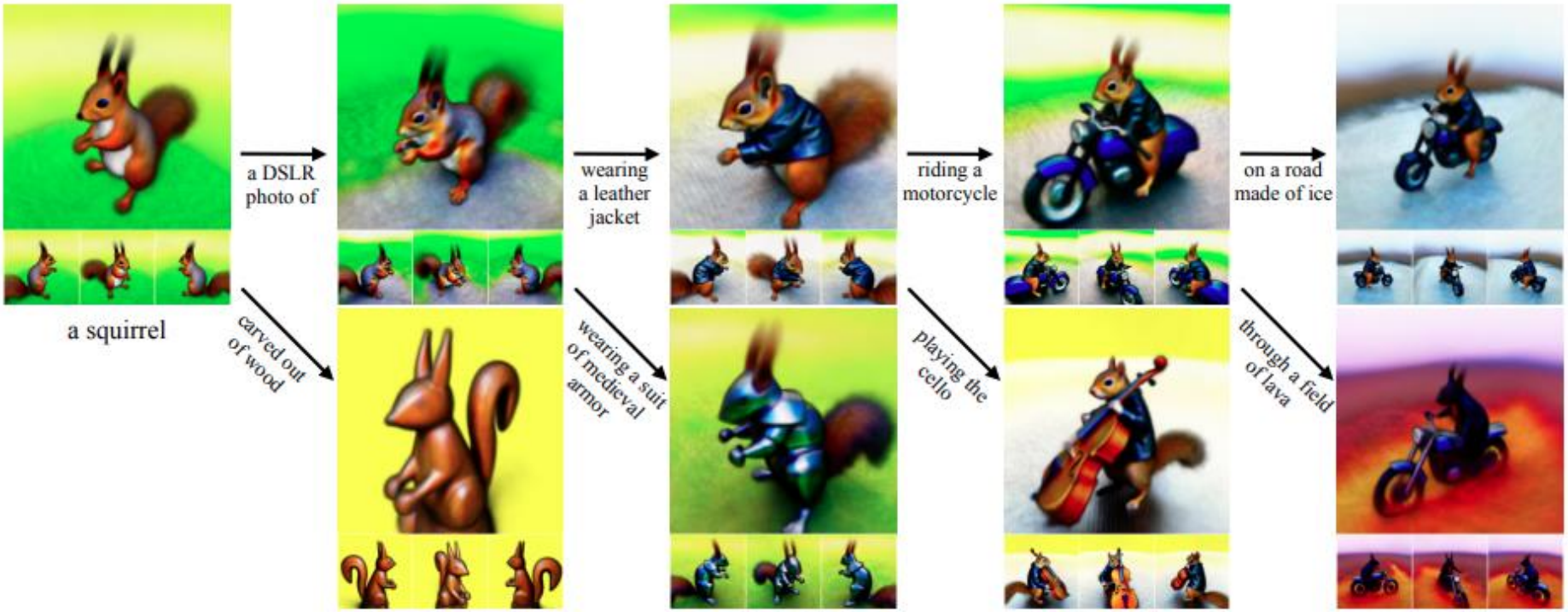
a classic Packard car*

zoomed out view of Tower Bridge made out of gingerbread and candy‡

a robot and dinosaur playing chess, high resolution*

a squirrel gesturing in front of an easel showing colorful pie charts

Result


a squirrel
a DSLR photo of
carved out of wood
wearing a leather jacket
wearing a suit of medieval armor
riding a motorcycle
playing the cello
on a road made of ice
through a field of lava

3D Gaussian Splatting for Real-Time Radiance Field Rendering

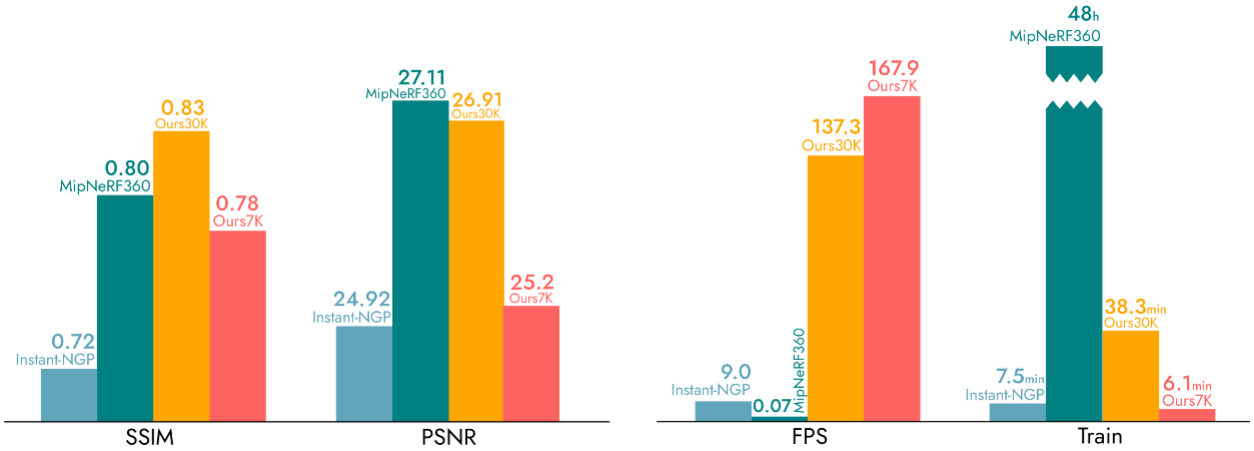
SIGGRAPH 2023

(ACM Transactions on Graphics)

Bernhard Kerbl* [1,2] Georgios Kopanas* [1,2] Thomas Leimkühler[3] George Drettakis[1,2]

* Denotes equal contribution

[1]Inria [2]Université Côte d'Azur [3]MPI Informatik



3D Gaussian Splatting for Real-Time Radiance Field Rendering Link

Method – When old meets new

NeRF | Gaussian Splatting

Slide credit: AI 甘安捏

3D Gaussian Splatting for Real-Time Radiance Field Rendering Link

Method – When old meets new

NeRF	Gaussian Splatting
Ray Tracing	**Rasterization**
＋ Quality 上限高 (因為基本跟人看東西道理一樣)	＋ 快
＋ 真實	＋ 預覽、視覺化簡單
＋ 反射、折射、陰影、材質	＋ 簡單
－ 慢	－ 真實
－ 難以預覽	－ 反射、折射 、陰影、材質
－ 複雜	

Slide credit: AI 甘安捏

3D Gaussian Splatting for Real-Time Radiance Field Rendering Link

Method – When old meets new

- Structure from Motion:
  - from multi-view image to sparse point cloud

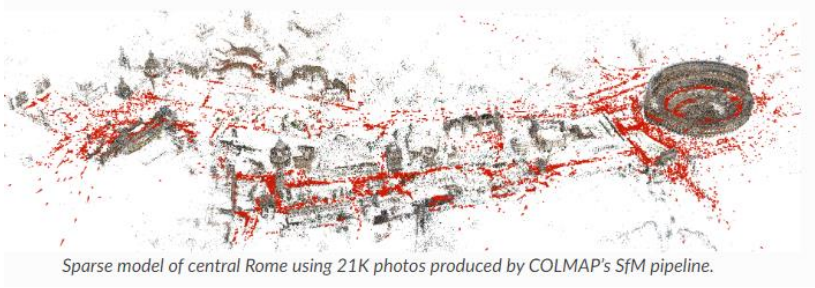
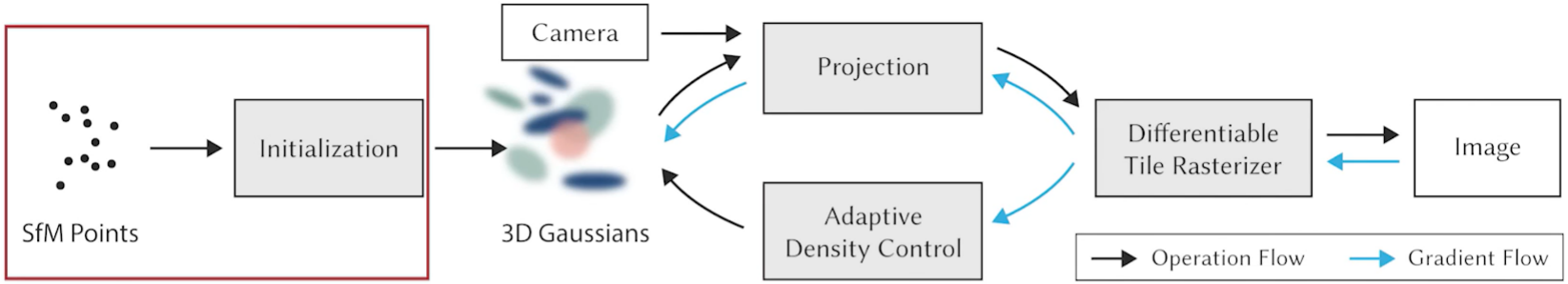


Photo credit: colmap

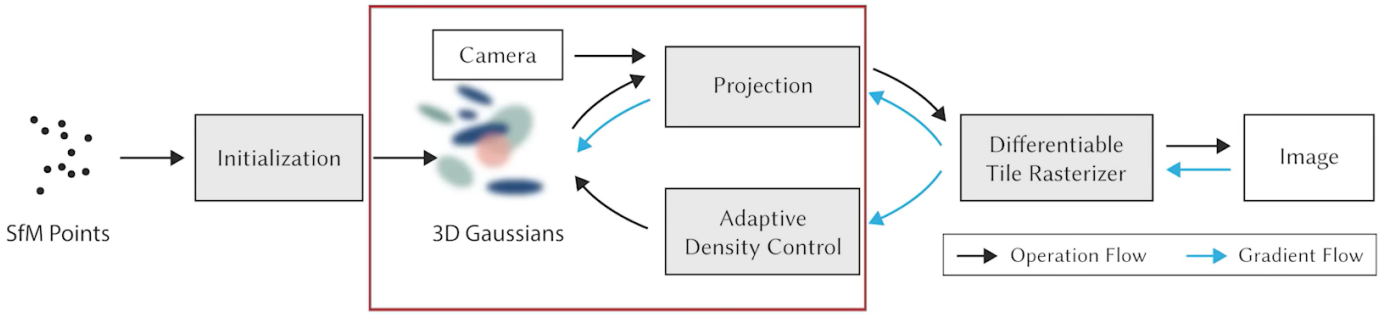


We start with a set of cameras and a point cloud provided by Structure from Motion during calibration.

3D Gaussian Splatting for Real-Time Radiance Field Rendering Link

Method

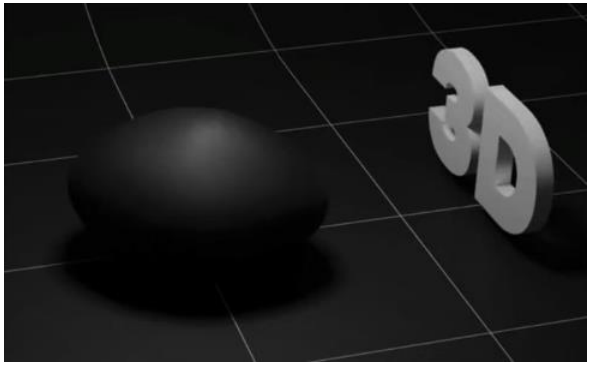
Overview of our Method



Next, we optimize the set of 3D Gaussians to represent the scene

- 3D Gaussians composed of four parameters:
  - position (x, y, z),
  - covariance (how it's stretched/scaled: 3x3),
  - color (RGB)
  - alpha (density)

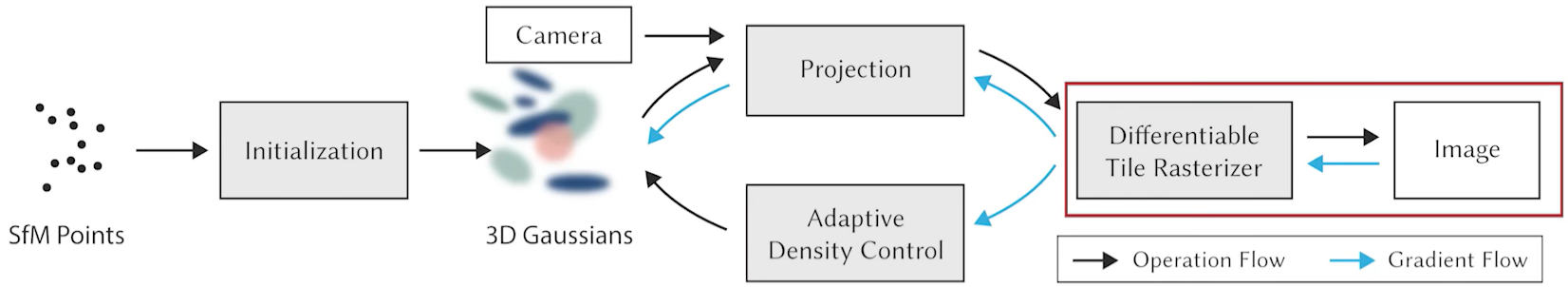
  to represent a 3D Gaussian



3D Gaussian Splatting for Real-Time Radiance Field Rendering Link

Method

Overview of our Method



Finally, we render transparent anisotropic Gaussians and backpropagate the gradients to their properties.

3D Gaussian Splatting for Real-Time Radiance Field Rendering Link

Result

- The render results

3D Gaussian Splatting for Real-Time Radiance Field Rendering Link

Result

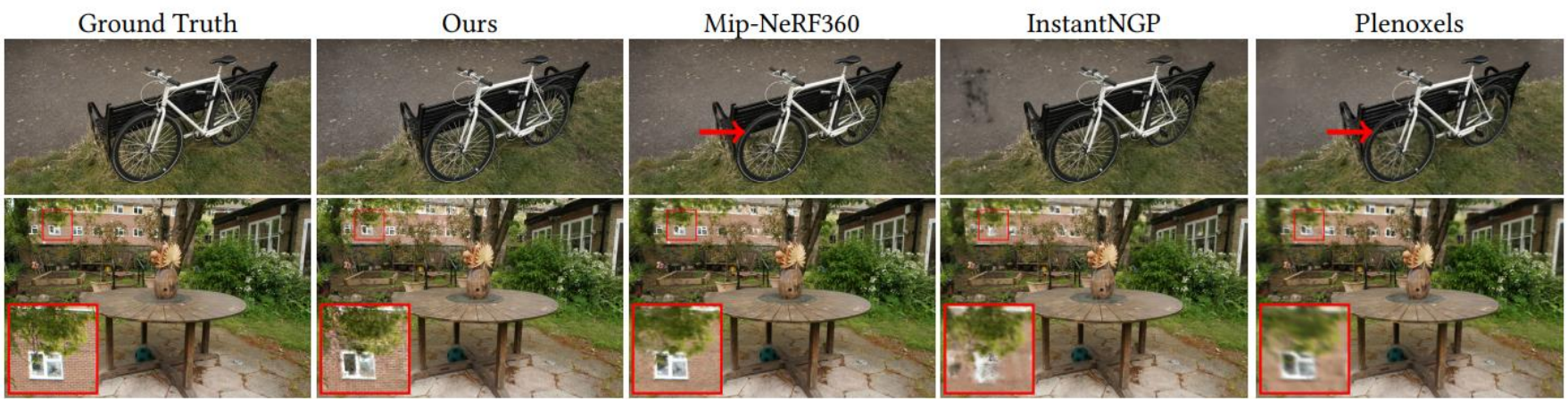
- The render results if we set all 3D gaussians' alpha to 1, without transparency.

3D Gaussian Splatting for Real-Time Radiance Field Rendering Link

Result

- Better visual quality with order of magnitude rendering speed difference.

Dataset	Mip-NeRF360						Tanks&Temples						Deep Blending					
Method|Metric	*SSIM*↑	*PSNR*↑	*LPIPS*↓	Train	FPS	Mem	*SSIM*↑	*PSNR*↑	*LPIPS*↓	Train	FPS	Mem	*SSIM*↑	*PSNR*↑	*LPIPS*↓	Train	FPS	Mem
Plenoxels	0.626	23.08	0.463	25m49s	6.79	2.1GB	0.719	21.08	0.379	25m5s	13.0	2.3GB	0.795	23.06	0.510	27m49s	11.2	2.7GB
INGP-Base	0.671	25.30	0.371	5m37s	11.7	13MB	0.723	21.72	0.330	5m26s	17.1	13MB	0.797	23.62	0.423	6m31s	3.26	13MB
INGP-Big	0.699	25.59	0.331	7m30s	9.43	48MB	0.745	21.92	0.305	6m59s	14.4	48MB	0.817	24.96	0.390	8m	2.79	48MB
M-NeRF360	0.792†	27.69†	0.237†	48h	0.06	8.6MB	0.759	22.22	0.257	48h	0.14	8.6MB	0.901	29.40	0.245	48h	0.09	8.6MB
Ours-7K	0.770	25.60	0.279	6m25s	160	523MB	0.767	21.20	0.280	6m55s	197	270MB	0.875	27.78	0.317	4m35s	172	386MB
Ours-30K	0.815	27.21	0.214	41m33s	134	734MB	0.841	23.14	0.183	26m54s	154	411MB	0.903	29.41	0.243	36m2s	137	676MB



3D Gaussian Splatting for Real-Time Radiance Field Rendering Link

More references about further topics of NeRF

- Editing Conditional Radiance Fields (EditNeRF)(ICCV 2021)
- pi-GAN: Periodic Implicit Generative Adversarial Networks for 3D-Aware Image Synthesis (CVPR 2021)
- FENeRF: Face Editing in Neural Radiance Fields (CVPR 2022)
- StyleNeRF: A Style-based 3D-Aware Generator for High-resolution Image Synthesis (ICLR 2022)
- MVSNeRF: Fast Generalizable Radiance Field Reconstruction from Multi-View Stereo (ICCV 2021)
- Panoptic Lifting for 3D Scene Understanding with Neural Fields (CVPR 2023)

What We've Covered Today?

- Introduction to 3D Vision
- Part I: 3D Perception
- Part II: 3D Reconstruction
- Neural Radiance Fields
  - Extension of NeRF
  - Advanced Topics of NeRF