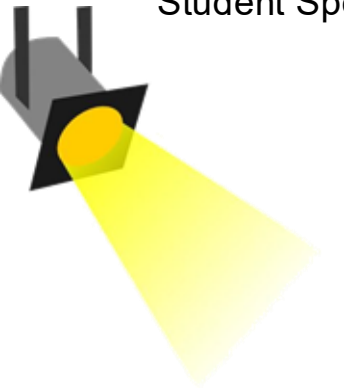
# Lecture 15: 3D Vision

## Administrative Announcements

- Milestone 2 deadline is tomorrow (May 22)! Make sure to book an appointment.



# Student Spotlights

Lectures:

- Mahda Soltani
- Ali Ahmad
- Luis
- Boyu

EdStem:

- Jeremy Hsieh
- Jessica Su
- Fisher Marks
- Adem Rimpapa
- Warren Chan
- Trang Vuong
- Ali Ahma

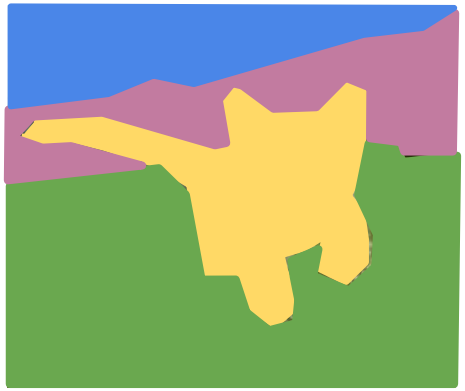
# **Recall**: 2D Detection and Segmentation

**Classification**

**CAT**

No spatial extent

**Semantic Segmentation**



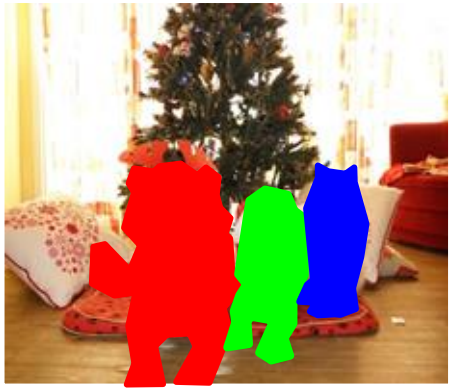
**GRASS**, **CAT**, **TREE**, **SKY**

No objects, just pixels

**Object Detection**

**DOG**, **DOG**, **CAT**

**Instance Segmentation**



**DOG**, **DOG**, **CAT**

Multiple Objects

# **Recall**: **Video** = 2D + Time

A video is a **sequence** of images
4D tensor: T x 3 x H x W
(or 3 x T x H x W)

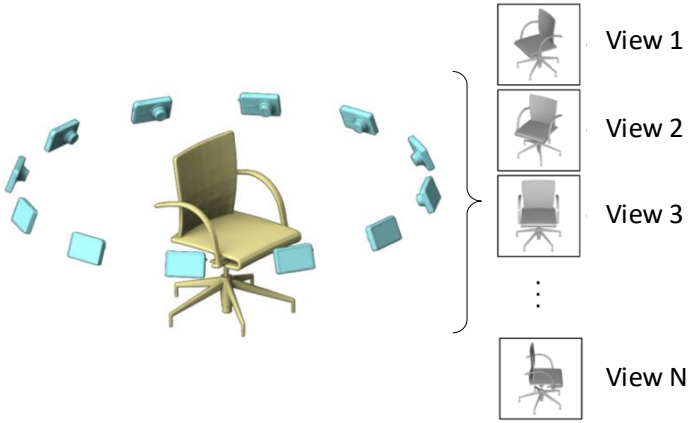
# Many topics in 3D Vision!

3D Representations
Computing Correspondences
Multi-view Stereo
Structure from Motion
3D Object Pose Estimation
Simultaneous Localization and Mapping (SLAM)
Differentiable Graphics
3D Sensors
……

# Multi-View CNN

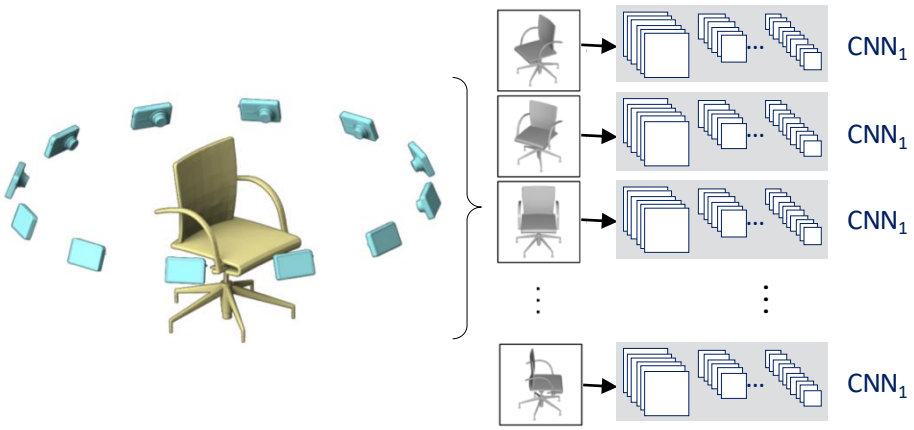
Su et al. ICCV 2015

# Multi-View CNN




Su et al. ICCV 2015

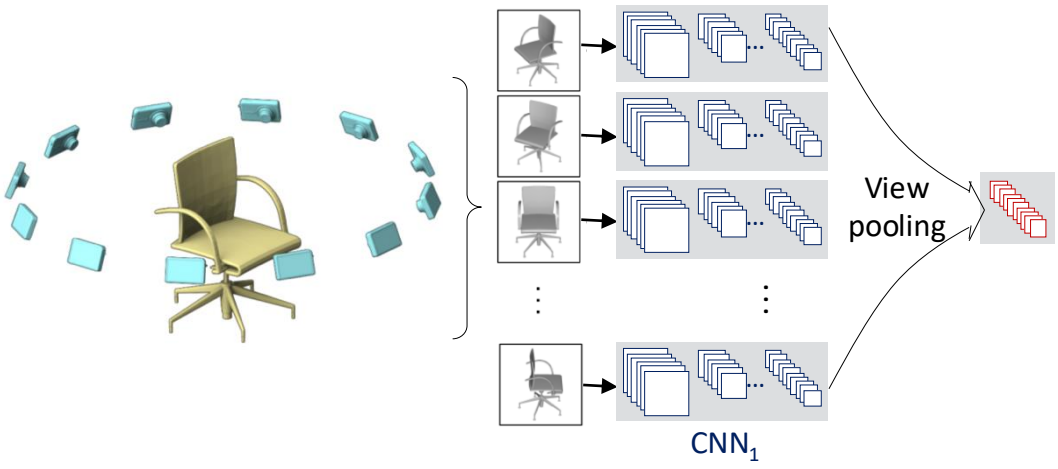
# Multi-View CNN




$CNN_1$: a ConvNet extracting image features

Su et al. ICCV 2015

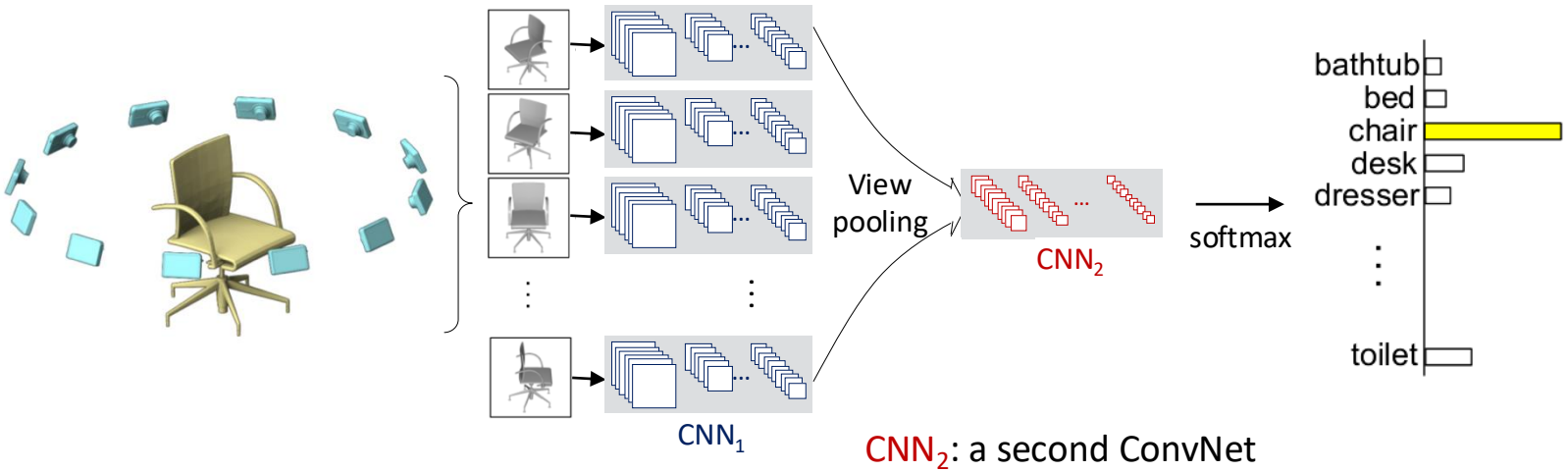
# Multi-View CNN




View pooling: element-wise max-pooling across all views

Su et al. ICCV 2015

# Multi-View CNN




$CNN_2$: a second ConvNet producing shape descriptors

Su et al. ICCV 2015

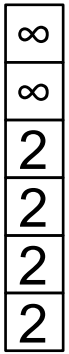
# Experiments – Classification & Retrieval

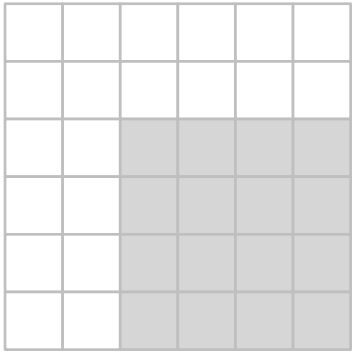
| | Method | Classification (Accuracy) | Retrieval (mAP) |
|---|---|---|---|
| Non-deep | SPH | 68.2% | 33.3% |
| | LFD | 75.5% | 40.9% |
| | 3D ShapeNets | 77.3% | 49.2% |
| | FV, 12 views | 84.8% | 43.9% |
| | CNN, 12 views | 88.6% | 62.8% |
| | MVCNN, 12 views | **89.9%** | 70.1% |
| | MVCNN+metric, 12 views | 89.5% | **80.2%** |
| | MVCNN, 80 views | 90.1% | 70.4% |
| | MVCNN+metric, 80 views | **90.1%** | **79.5%** |

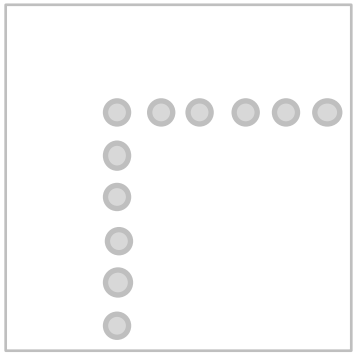
On ModelNet 40

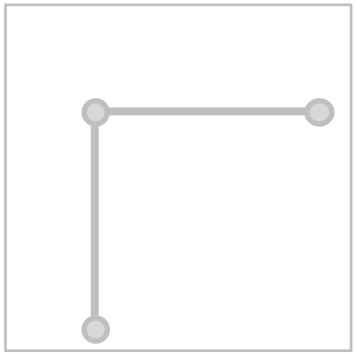
Su et al. ICCV 2015

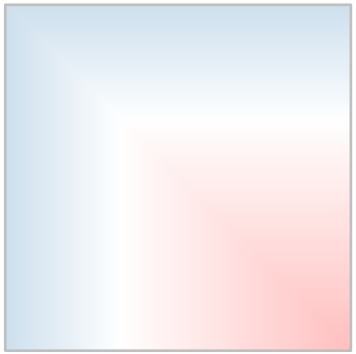
# 3D Shape Representations









Depth Map

Voxel Grid

Pointcloud

Mesh

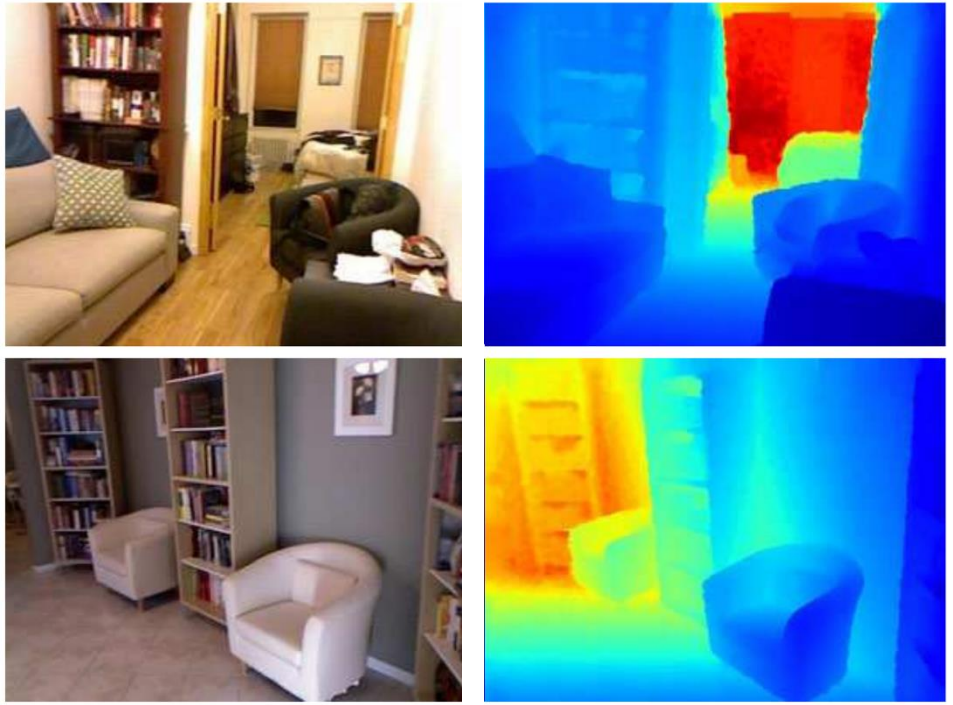
Implicit Surface

# 3D Shape Representations: Depth Map

For each pixel, **depth map** gives distance from the camera to the object in the world at that pixel

RGB image + Depth image
= RGB-D Image (2.5D)

This type of data can be recorded directly for some types of 3D sensors (e.g. Intel Realsense, Microsoft Kinect)

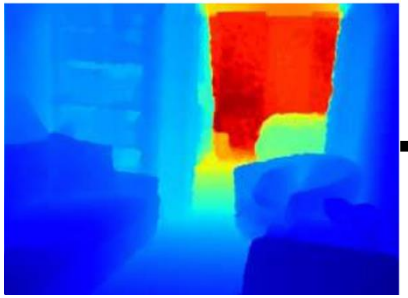


RGB Image: 3 x H x W  Depth Map: H x W

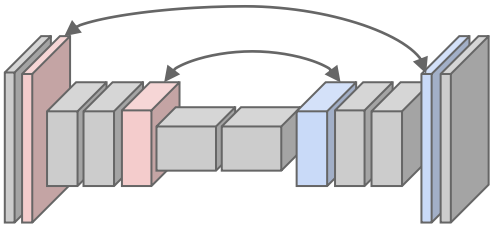
Eigen and Fergus, "Predicting Depth, Surface Normals and Semantic Labels with a Common Multi-Scale Convolutional Architecture", ICCV 2015

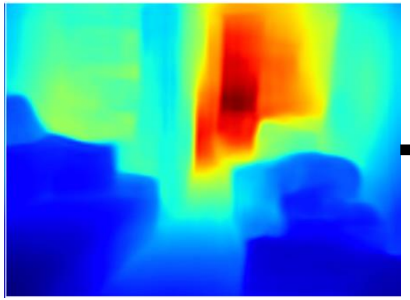
# Predicting Depth Maps

**Ground-truth Depth:**
1 x H x W



**Per-Pixel Loss**
(L2 Distance)





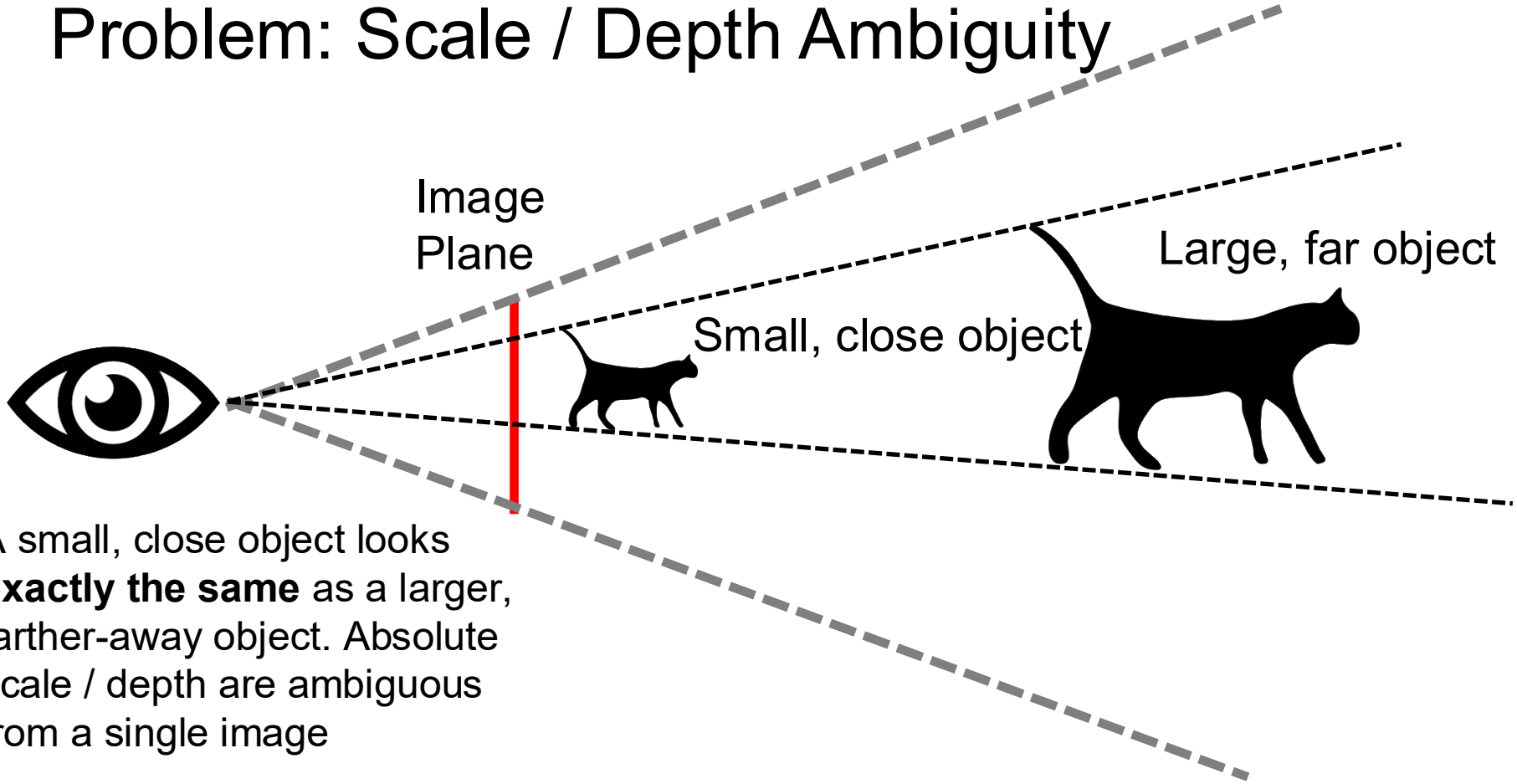
**RGB Input Image:**
3 x H x W

**Fully Convolutional network**

**Predicted Depth Image:**
1 x H x W

Eigen, Puhrsh, and Fergus, "Depth Map Prediction from a Single Image using a Multi-Scale Deep Network", NeurIPS 2014
Eigen and Fergus, "Predicting Depth, Surface Normals and Semantic Labels with a Common Multi-Scale Convolutional Architecture", ICCV 2015

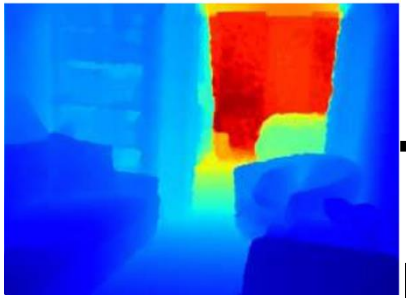
# Problem: Scale / Depth Ambiguity




A small, close object looks **exactly the same** as a larger, farther-away object. Absolute scale / depth are ambiguous from a single image

# Predicting Depth Maps

**Ground-truth Depth:**
1 x H x W

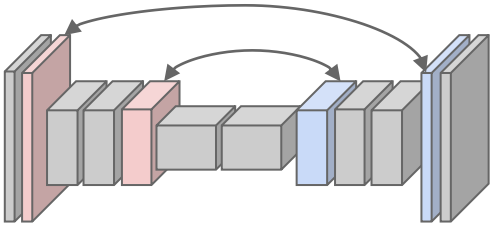


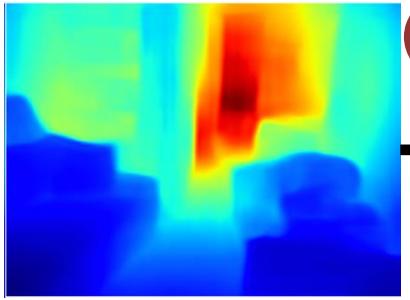
**Scale invariant loss**

$$D(y, y^*) = \frac{1}{2n^2} \sum_{i,j} \left( (\log y_i - \log y_j) - (\log y_i^* - \log y_j^*) \right)^2$$

$$= \frac{1}{n} \sum_i d_i^2 - \frac{1}{n^2} \sum_{i,j} d_i d_j = \frac{1}{n} \sum_i d_i^2 - \frac{1}{n^2} \left( \sum_i d_i \right)^2$$

**Per-Pixel Loss**
**(Scale invariant)**





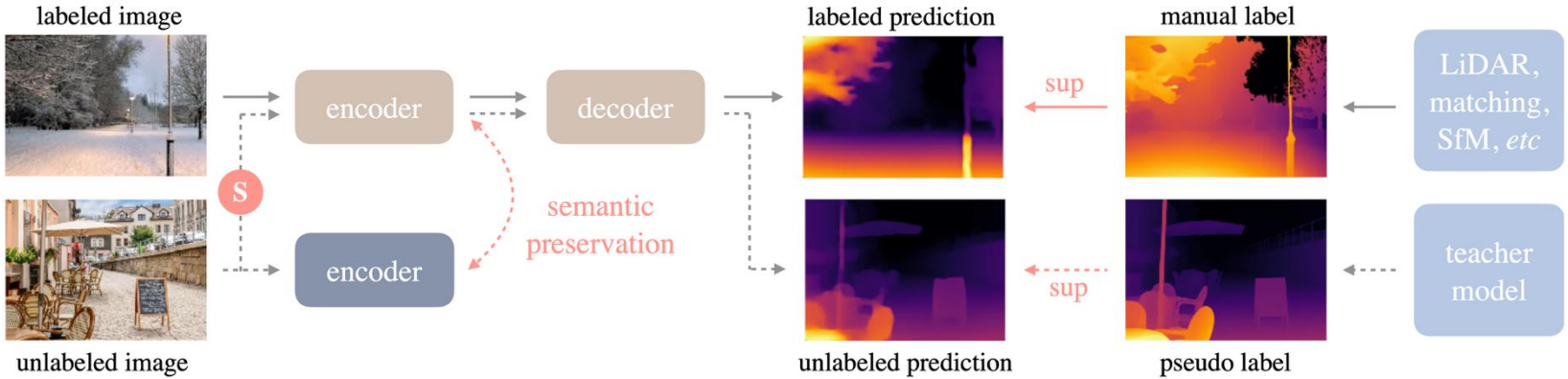
**RGB Input Image:**
3 x H x W

**Fully Convolutional network**

**Predicted Depth Image:**
1 x H x W

Eigen, Puhrsh, and Fergus, "Depth Map Prediction from a Single Image using a Multi-Scale Deep Network", NeurIPS 2014
Eigen and Fergus, "Predicting Depth, Surface Normals and Semantic Labels with a Common Multi-Scale Convolutional Architecture", ICCV 2015

# Modern Depth Prediction Models: DepthAnything Series



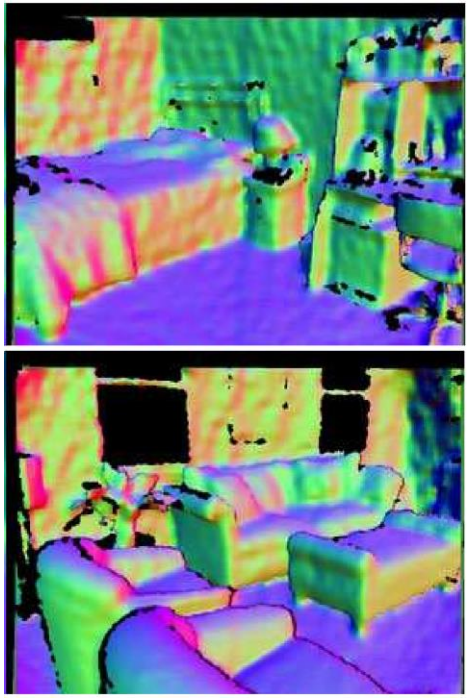

Yang, Lihe, et al. "Depth anything: Unleashing the power of large-scale unlabeled data." Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. 2024.

# Modern Depth Prediction Models: DepthAnything Series

Yang, Lihe, et al. "Depth anything: Unleashing the power of large-scale unlabeled data." Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. 2024.

# 3D Shape Representations: Surface Normals

For each pixel, **surface normals** give a vector that is perpendicular to the surface at the given point



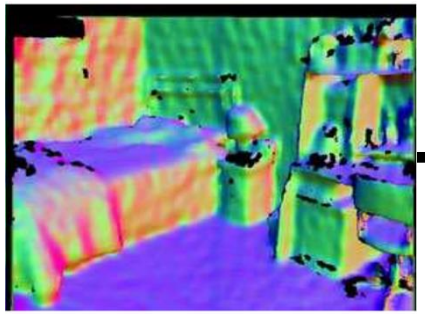
RGB Image: 3 x H x W

Normals: 3 x H x W

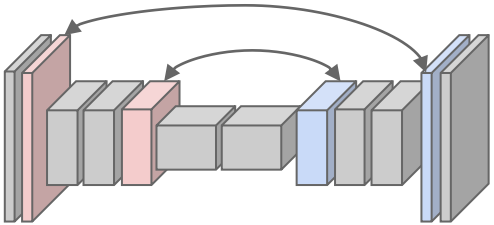
Eigen and Fergus, "Predicting Depth, Surface Normals and Semantic Labels with a Common Multi-Scale Convolutional Architecture", ICCV 2015

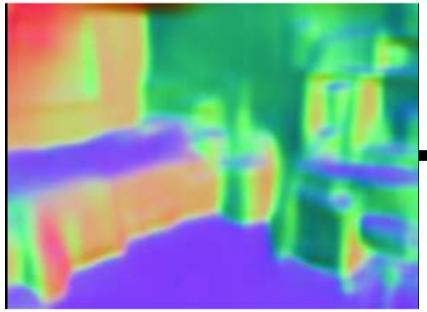
# Predicting Normals

**Ground-truth Normals:**
3 x H x W



**Per-Pixel Loss:**
$(x \cdot y) / (|x||y|)$





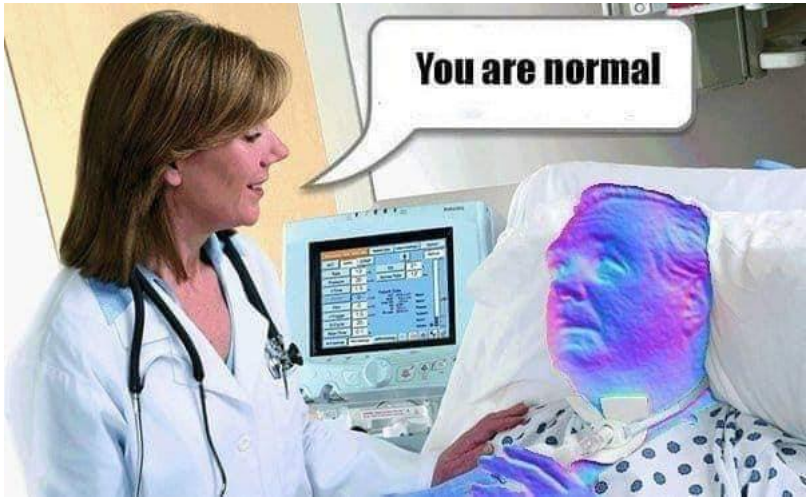
**RGB Input Image:**
3 x H x W

**Fully Convolutional network**

**Predicted Normals:**
3 x H x W

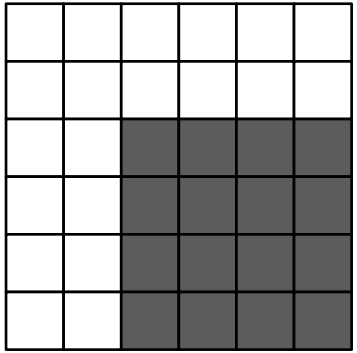
Recall:
$x \cdot y$
$= |x|\,|y| \cos \theta$

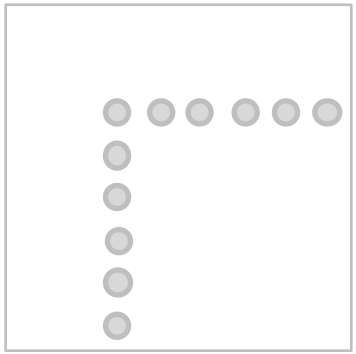
Eigen and Fergus, "Predicting Depth, Surface Normals and Semantic Labels with a Common Multi-Scale Convolutional Architecture", ICCV 2015

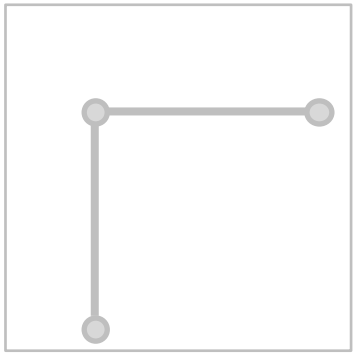

You are normal

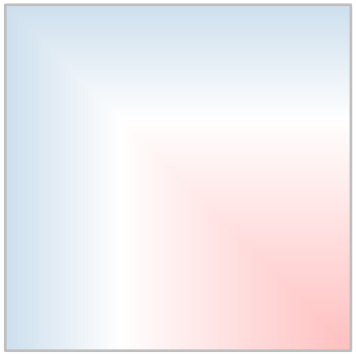

When I grow up, I want to have a normal life and a normal family
25 years later

# 3D Shape Representations







Depth Map

Voxel Grid

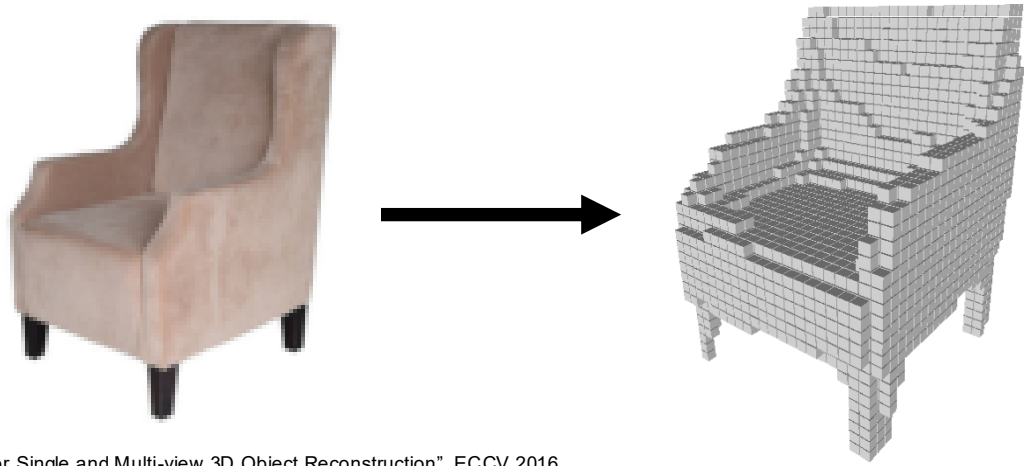
Pointcloud

Mesh

Implicit Surface

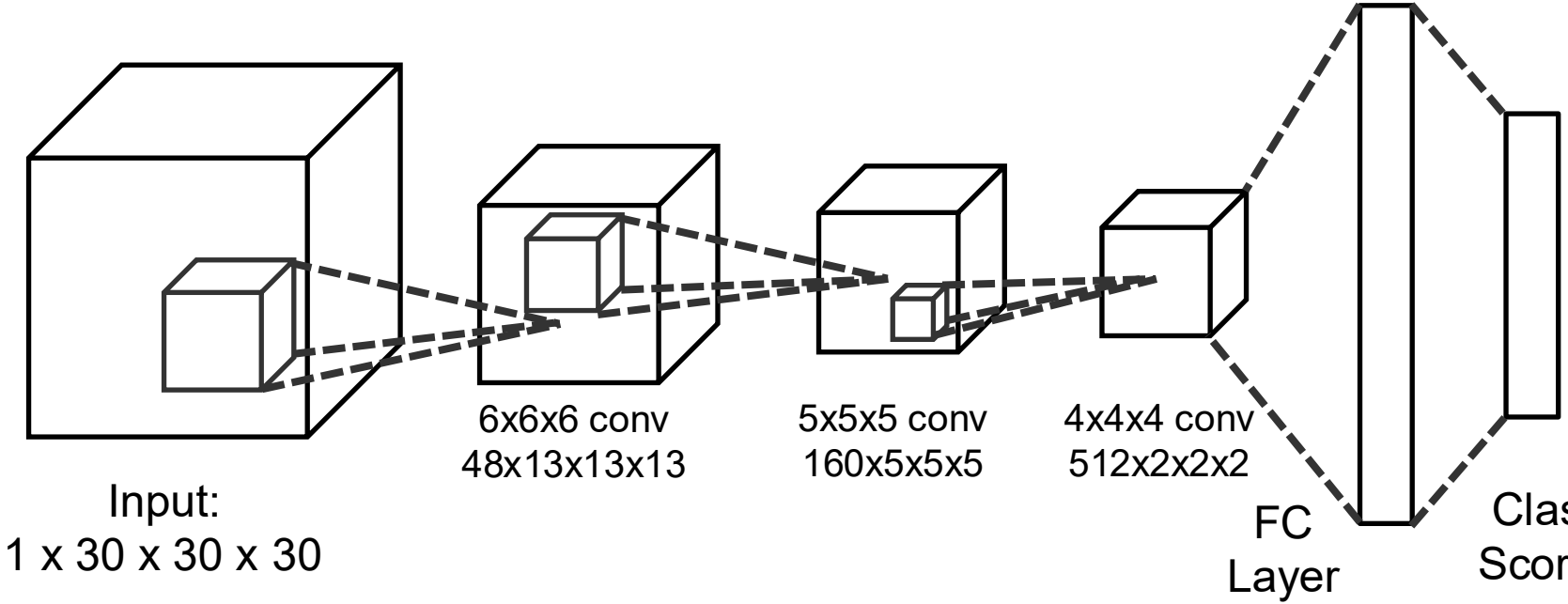
# 3D Shape Representations: Voxels

- Represent a shape with a V x V x V grid of occupancies
- Just like segmentation masks in Mask R-CNN, but in 3D!
- (+) Conceptually simple: just a 3D grid!
- (-) Need high spatial resolution to capture fine structures
- (-) Scaling to high resolutions is nontrivial!



Choy et al, "3D-R2N2: A Unified Approach for Single and Multi-view 3D Object Reconstruction", ECCV 2016

# Processing Voxel Inputs: 3D Convolution

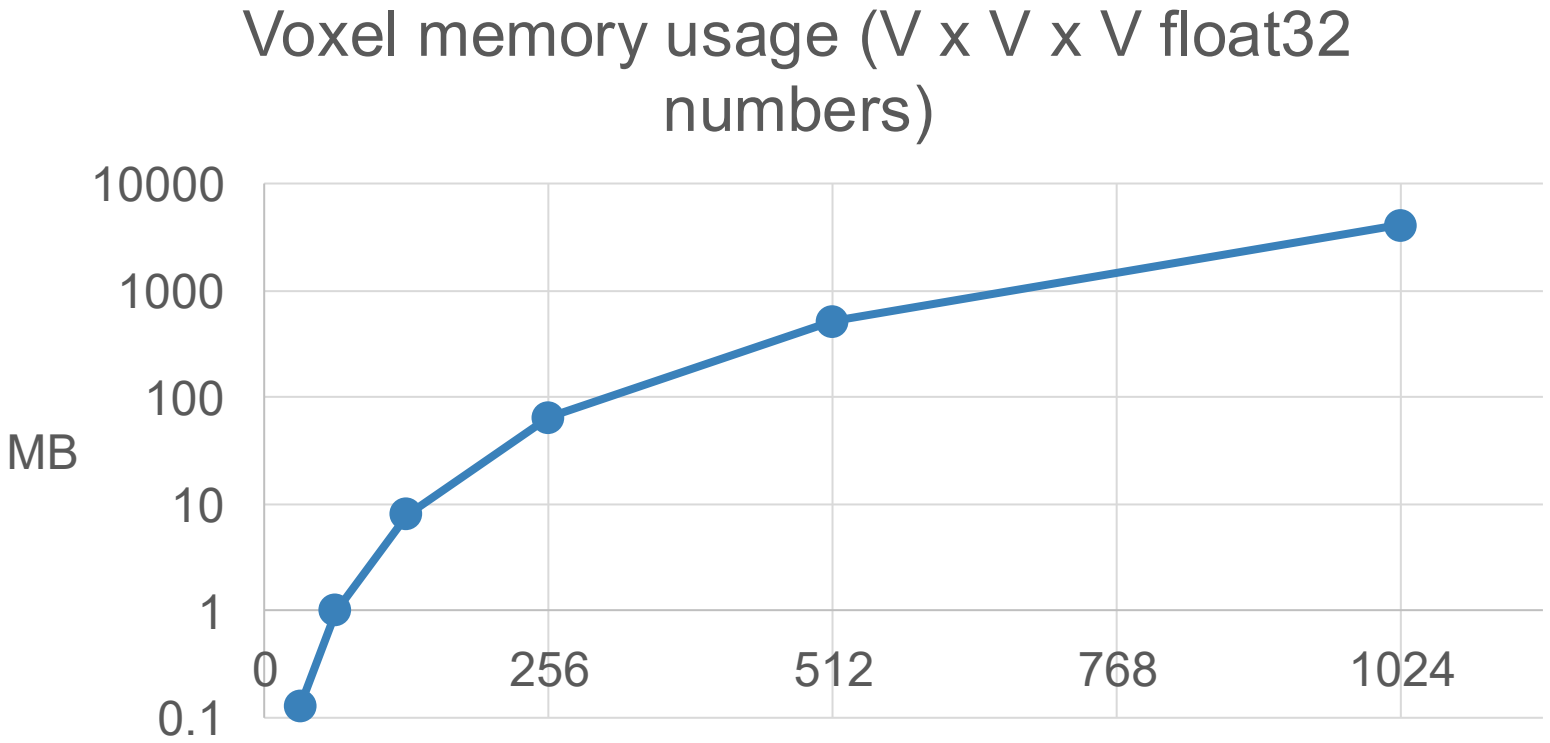



Train with classification loss

Wu et al, "3D ShapeNets: A Deep Representation for Volumetric Shapes", CVPR 2015

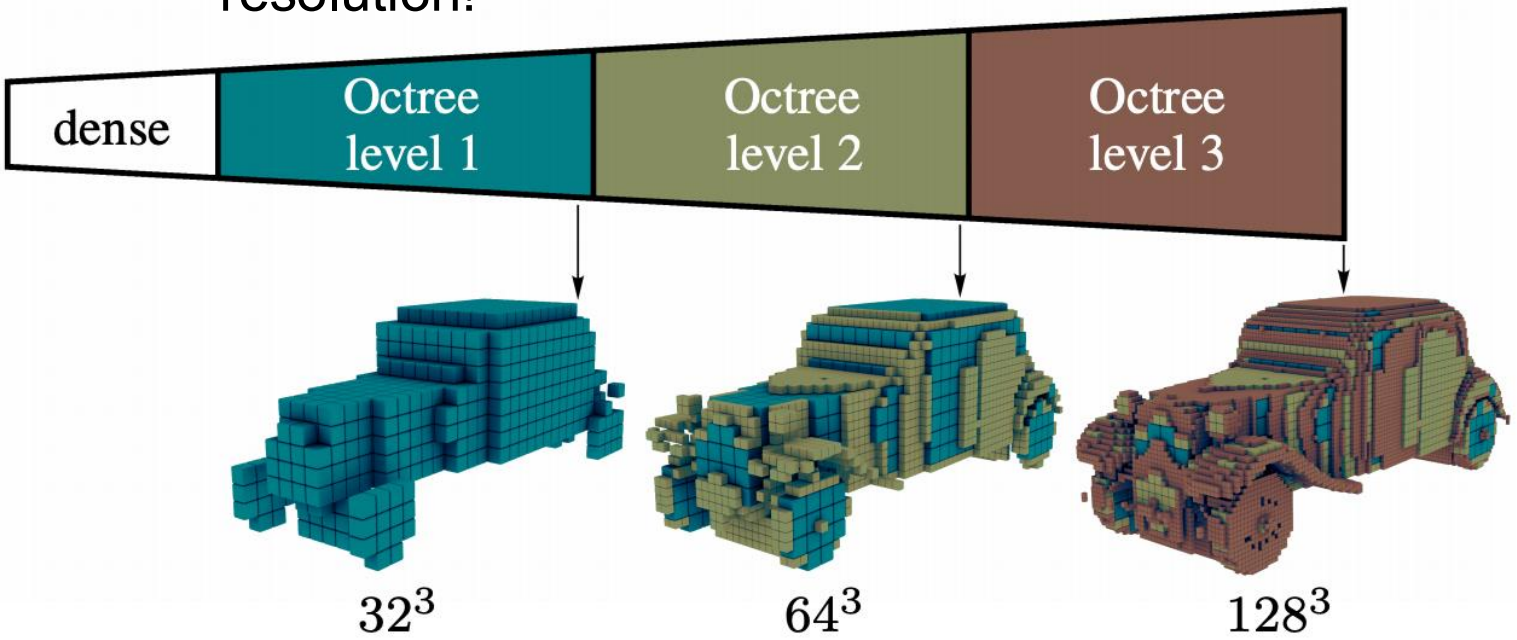
# Voxel Problems: Memory Usage

Storing $1024^3$ voxel grid takes 4GB of memory!



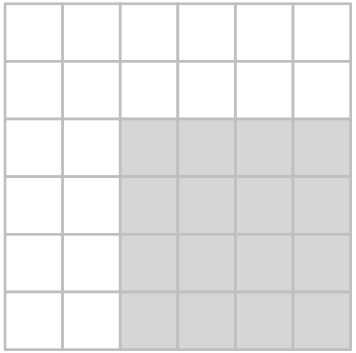
# Scaling Voxels: Oct-Trees

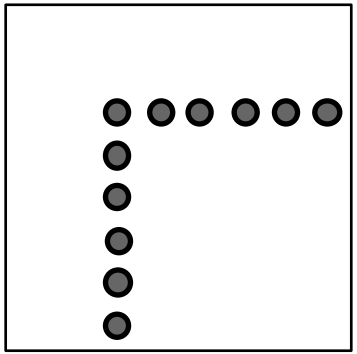
Use voxel grids with heterogenous resolution!

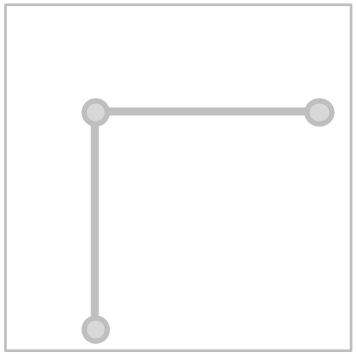


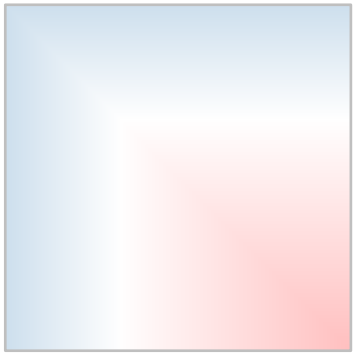

Tatarchenko et al, "Octree Generating Networks: Efficient Convolutional Architectures for High-resolution 3D Outputs", ICCV 2017

# 3D Shape Representations







Depth Map

Voxel Grid

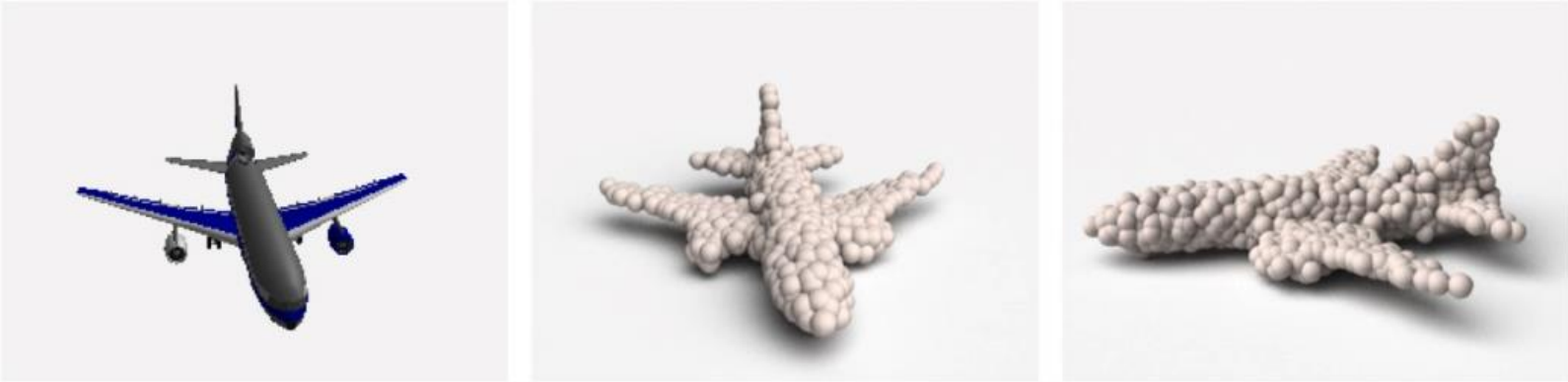
Pointcloud

Mesh

Implicit Surface

# 3D Shape Representations: Point Cloud

- Represent shape as a set of P points in 3D space
- (+) Can represent fine structures without huge numbers of points
- ( ) Requires new architecture, losses, etc
- (-) Doesn't explicitly represent the surface of the shape: extracting a mesh for rendering or other applications requires post-processing



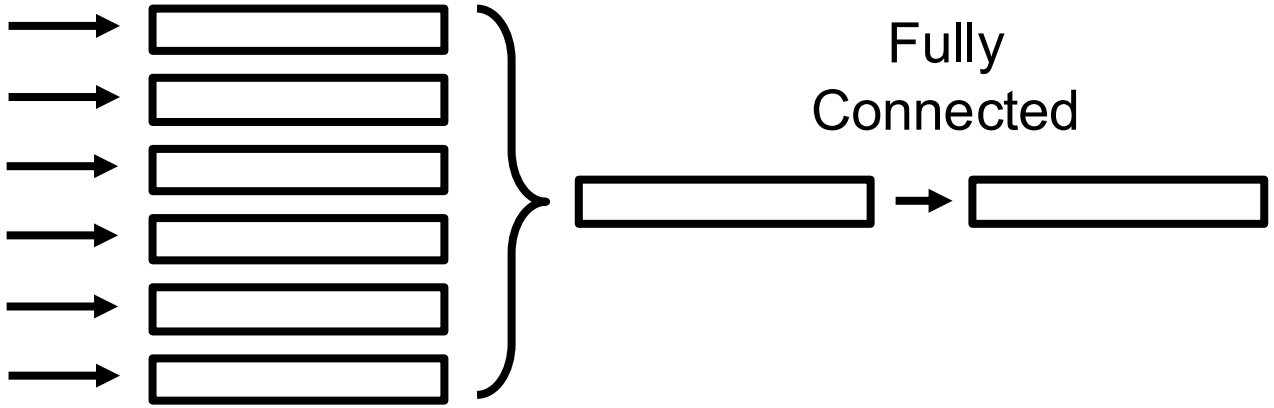
Fan et al, "A Point Set Generation Network for 3D Object Reconstruction from a Single Image", CVPR 2017

# Processing Pointcloud Inputs: PointNet

Run MLP on each point

Max-Pool

Want to process pointclouds as **sets**: order should not matter




**Input pointcloud:** P x 3

**Point features:** P x D

**Pooled vector:** D

**Class score:** C

Qi et al, "PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation", CVPR 2017
Qi et al, "PointNet++: Deep Hierarchical Feature Learning on Point Sets in a Metric Space", NeurIPS 2017

# PointNet applications:

PointNet

airplane?

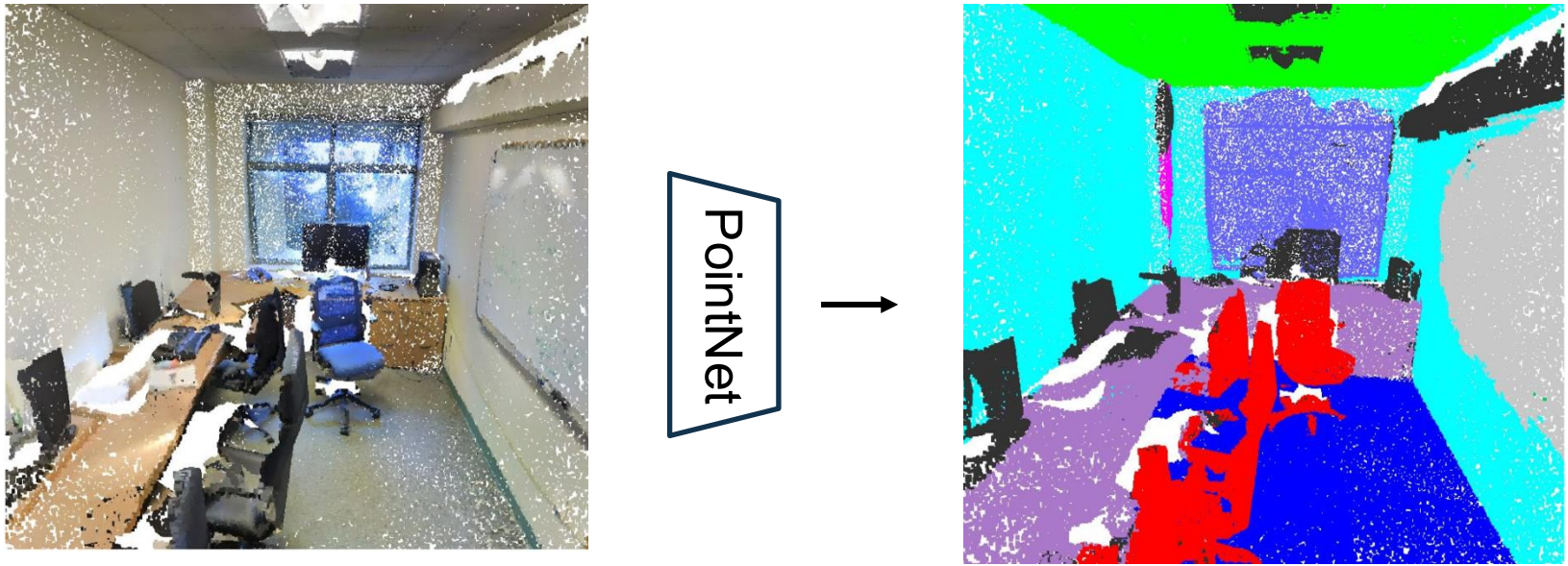
mug?

table?

car?

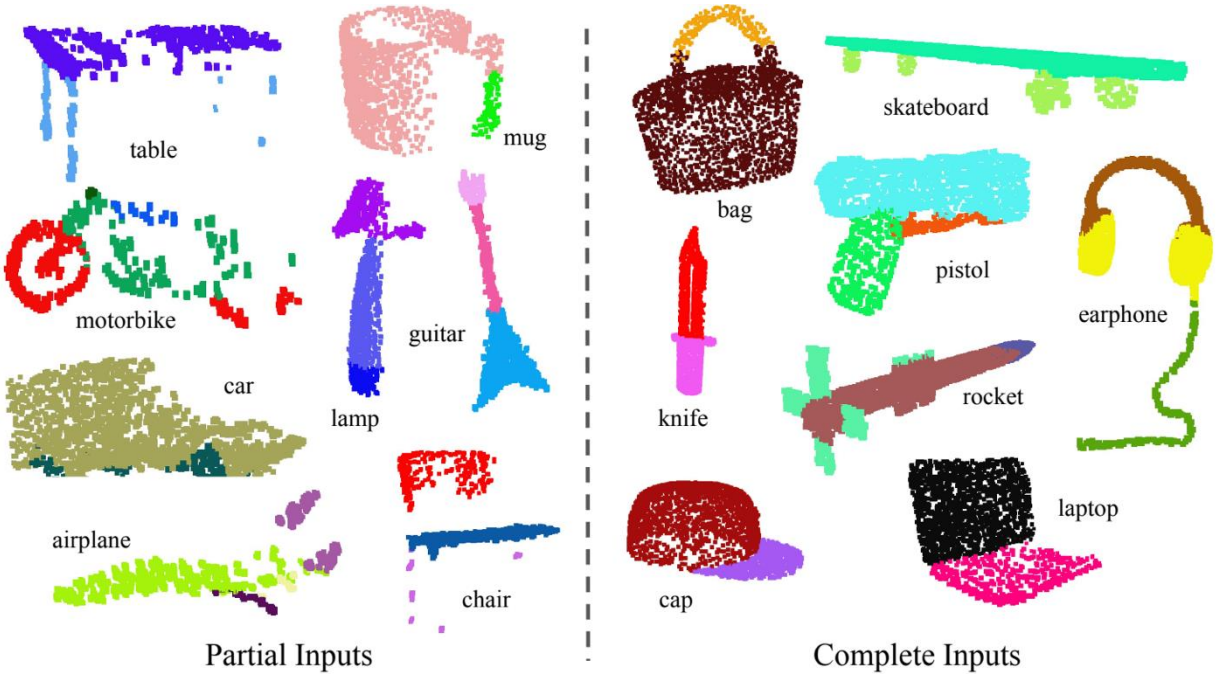
3D Object Classification

# PointNet applications:



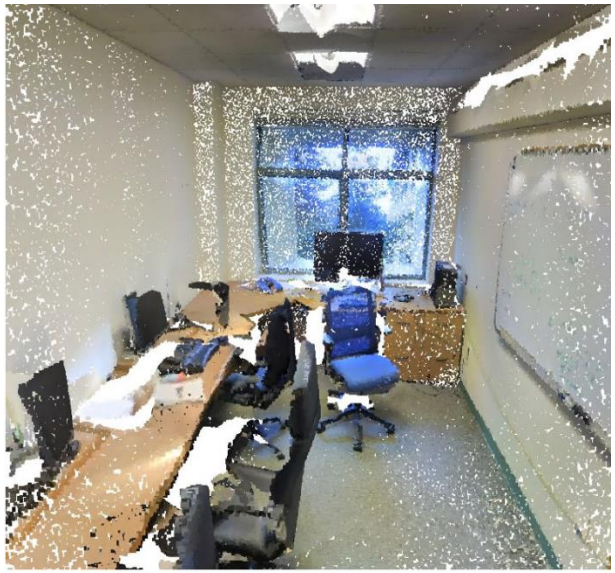

Semantic Segmentation on Point Clouds

# PointNet applications:



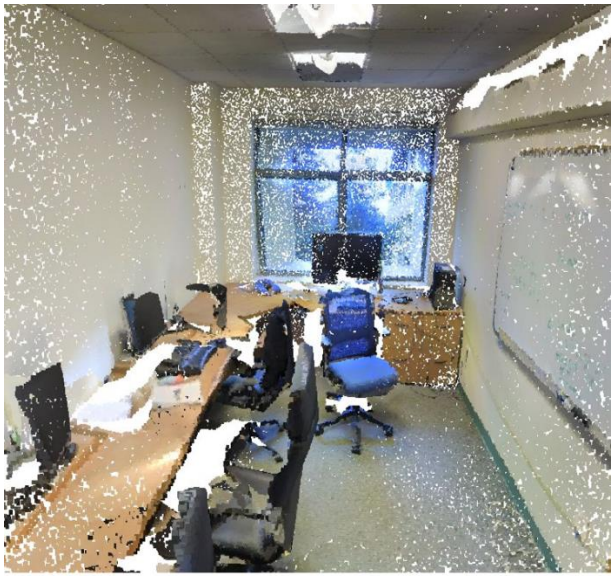

Object Part Segmentation

# Sensor Fusion: Pointclouds + RGB



Point Clouds

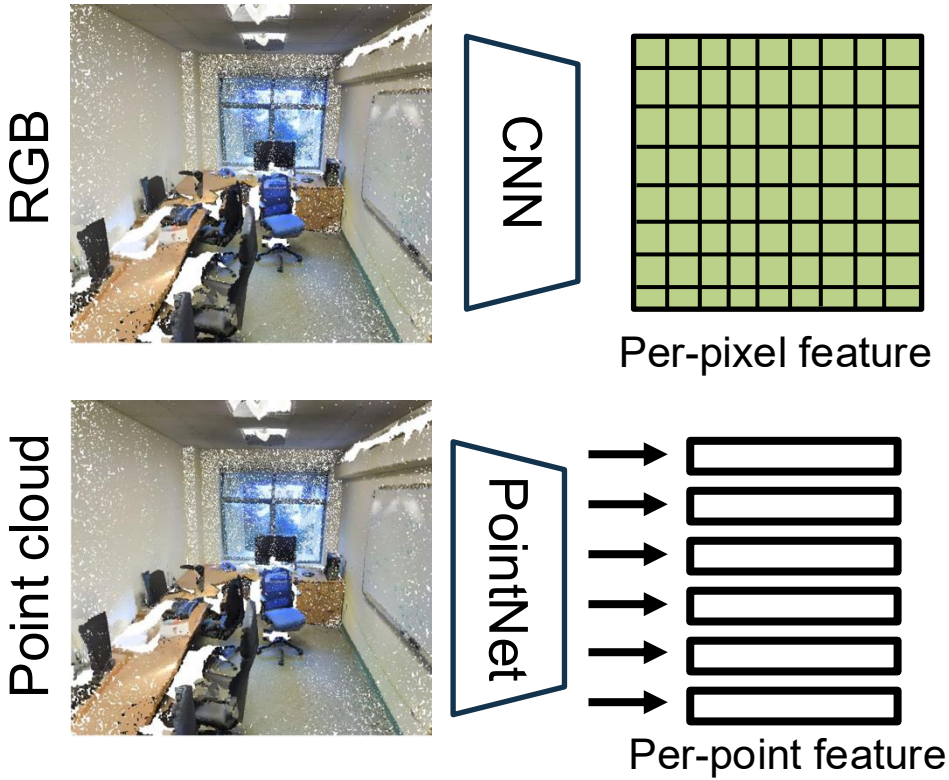
(x, y, z)



Point Clouds + RGB

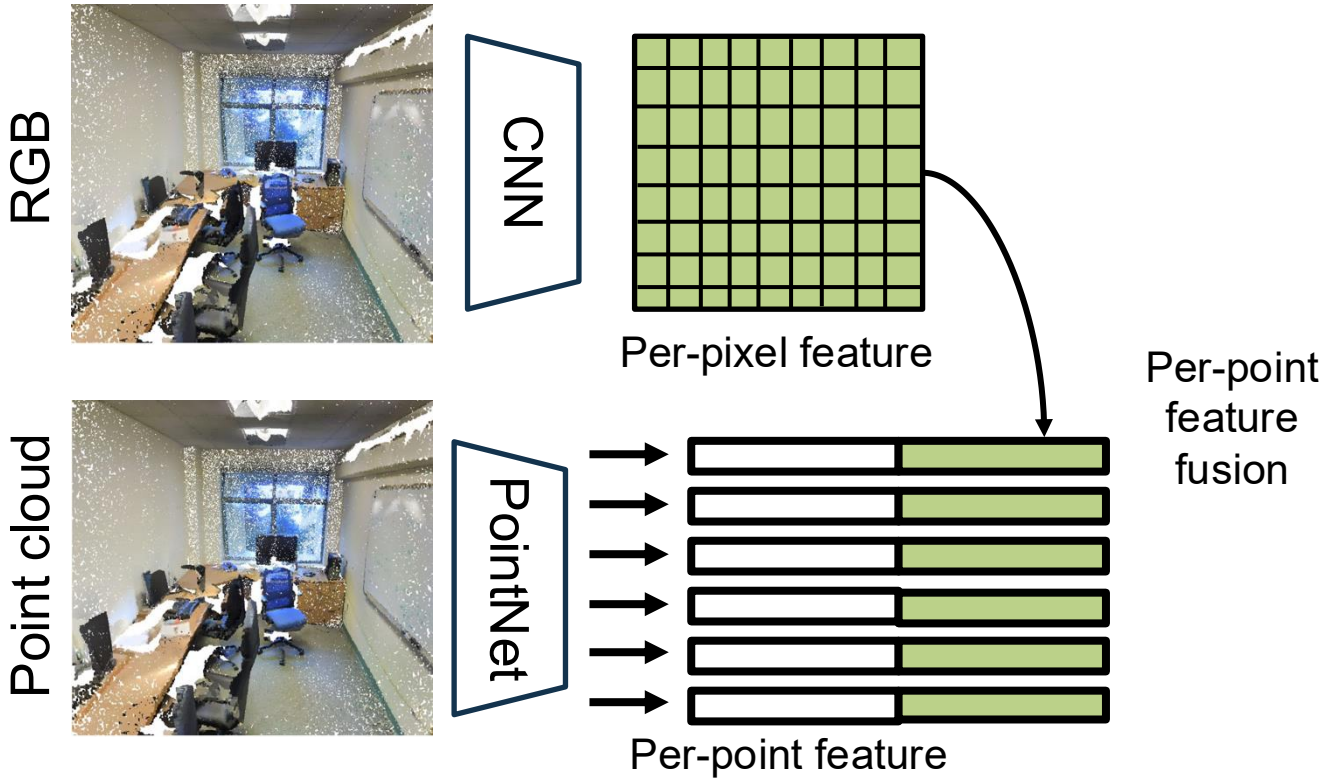
(x, y, z,
r, g, b)

# Processing Pointcloud+RGB: DenseFusion



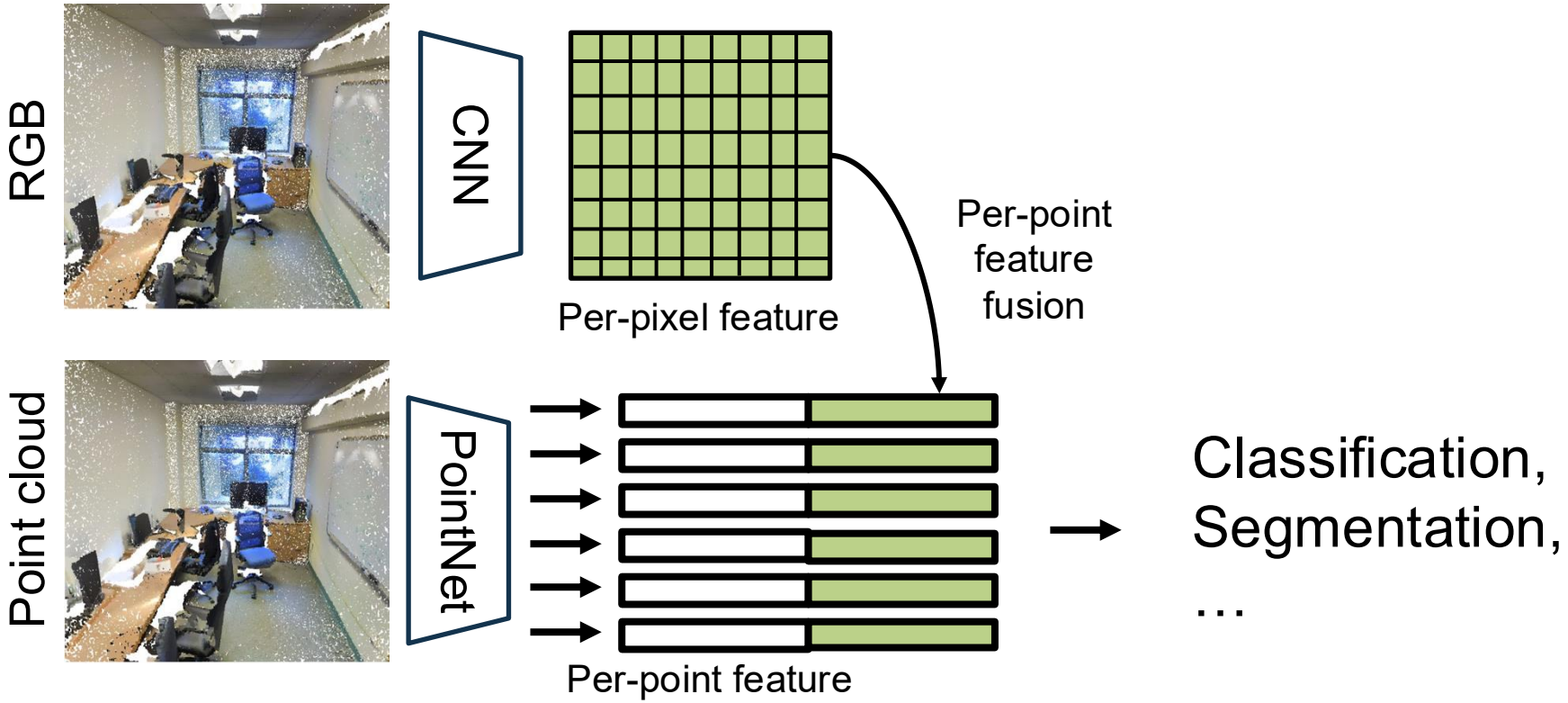

Wang et al, “DenseFusion: 6D Object Pose Estimation by Iterative Dense Fusion”, CVPR 2019

# Processing Pointcloud+RGB: DenseFusion




Wang et al, “DenseFusion: 6D Object Pose Estimation by Iterative Dense Fusion”, CVPR 2019

# Processing Pointcloud+RGB: DenseFusion




Wang et al, “DenseFusion: 6D Object Pose Estimation by Iterative Dense Fusion”, CVPR 2019

# Generating Pointcloud Outputs

**Input Image**:
3 x H x W

2D CNN

**Image Features**:
C x H' x W'

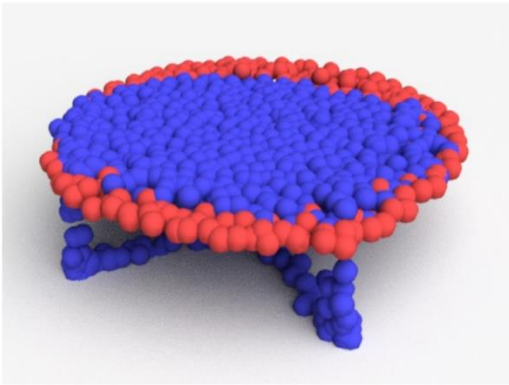
Fully connected branch

Convolutional branch

2D CNN

**Points**:
$P_1$ x 3

**Points**:
($P_2$x3) x H' x W'



**Pointcloud**:
($P_1$ + H'W'$P_2$) x 3

Fan et al, "A Point Set Generation Network for 3D Object Reconstruction from a Single Image", CVPR 2017

# Predicting Point Clouds: Loss Function

We need a (differentiable) way to compare pointclouds **as sets**!

Fan et al, “A Point Set Generation Network for 3D Object Reconstruction from a Single Image”, CVPR 2017

# Predicting Point Clouds: Loss Function

We need a (differentiable) way to compare pointclouds **as sets**!

**Chamfer distance** is the sum of L2 distance to each point's nearest neighbor in the other set

$$d_{CD}(S_1, S_2) = \sum_{x \in S_1} \min_{y \in S_2} \|x - y\|_2^2 + \sum_{y \in S_2} \min_{x \in S_1} \|x - y\|_2^2$$

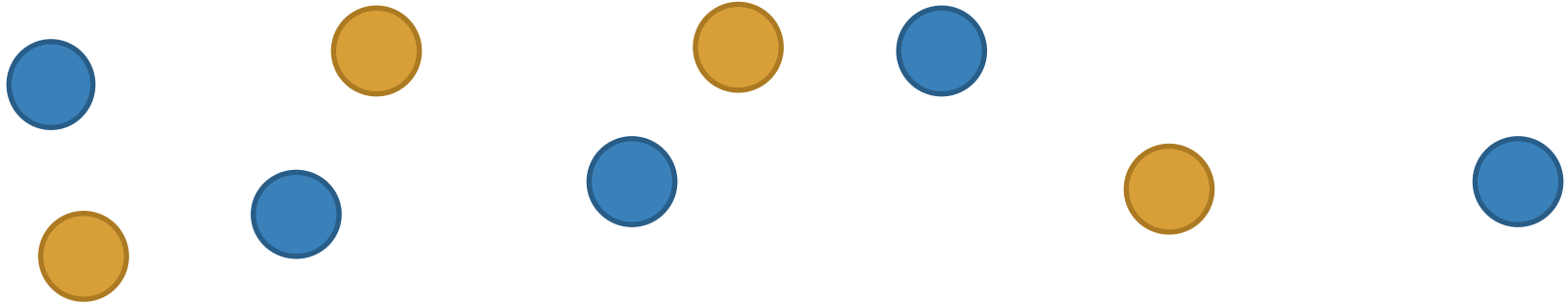
Fan et al, "A Point Set Generation Network for 3D Object Reconstruction from a Single Image", CVPR 2017

# Predicting Point Clouds: Loss Function

We need a (differentiable) way to compare pointclouds **as sets**!

**Chamfer distance** is the sum of L2 distance to each point's nearest neighbor in the other set

$$d_{CD}(S_1, S_2) = \sum_{x \in S_1} \min_{y \in S_2} \|x - y\|_2^2 + \sum_{y \in S_2} \min_{x \in S_1} \|x - y\|_2^2$$



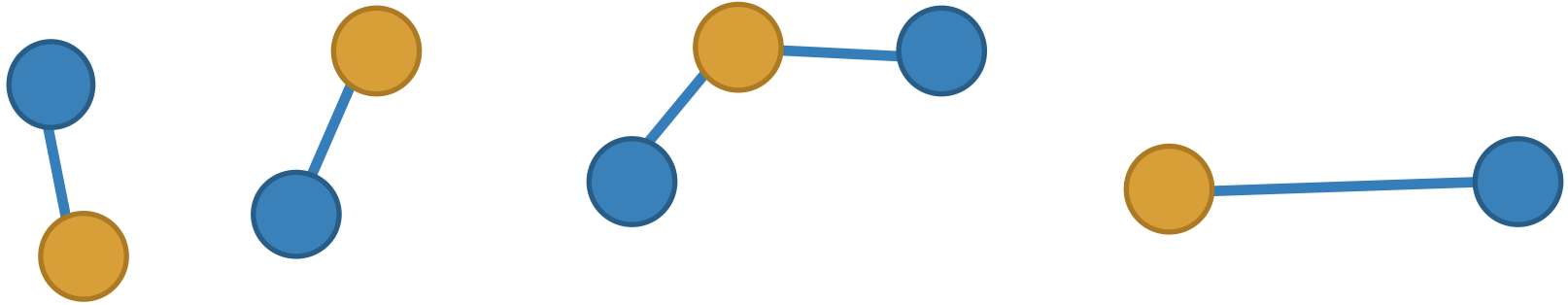
Fan et al, "A Point Set Generation Network for 3D Object Reconstruction from a Single Image", CVPR 2017

# Predicting Point Clouds: Loss Function

We need a (differentiable) way to compare pointclouds **as sets**!

**Chamfer distance** is the sum of L2 distance to each point's nearest neighbor in the other set

$$d_{CD}(S_1, S_2) = \sum_{x \in S_1} \min_{y \in S_2} \|x - y\|_2^2 + \sum_{y \in S_2} \min_{x \in S_1} \|x - y\|_2^2$$



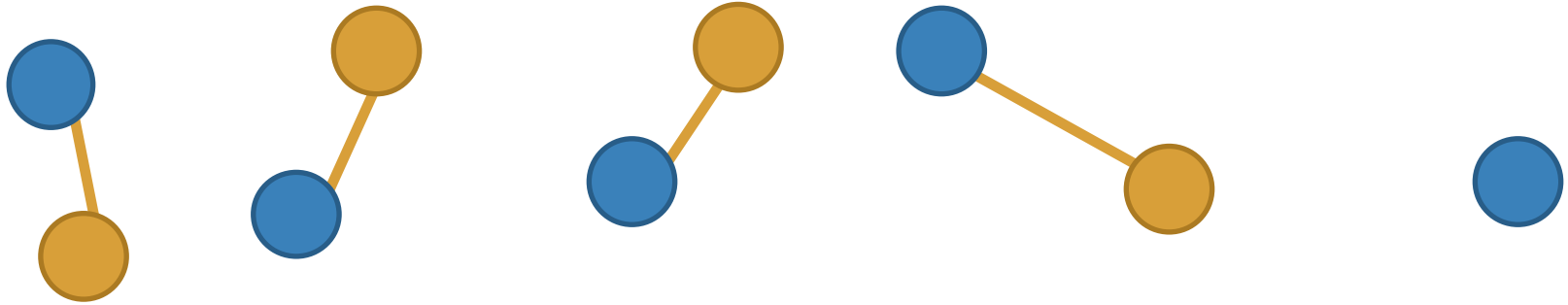
Fan et al, "A Point Set Generation Network for 3D Object Reconstruction from a Single Image", CVPR 2017

# Predicting Point Clouds: Loss Function

We need a (differentiable) way to compare pointclouds **as sets**!

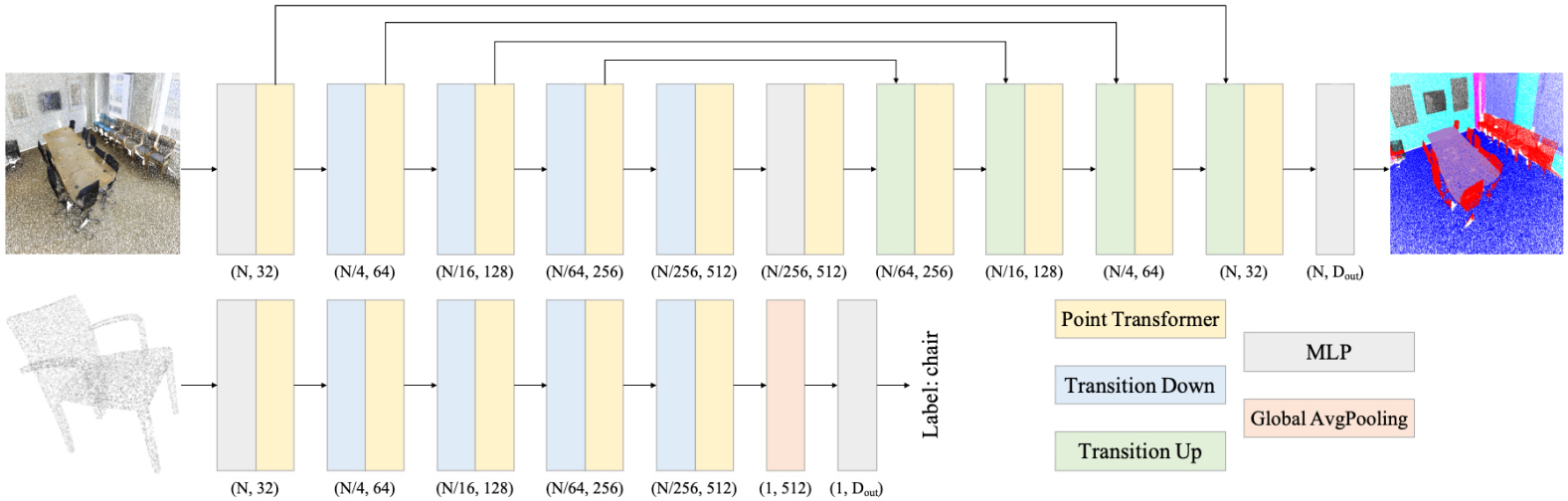
**Chamfer distance** is the sum of L2 distance to each point's nearest neighbor in the other set

$$d_{CD}(S_1, S_2) = \sum_{x \in S_1} \min_{y \in S_2} \|x - y\|_2^2 + \sum_{y \in S_2} \min_{x \in S_1} \|x - y\|_2^2$$



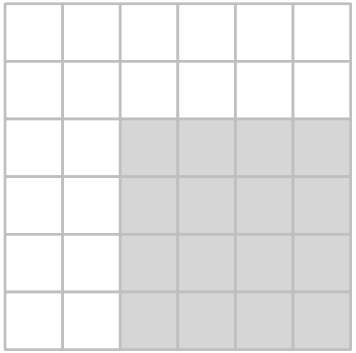
Fan et al, "A Point Set Generation Network for 3D Object Reconstruction from a Single Image", CVPR 2017

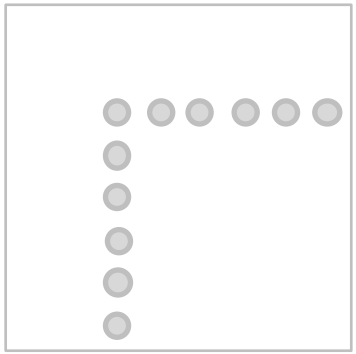
# Modern Point Coud Models: Point Transformer Series

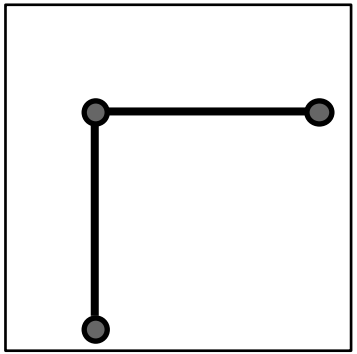


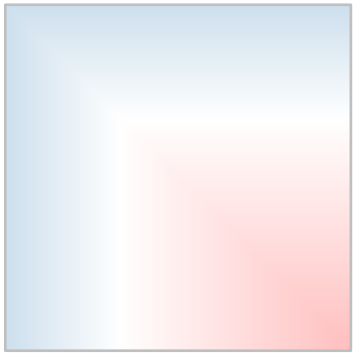

Zhao, Hengshuang, et al. "Point transformer." Proceedings of the IEEE/CVF international conference on computer vision. 2021.

# 3D Shape Representations







Depth Map

Voxel Grid

Pointcloud

Mesh

Implicit Surface

# 3D Shape Representations: Triangle Mesh

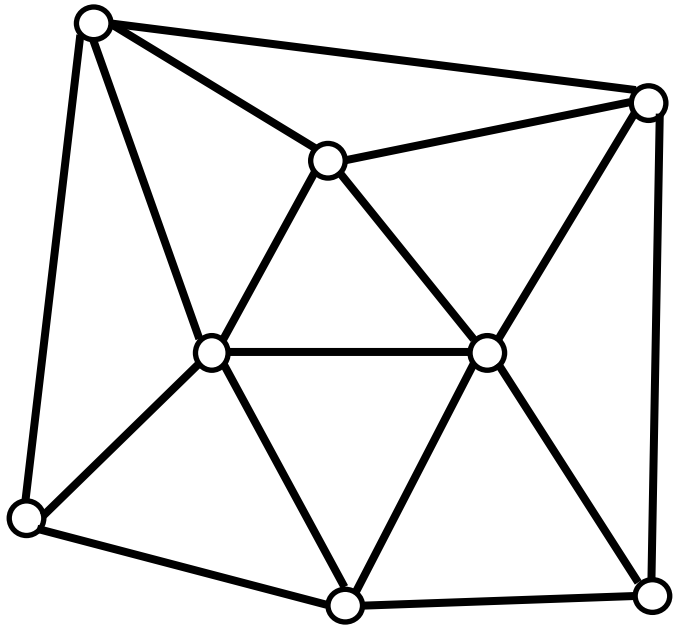
Represent a 3D shape as a set of triangles

**Vertices**: Set of V points in 3D space

**Faces**: Set of triangles over the vertices

(+) Standard representation for graphics

(+) Explicitly represents 3D shapes



# 3D Shape Representations: Triangle Mesh

Represent a 3D shape as a set of triangles

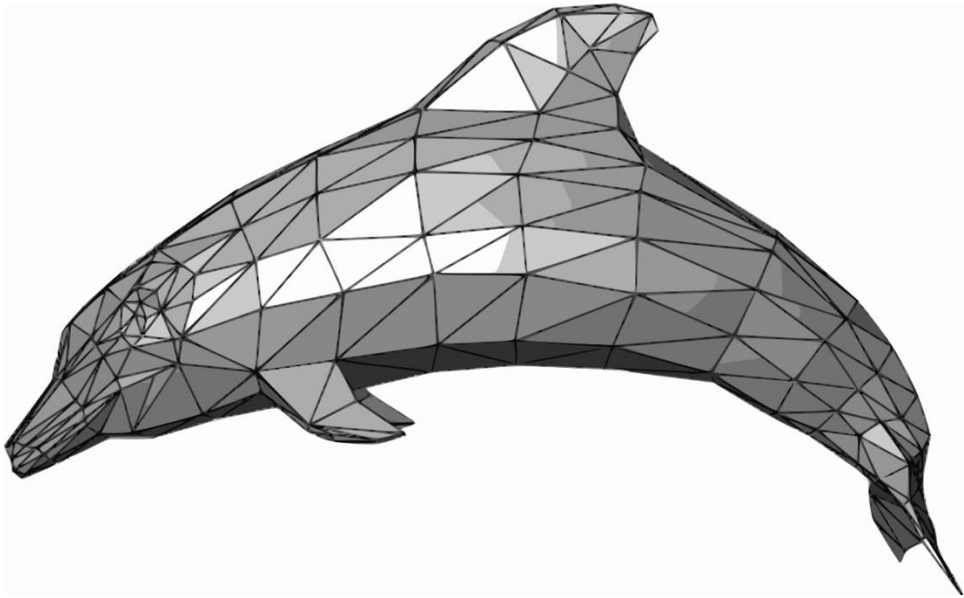
**Vertices**: Set of V points in 3D space

**Faces**: Set of triangles over the vertices

(+) Standard representation for graphics

(+) Explicitly represents 3D shapes

(+) Adaptive: Can represent flat surfaces very efficiently, can allocate more faces to areas with fine detail



# 3D Shape Representations: Triangle Mesh

Represent a 3D shape as a set of triangles

**Vertices**: Set of V points in 3D space

**Faces**: Set of triangles over the vertices

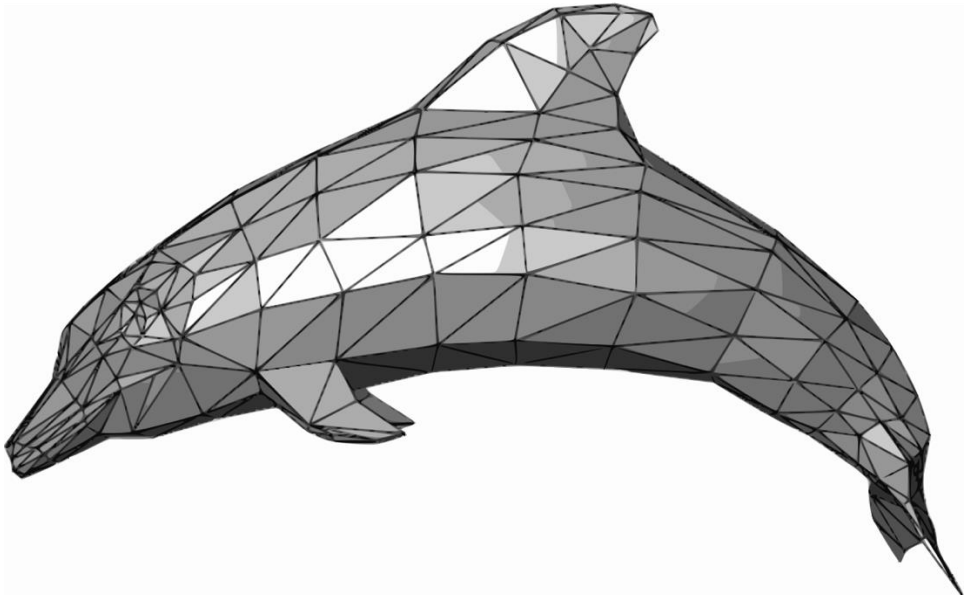
(+) Standard representation for graphics

(+) Explicitly represents 3D shapes

(+) Adaptive: Can represent flat surfaces very efficiently, can allocate more faces to areas with fine detail

(+) Can attach data on verts and interpolate over the whole surface: RGB colors, texture coordinates, normal vectors, etc.

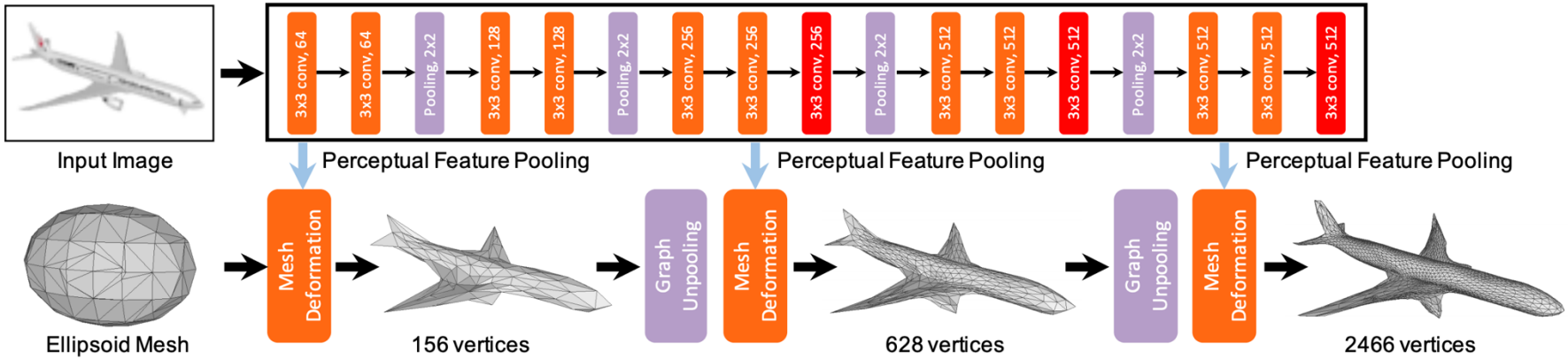
(-) Nontrivial to process with neural networks



# Predicting Meshes: Pixel2Mesh

**Input**: Single RGB Image of an object

**Output**: Triangle mesh for the object



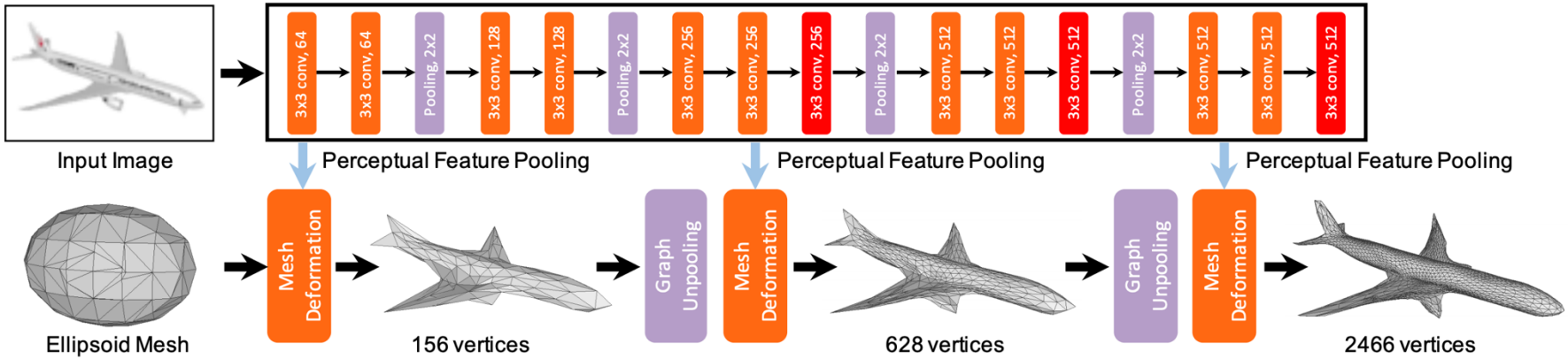

Wang et al, "Pixel2Mesh: Generating 3D Mesh Models from Single RGB Images", ECCV 2018

# Predicting Meshes: Pixel2Mesh

**Input**: Single RGB Image of an object

**Key ideas**:
Iterative Refinement
Graph Convolution
Vertex Aligned-Features
Chamfer Loss Function

**Output**: Triangle mesh for the object



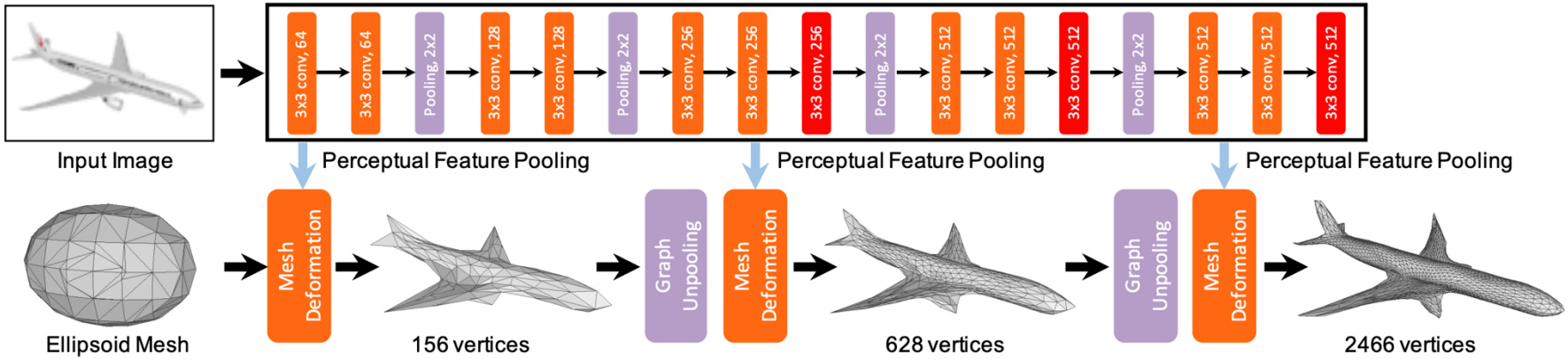

Wang et al, "Pixel2Mesh: Generating 3D Mesh Models from Single RGB Images", ECCV 2018

# Predicting Meshes: Pixel2Mesh

**Input**: Single RGB Image of an object

**Key ideas**:
Iterative Refinement
Graph Convolution
Vertex Aligned-Features
Chamfer Loss Function

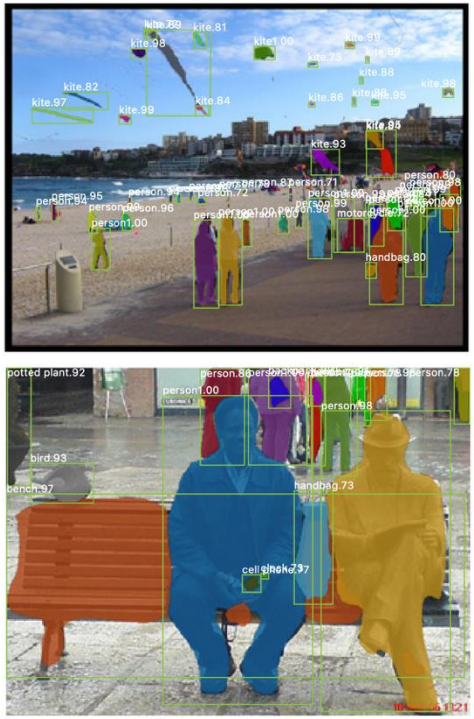
**Output**: Triangle mesh for the object



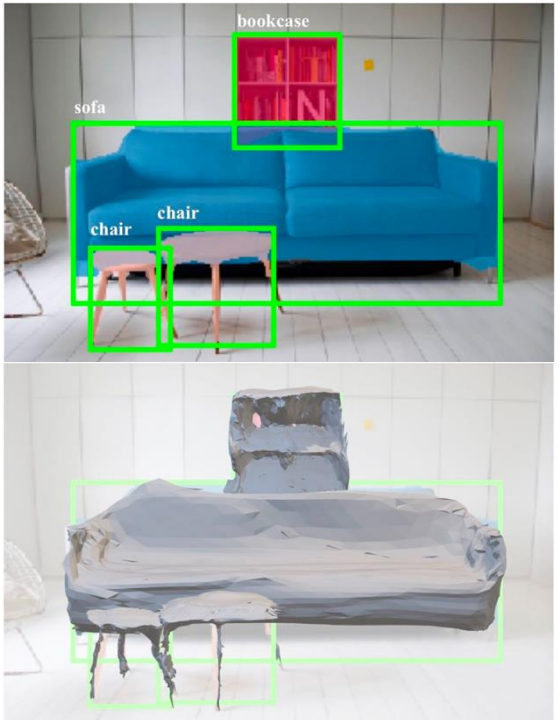

Wang et al, "Pixel2Mesh: Generating 3D Mesh Models from Single RGB Images", ECCV 2018

# 3D Shape Prediction: Mesh R-CNN

Mask R-CNN:
2D Image -> 2D shapes




**Mesh** R-CNN:
2D Image -> **Triangle Meshes**




He, Gkioxari, Dollár, and Girshick, "Mask R-CNN", ICCV 2017

Gkioxari, Malik, and Johnson, "Mesh R-CNN", ICCV 2019

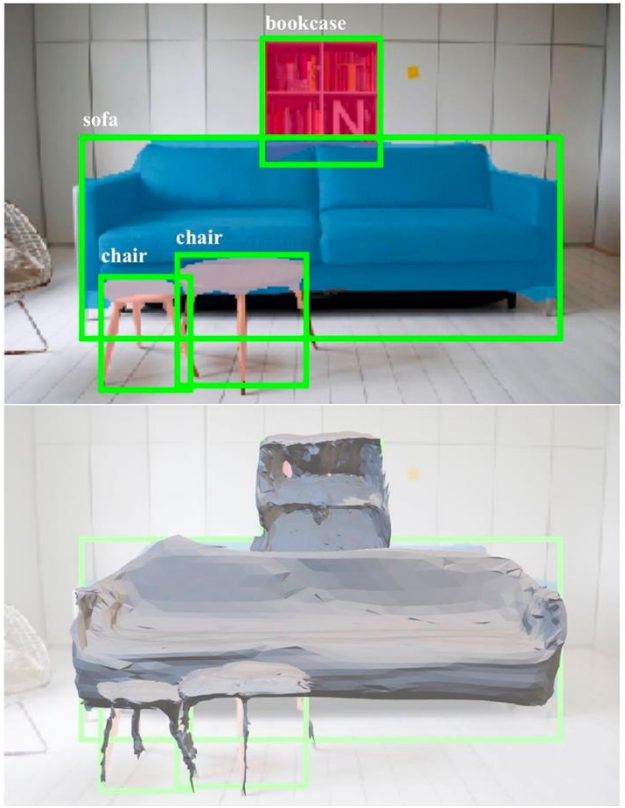
# Mesh R-CNN: Task

**Input**: Single RGB image

**Output**:
A set of detected objects
For each object:
- Bounding box
- Category label
- Instance segmentation
- 3D triangle mesh

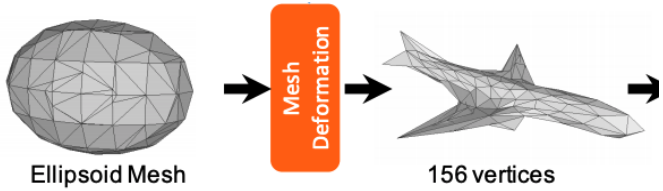
Mask R-CNN

Mesh head

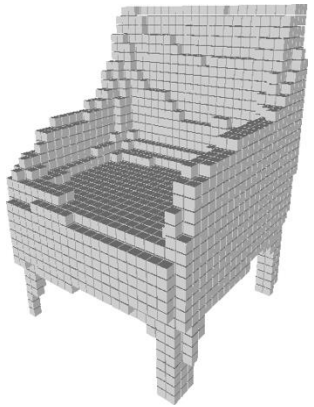


# Mesh R-CNN: Hybrid 3D shape representation

**Mesh deformation** gives good results, but the topology (verts, faces, genus, connected components) fixed by the initial mesh




**Mesh R-CNN**: Use voxel predictions to create initial mesh prediction!



# Mesh R-CNN Pipeline

Input image

# Mesh R-CNN Pipeline

Input image

2D object recognition

# Mesh R-CNN Pipeline

Input image

2D object recognition

3D object voxels

# Mesh R-CNN Pipeline

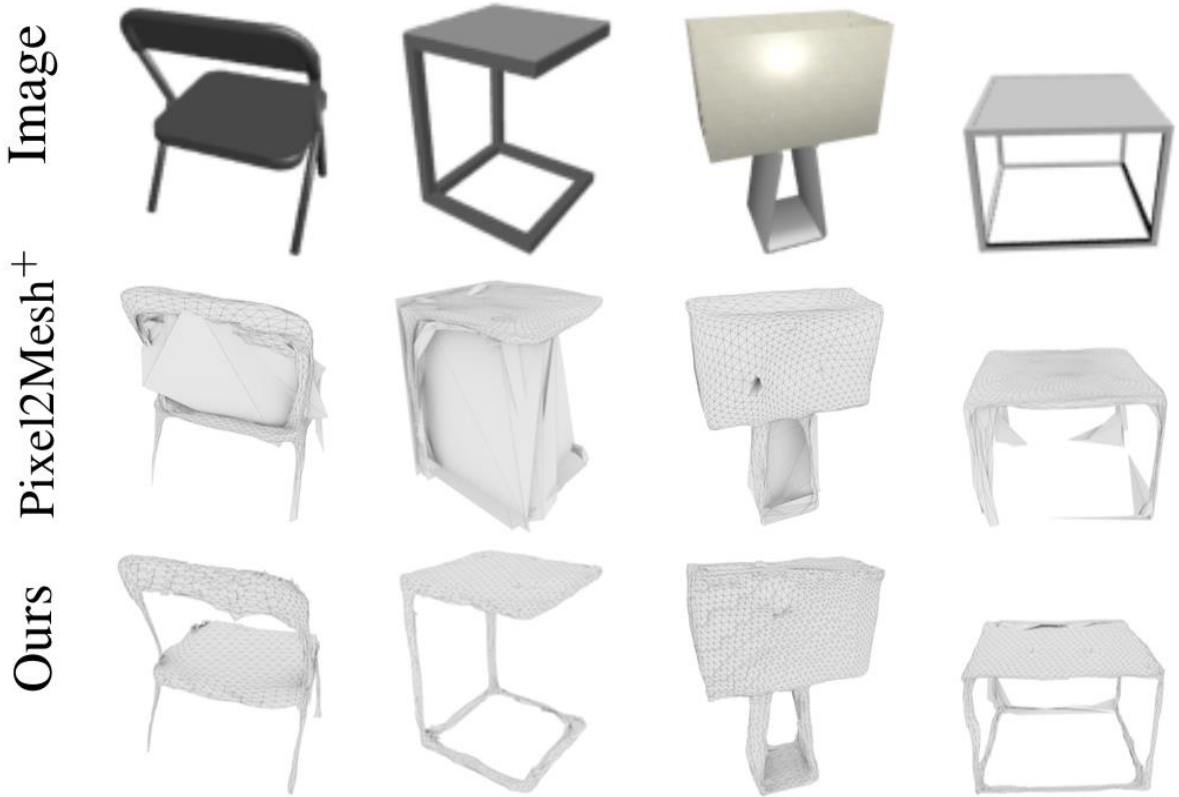
Input image

2D object recognition

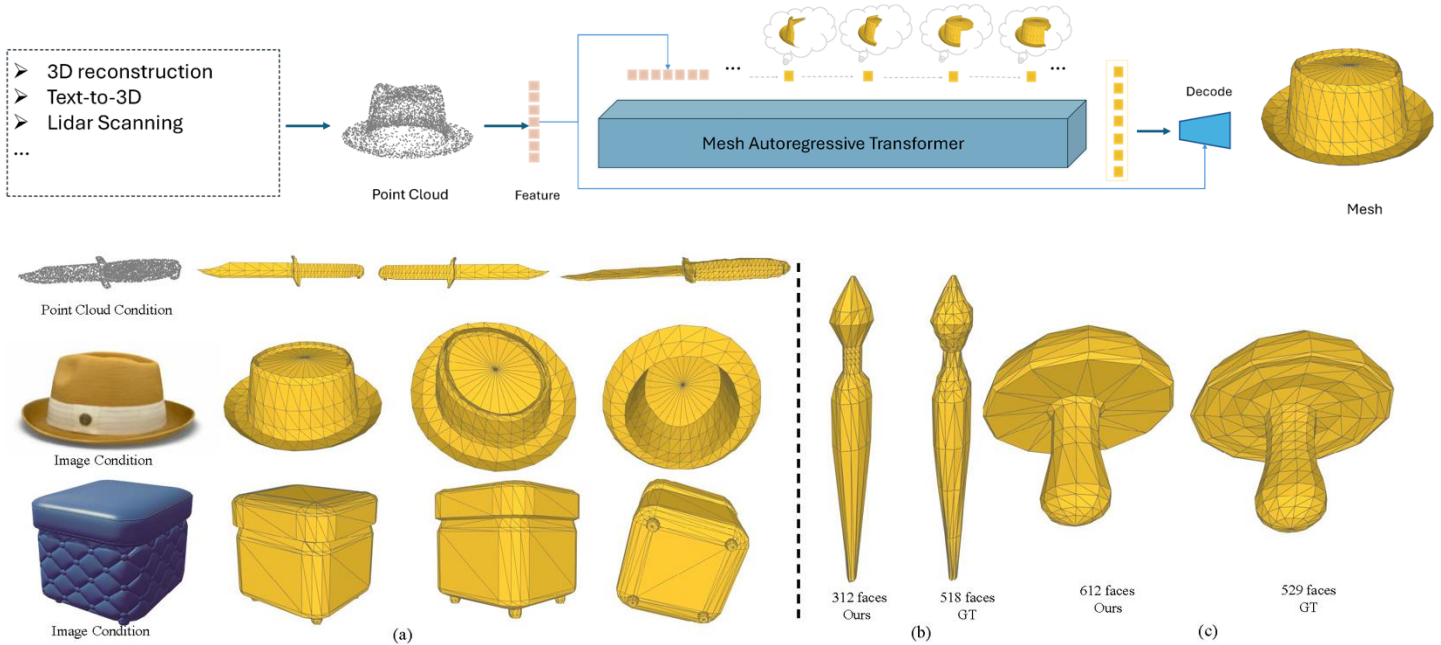
3D object meshes

3D object voxels

# Mesh R-CNN: ShapeNet Results

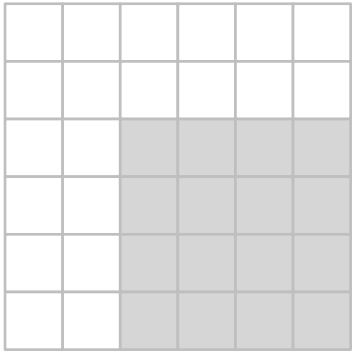


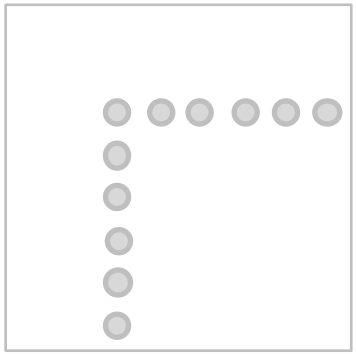
# Modern Mesh Prediction Models: Mesh Anything Series

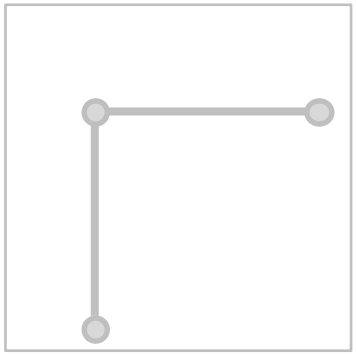


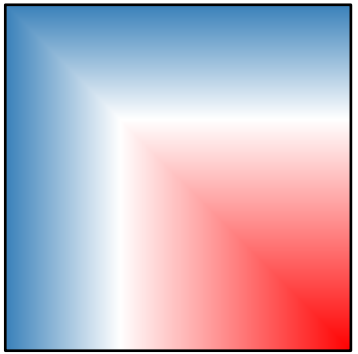

Chen, Yiwen, et al. "Meshanything: Artist-created mesh generation with autoregressive transformers." International Conference on Learning Representations. Vol. 2025. 2025.

# 3D Shape Representations







Depth Map

Voxel Grid

Pointcloud

Mesh

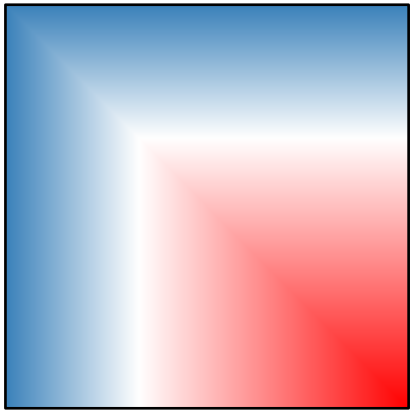
Implicit Surface

# 3D Shape Representations: Implicit Functions

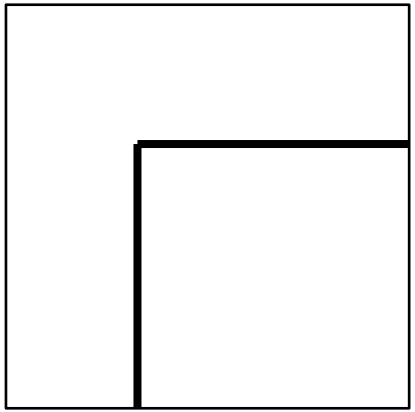
Learn a function to classify arbitrary 3D points as inside / outside the shape

$$o : \mathbb{R}^3 \rightarrow \{0, 1\}$$

The surface of the 3D object is the level set $\{x : o(x) = ½\}$



Implicit function



Explicit Shape

# Algebraic Surfaces (Implicit)

**Surface is zero set of a polynomial in x, y, z**

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

$(R-\sqrt{x^2+y^2})^2+z^2=r^2$

$(x^2+\frac{9y^2}{4}+z^2-1)^3=x^2z^3+\frac{9y^2z^3}{80}$

Slide credit: Ren Ng

# Algebraic Surfaces (Implicit)

**Surface is zero set of a polynomial in x, y, z**

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

$$(R-\sqrt{x^2+y^2})^2+z^2=r^2$$

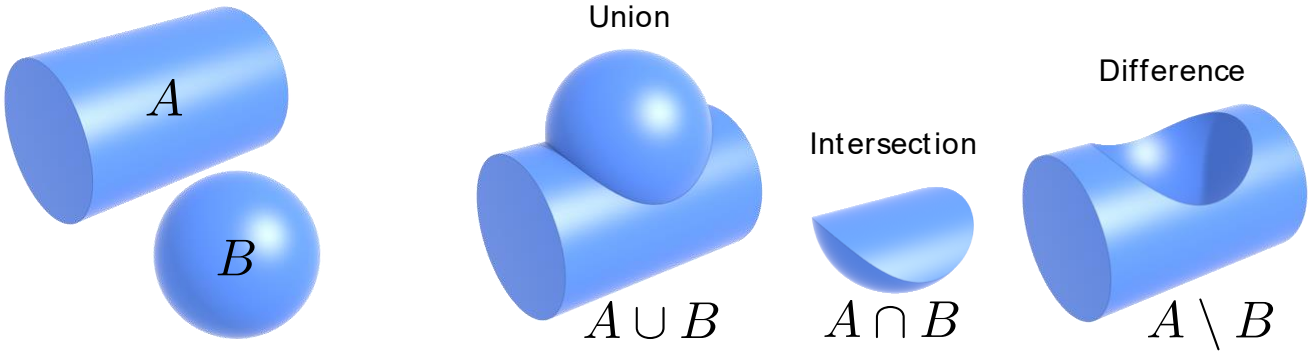
$$(x^2+\frac{9y^2}{4}+z^2-1)^3=x^2z^3+\frac{9y^2z^3}{80}$$

**More complex shapes?**

Slide credit: Ren Ng

# Constructive Solid Geometry (Implicit)

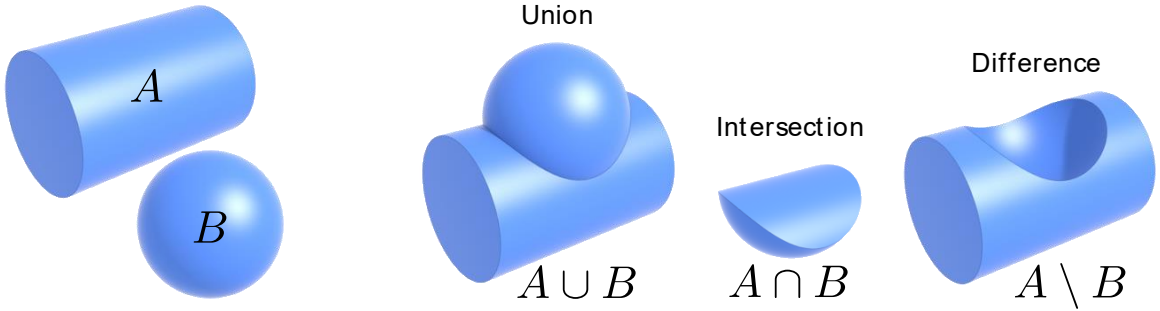
Combine implicit geometry via Boolean operations



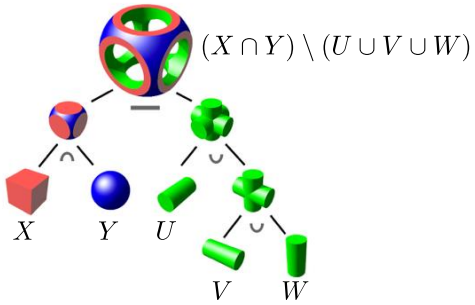

Slide credit: Ren Ng

# Constructive Solid Geometry (Implicit)

Combine implicit geometry via Boolean operations



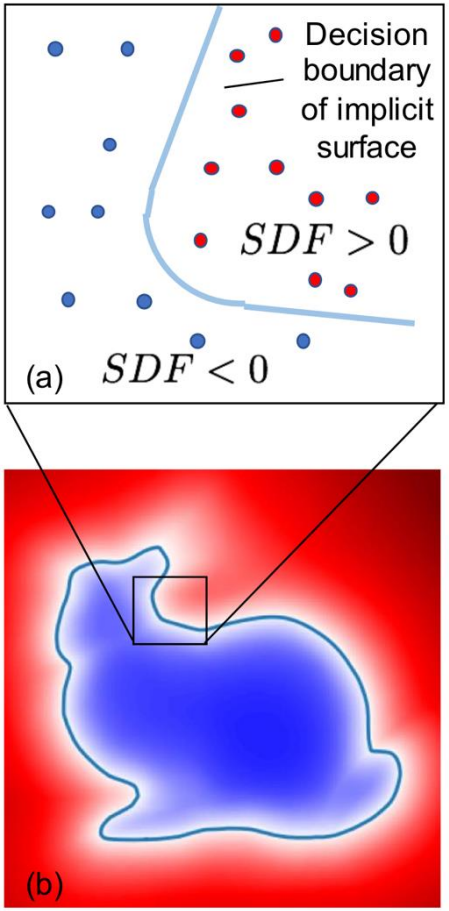

Boolean expressions:





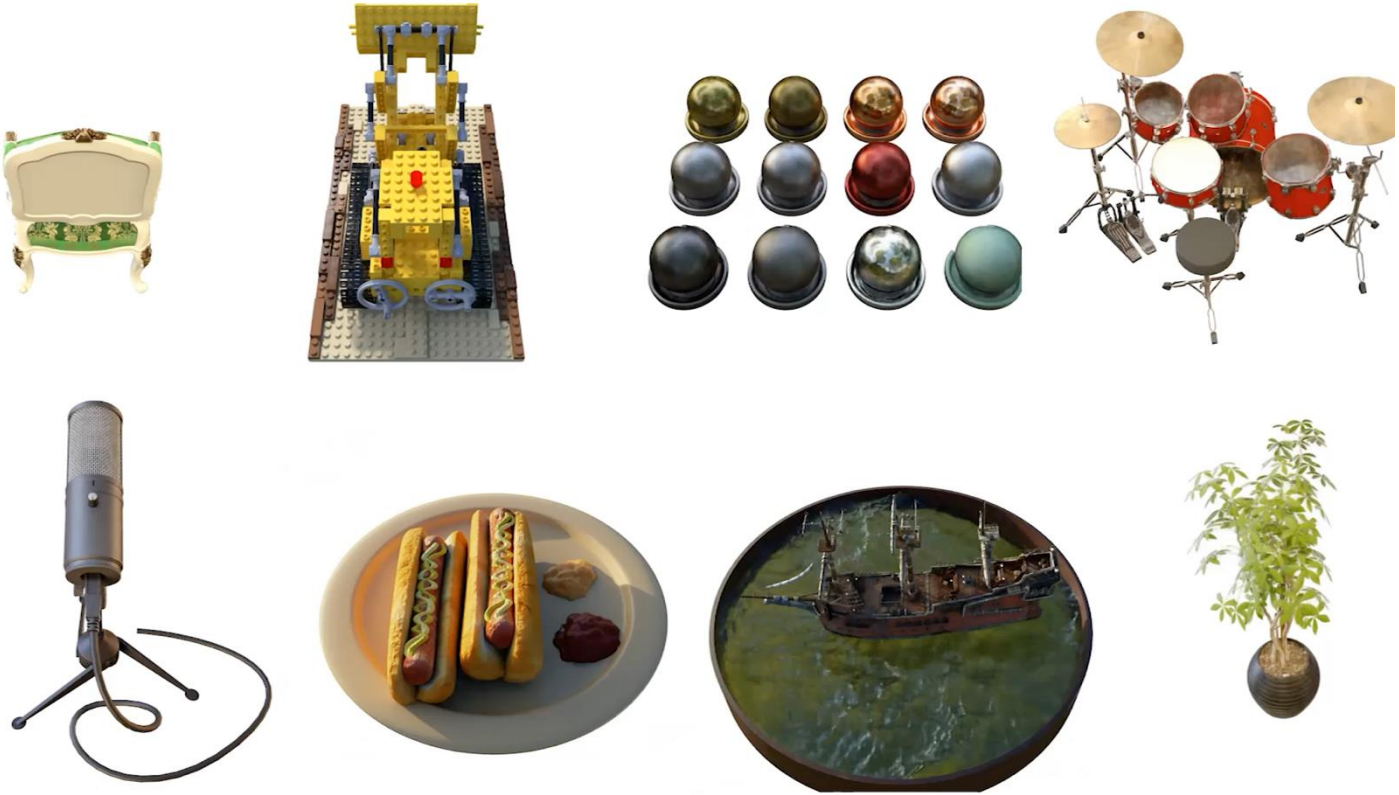

Slide credit: Ren Ng

# DeepSDF



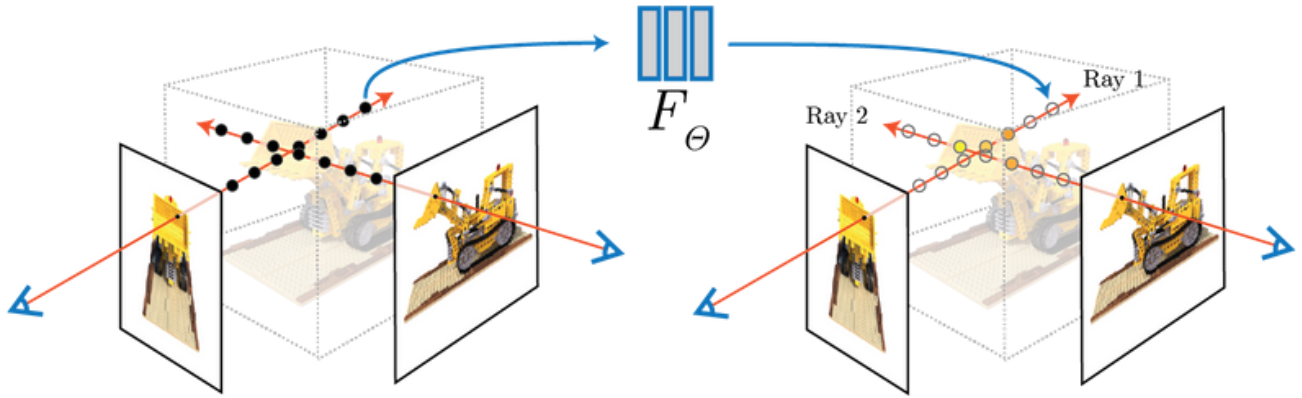

Park et al., “DeepSDF: Learning Continuous Signed Distance Functions for Shape Representation”, CVPR 2019

# Neural Radiance Fields (NeRF)



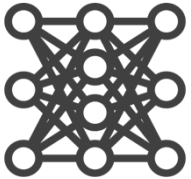
Mildenhall et al, “Representing Scenes as Neural Radiance Fields for View Synthesis”, ECCV 2020

# NeRF: Representing Scenes as Neural Radiance Fields




Novel view synthesis

Scene location: $(x, y, z)$ Viewing direction: $(\theta, \emptyset)$




Output color: $(r, g, b)$ Output density: $(\sigma)$

Mildenhall et al, "Representing Scenes as Neural Radiance Fields for View Synthesis", ECCV 2020

# Neural Radiance Fields

Mildenhall et al, “Representing Scenes as Neural Radiance Fields for View Synthesis”, ECCV 2020

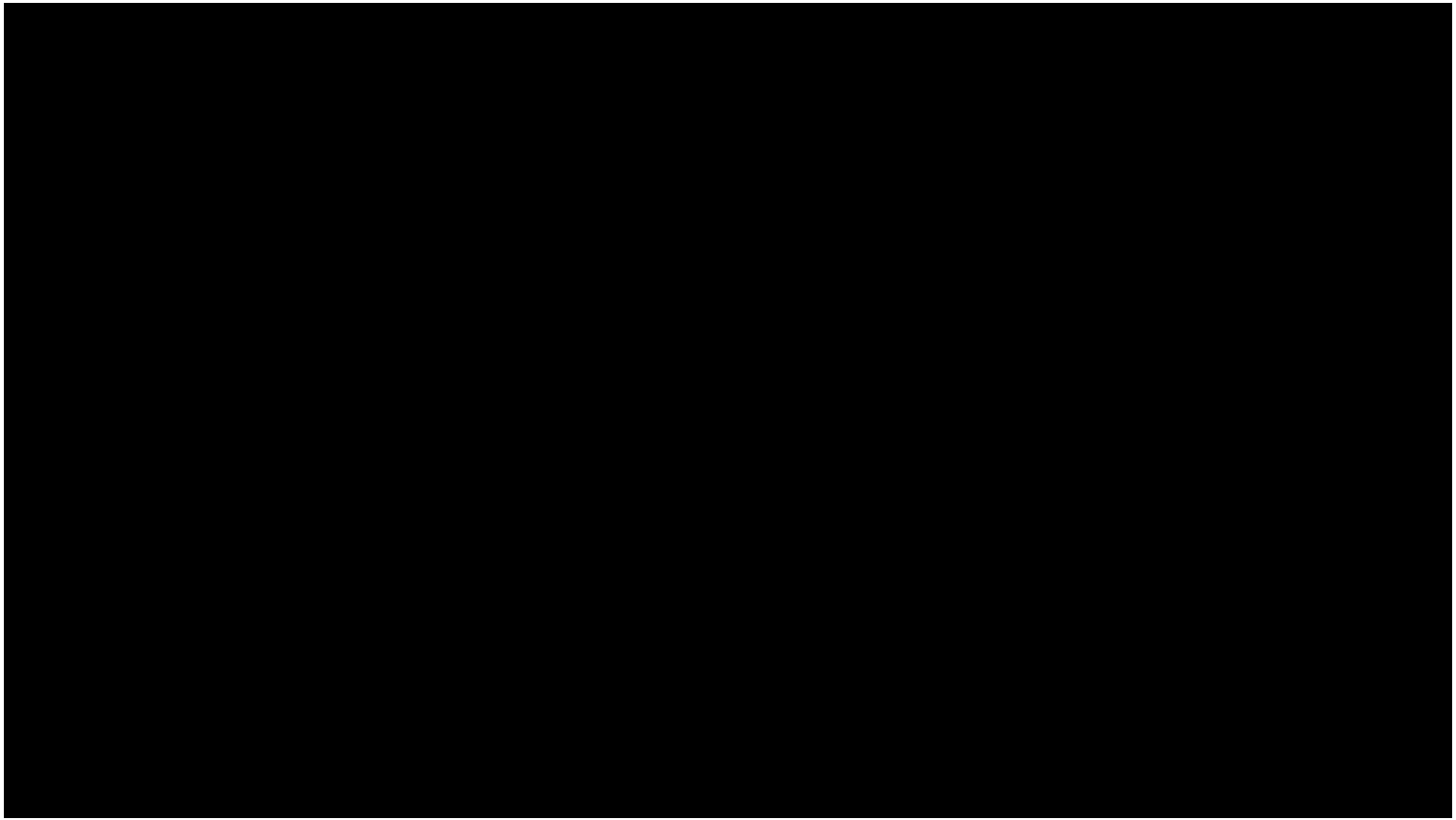
# Neural Radiance Fields

Mildenhall et al, “Representing Scenes as Neural Radiance Fields for View Synthesis”, ECCV 2020

# Dynamic NeRF: Deformable Scenes

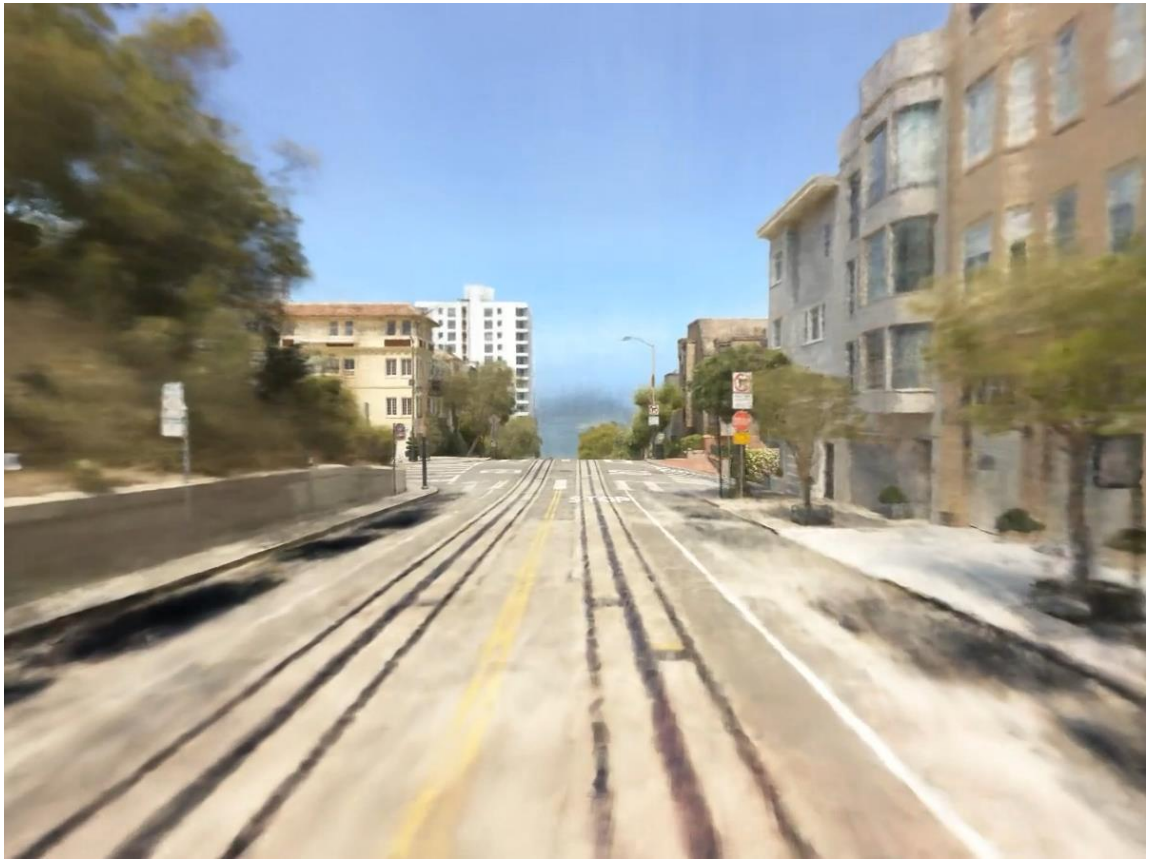

Park et al, “Nerfies: Deformable Neural Radiance Fields”, ICCV 2021

# RawNeRF: High-Dynamic Range Imagery



Mildenhall et al, “NeRF in the Dark: High Dynamic Range View Synthesis from Noisy Raw Images”, CVPR 2022

# BlockNeRF: A Neighborhood of San Francisco



Tancik et al, “Block-NeRF: Scalable Large Scene Neural View Synthesis”, CVPR 2022

Main Problem: Very slow!

**Training**: 1-2 days on a V100 GPU, for just a single scene!

**Inference**: Sampling an image from a trained model:
(256 x 256 pixels) x (224 samples per pixel)
= 14.6M forward passes through MLP

Mildenhall et al, “Representing Scenes as Neural Radiance Fields for View Synthesis”, ECCV 2020

# 3D Gaussian Splatting


**Kerbl and Kopanas et al., 2023**

# 3D Gaussian Splatting


Query a **continuous MLP** along the ray

$$(x,y,z,\theta,\phi) \rightarrow F_\Theta \rightarrow (RGB\sigma)$$

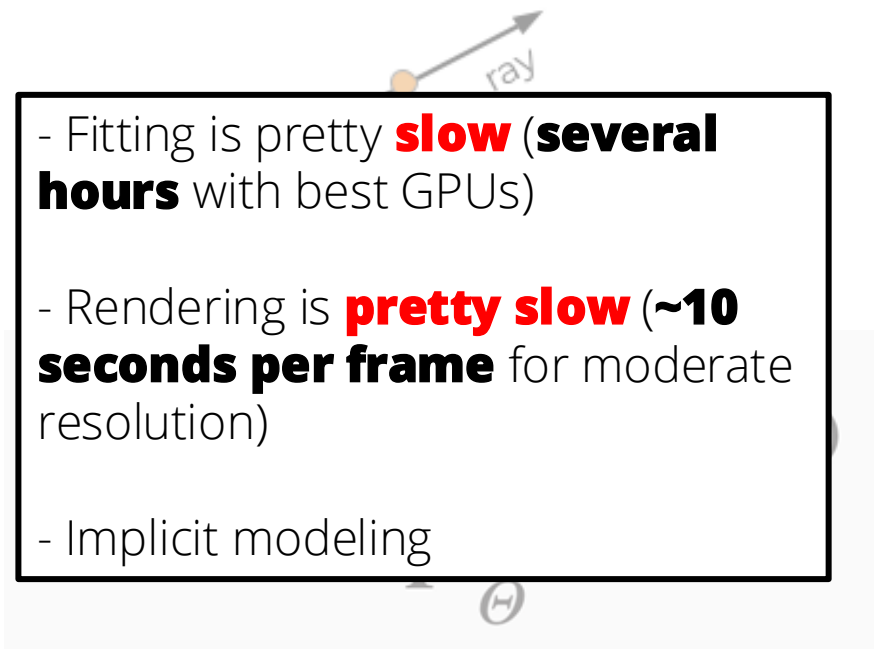
**NeRF**


Blend a **discrete set of Gaussians** along the ray


$$\rightarrow (RGB\sigma)$$

**3D Gaussian Splatting**

# 3D Gaussian Splatting

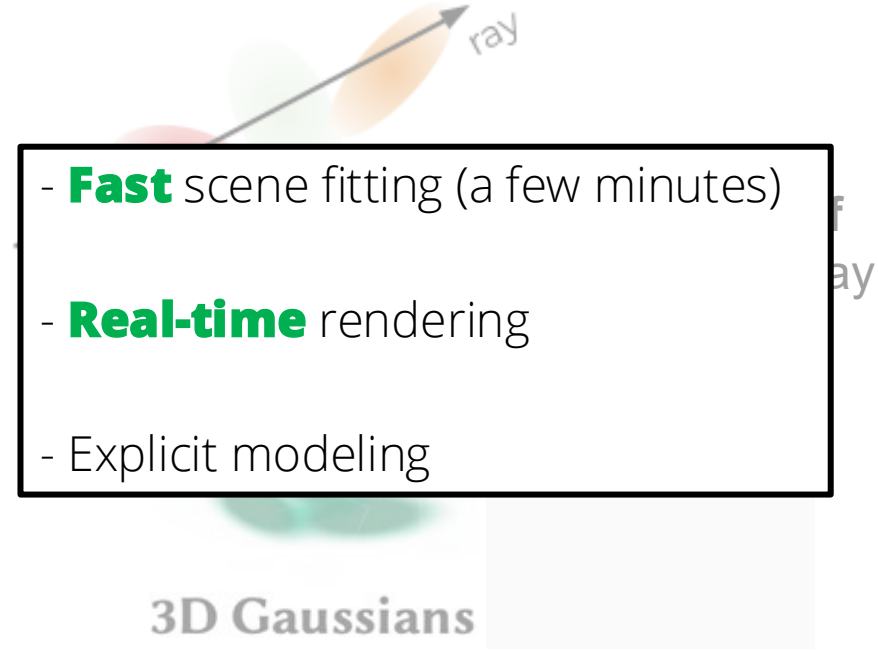



- Fitting is pretty **slow** (**several hours** with best GPUs)

- Rendering is **pretty slow** (**~10 seconds per frame** for moderate resolution)

- Implicit modeling

**NeRF**




- **Fast** scene fitting (a few minutes)

- **Real-time** rendering

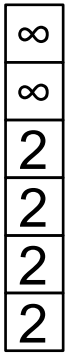
- Explicit modeling

**3D Gaussian Splatting**

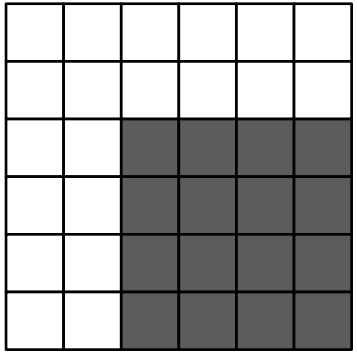
# Dynamic 3D Gaussians

Luiten et al, “Dynamic 3D Gaussians: Tracking by Persistent Dynamic View Synthesis”, 3DV 2024

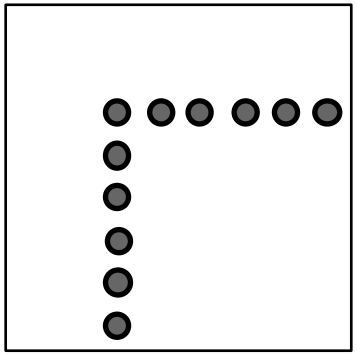
# Summary: 3D Shape Representations



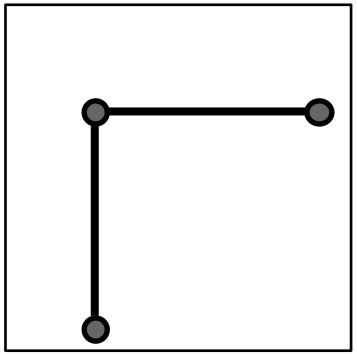

Depth Map



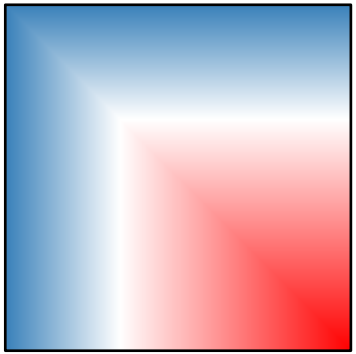
Voxel Grid



Pointcloud

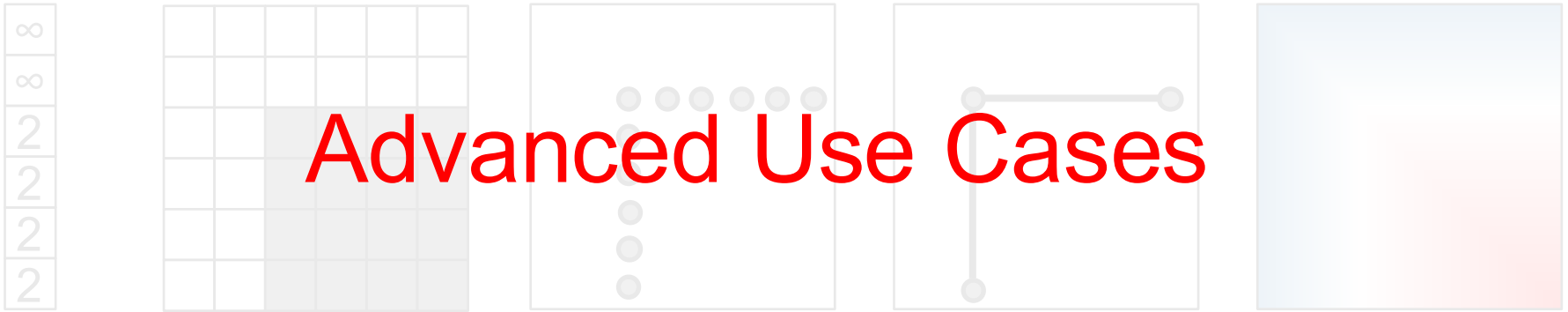


Mesh



Implicit Surface

# Summary: 3D Shape Representations




Depth Map

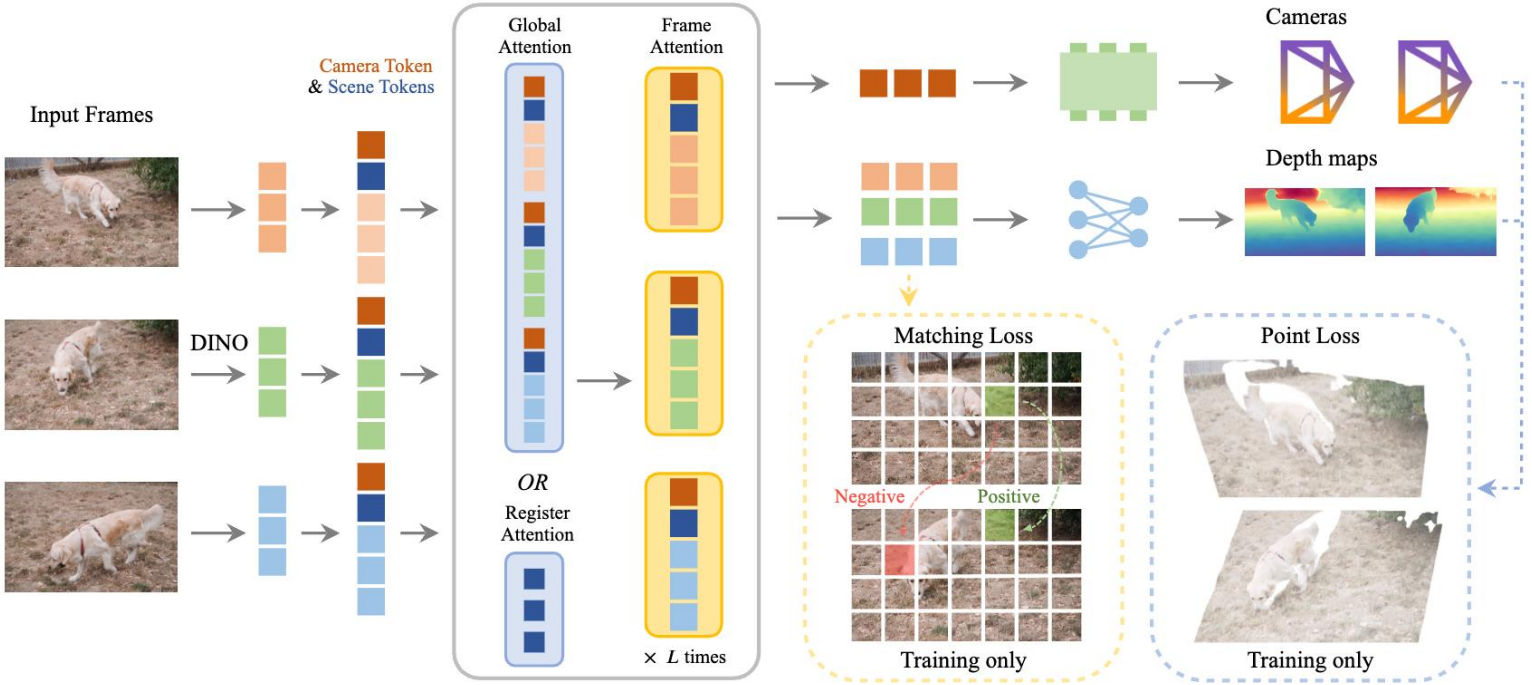
Voxel Grid

Pointcloud

Mesh

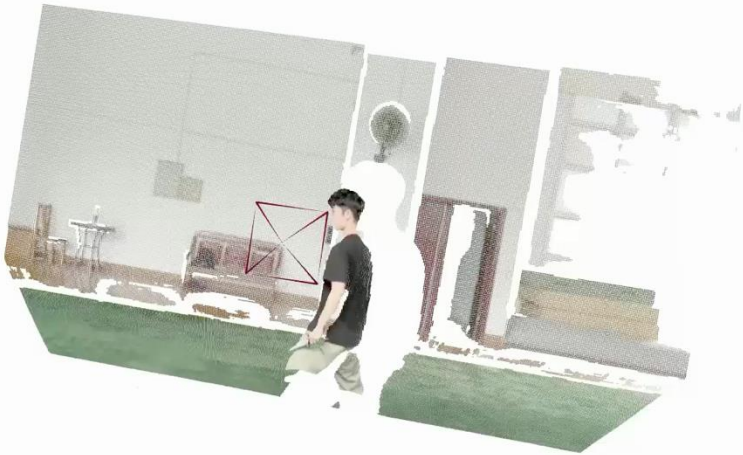
Implicit Surface

# Modern SLAM Models: VGGT Series



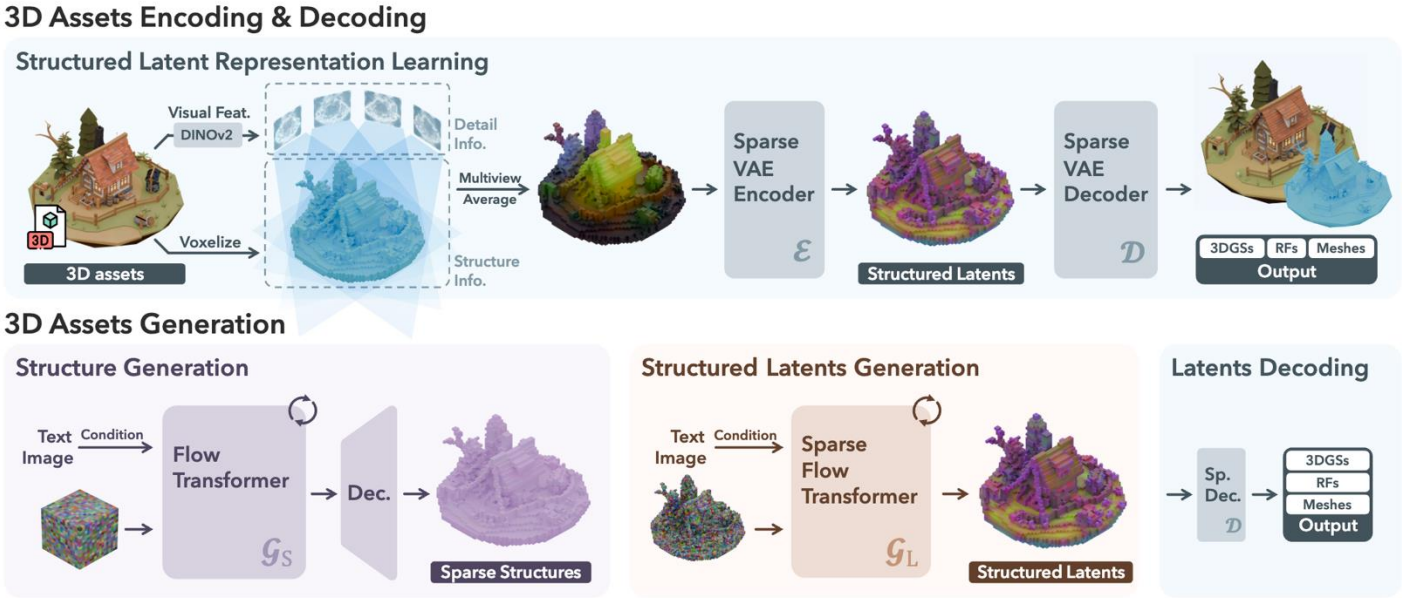

Wang, Jianyuan, et al. "VGGT-$\Omega$." *arXiv preprint arXiv:2605.15195* (2026).

# Modern SLAM Models: VGGT Series



Wang, Jianyuan, et al. "VGGT-$\Omega$." *arXiv preprint arXiv:2605.15195* (2026).

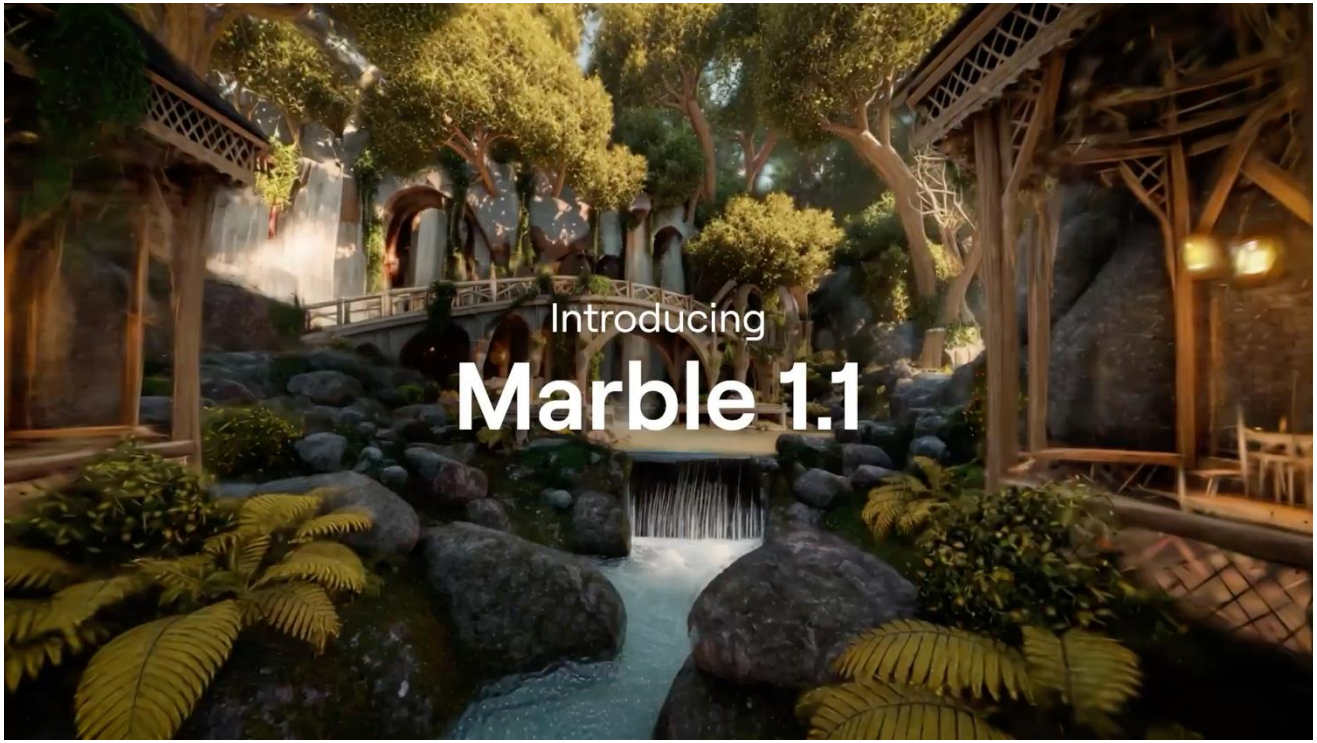
# Modern 3D Object Generation Models: TRELLIS Series




Xiang, Jianfeng, et al. "Native and compact structured latents for 3d generation." *arXiv preprint arXiv:2512.14692* (2025).


TRELLIS.2
AN OPEN-SOURCE
IMAGE TO 3D GENERATION MODEL.

# Modern 3D World Generation Models: Marble




Marble1.1 @ The World Labs

# Next time: Vision and Language