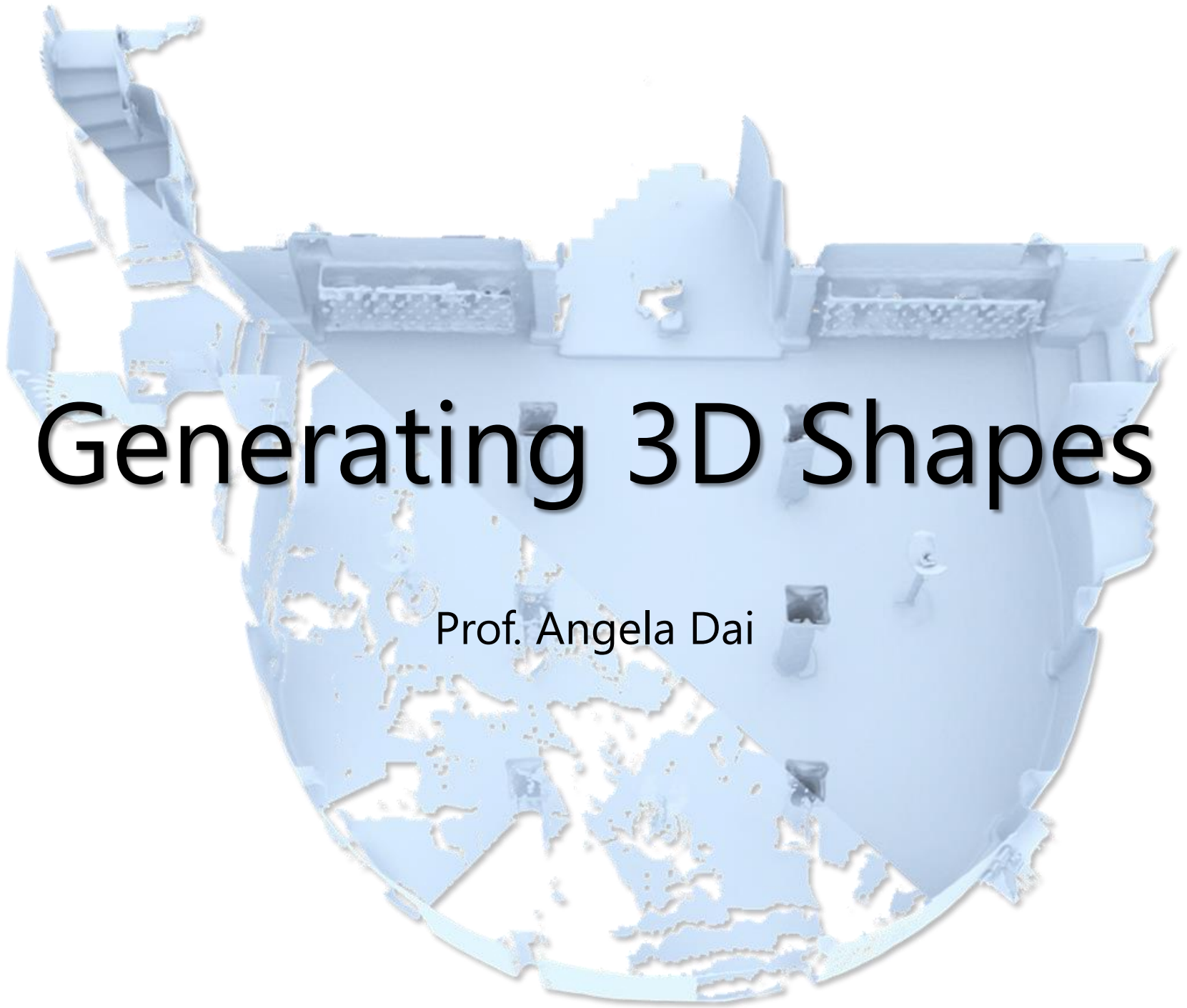
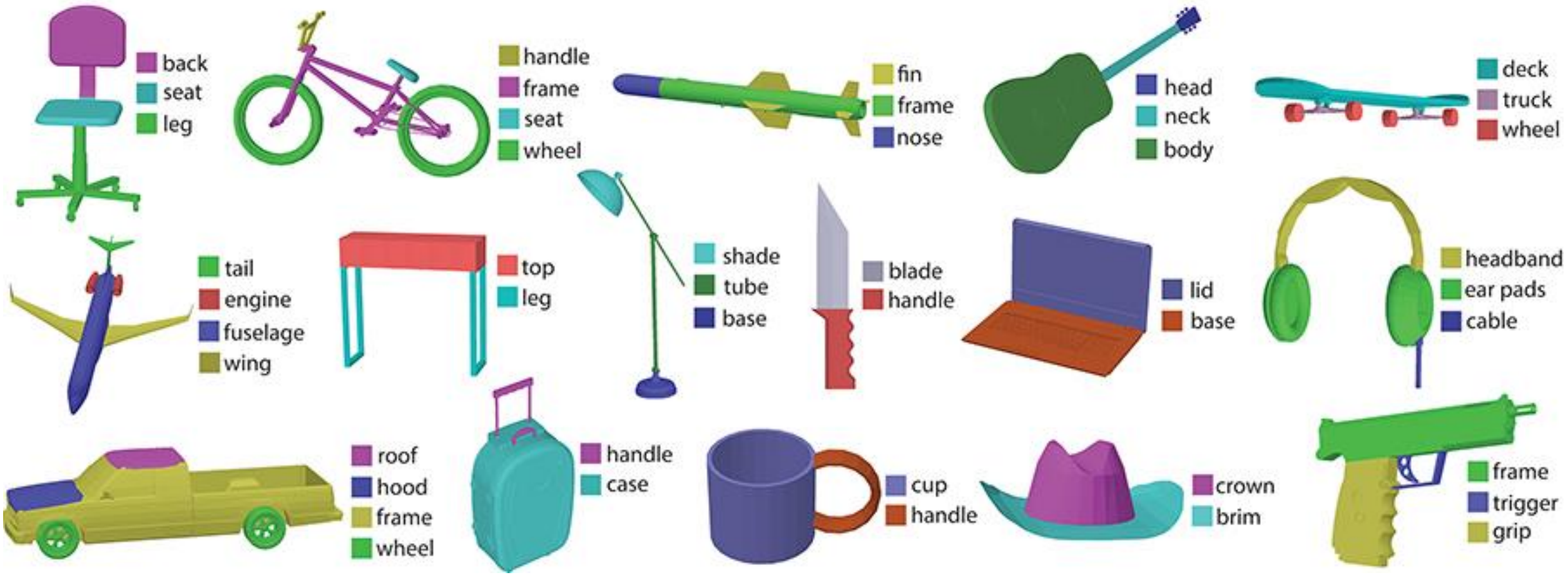


# Generating 3D Shapes

Prof. Angela Dai

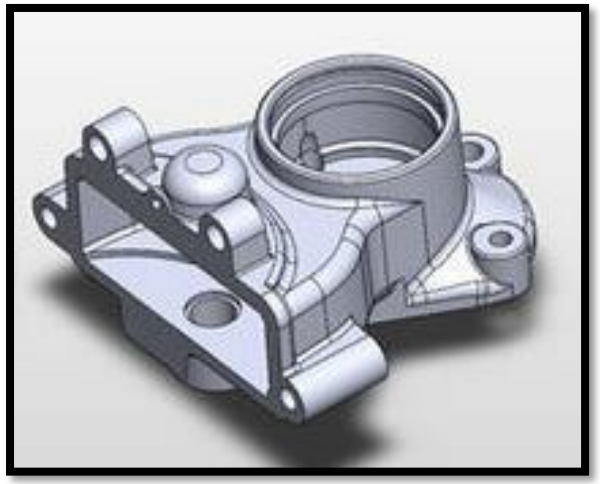
# Last time

# Shape segmentation into parts



# Generating Shapes

Professional 3D Modeling (e.g., Autodesk Maya)



Mechanical CAD Design

3D Perception

# Generating Shapes

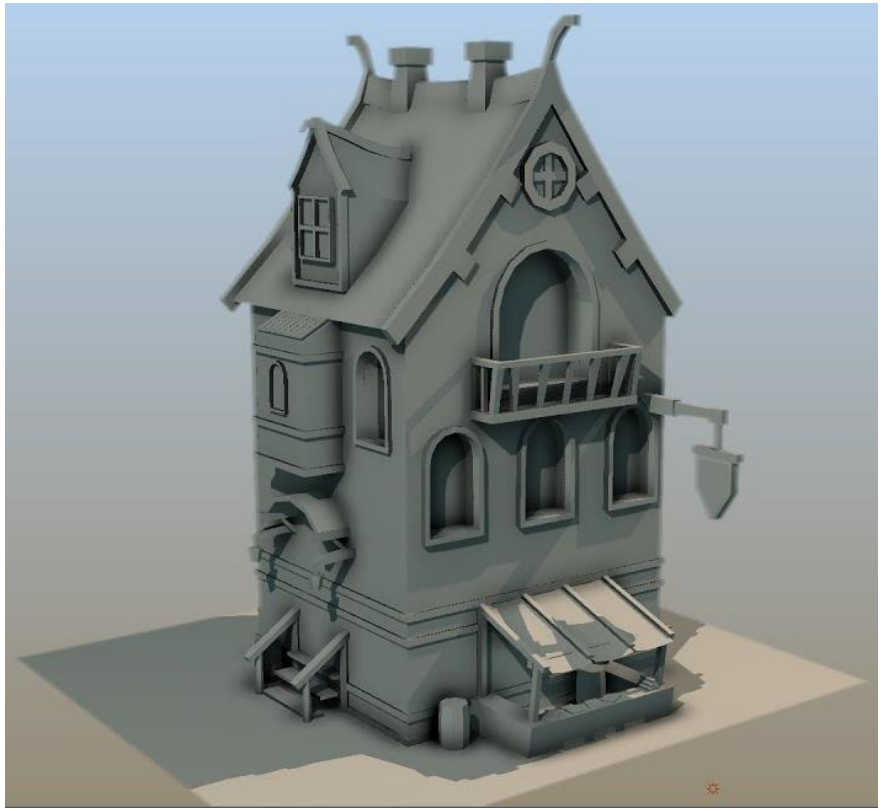
- Democratizing content creation

➢Enable amateurs to create quality 3D models

- Help guide professionals

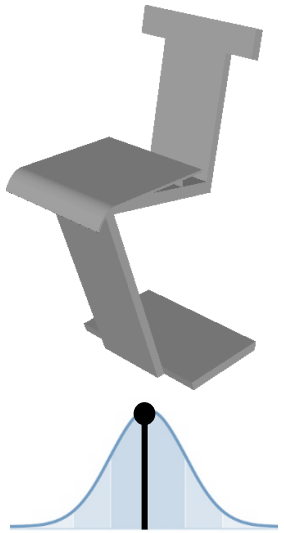
➢Reduce tedious modeling work

- Perceiving 3D structures from partial observations (real-world scenarios)

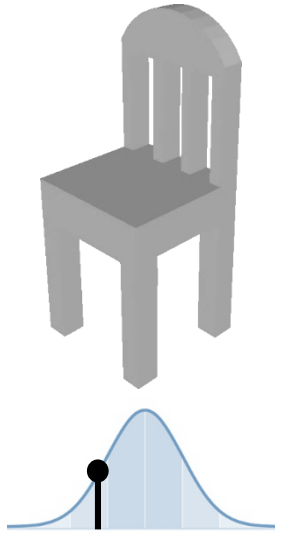


# Generative Shape Tasks

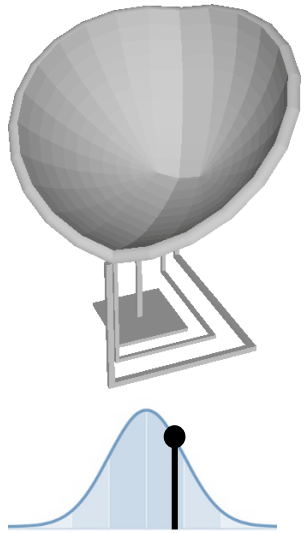
- Unconditional generative model:
  - Sample new geometry and/or appearance



$z_0$



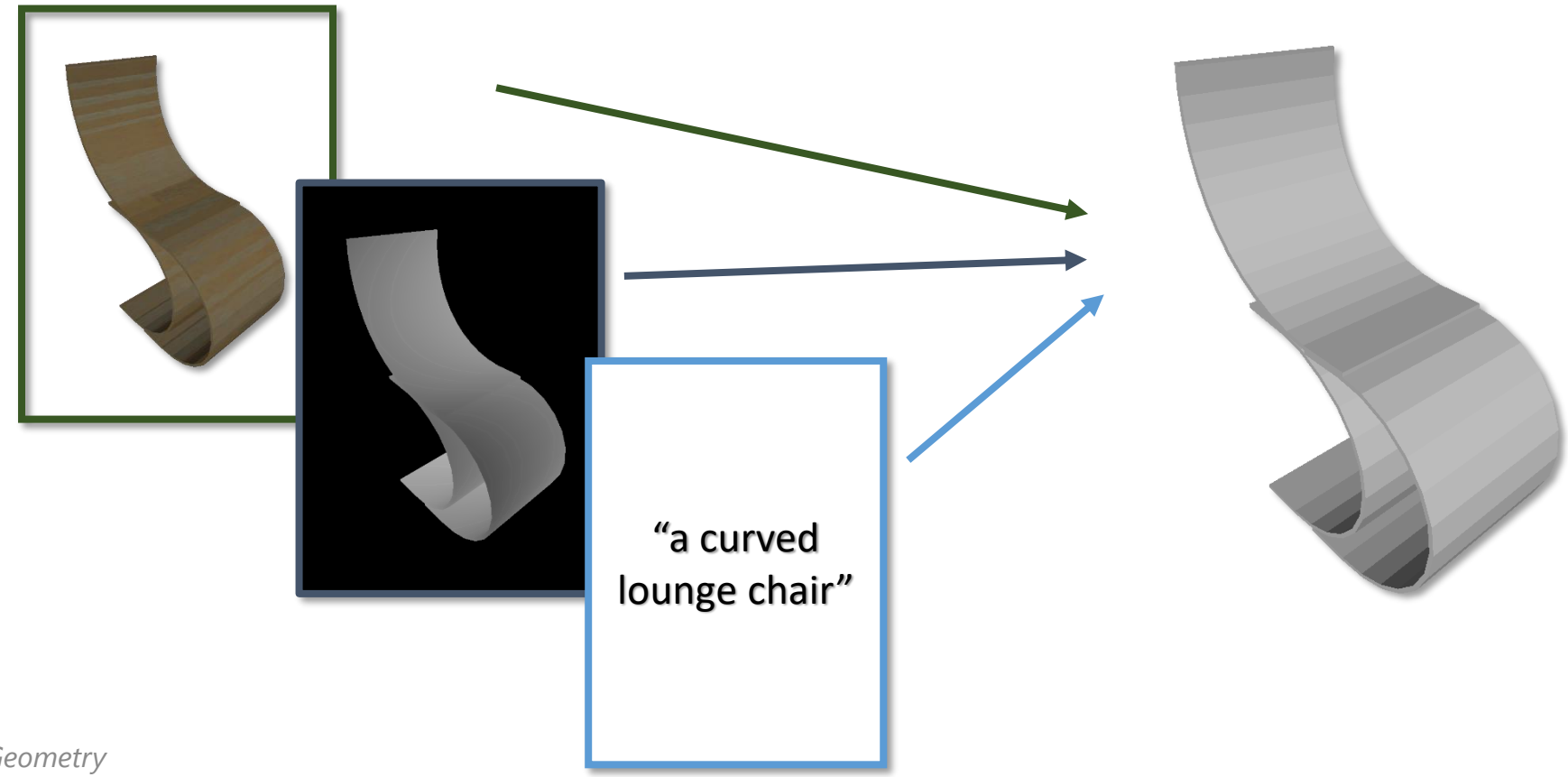
$z_1$



$z_2$

# Generative Shape Tasks

- Conditional generative model:
  - Reconstruct geometry and/or appearance from input observation



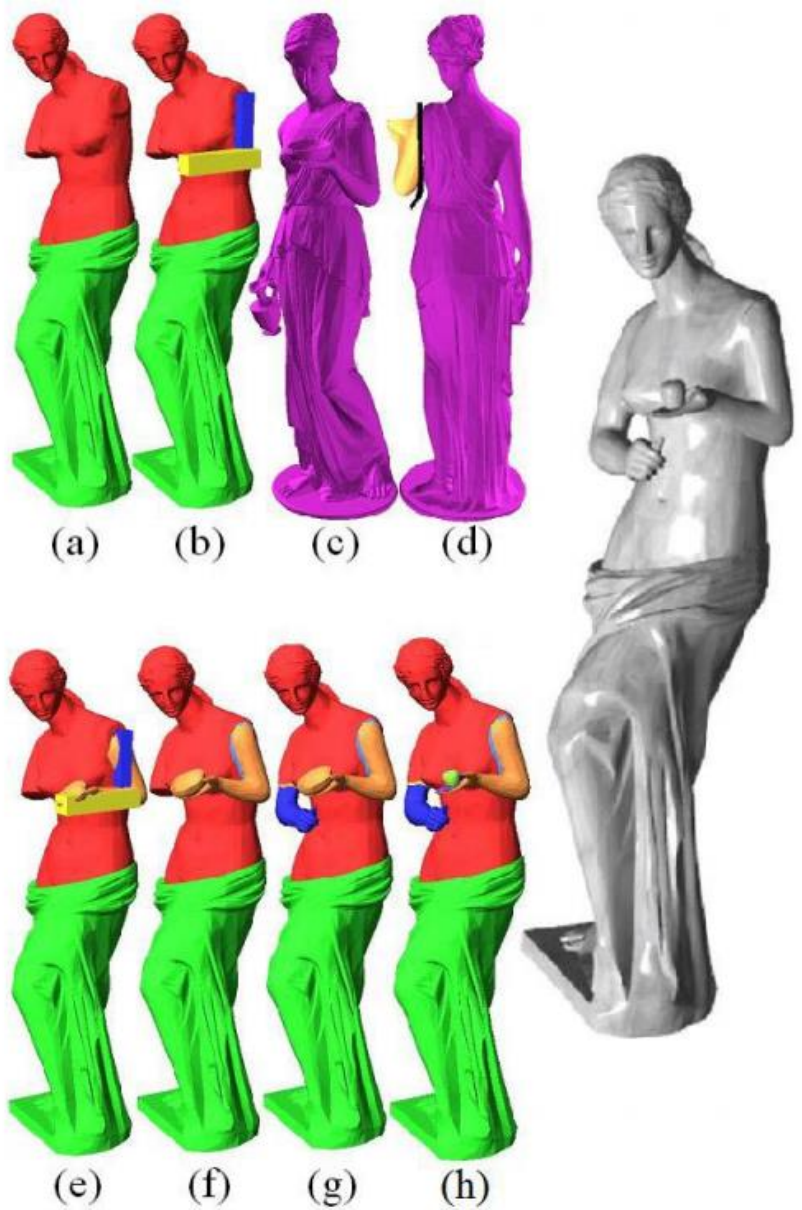
# Modeling by Example

- From an initial model:
- Select a part of the model to edit
- Search 3D database for similar parts
- Composite selected part into model
- Repeat

[Funkhouser et al. '04]

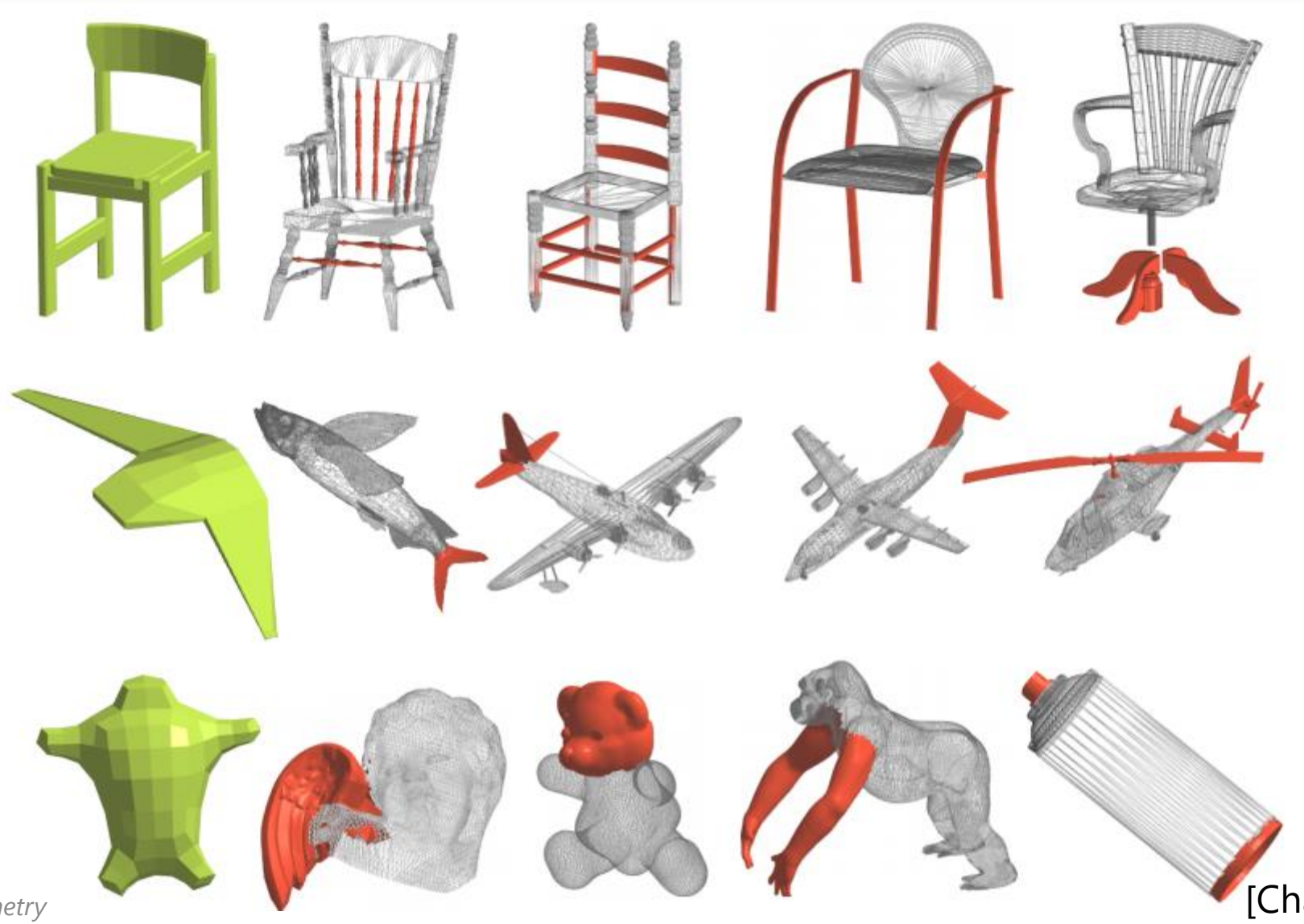
# Modeling by Example

- From an initial model:
- Select a part of the model to edit
- Search 3D database for similar parts
- Composite selected part into model
- Repeat




[Funkhouser et al. ‘04]

# Part Suggestions for Creativity Support



[Chaudhuri and Koltun ‘10]

# Spore Creature Creator

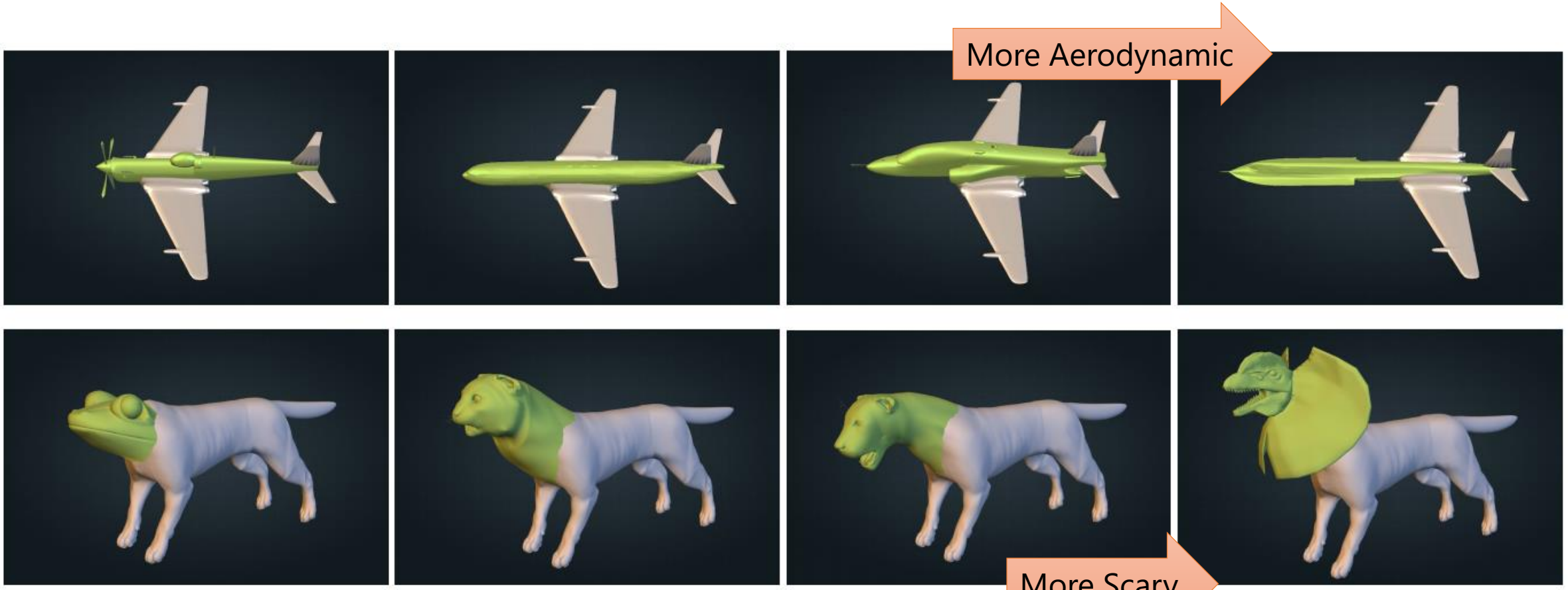
- Modeling by parts

# Semantic Basis for Part Suggestions


[Chaudhuri et al. '13]

# Semantic Basis for Part Suggestions

- Crowdsourced annotations for semantic attributes




[Chaudhuri et al. '13]

# Object Reconstruction

- Perceiving object structures

From a 3D scan

From a single RGB image

# Shape Reconstruction from a Depth Image

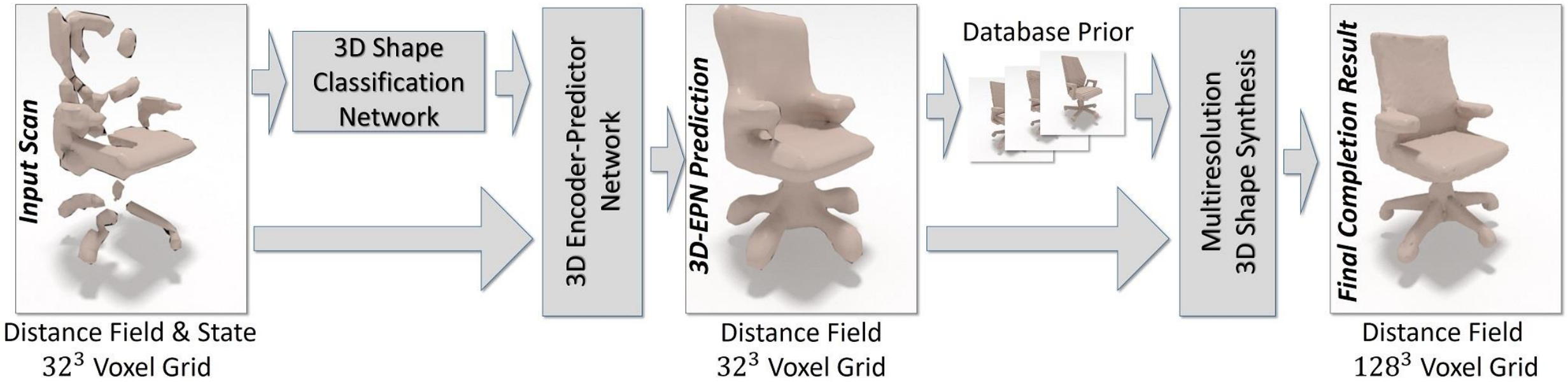
- Commodity range sensors are increasingly available
- Depth image data gives strong 3D prior
- Depth data is noisy and incomplete


Microsoft Kinect


iPhone 12 Pro

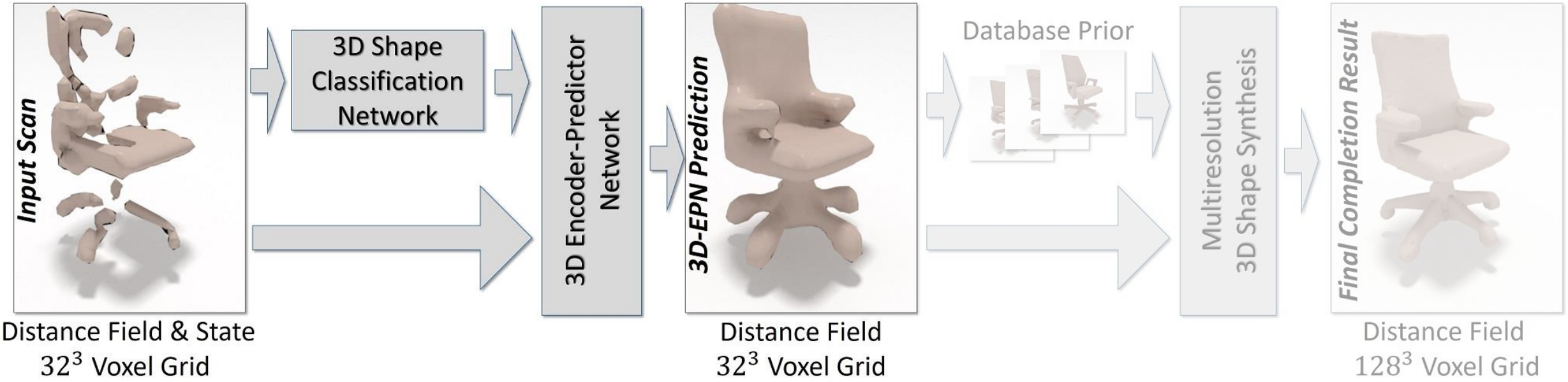
Intel RealSense

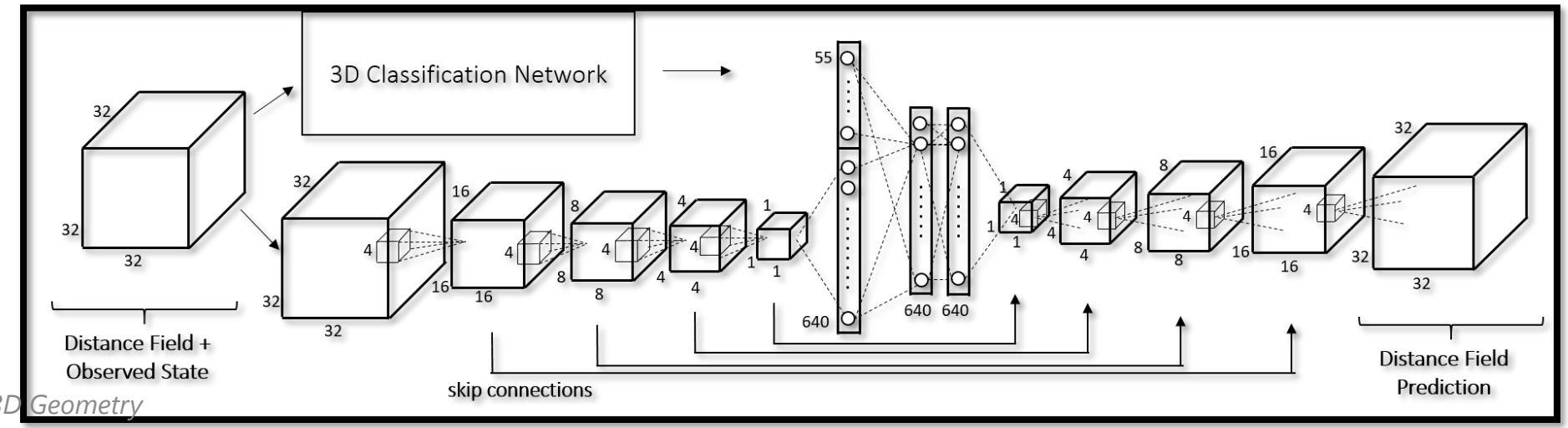
# 3D-EPN for Shape Completion




[Dai et al. '17]

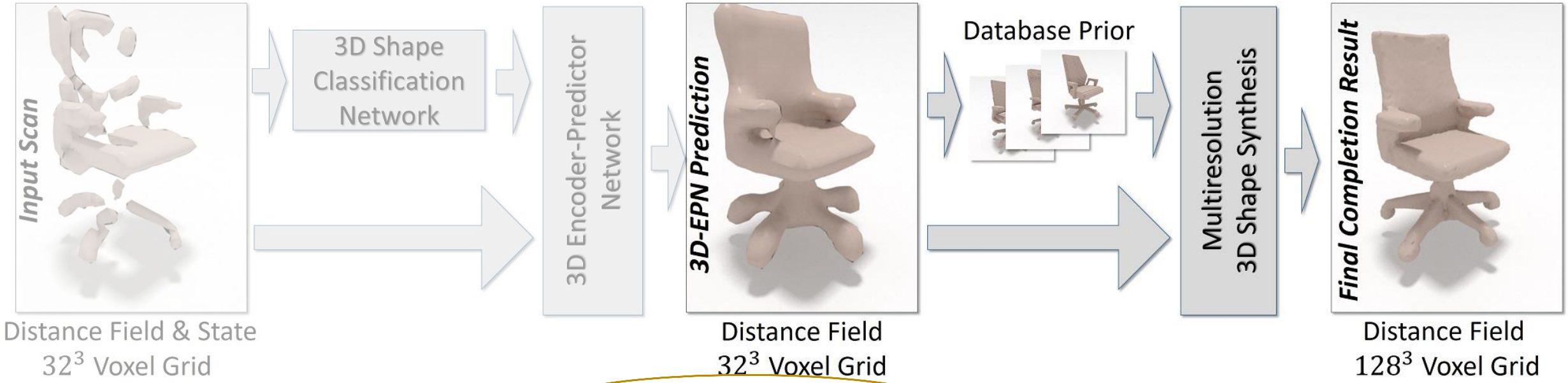
# 3D-EPN for Shape Completion




[Dai et al. '17]

# 3D-EPN for Shape Completion





[Dai et al. '17]

# 3D-EPN for Shape Completion

- 3D Representation?

| Surface Rep. | $\ell_1$-Error ($32^3$) | $\ell_2$-Error ($32^3$) |
|---|---|---|
| Binary Grid | 0.653 | 1.160 |
| Ternary Grid | 0.567 | 0.871 |
| Distance Field | 0.417 | 0.483 |
| Signed Distance Field | **0.379** | **0.380** |



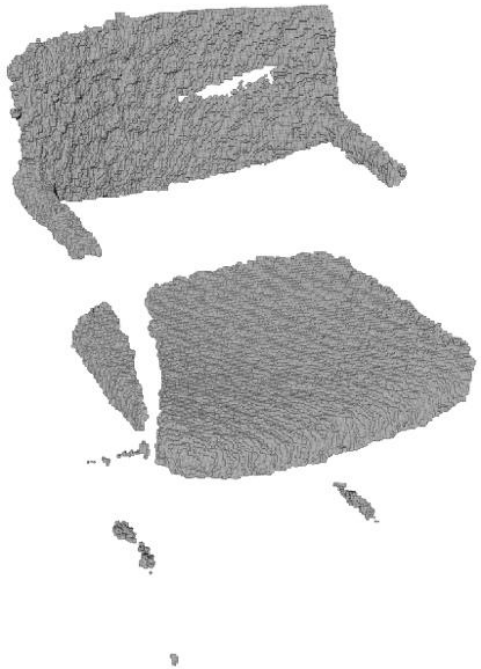
[Dai et al. '17]

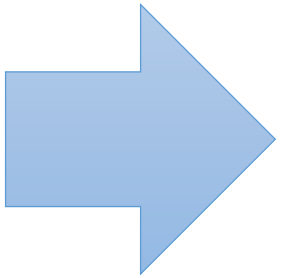
# 3D-EPN for Shape Completion

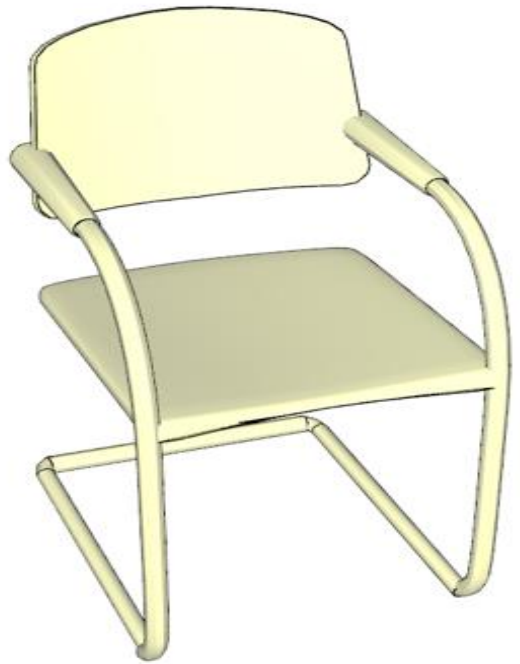

[Dai et al. '17]

# Shape Reconstruction from an Image

- No depth signal
- Color images are more commonly available
- Color images often higher resolution, less noise

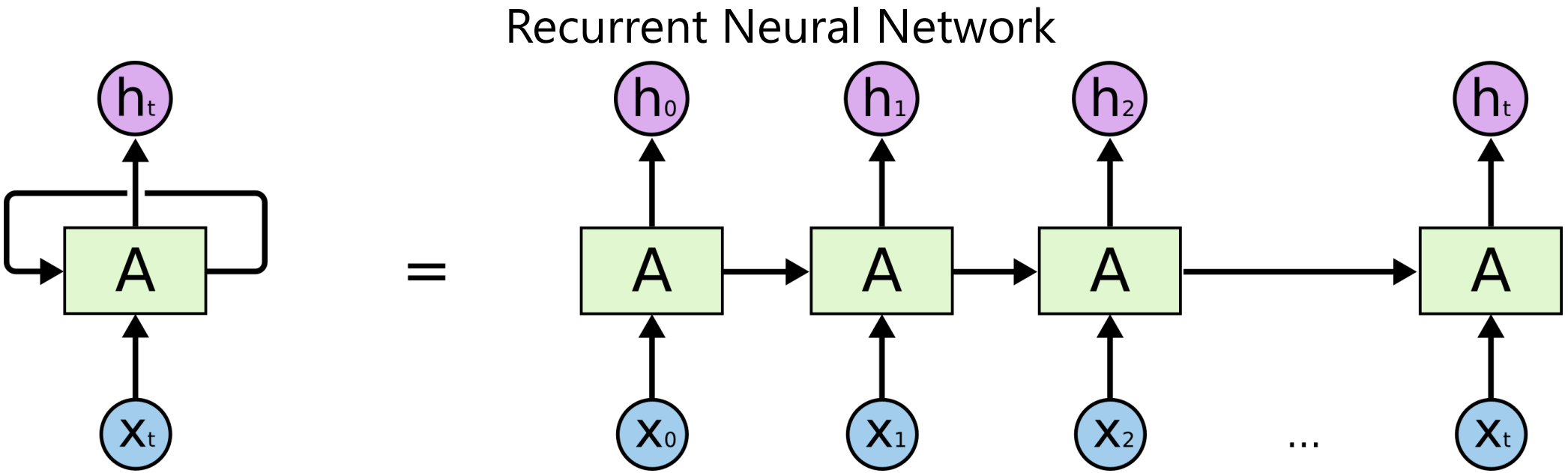






# 3D-R2N2: 3D Recurrent Reconstruction

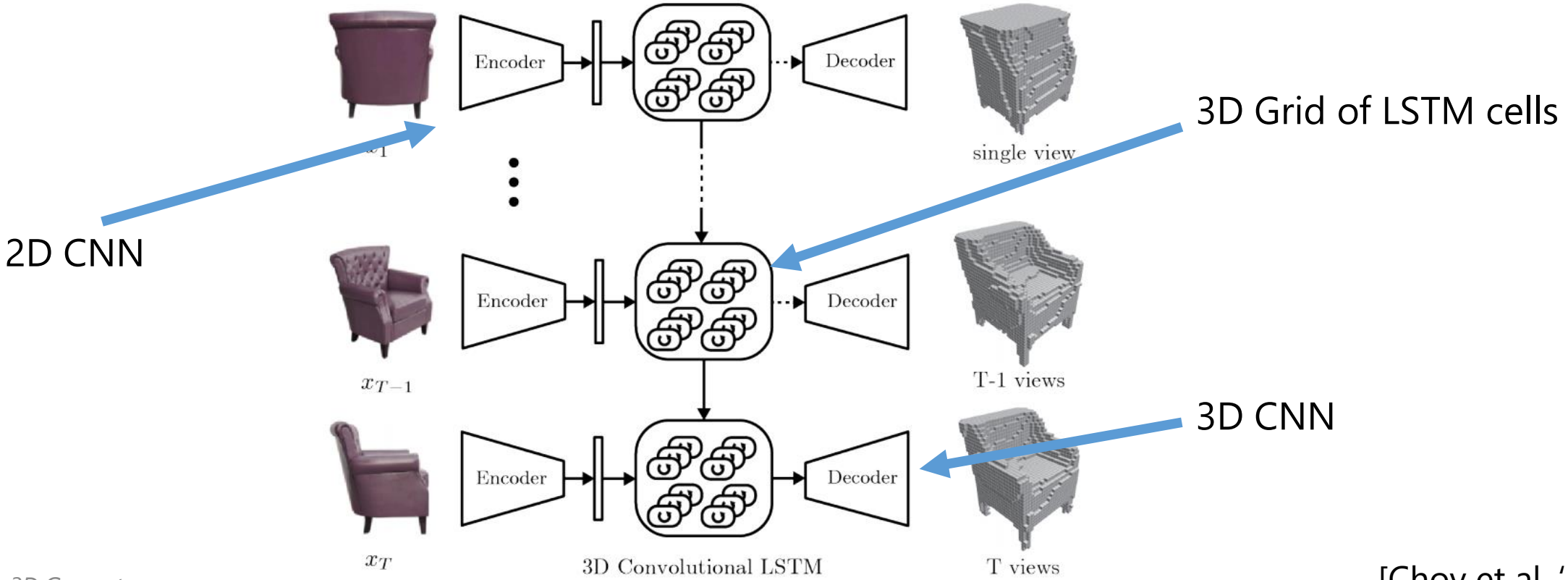
- Multi-view images → 3D volumetric occupancy grid
- Recurrent neural network to fuse multiple color images




http://colah.github.io/posts/2015-08-Understanding-LSTMs/

# 3D-R2N2: 3D Recurrent Reconstruction

- Multi-view images → 3D volumetric occupancy grid
- Recurrent neural network to fuse multiple color images



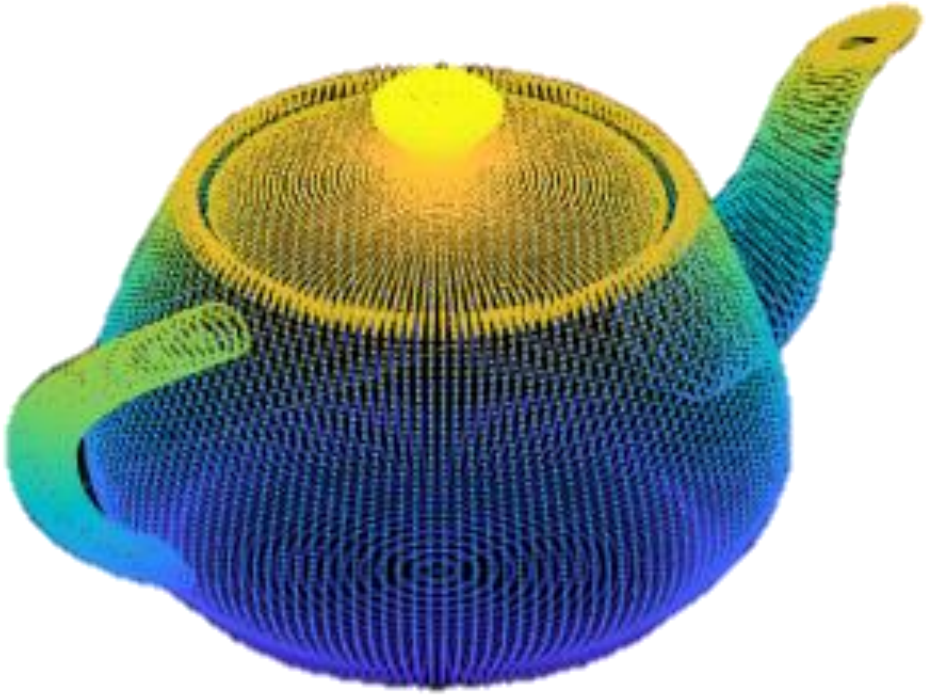

[Choy et al. '16]

# 3D-R2N2: 3D Recurrent Reconstruction

[Choy et al. ‘16]

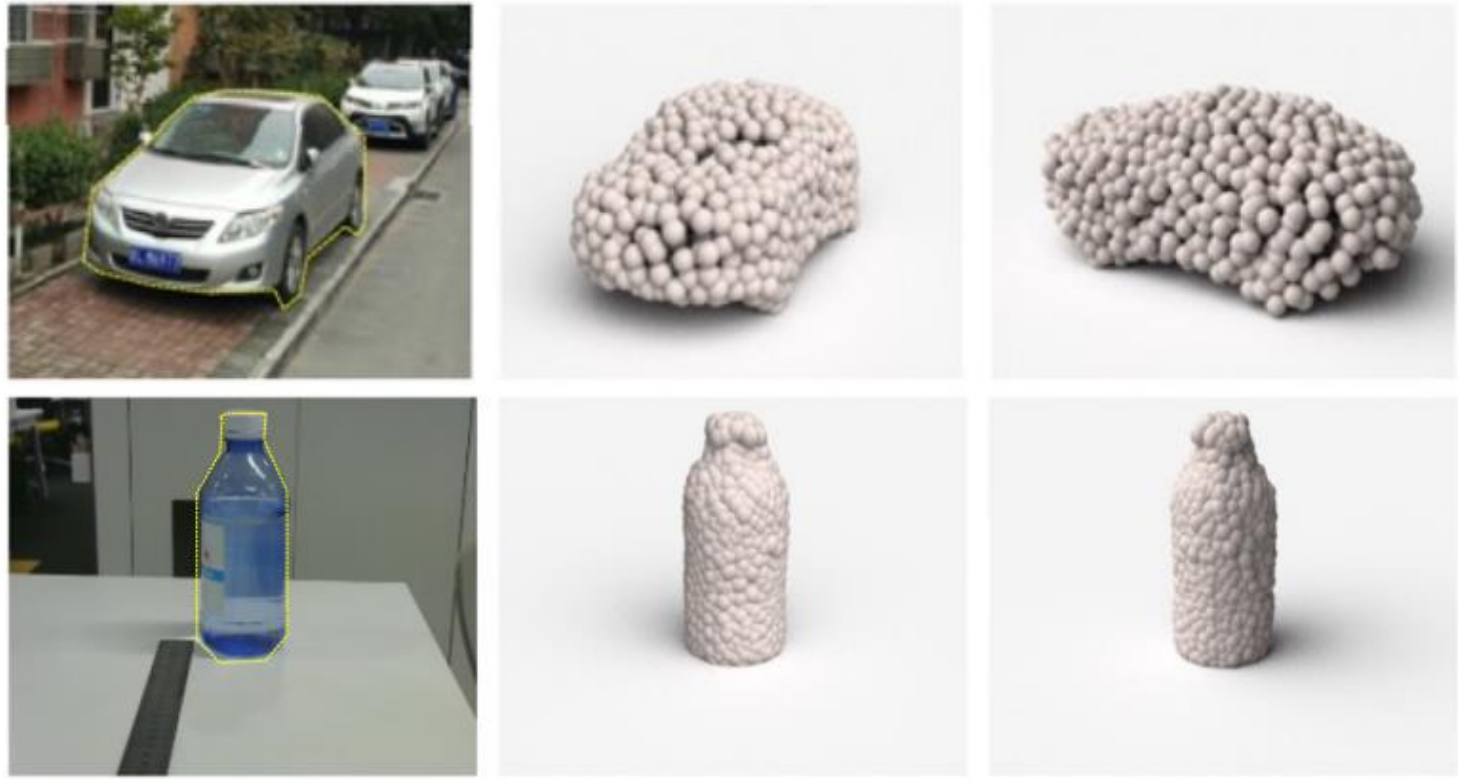
# Generating Point Clouds

- Point Clouds: no surface connectivity, but structural information
- Vs volumetric representations: generate points on surface only



# PSGN: Point Set Generation

- From a image and object segmentation, generate point set



[Fan et al. ‘17]

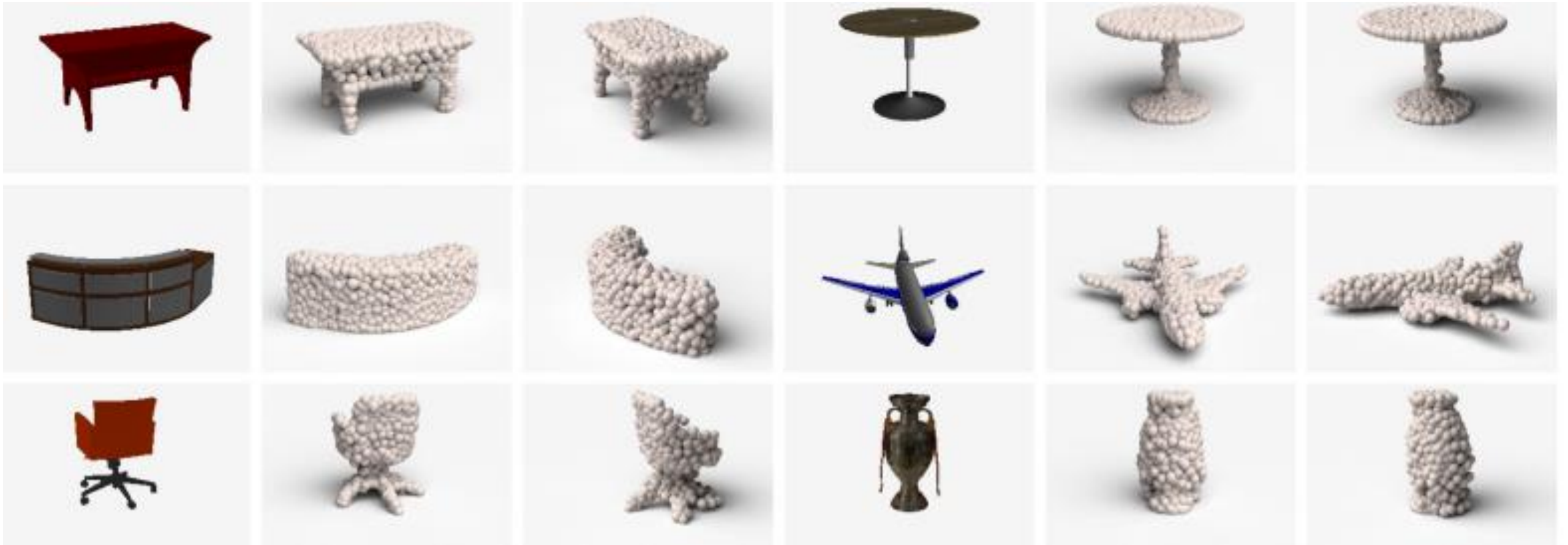
# PSGN: Point Set Generation

- From a image and object segmentation, generate point set
  - 2D CNN for encoding image
  - MLP for point set generation
  - Chamfer distance loss between generated point set $S_p$ vs. target point set $S_t$; $S_p, S_t \subseteq \mathbb{R}^3$:

$$d_{chamfer}(S_p, S_t) = \sum_{x \in S_p} \min_{y \in S_t} \|x - y\|_2^2 + \sum_{y \in S_t} \min_{x \in S_p} \|x - y\|_2^2$$

[Fan et al. '17]

# PSGN: Point Set Generation

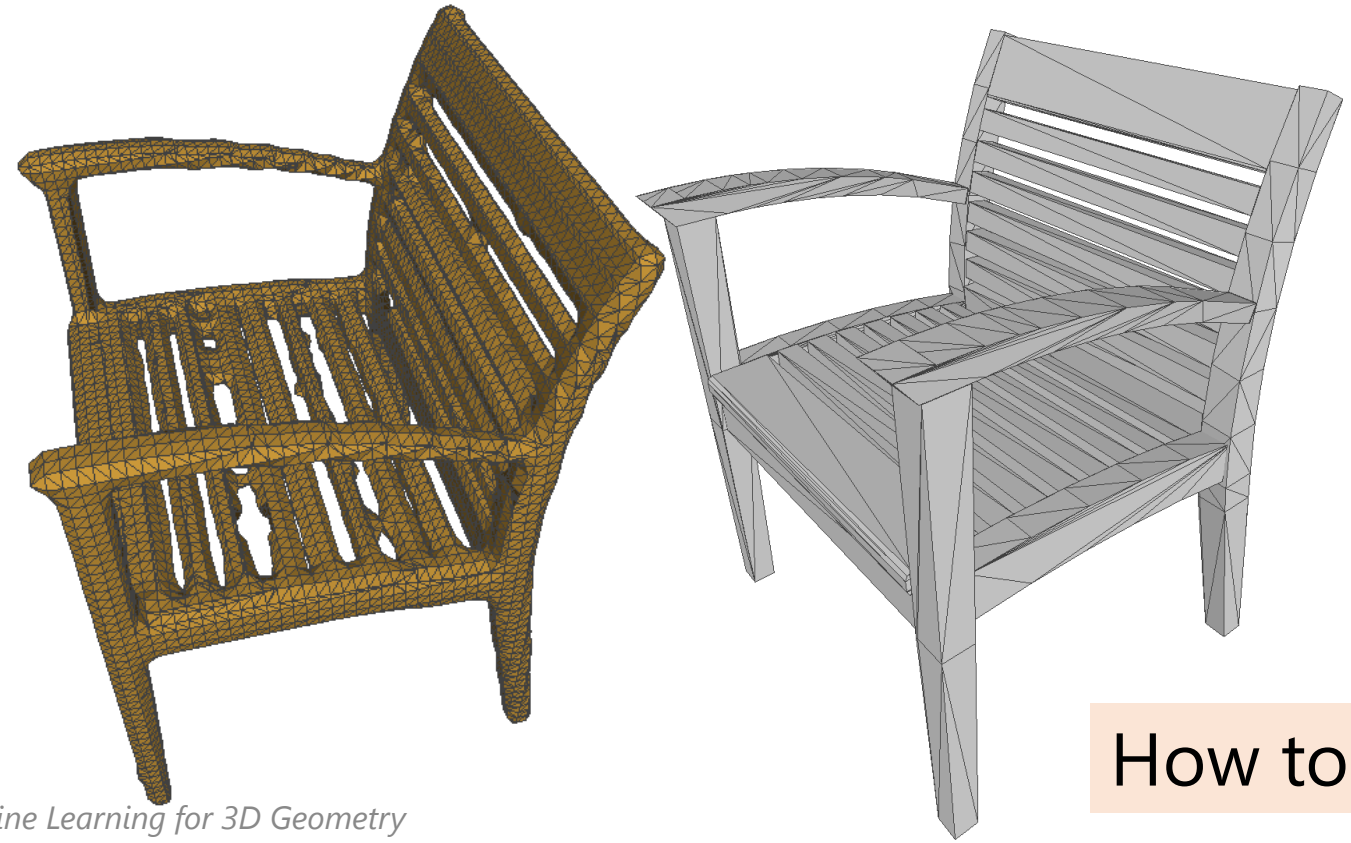


[Fan et al. ‘17]

# Reconstructing Explicit 3D Object Meshes

- Predicting volumetric signed distance fields or occupancy grids

➢Extract an output mesh with Marching Cubes



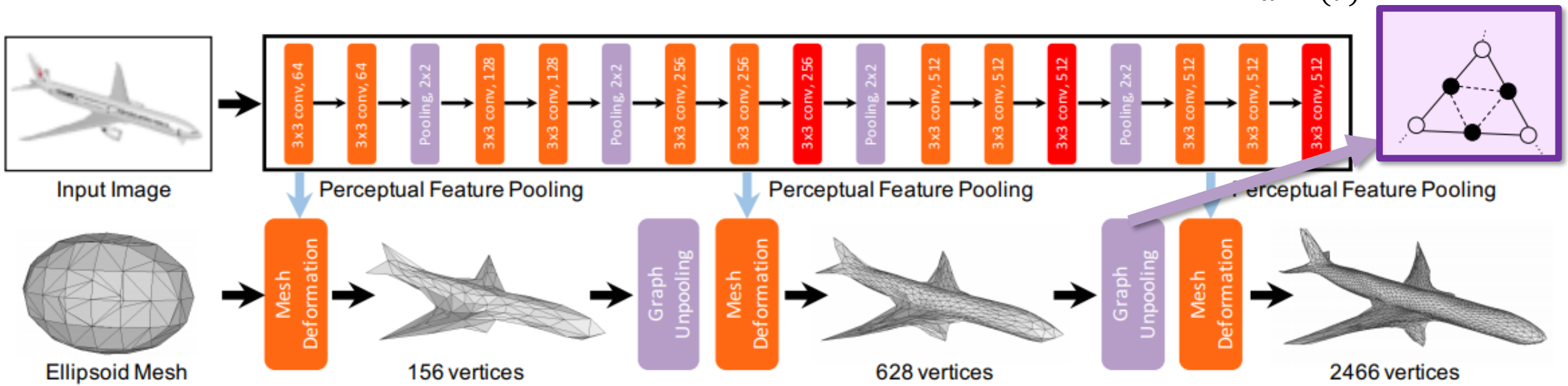
Predicting a mesh directly:

- Enable loss on actual output mesh
- Possibility for more efficient mesh construction

How to predict a mesh representation?

# Deforming a Template Mesh

- Pixel2Mesh: Generating 3D Mesh Models from Single RGB Images
  - Graph neural network to predict vertex displacements
  - Vertex features $f_v \rightarrow$ Apply convs $wf_v \rightarrow$ update $f_v' = w_0 f_v + \sum_{u \in N(v)} w_1 f_u$




Limited to topology of the initial template mesh

[Wang et al. '18]

# Predict a Template + Deformation

- Mesh R-CNN

Input image


2D detection and segmentation

Estimate per-detection shapes as coarse occupancy grids

Convert occupancy grid to mesh and refine

[Gkioxari et al. '19]

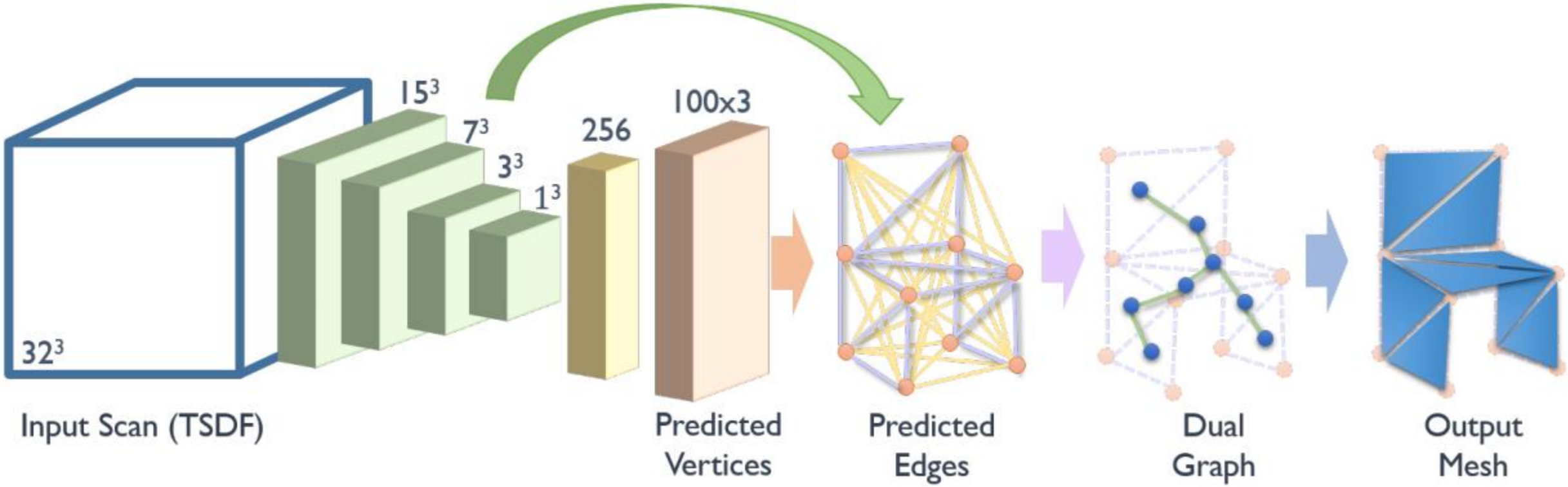
# Freeform Mesh Generation

- Scan2Mesh: From Unstructured Range Scans to 3D Meshes


[Dai and Niessner '19]

# Freeform Mesh Generation

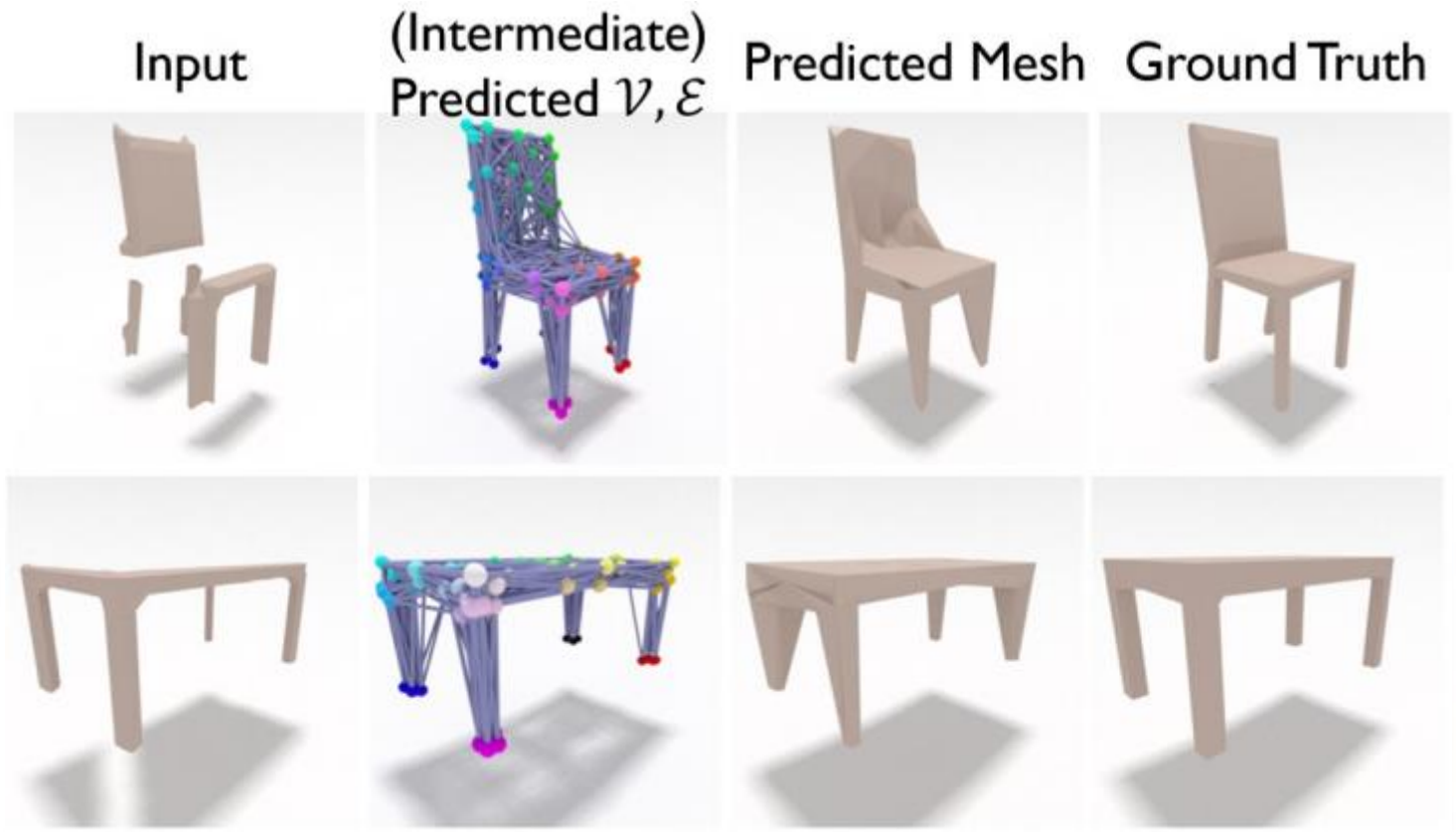
- Scan2Mesh: From Unstructured Range Scans to 3D Meshes




[Dai and Niessner '19]

# Freeform Mesh Generation

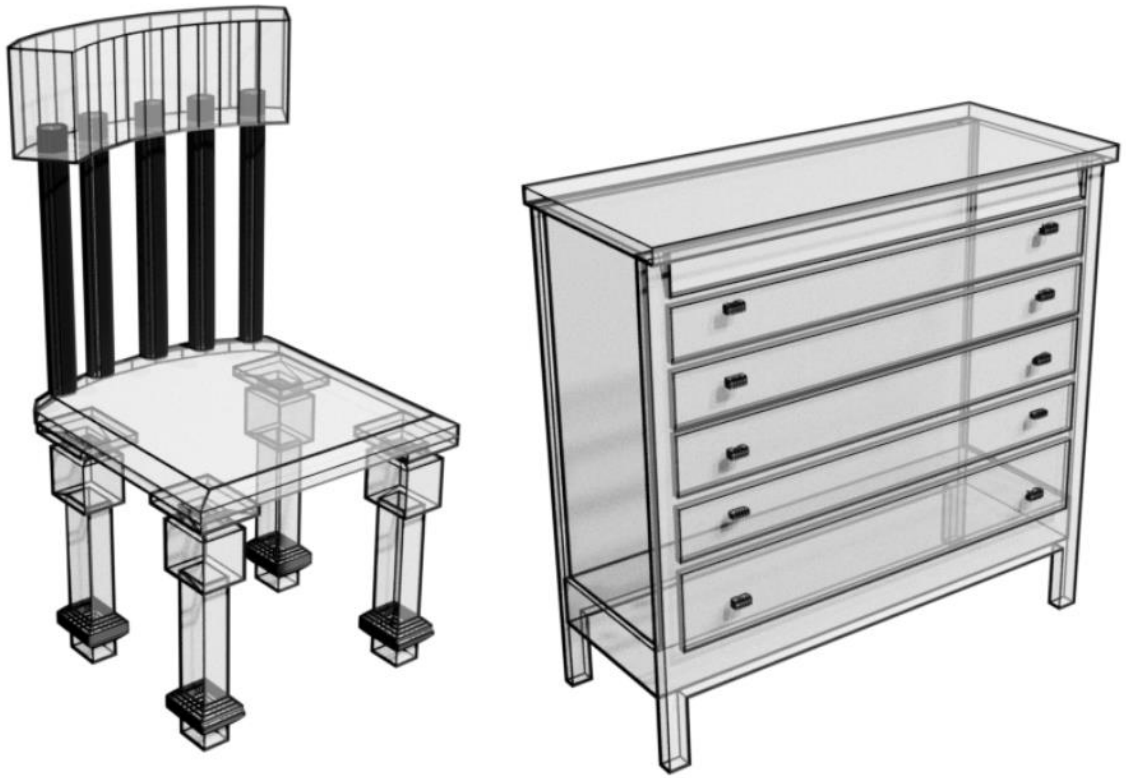
- Scan2Mesh: From Unstructured Range Scans to 3D Meshes




[Dai and Niessner '19]

# Polygon Mesh Generation

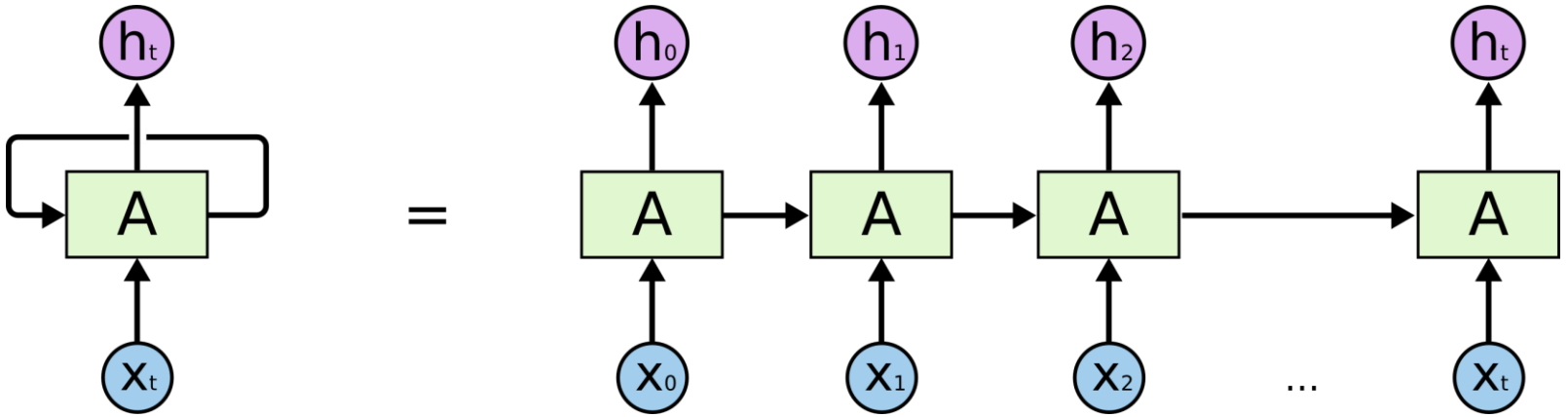
- $n$-gon mesh generation (per class category)
- First generate vertices, then faces conditioned on vertices



[Nash et al. ‘20]

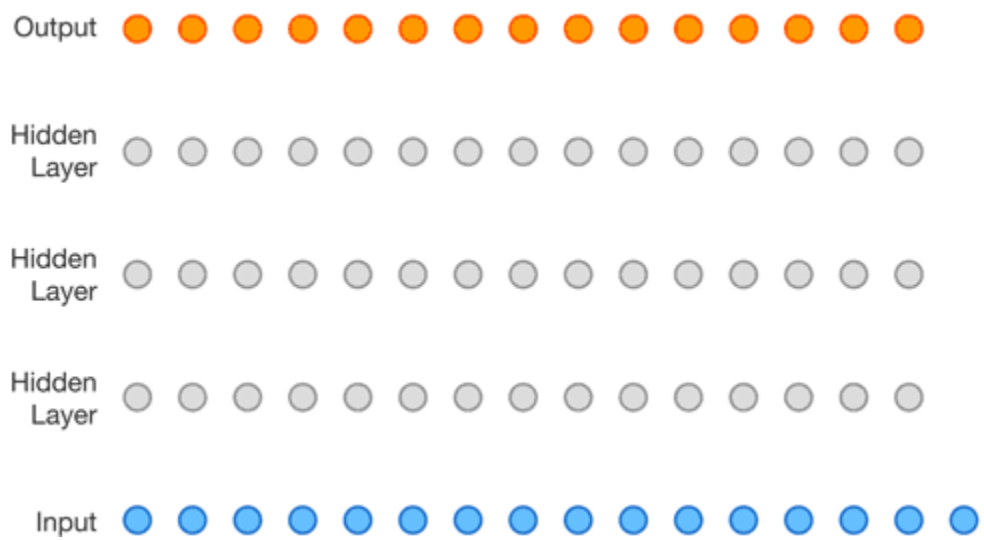
# Autoregressive Models

- Generative model, describing sequence data
- Output depends on previous values
- Recall: RNNs
  - RNN output depends on previous time steps via hidden state



# Autoregressive Models

- Generative model, describing sequence data
- Recall: RNNs
  - RNN output depends on previous time steps via hidden state
- Autoregressive output depends on previous values given as input

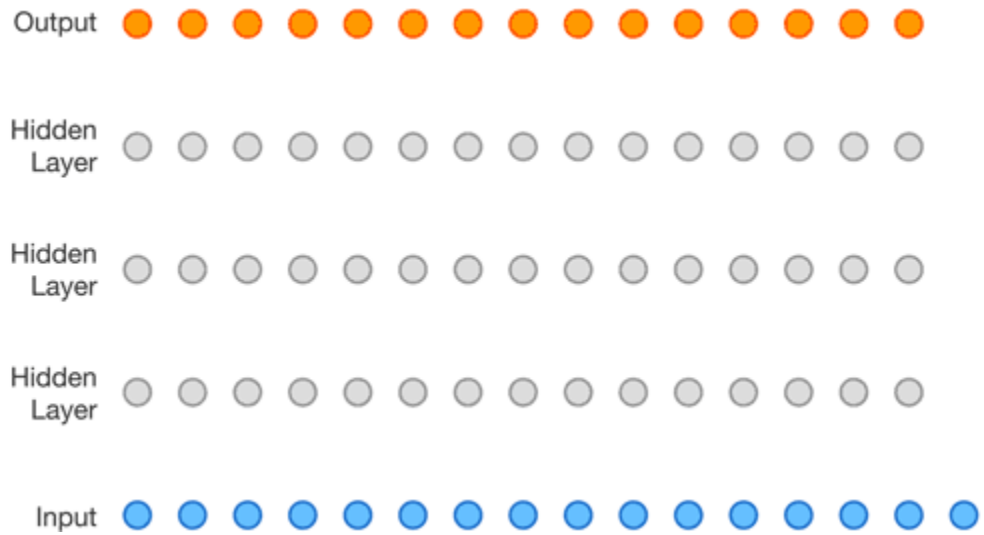





https://www.deepmind.com/blog/wavenet-a-generative-model-for-raw-audio

# Autoregressive Models

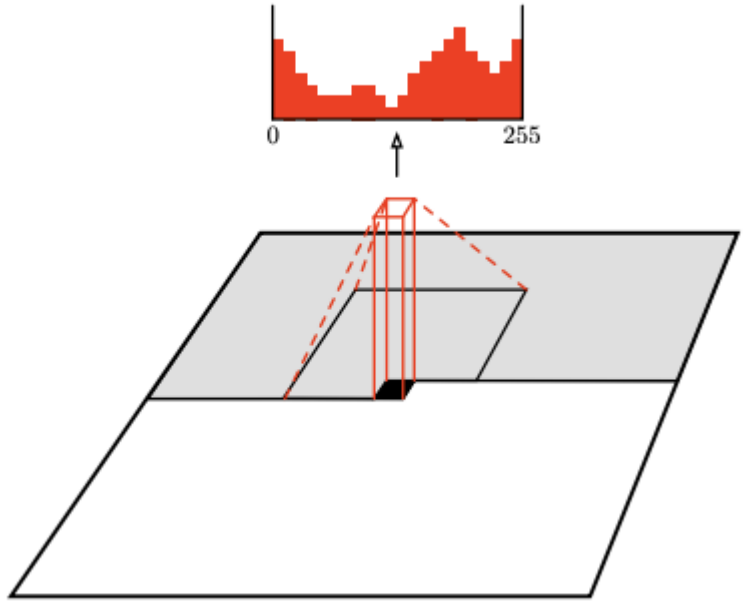
- Generative model, describing sequence data
- Autoregressive output depends on previous values given as input
- Feed-forward model predicting future values from past values




https://www.deepmind.com/blog/wavenet-a-generative-model-for-raw-audio

# Autoregressive Models

- Can also be applied to data that is not sequential in nature
- For example: images
  - Generate pixel by pixel from top-left to bottom-right




[Van den Oord et al. '16]

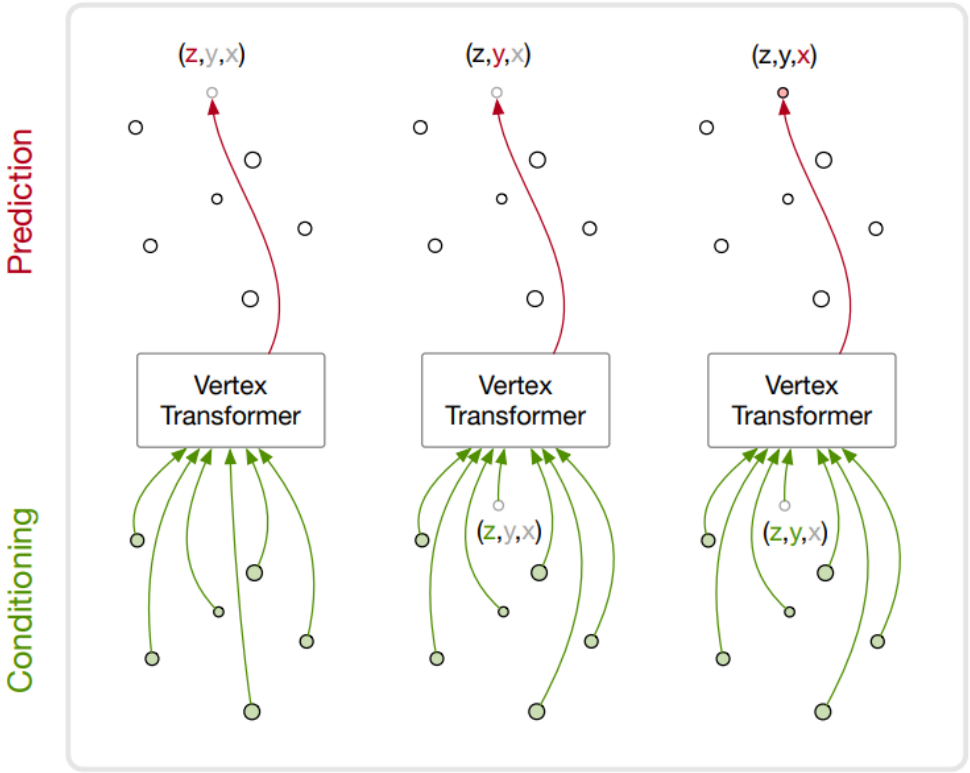
# Autoregressive Models

- Statistically, an autoregressive model characterizes random process
- Given previous values $x_1, x_2, \ldots, x_t$, output predictive probability distribution $P(x_{t+1}|x_1, x_2, \ldots, x_t)$ for $x_{t+1}$
  - If $x$ discrete, model probability distribution with softmax
  - If $x$ continuous, can discretize and follow the above

[Van den Oord et al. '16]

# Polygon Mesh Generation: PolyGen

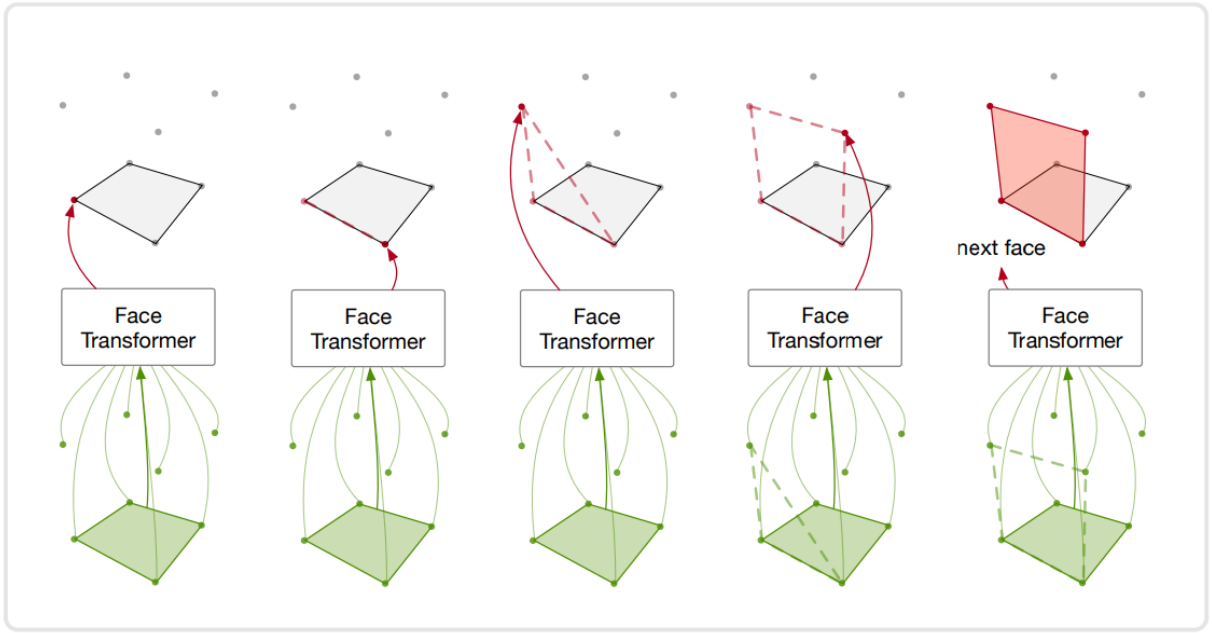
- $n$-gon mesh generation (per class category)
- First generate vertices autoregressively




[Nash et al. '20]

# Polygon Mesh Generation: PolyGen

- $n$-gon mesh generation (per class category)
- Then generate faces autoregressively conditioned on vertices
  - Face model takes vertices and current faces as input and outputs distribution over vertex indices



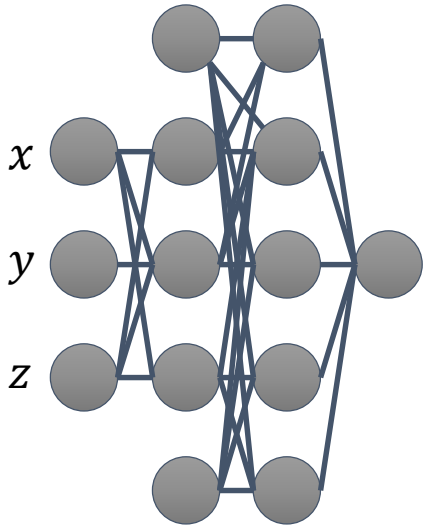

[Nash et al. '20]

# Coordinate-field Models for Shapes

- Recall: implicit surfaces

$$S = \{x \in \mathbb{R}^3 \mid f(x) = 0\}$$

- What if we model $f$ as a deep neural network?




$f$

Extract a mesh representation with Marching Cubes!

Must sample $f$ many times – MC grid resolution

# Coordinate-field Models for Shapes

- Recall: implicit surfaces

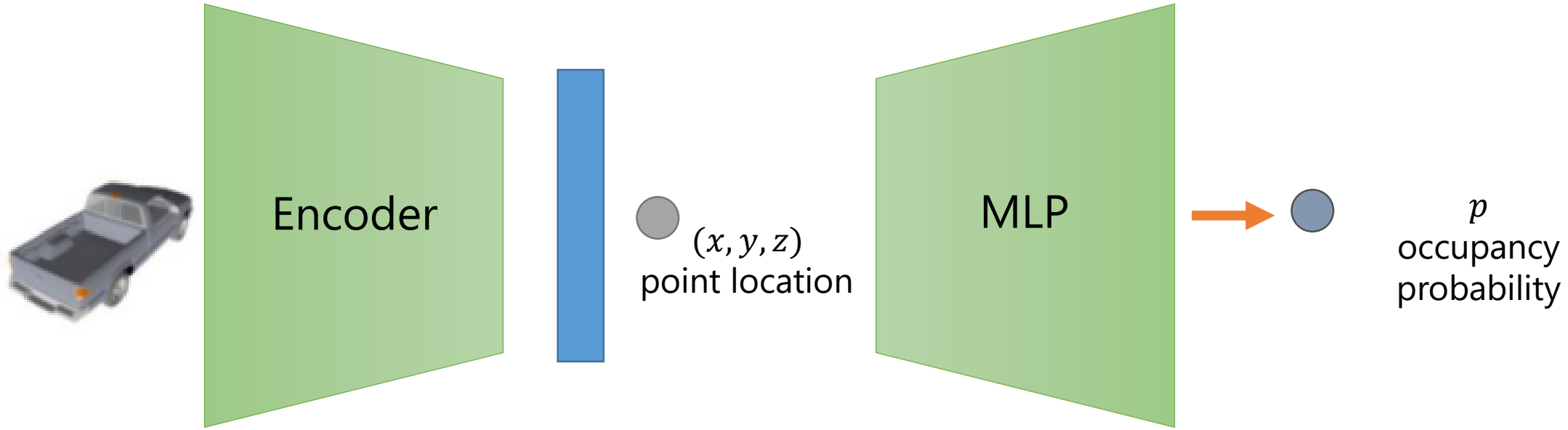
$$S = \{x \in \mathbb{R}^3 \mid f(x) = 0\}$$

- What if we model $f$ as a deep neural network?

- Note: can use deep network $f$ to model a non-implicit surface, e.g., surface occupancy

# Occupancy Networks

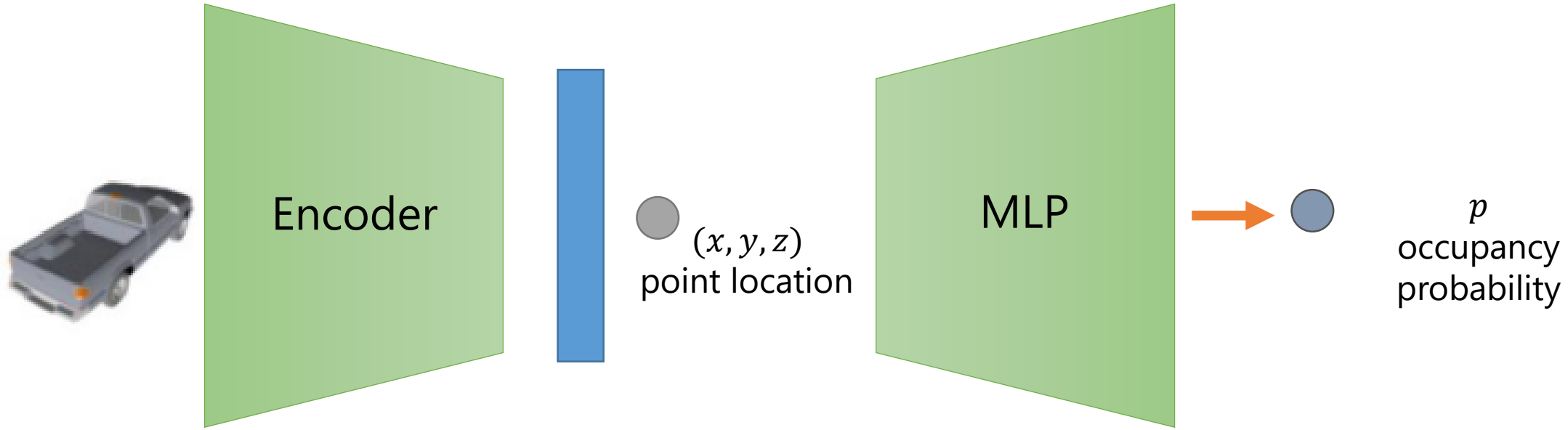
- 3D reconstruction as coordinate-field occupancy
- Encoder-decoder training




[Mescheder et al. '19]

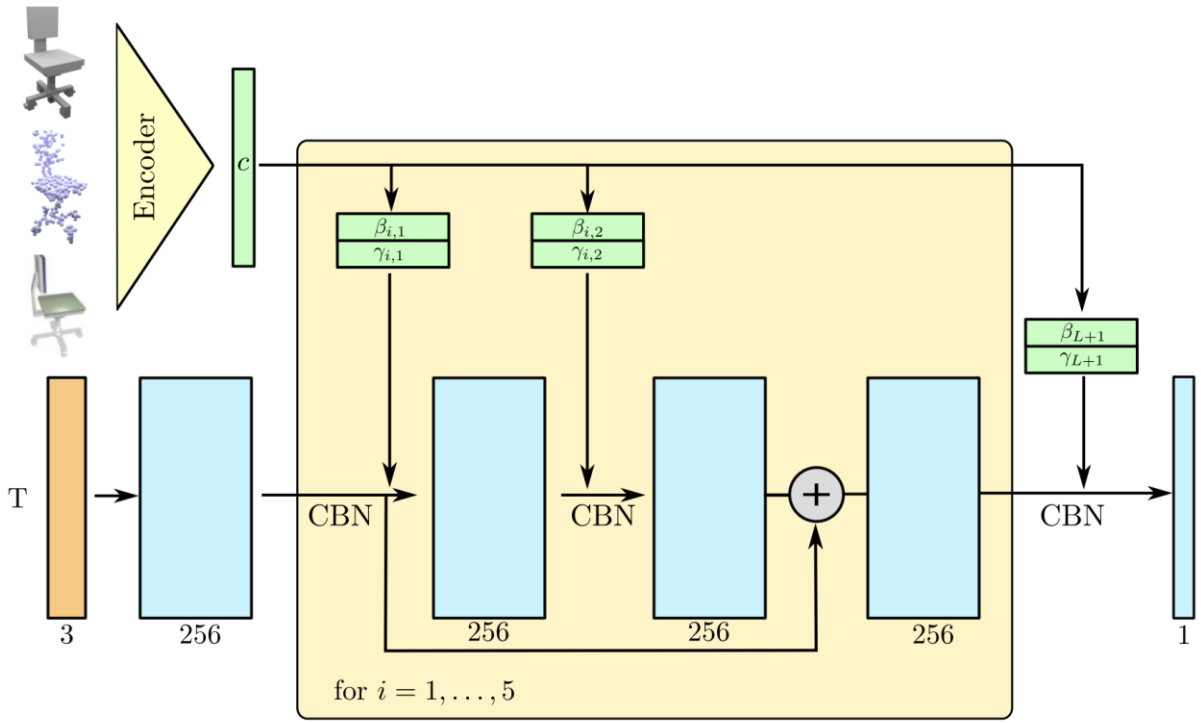
# Occupancy Networks

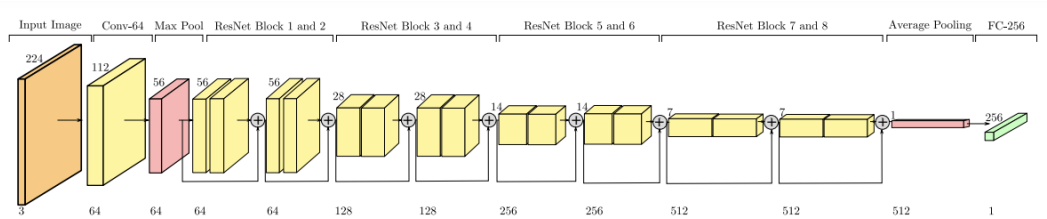
- 3D reconstruction as coordinate-field occupancy
- Encoder-decoder training



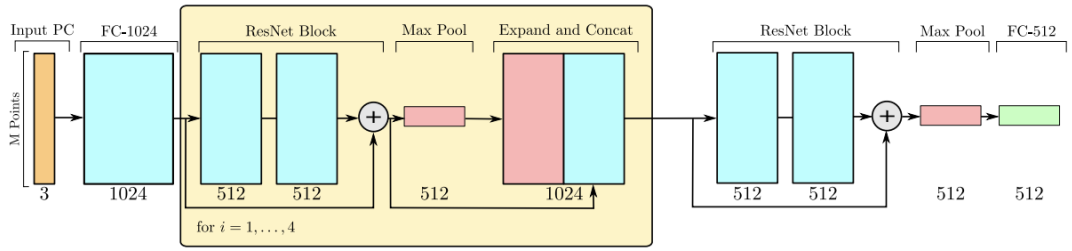

[Mescheder et al. '19]

# Occupancy Networks

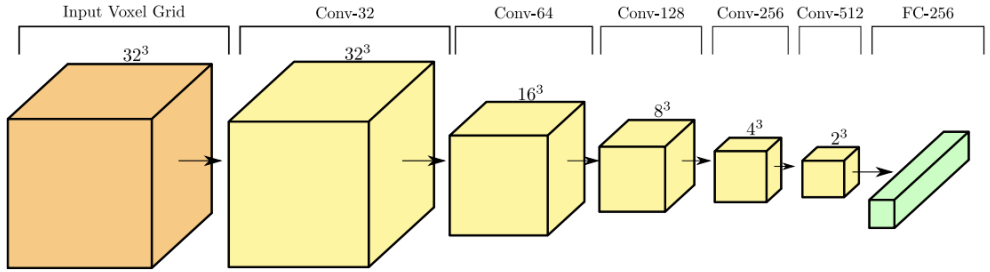



(a) Single Image 3D Reconstruction.




(b) Point Cloud Completion.

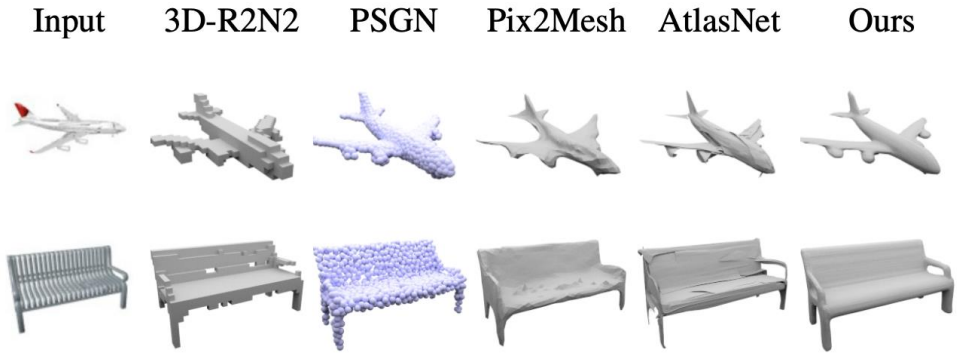



(c) Voxel Super-Resolution.

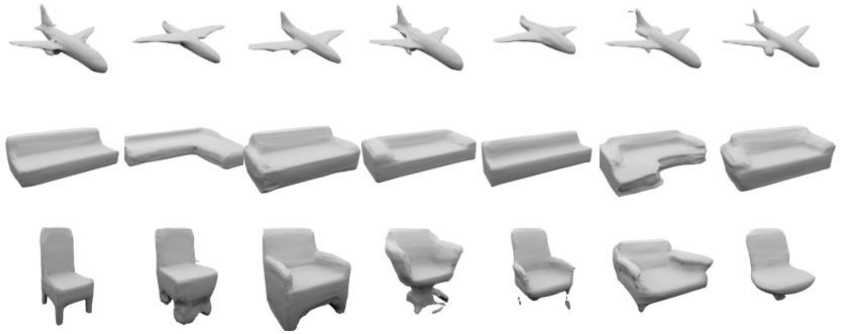
[Mescheder et al. '19]

# Occupancy Networks

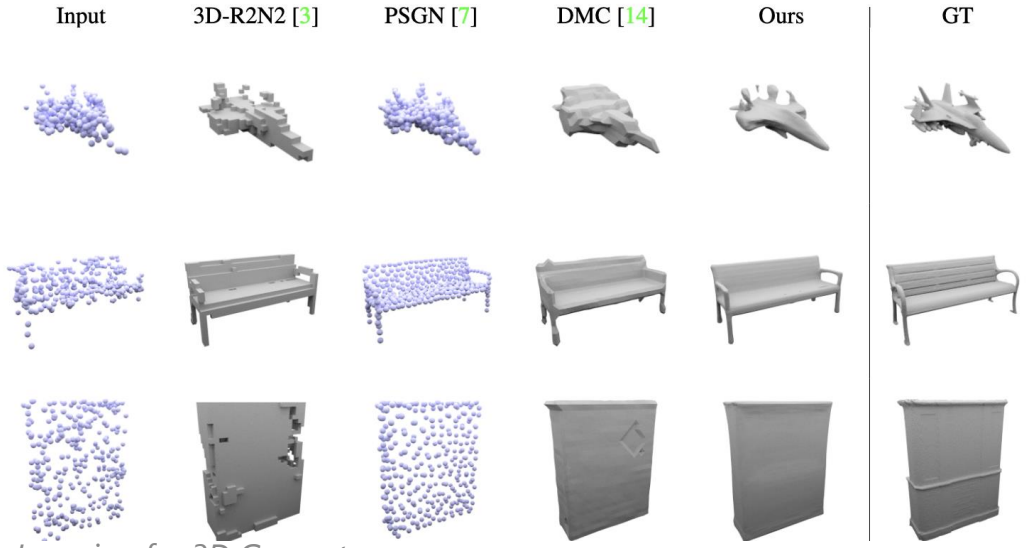
## Single-Image Reconstruction



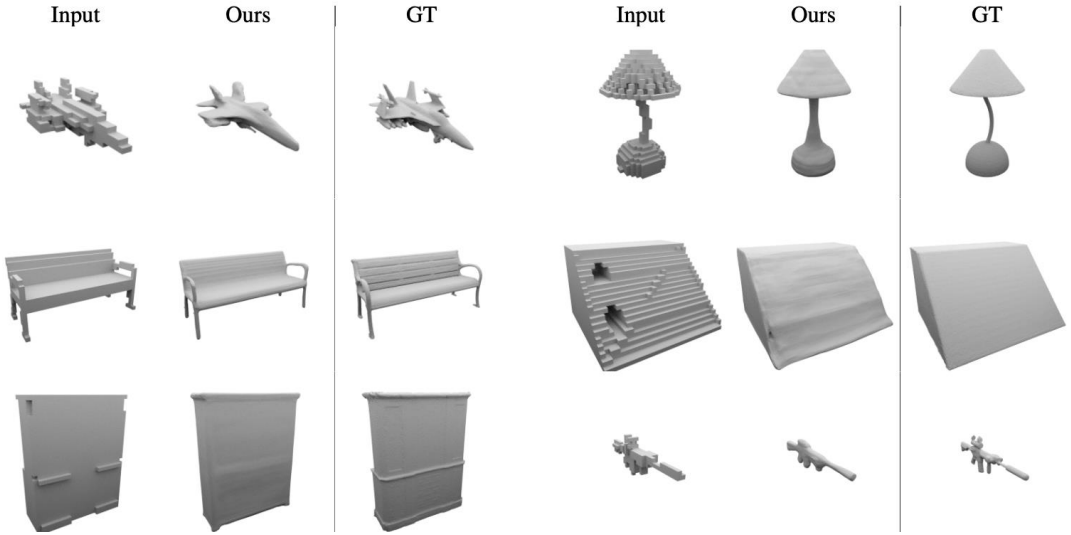

## Latent Space Interpolation



## Point Cloud Completion



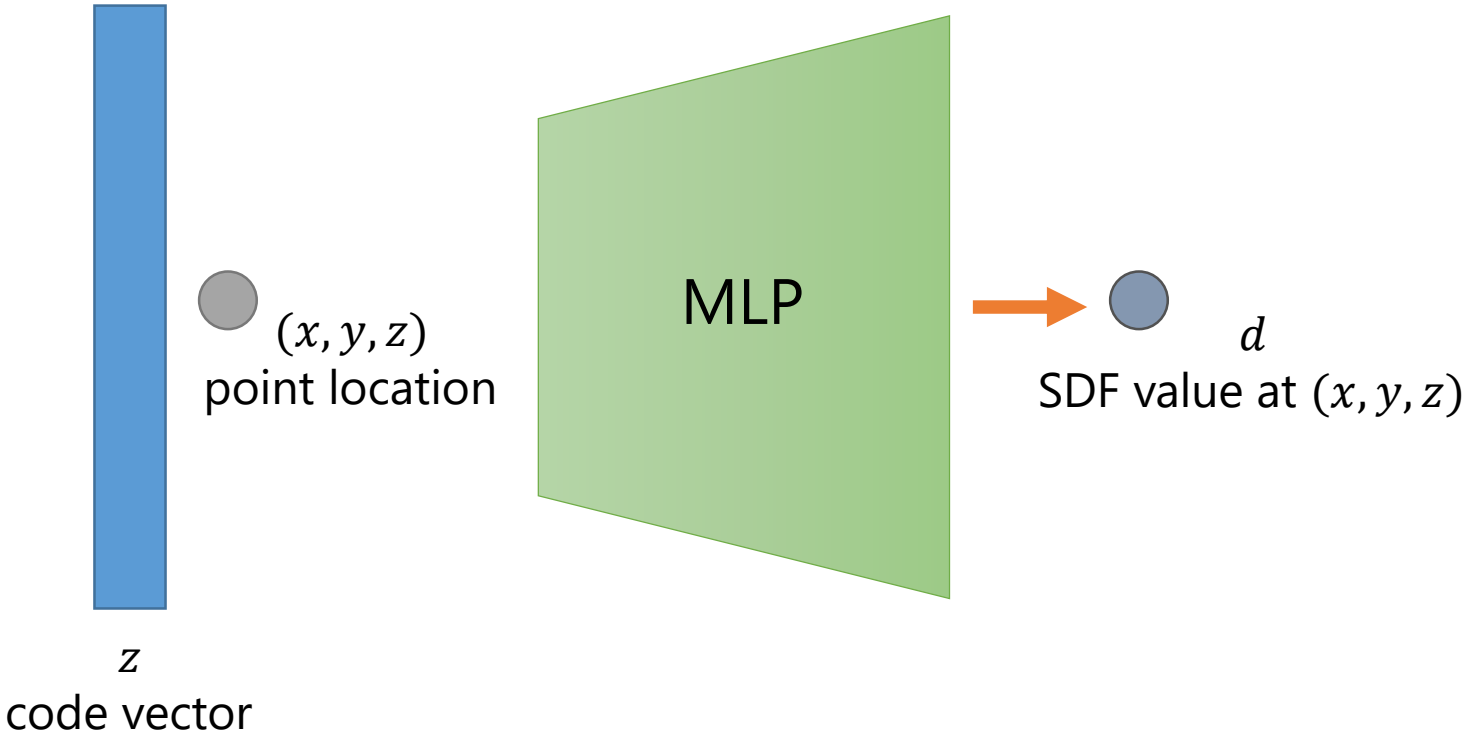

## Voxel Super Resolution




[Mescheder et al. ‘19]

# DeepSDF

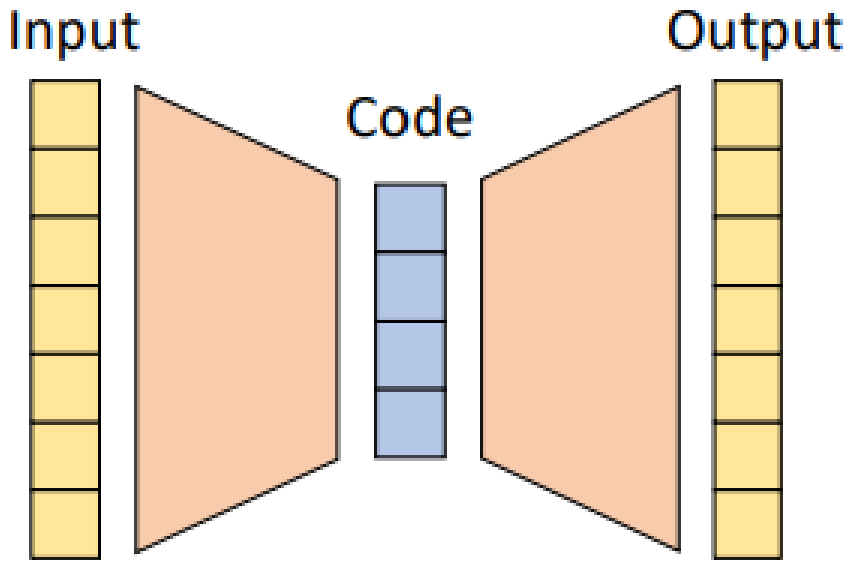
- Implicit 3D reconstruction



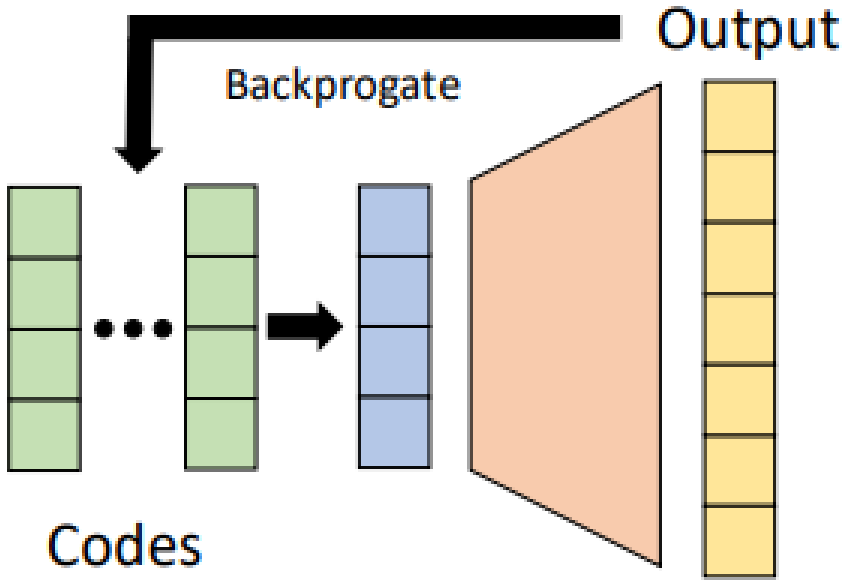

[Park et al. '19]

# DeepSDF

- Auto-decoder training




auto-encoder

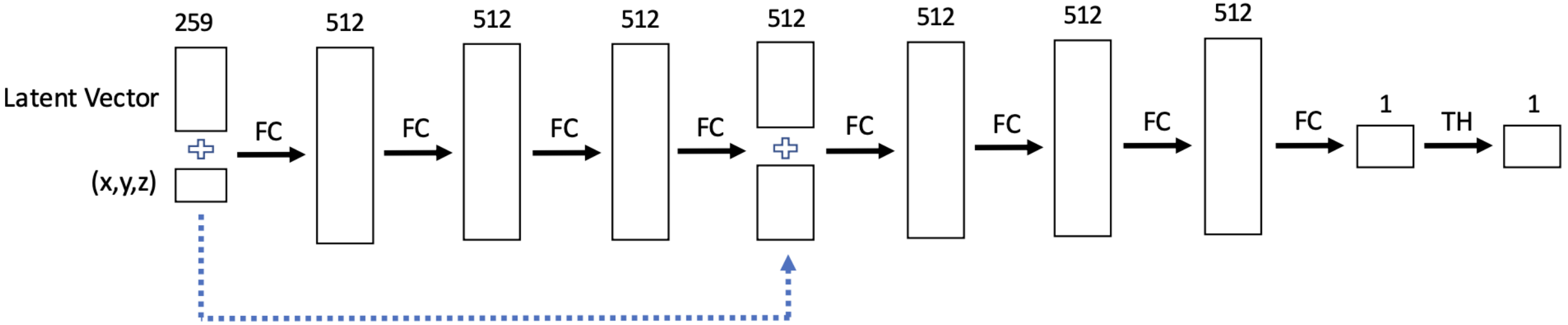



auto-decoder

[Park et al. '19]

# DeepSDF

- Auto-decoder training




$$\arg\min_{\theta,\{\boldsymbol{z}_i\}_{i=1}^{N}} \sum_{i=1}^{N} \left( \sum_{j=1}^{K} \mathcal{L}(f_\theta(\boldsymbol{z}_i, \boldsymbol{x}_j), s_j) + \frac{1}{\sigma^2} ||\boldsymbol{z}_i||_2^2 \right)$$

[Park et al. '19]

# DeepSDF

- Training: learn shape manifold, sampled code $z$ can be decoded to a shape
- Inference: optimize for $z$ that best fits input condition (e.g., partial point cloud)

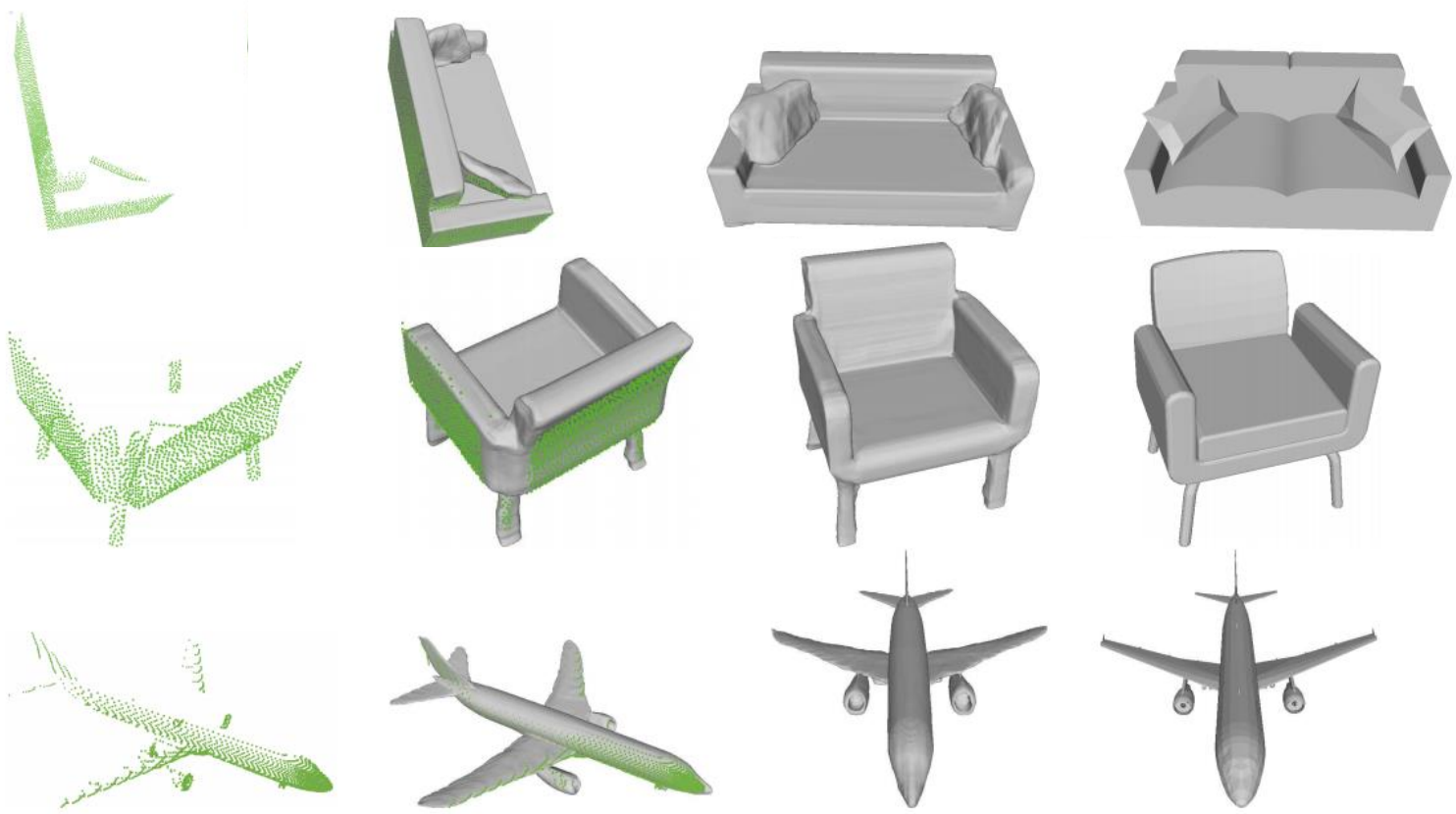
$$\hat{\boldsymbol{z}} = \arg\min_{\boldsymbol{z}} \sum_{(\boldsymbol{x}_j, \boldsymbol{s}_j) \in X} \mathcal{L}(f_\theta(\boldsymbol{z}, \boldsymbol{x}_j), s_j) + \frac{1}{\sigma^2} ||\boldsymbol{z}||_2^2$$

- Shape manifold must produce valid shapes for good reconstruction in unobserved areas
- Test-time optimization: many forward+backward passes but can fit input more closely

[Park et al. '19]

# DeepSDF

- Shape completion from single depth image

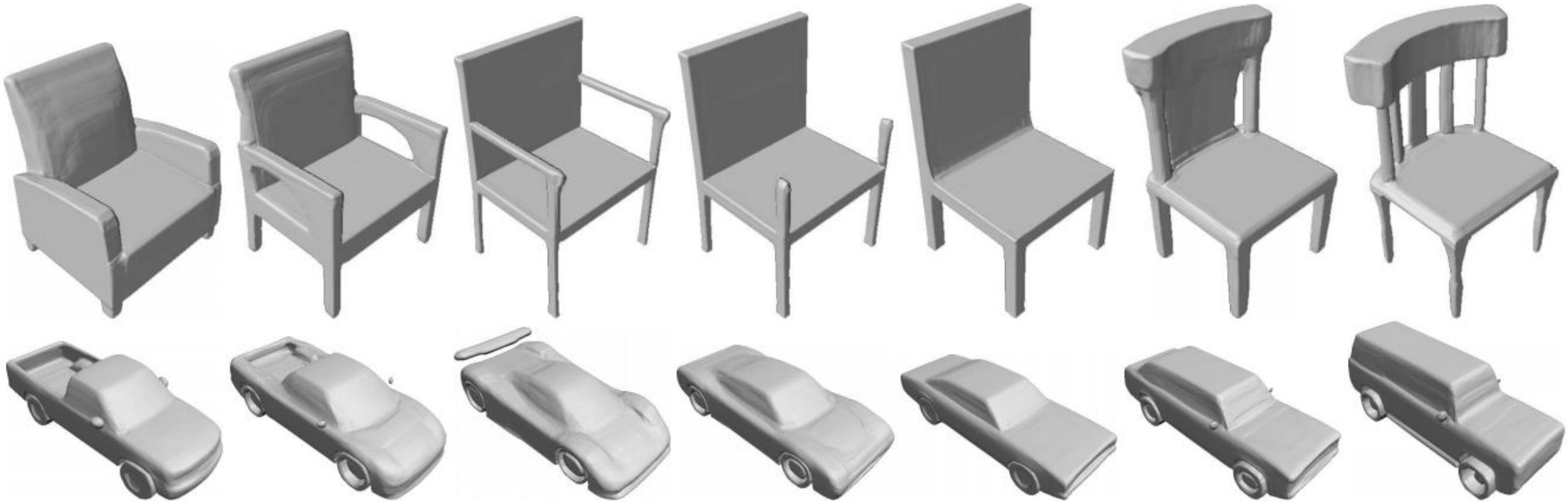


Input Depth | DeepSDF Completion | Ground Truth

[Park et al. ‘19]

# DeepSDF

- Shape interpolation (interpolating latent codes)

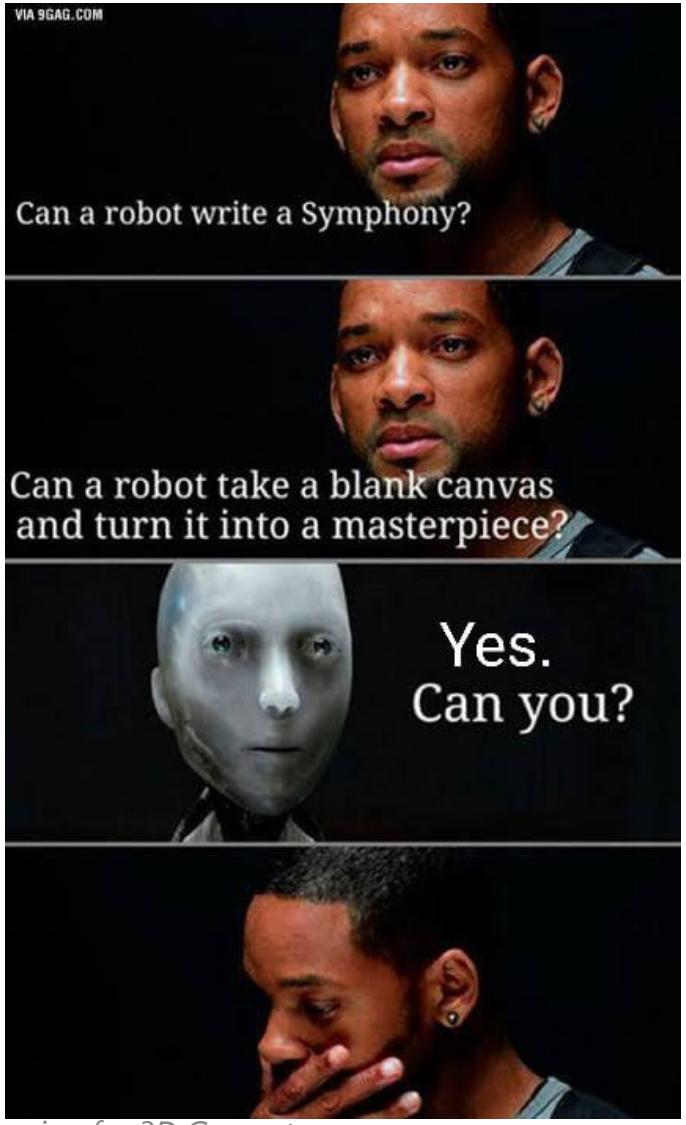


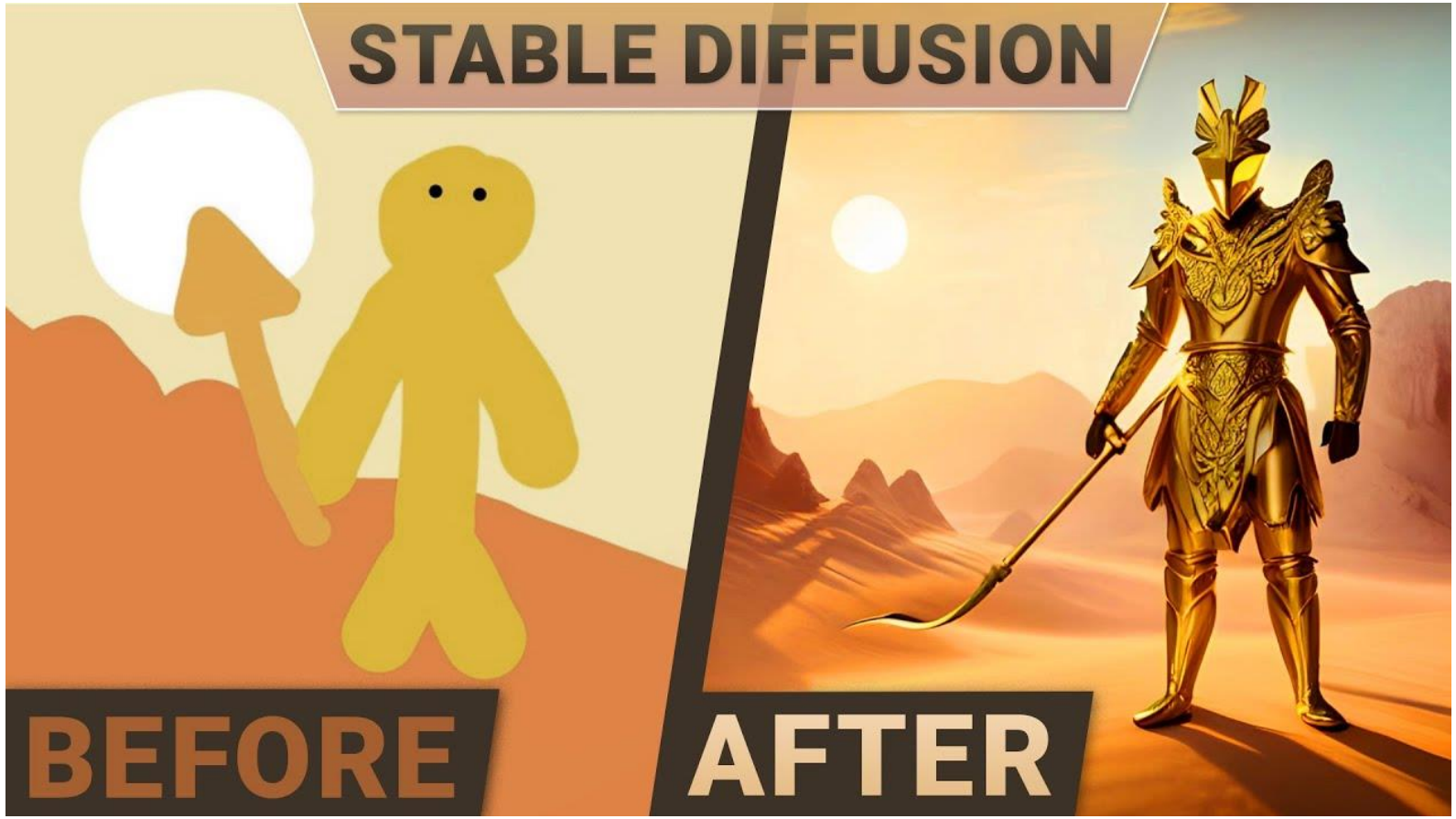
[Park et al. ‘19]

# Coordinate-field Models for Shapes

- Why coordinate-field neural network representation?
- No ties to any explicit resolution (e.g., voxel resolution, point density, etc.)
  - NOT infinite resolution – this depends on train sampling!
- Potentially compact representation
  - E.g., 5-layer MLP with hidden dimension 128: ~82k parameters
- But: how to learn general local structures?

# Diffusion Generative Models



# Diffusion Generative Models


# Diffusion Models

- Define Markov chain of diffusion steps
- Slowly add random noise to data
- Learn reverse diffusion process to construct data samples from noise

- Need: ground truth data samples
  - If conditional model, corresponding condition to gt

# Forward Diffusion

- Noising process
- $x_0 \rightarrow x_1 \rightarrow \cdots x_T$
- $x_0 \sim q(x)$ from data distribution we want to model
- $x_T \sim \mathcal{N}(0, \sigma^2 I)$

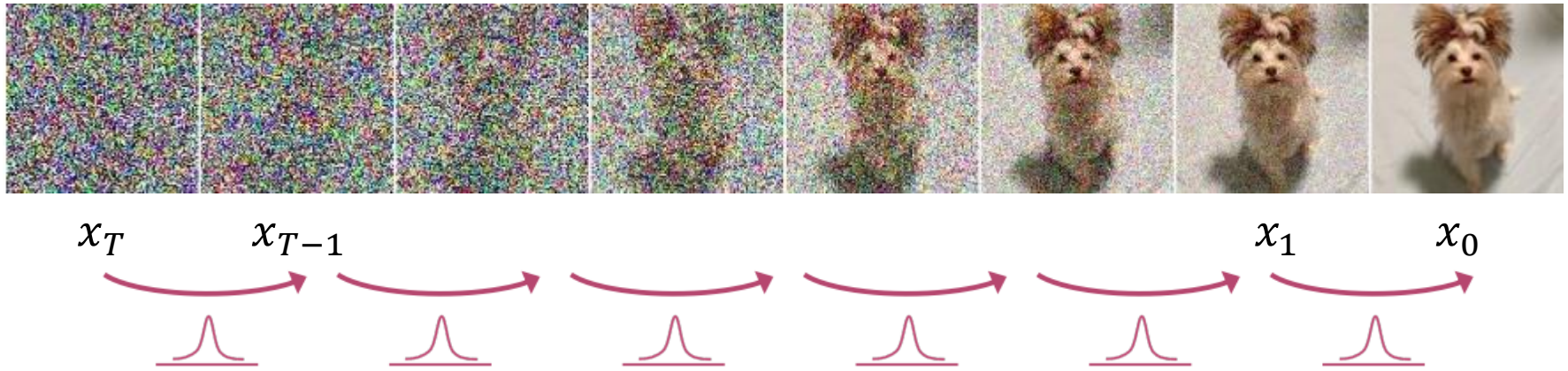
# Reverse Diffusion

- De-noising process
- $x_T \rightarrow x_{T-1} \rightarrow \cdots x_0$
- Sample from noise distribution: $x_T \sim \mathcal{N}(0, \sigma^2 I)$
- Reverse diffusion and sample $q(x_{t-1}|x_t)$ to re-create true sample

$x_T$ $x_{T-1}$ $x_1$ $x_0$

- Approximate by learning $p_\theta$

# Reverse Diffusion




$$p(x_T) = \mathcal{N}(x_T; 0, I)$$

$$p(x_{t-1}|x_t) = \mathcal{N}(x_{t-1}; \mu_\theta(x_t, t), \sigma_t^2 I)$$

trainable network
(e.g., U-Net)

$$p_\theta(x_{0:T}) = p(x_T) \prod_{t=1}^{T} p_\theta(x_{t-1}|x_t)$$

# Learning Denoising

- Form variational upper bound

$$\mathbb{E}_{q(x_0)}[-\log p_\theta(x_0)] \leq \mathbb{E}_{q(x_0)q(x_{1:T}|x_0)}\left[-\log \frac{p_\theta(x_{0:T})}{q(x_{1:T}|x_0)}\right] =: L$$

$$L = \mathbb{E}_q\Big[\underbrace{D_{KL}(q(x_T|x_0)||p(x_T))}_{L_T} + \sum \underbrace{D_{KL}(q(x_{t-1}|x_t,x_0)||p_\theta(x_{t-1}|x_t))}_{L_{t-1}} \underbrace{-\log p_\theta(x_0|x_1)}_{L_0}\Big]$$

# Diffusion for 3D?

- Dense voxel grid
  - Cubic growth in representation
  - Difficult to model high resolutions

- Points
  - Can apply noising/denoising directly to points

# Latent Diffusion

- Technique empowering StableDiffusion

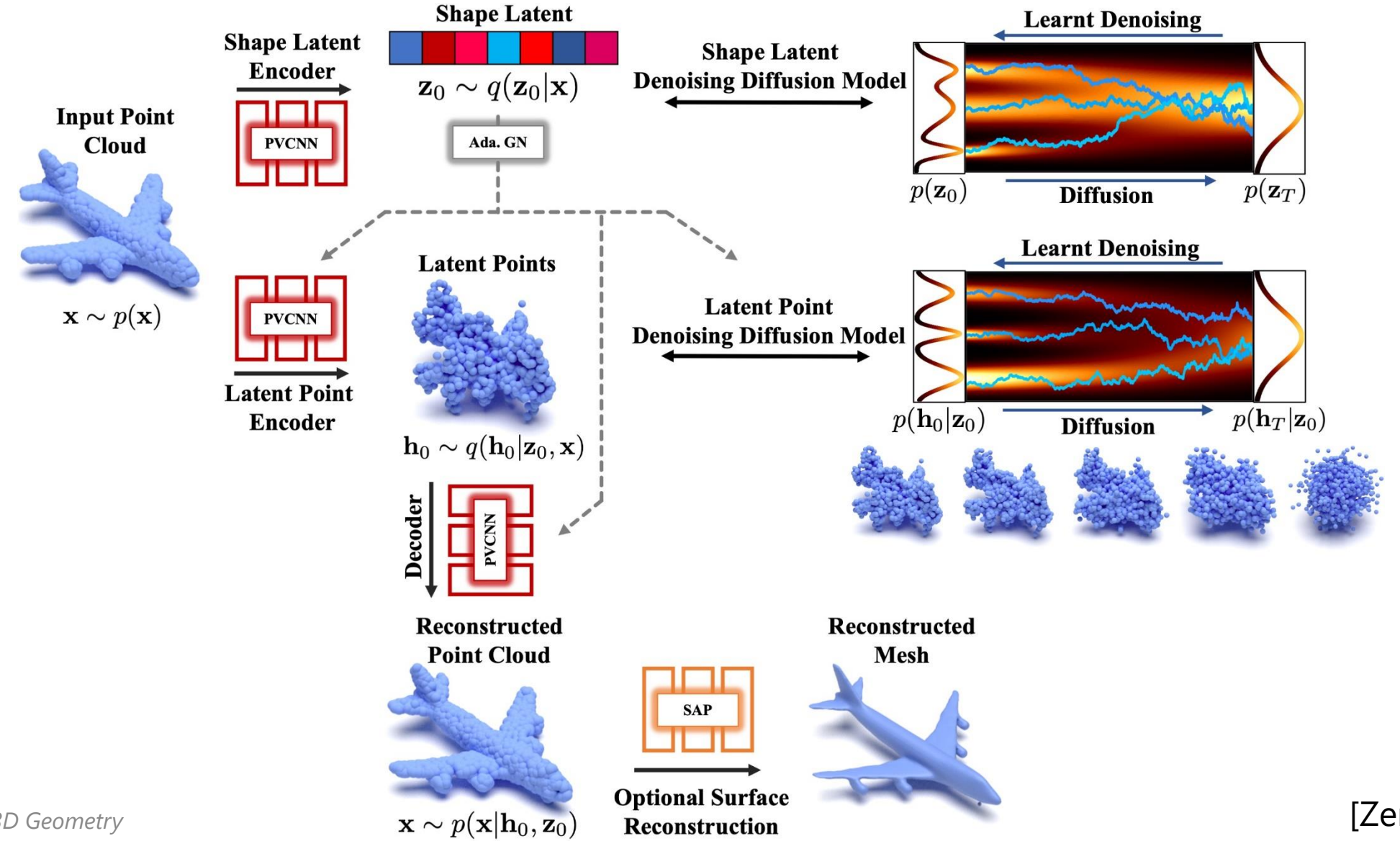
- Observation: images are high-dimensional but contain lower-dimensional semantic content, learned models spend significant effort learning imperceptible details

- First train VAE to compress images in a latent space
- Train diffusion in latent space



[Rombach et al. ‘22]

# Latent Point Diffusion: LION

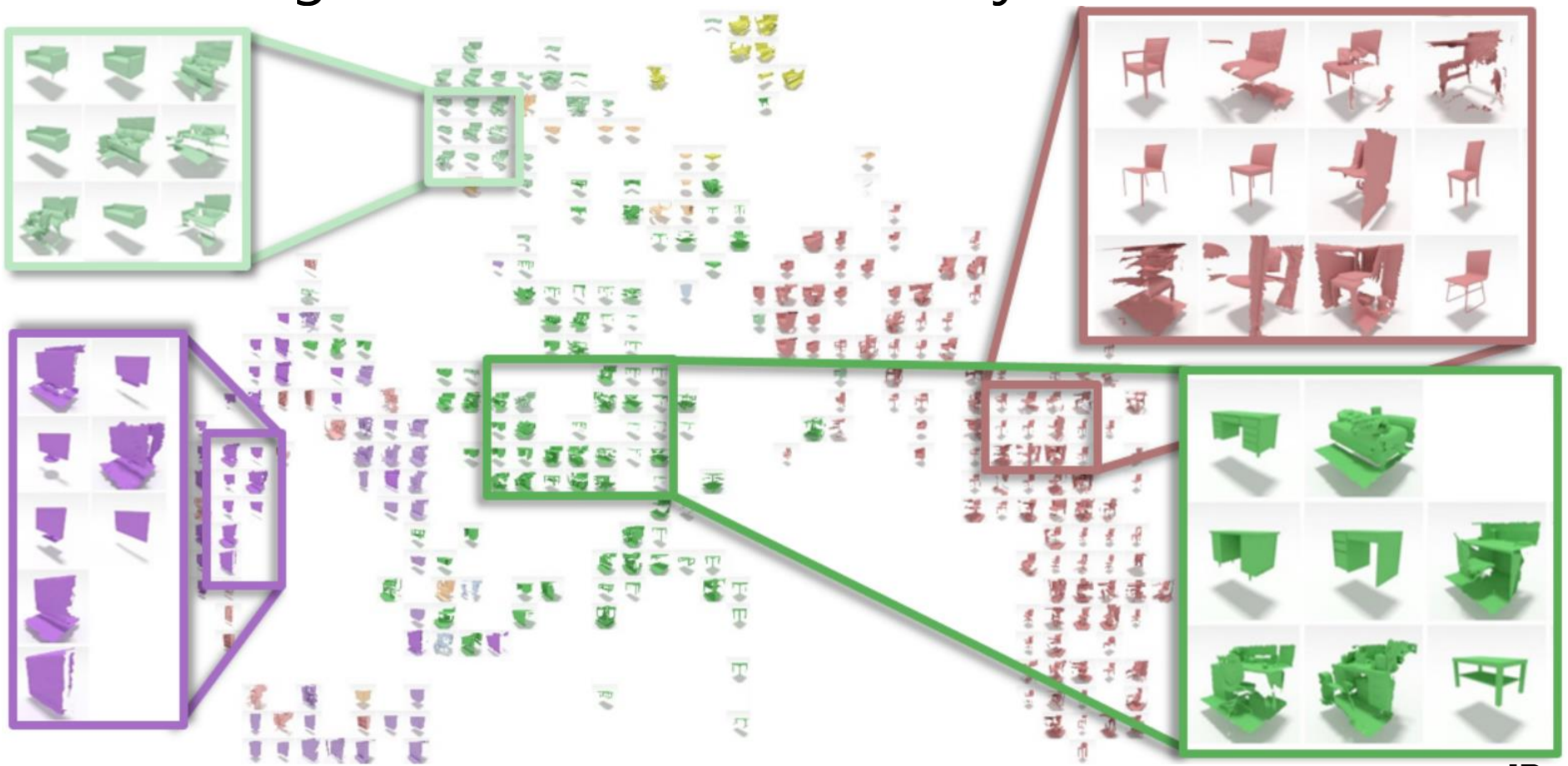



[Zeng et al. '22]

# Retrieval-based Object Representation

# Joint Embedding for Retrieval

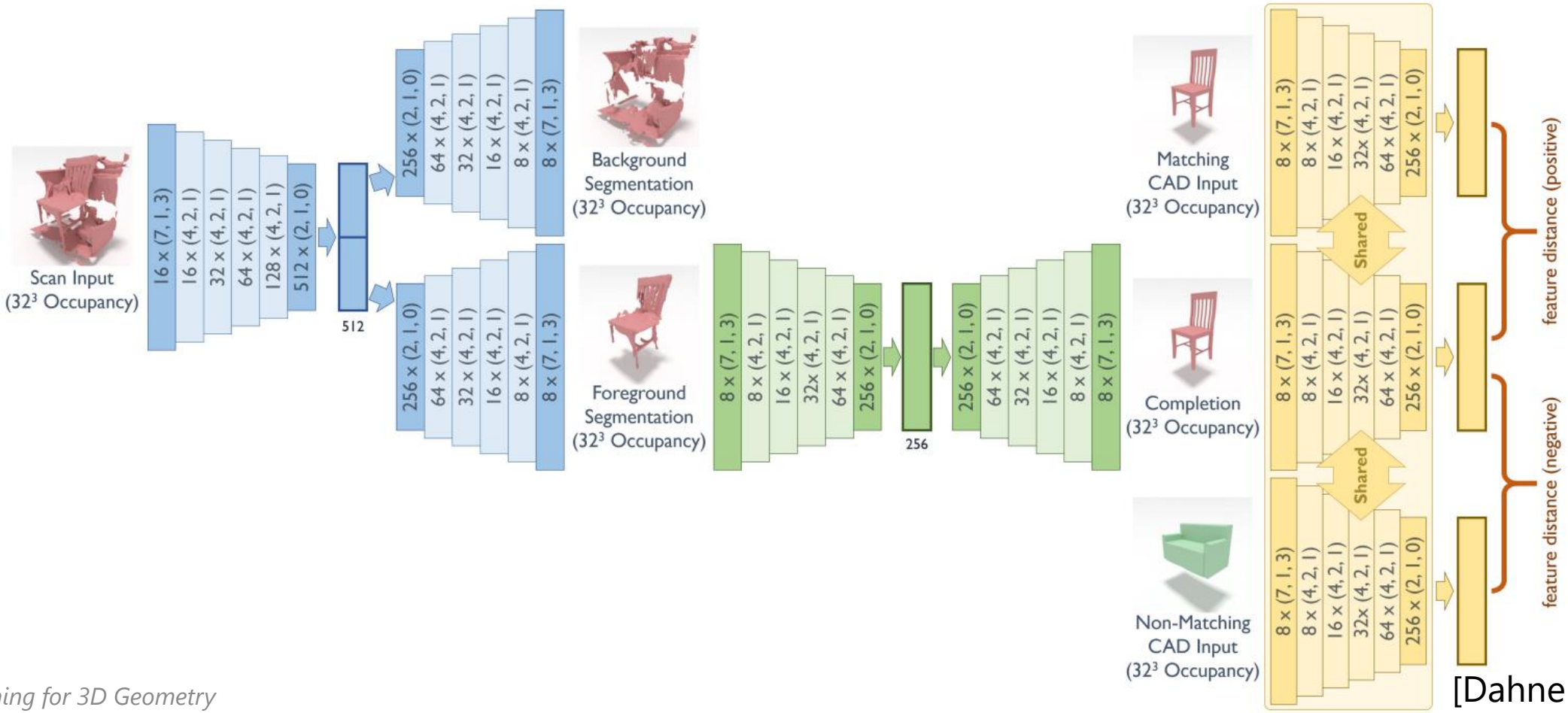
- Joint Embedding of 3D Scan and CAD Objects



[Dahnert et al. '19]

# Joint Embedding for Retrieval

- Joint Embedding of 3D Scan and CAD Objects




[Dahnert et al. '19]

# Joint Embedding for Retrieval

- Joint Embedding of 3D Scan and CAD Objects
  - End-to-end construction of embedding space
  - Triplet loss for metric learning to construct embedding space:
    - Compare an input with a known positive correspondence and known negative correspondence

$$L = \max(d\left(f(S), g\left(C_p\right)\right) - d\left(f(S), g(C_n)\right) + \text{margin}, 0)$$

    - $f(S)$: encoding of scan object
    - $g(C)$: encoding of CAD object
    - Minimize distance between input and positive small while maximizing distance between input and negative

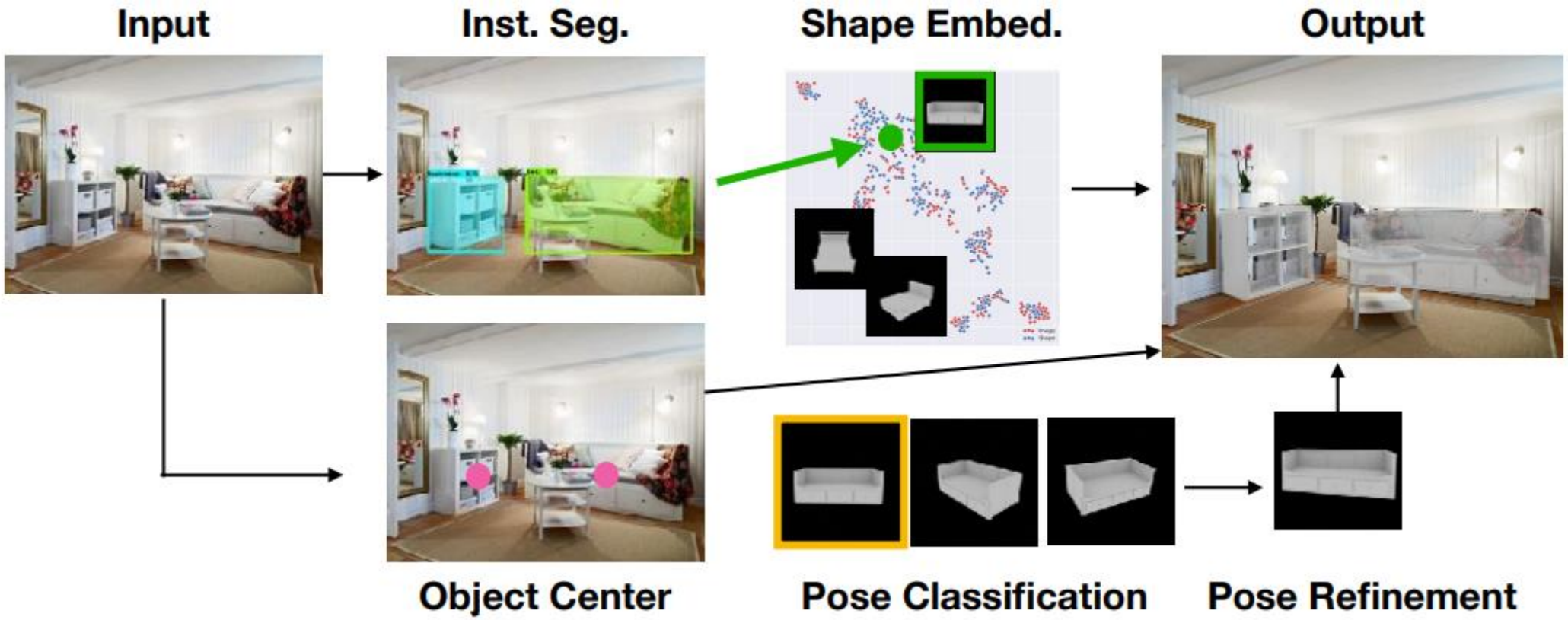
# Joint Embedding for Retrieval

- Joint Embedding of 3D Scan and CAD Objects


[Dahnert et al. '19]

# Retrieval-based Object Representation

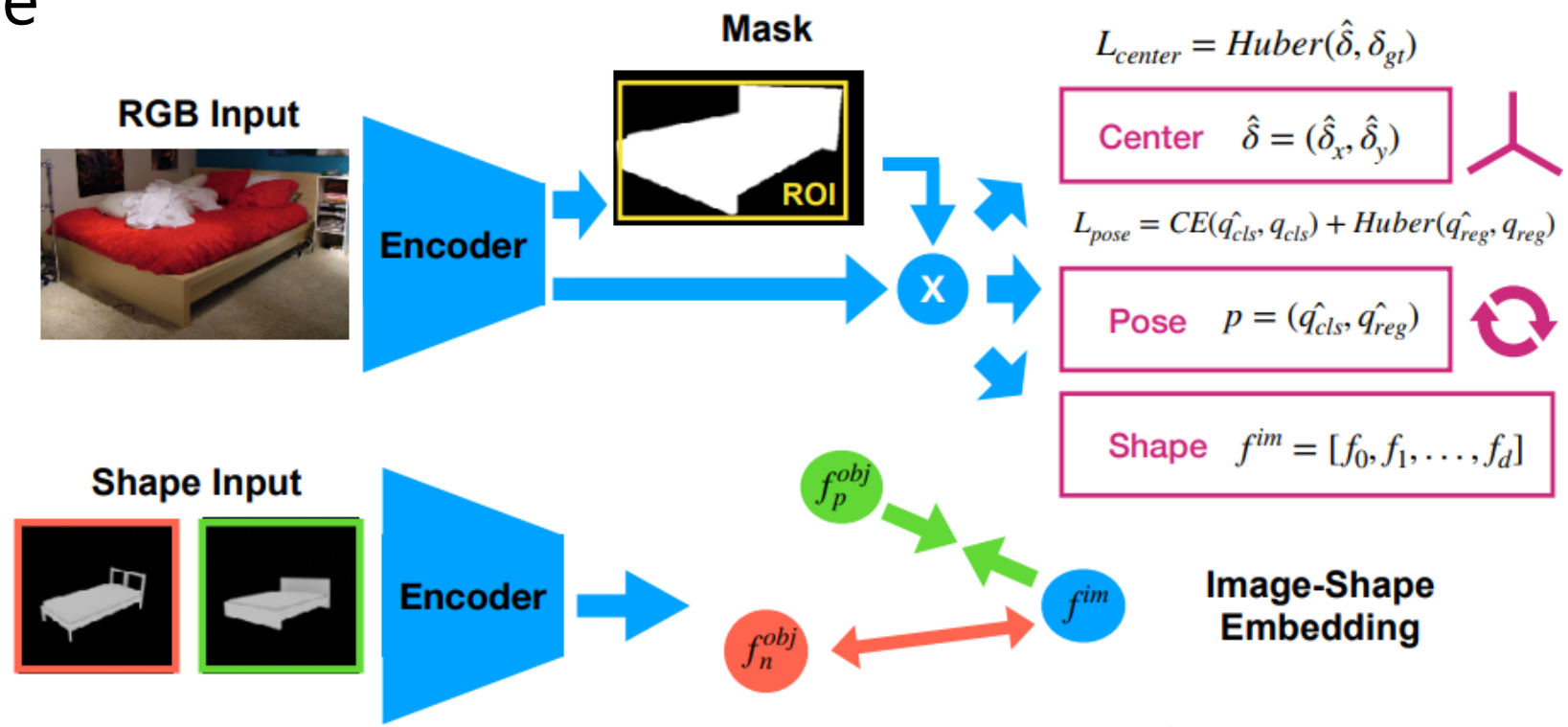
- Mask2CAD: 3D Shape Prediction by Learning to Segment and Retrieve




[Kuo et al. '20]

# Retrieval-based Object Representation

- Mask2CAD: 3D Shape Prediction by Learning to Segment and Retrieve




[Kuo et al. '20]

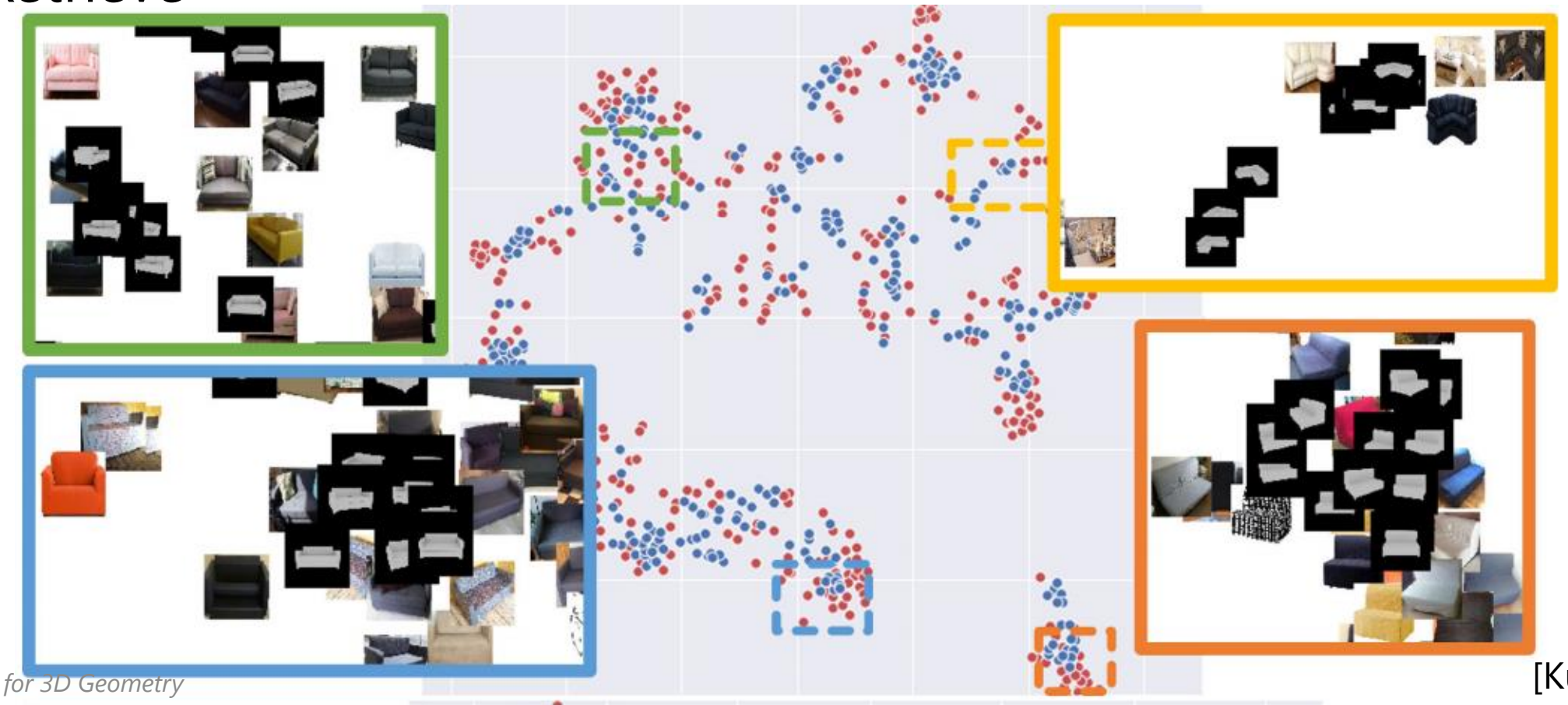
# Retrieval-based Object Representation

- Mask2CAD: 3D Shape Prediction by Learning to Segment and Retrieve


[Kuo et al. '20]

# Retrieval-based Object Representation

- Mask2CAD: 3D Shape Prediction by Learning to Segment and Retrieve



[Kuo et al. '20]

# Retrieval-based Object Representation

- Mask2CAD: 3D Shape Prediction by Learning to Segment and Retrieve

| | AP | AP50 | AP75 |
|---|---|---|---|
| original set | 6.5 | 17.3 | 3.8 |
| + more CADs | **8.2** | **20.7** | **4.8** |

Generalize to additional CAD models at inference time

[Kuo et al. '20]

# Useful References

- Real Image + 3D Shape Datasets
  - Pix3D: http://pix3d.csail.mit.edu
  - ScanNet+Scan2CAD: https://github.com/skanti/Scan2CAD

- Deeper dive into mathematical background of diffusion models
  - https://lilianweng.github.io/posts/2021-07-11-diffusion-models/
  - https://yang-song.net/blog/2021/score/