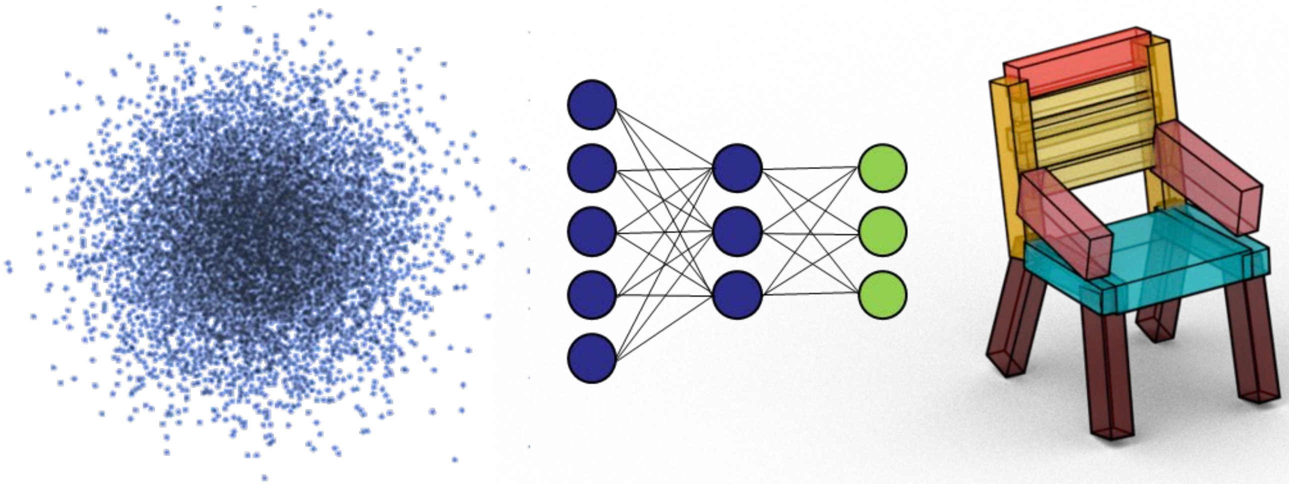
# CS348n: Neural Representations and Generative Models for 3D Geometry



Leonidas Guibas
Computer Science Department
Stanford University

# Tidbits

- The class will continue in Zoom format next week.
- Extended office hours this Friday (Jan 21): 1:30-3:00 pm. Can be in person. Please send e-mail to request a time slot.
- Please ask Kaichun for Google Cloud for Education coupons.

# Homework 1

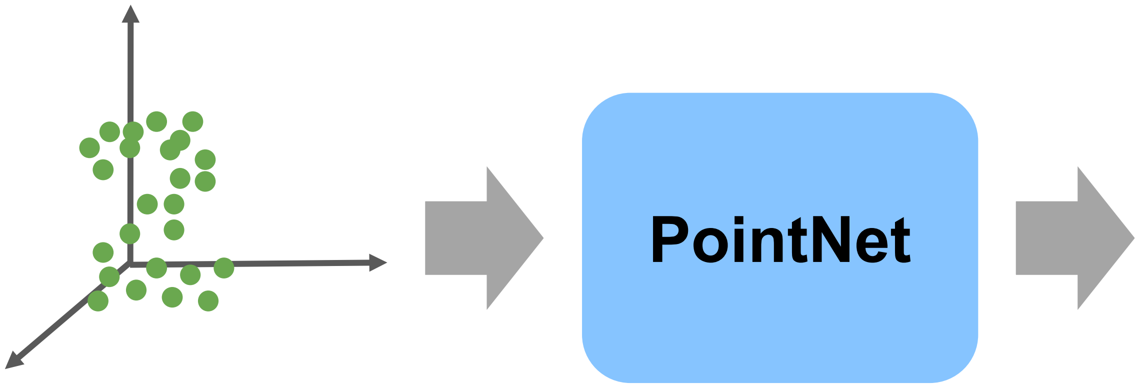
due, Wed, Jan 26, 2022

# Homework Policies

- Can work in groups of up to 3 students – single shared writeup and code submission OK.
- Writeup must be in digital form, typeset (LaTeX or Word), and submitted through Gradescope.
- Two “grace class periods” for late homeworks – after that, 20% penalty per period.
- Respect the honor code: all submitted work must be your own and properly reference materials used.

# Last Time: Deep Learning on Point Clouds (PCs)

# Deep Nets for PCs: PointNet and PointNet++




*Object Classification*

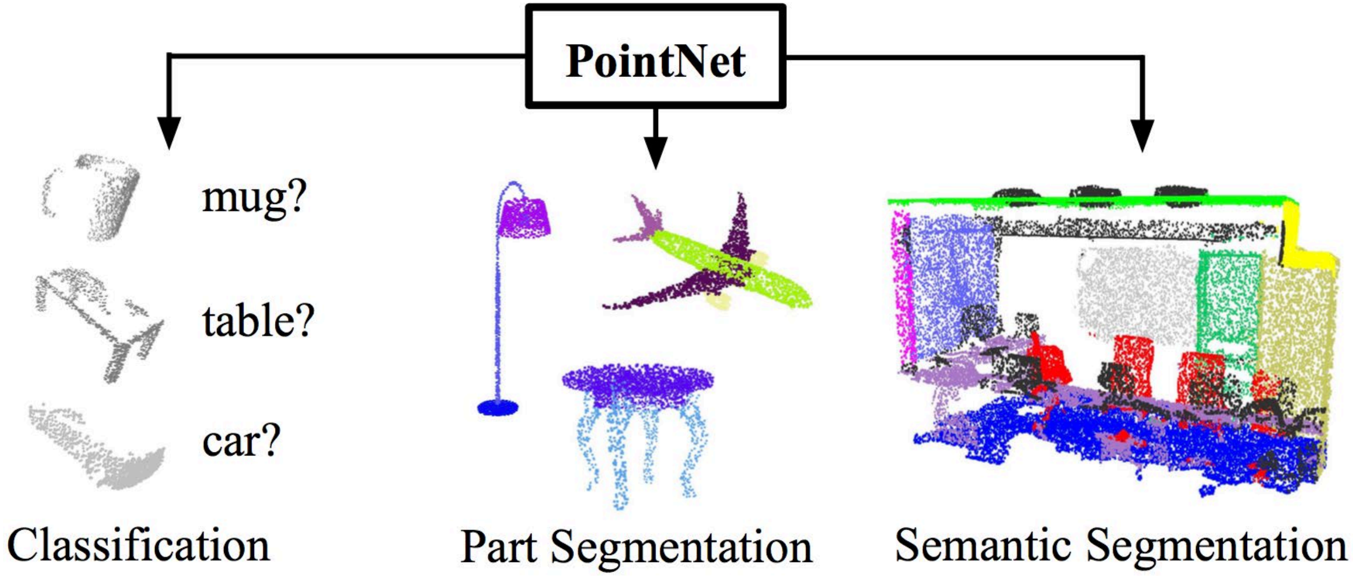
*Object Part Segmentation*

*Semantic Scene Parsing*

...

**End-to-end learning** for irregular point data

**Unified** framework for various tasks




Charles R. Qi, Hao Su, Kaichun Mo, Leonidas J. Guibas. PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation. (CVPR'17)

# Invariances

*The model has to respect key desiderata for point clouds:*

## Point Permutation Invariance

Point cloud is a set of unordered points

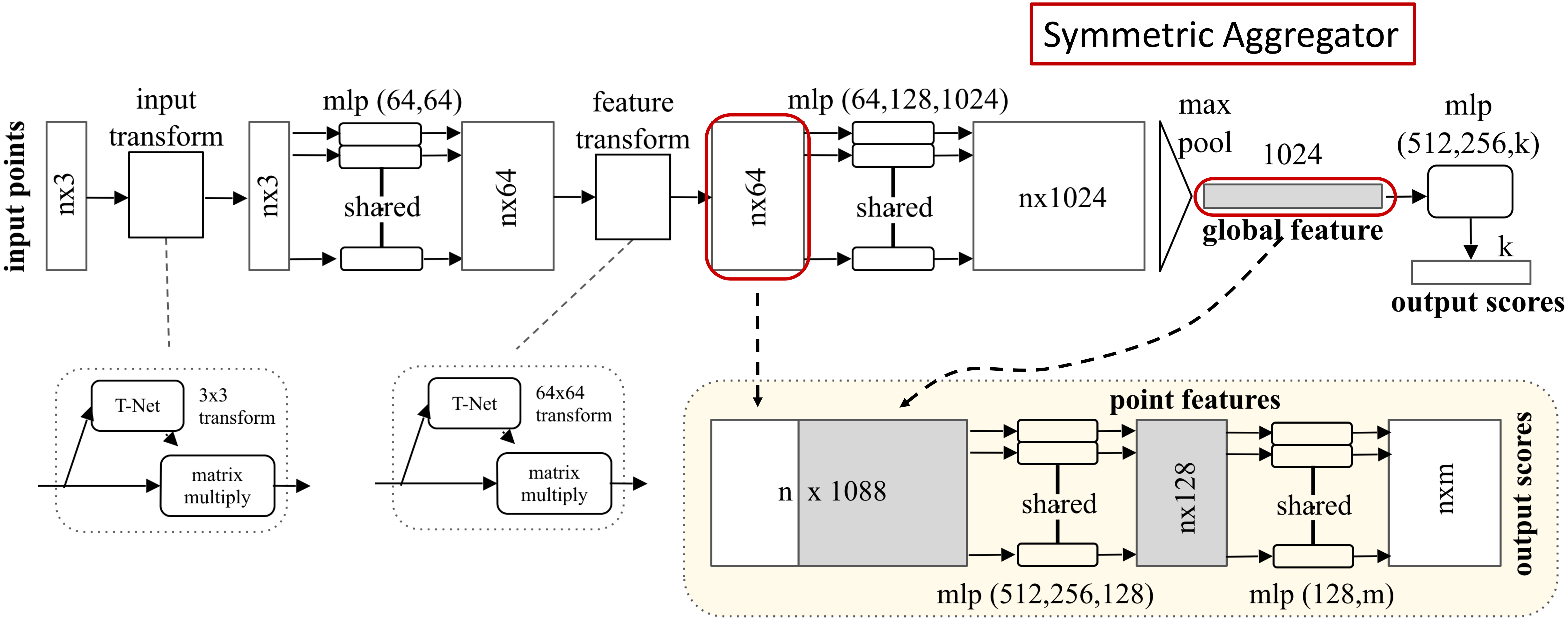
## Spatial Transformation Invariance

Point cloud rigid motions should not alter classification results

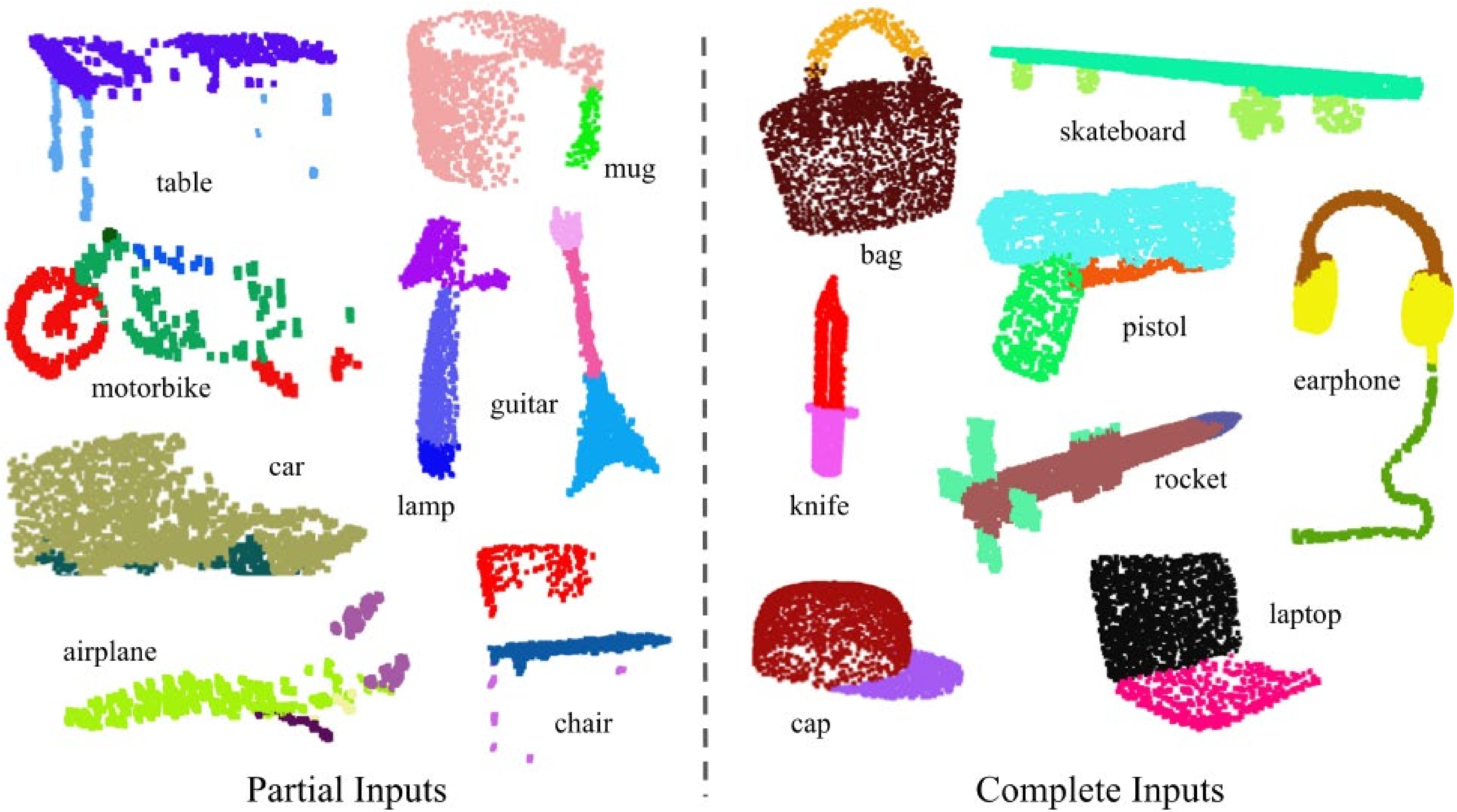
## Sampling Invariance

Output a function of the underlying geometry and not the sampling

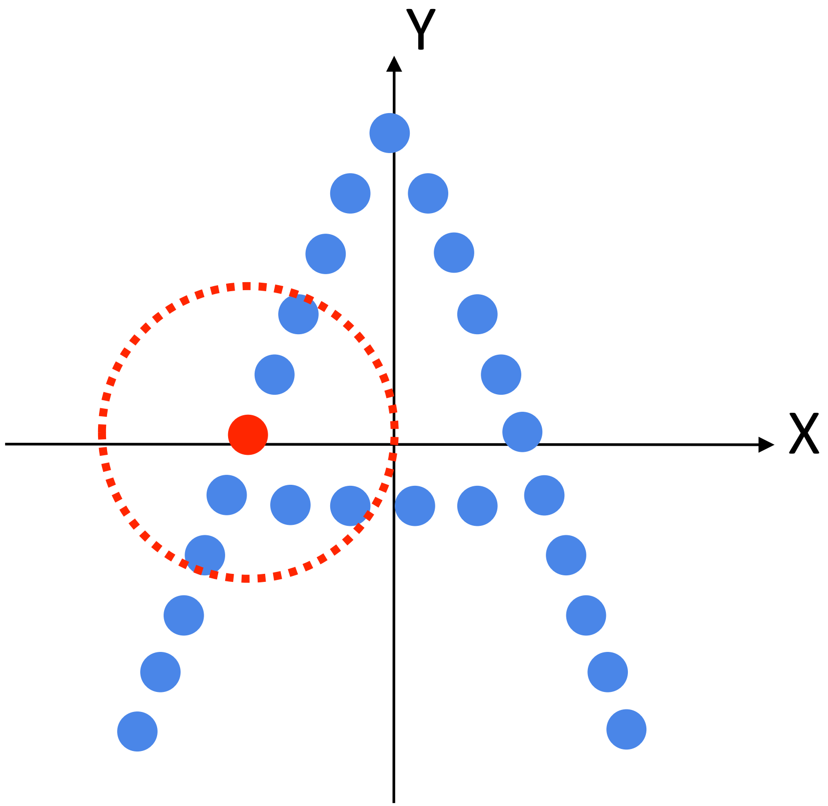
# PointNet for Classification and Segmentation



# Results on Object Part Segmentation

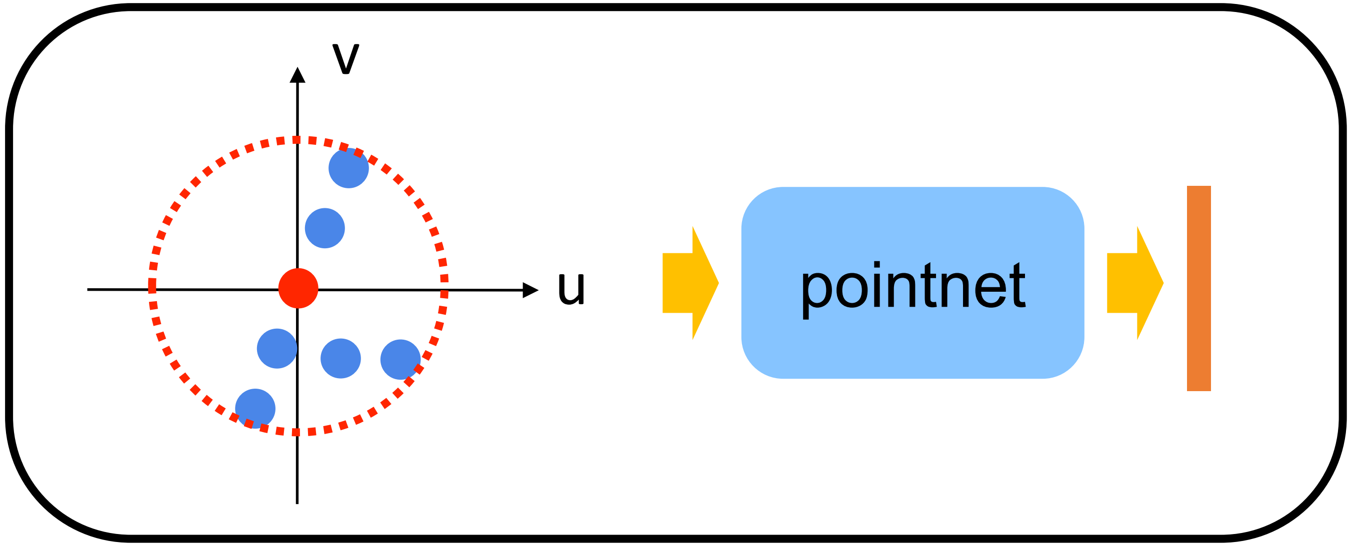


# PointNet++: Hierarchical Point Feature Learning



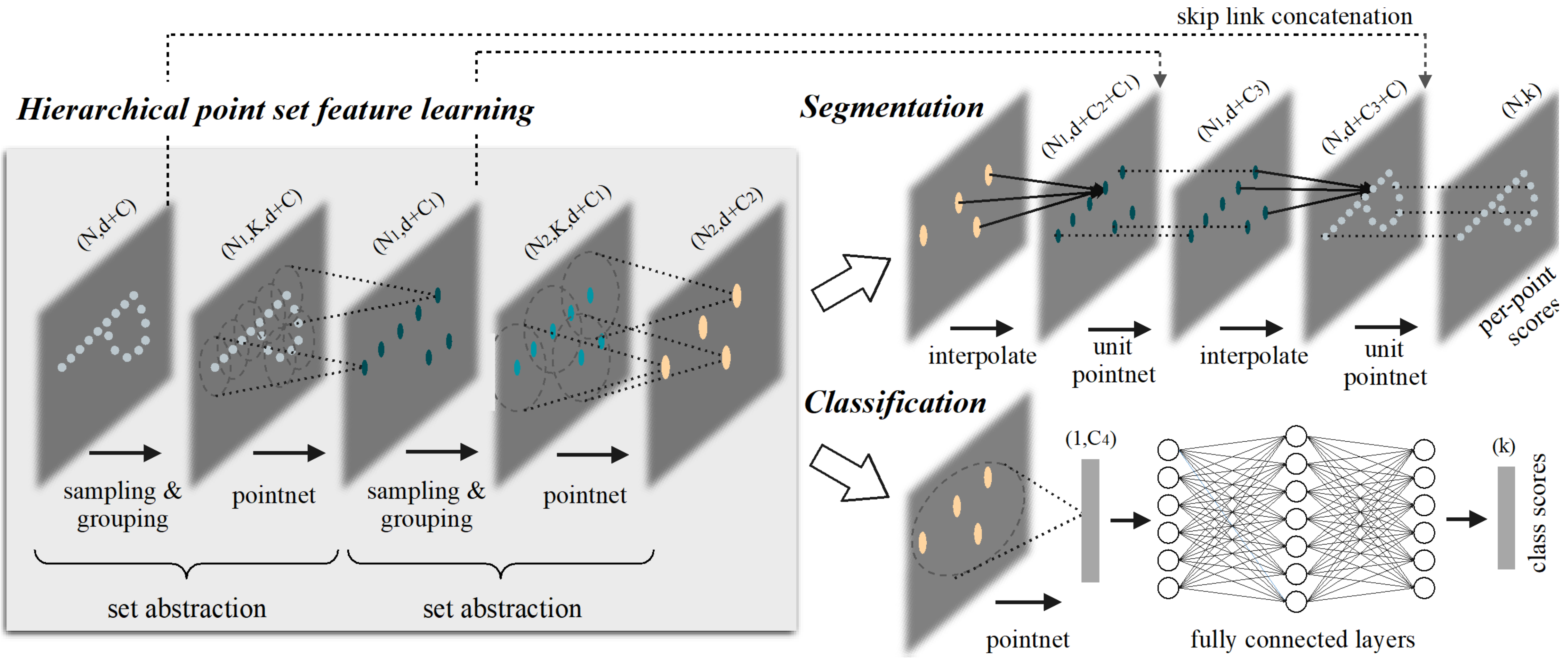

N points in (X,Y)

**Apply pointnet at a local region**




k points in local coordinates (u,v)

# PointNet++ for Classification and Segmentation

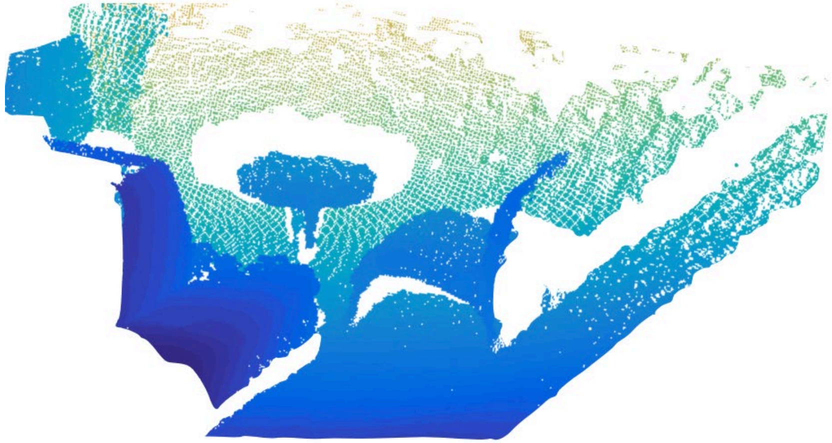


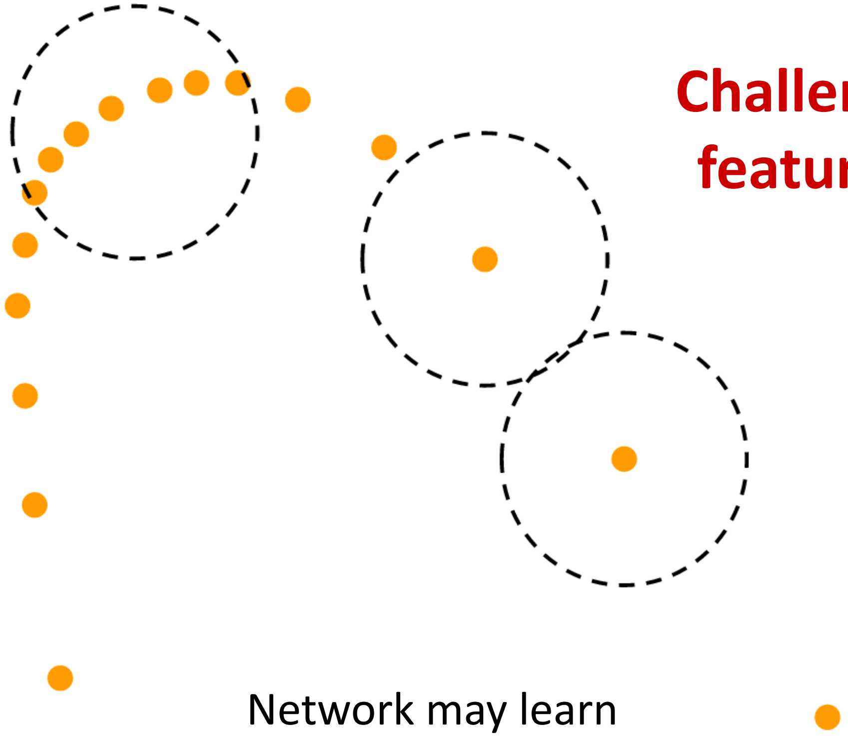

Aggregation pattern is only a function of the spatial locations of the points

# Non-uniform Sampling Density in Point Clouds

Density variation is a common issue in 3D point cloud processing

- perspective effect, radial density variation, motion etc.





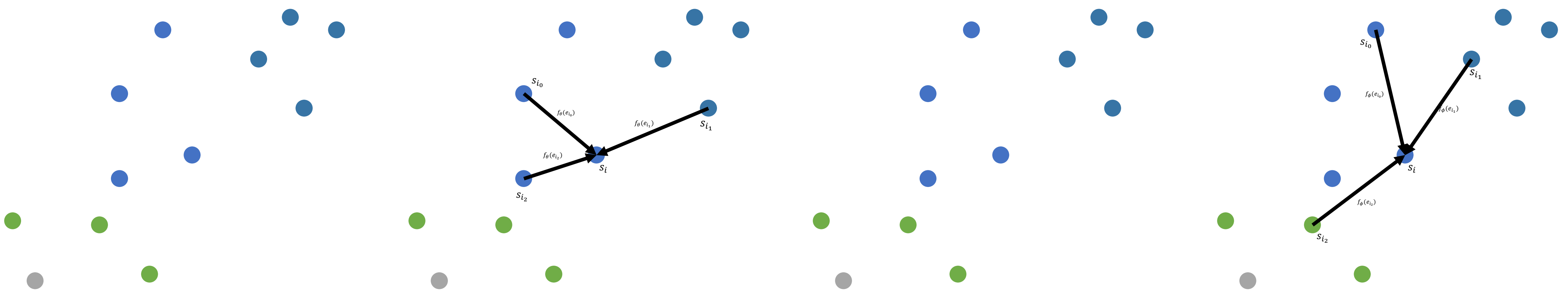
**Challenge for local feature learning!**

Network may learn sampling pattern – not underlying geometry

# Dynamic Graph CNN: EdgeConv

Graph convolutions: $\mathbf{x}'_i = \square_{j:(i,j)\in\mathcal{E}} h_{\Theta}(\mathbf{x}_i, \mathbf{x}_j)$

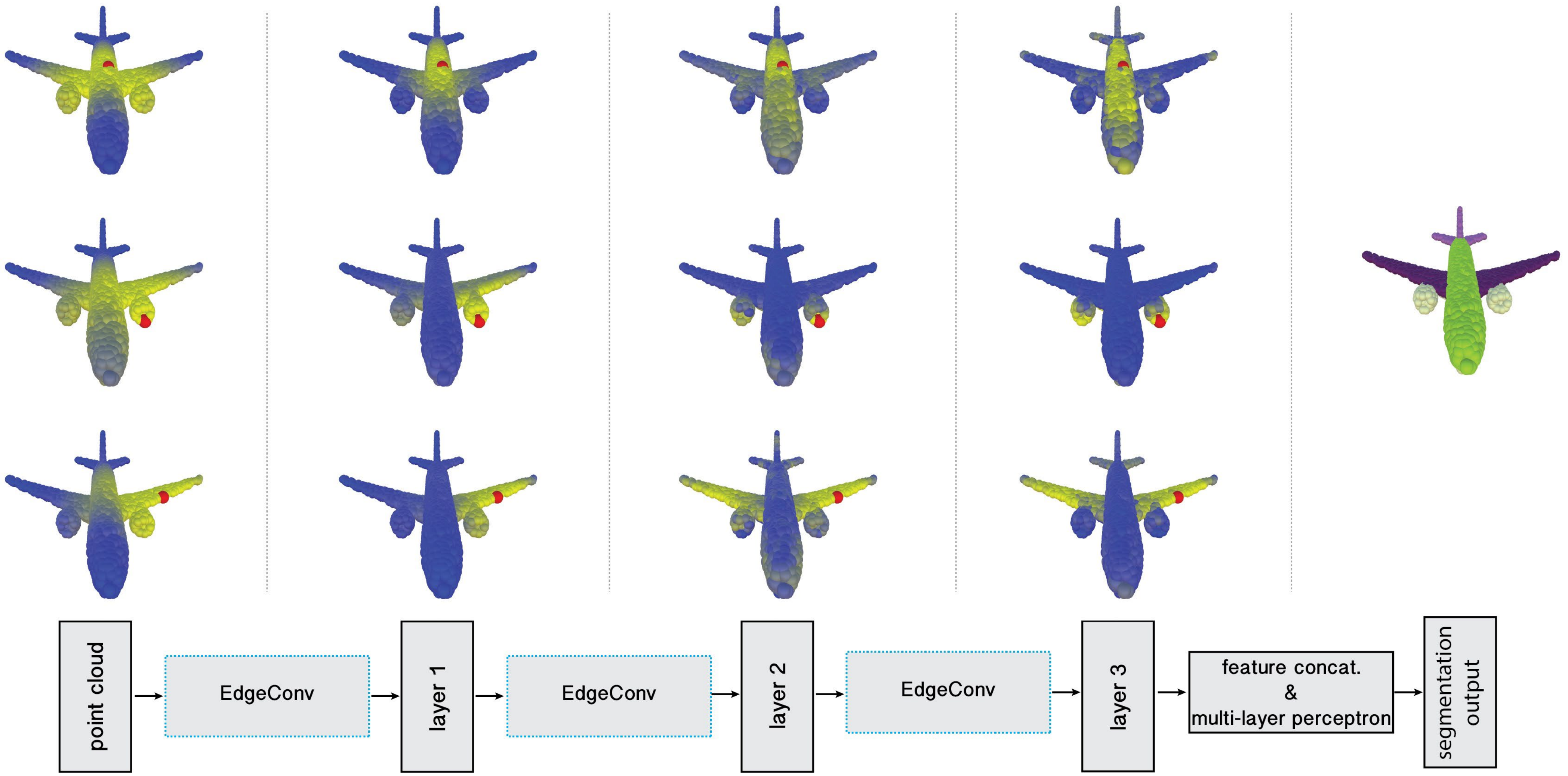
$\mathbf{x}'_i = \square_{j:(i,j)\in\mathcal{E}} h_{\Theta}(\mathbf{x}_i, \mathbf{x}_j)$



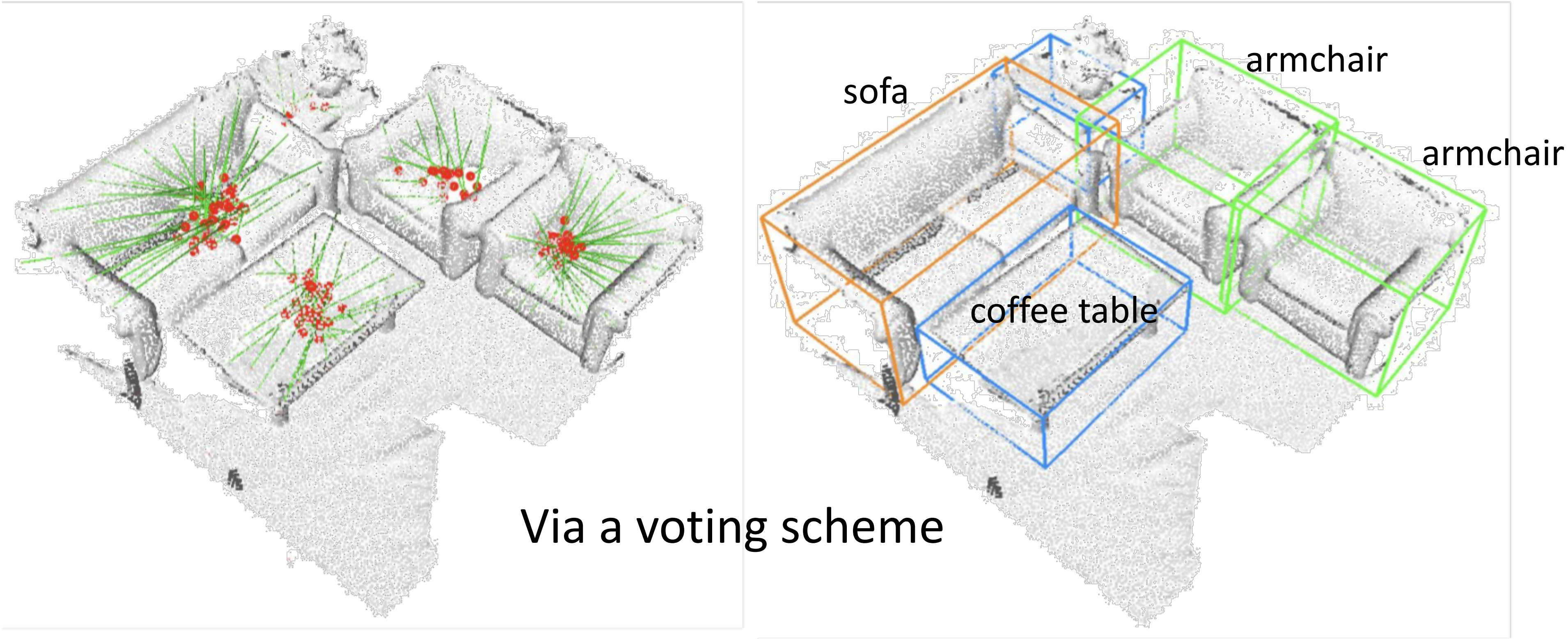
DGCNN alternates feature learning (EdgeConvs) and graph reconstruction (neighbor computation)

[Wang et al., TOG 2019]

# From Geometry to Semantics



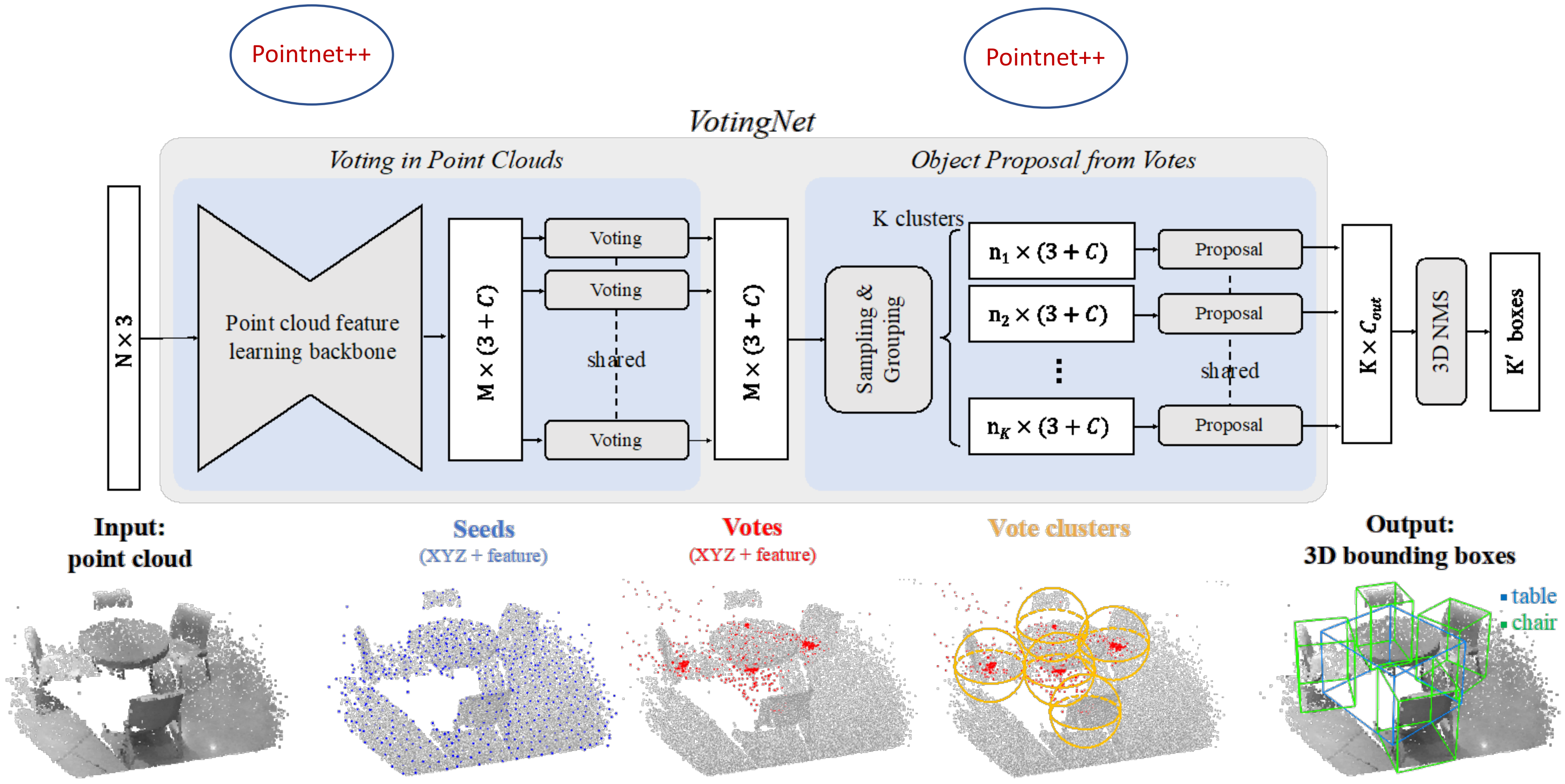
# Point Cloud Object Amodal Bounding Box Detection



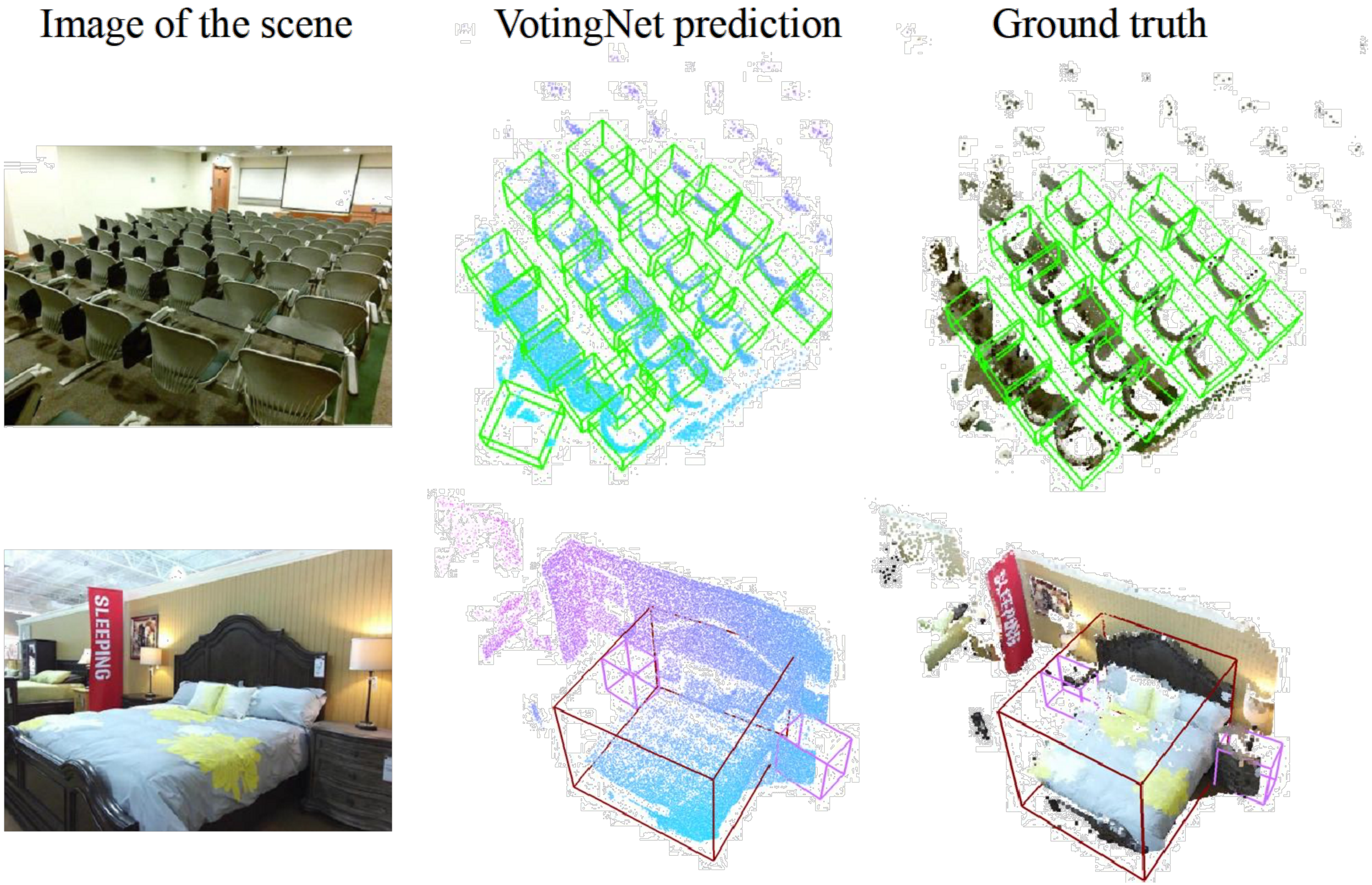

Via a voting scheme


- Charles R. Qi, Or Litany, Kaiming He, Leonidas J. Guibas. *Deep Hough Voting for 3D Object Detection in Point Clouds*. ICCV 2019.
- Charles R. Qi, Xinlei Chen, Or Litany, Leonidas J. Guibas. *ImVoteNet: Boosting 3D Object Detection in Point Clouds with Image Votes*. CVPR 2020.

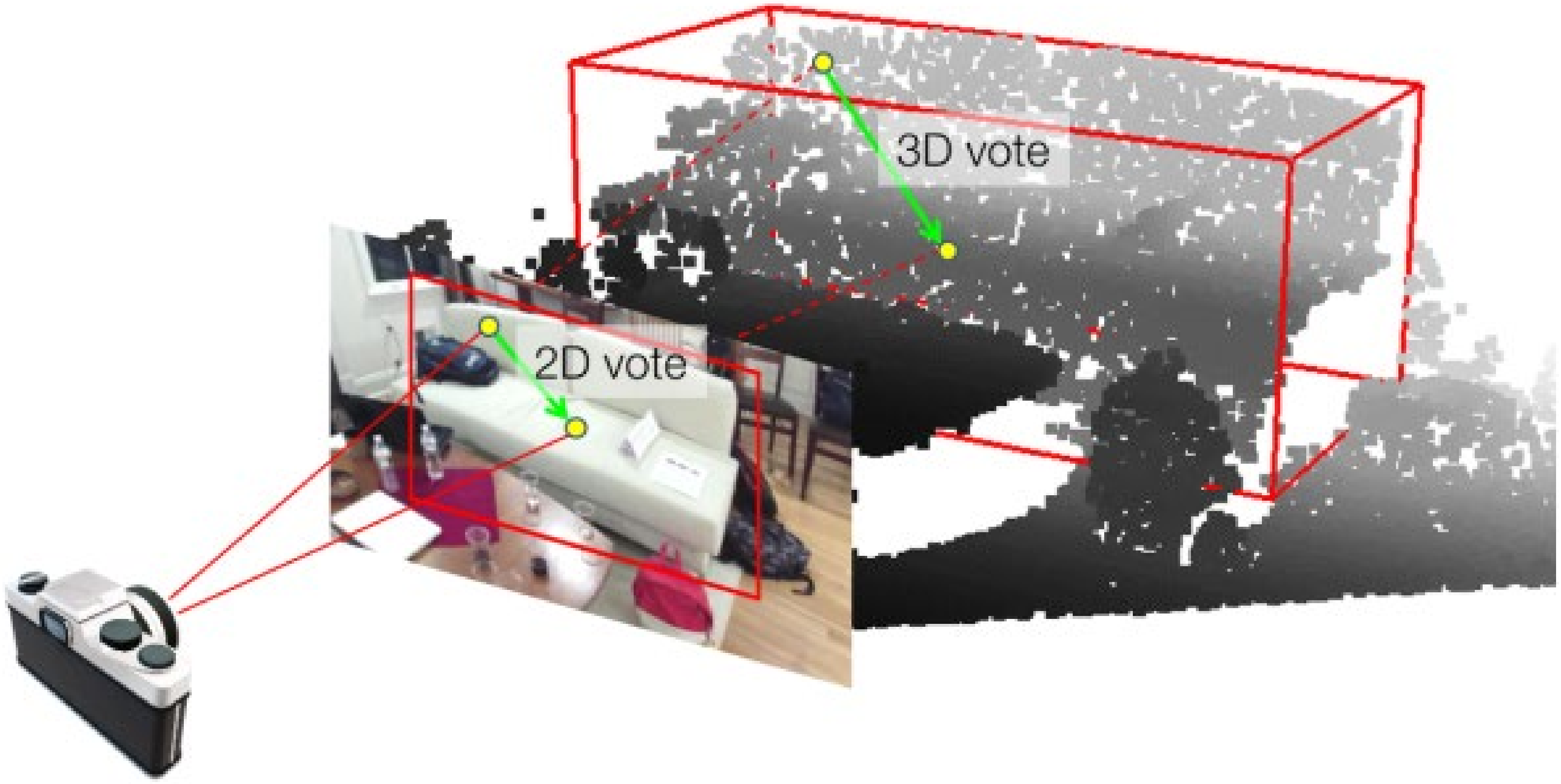
# VoteNet – A Two-Stage Approach



# VoteNet Results on SUN RGB-D

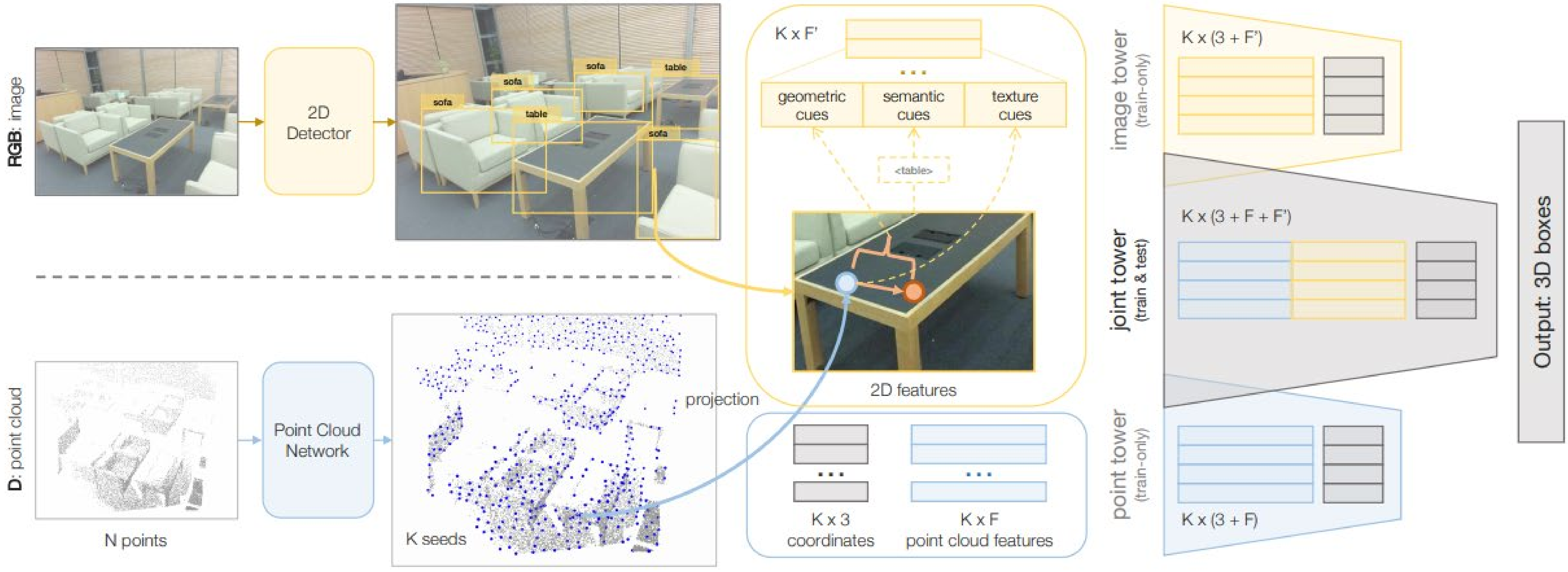


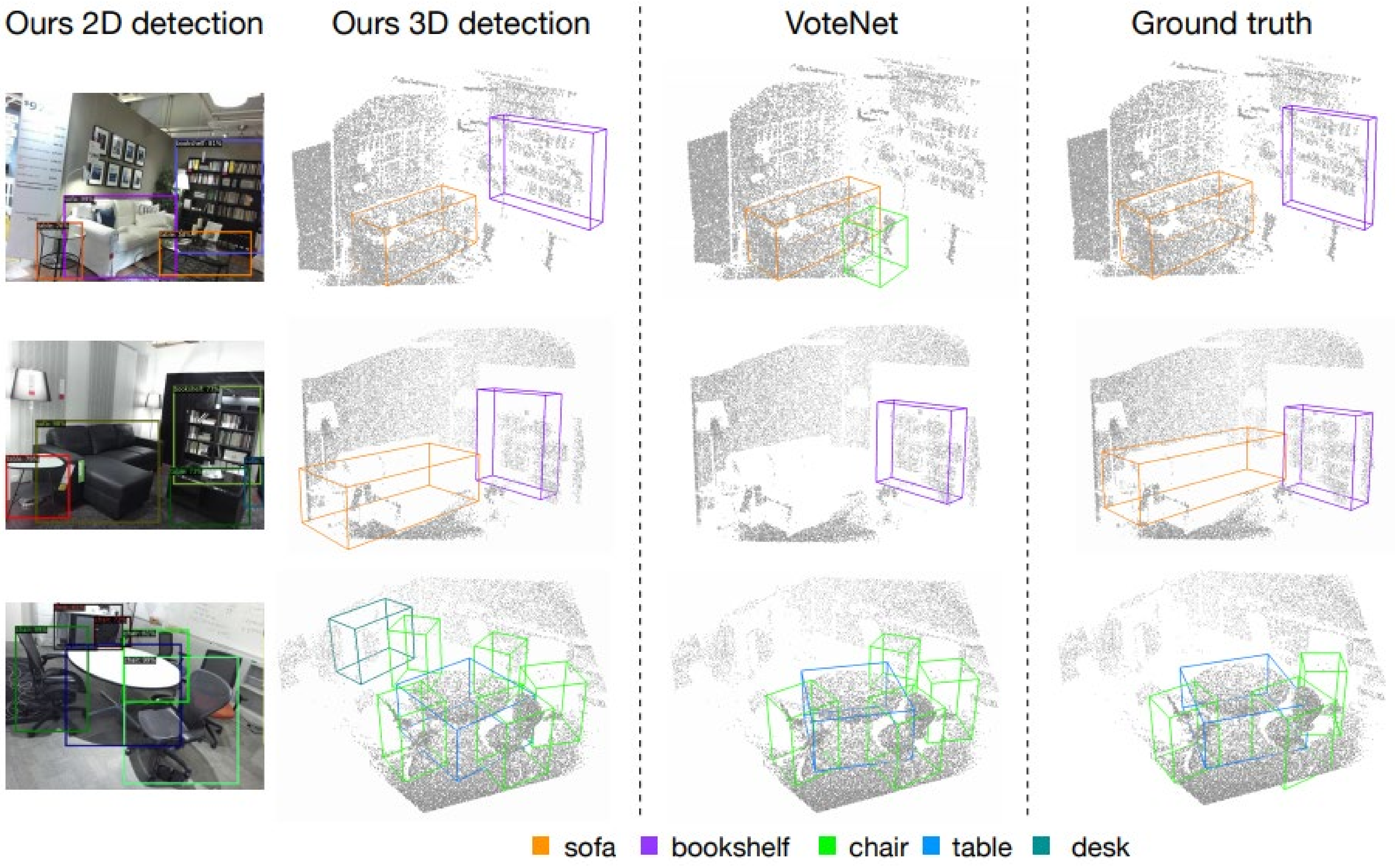
# Basic idea: *ImVoteNet*



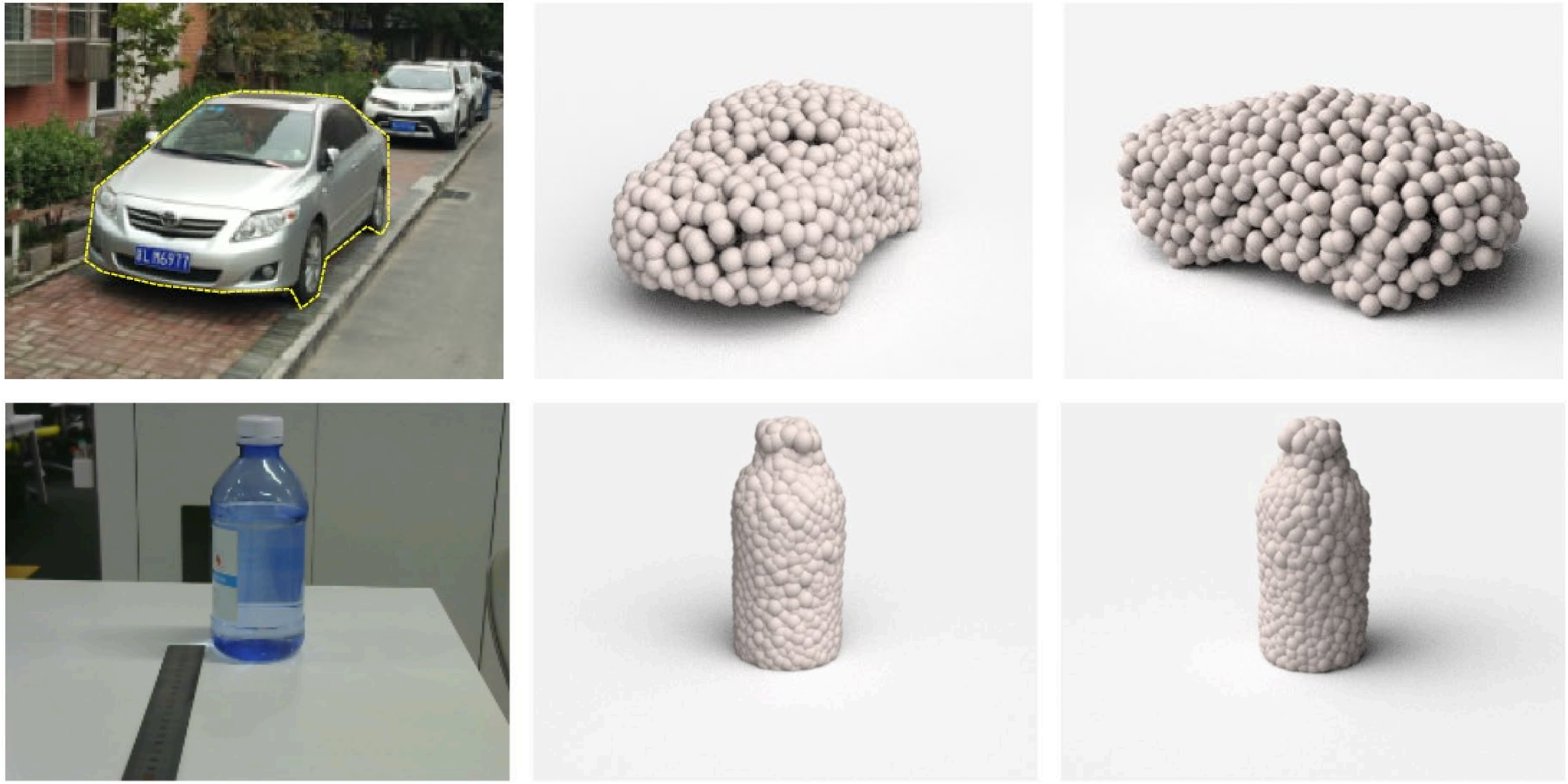

C. Qi, X. Chen, O. Litany, L.J. Guibas CVPR 2020

# *ImVoteNet* Architecture




Ours 2D detection
Ours 3D detection
VoteNet
Ground truth
sofa
bookshelf
chair
table
desk

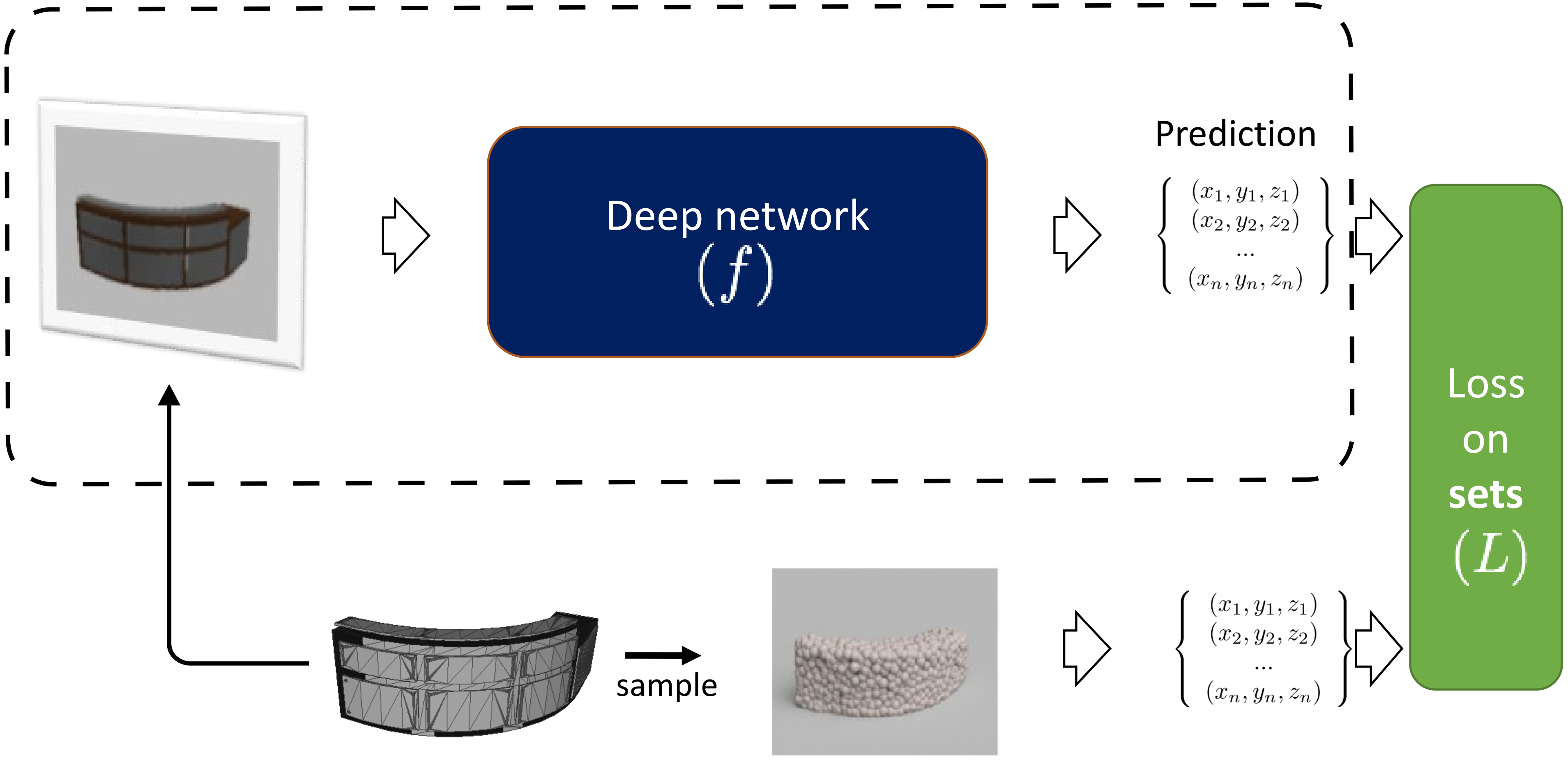
# Point Cloud Synthesis from a Single Image



Input Reconstructed 3D point cloud

[H. Su, H. Fan, LG, 2017]

# End-to-End Learning

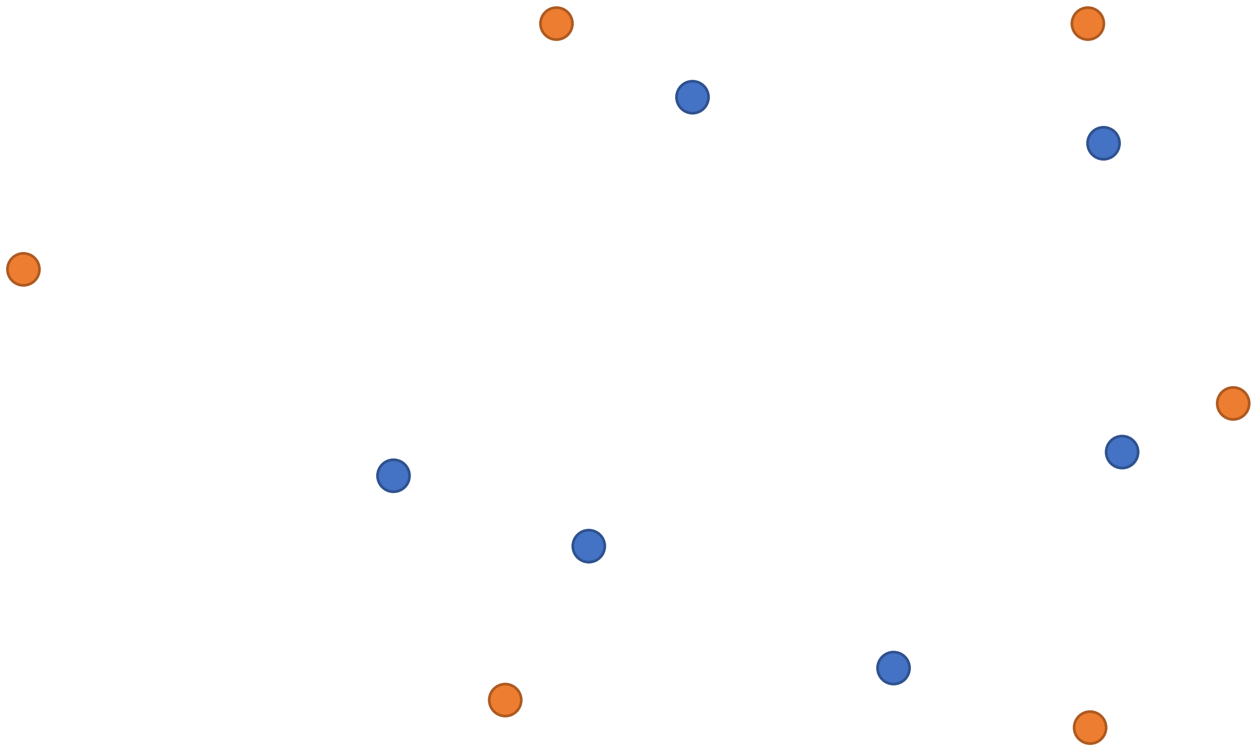


# Common Distance Metrics

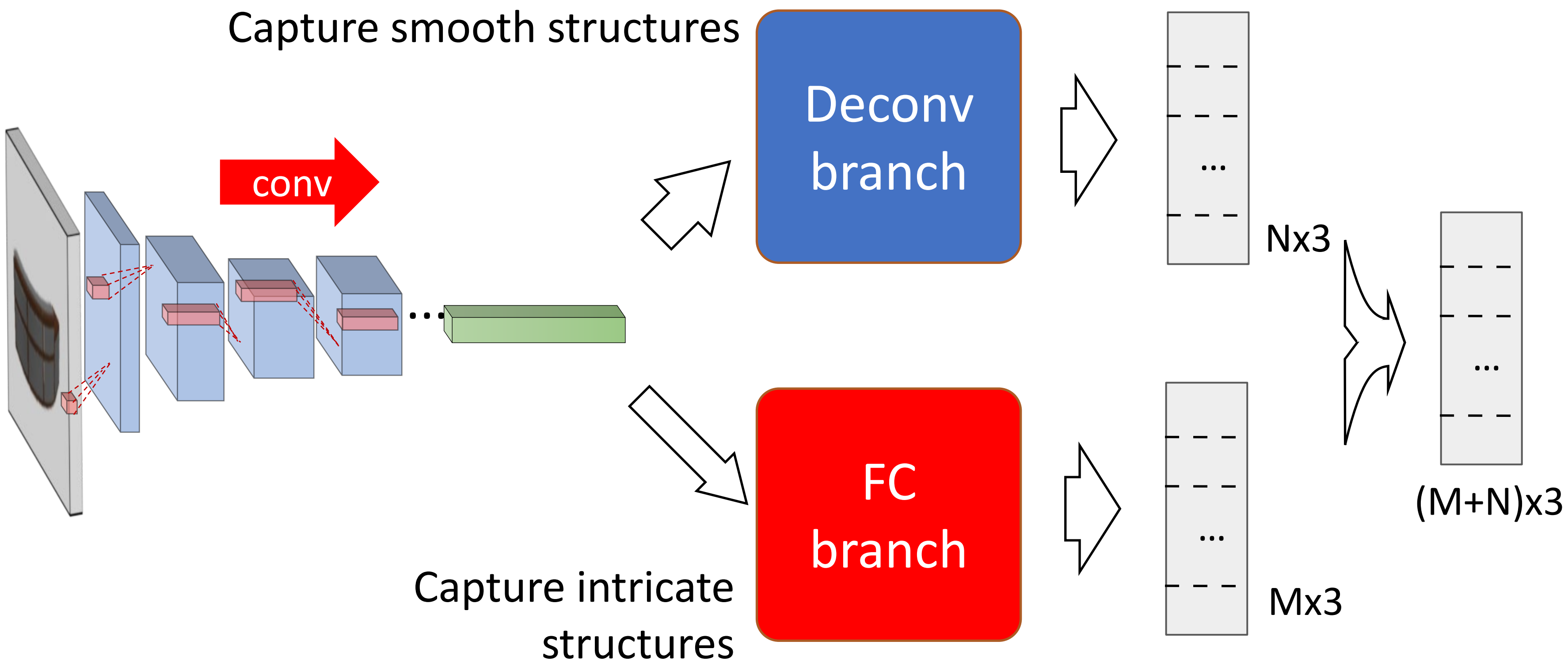
Worst case: Hausdorff distance (HD)

Average case: Chamfer distance (CD)

Optimal case: Earth Mover's distance (EMD)

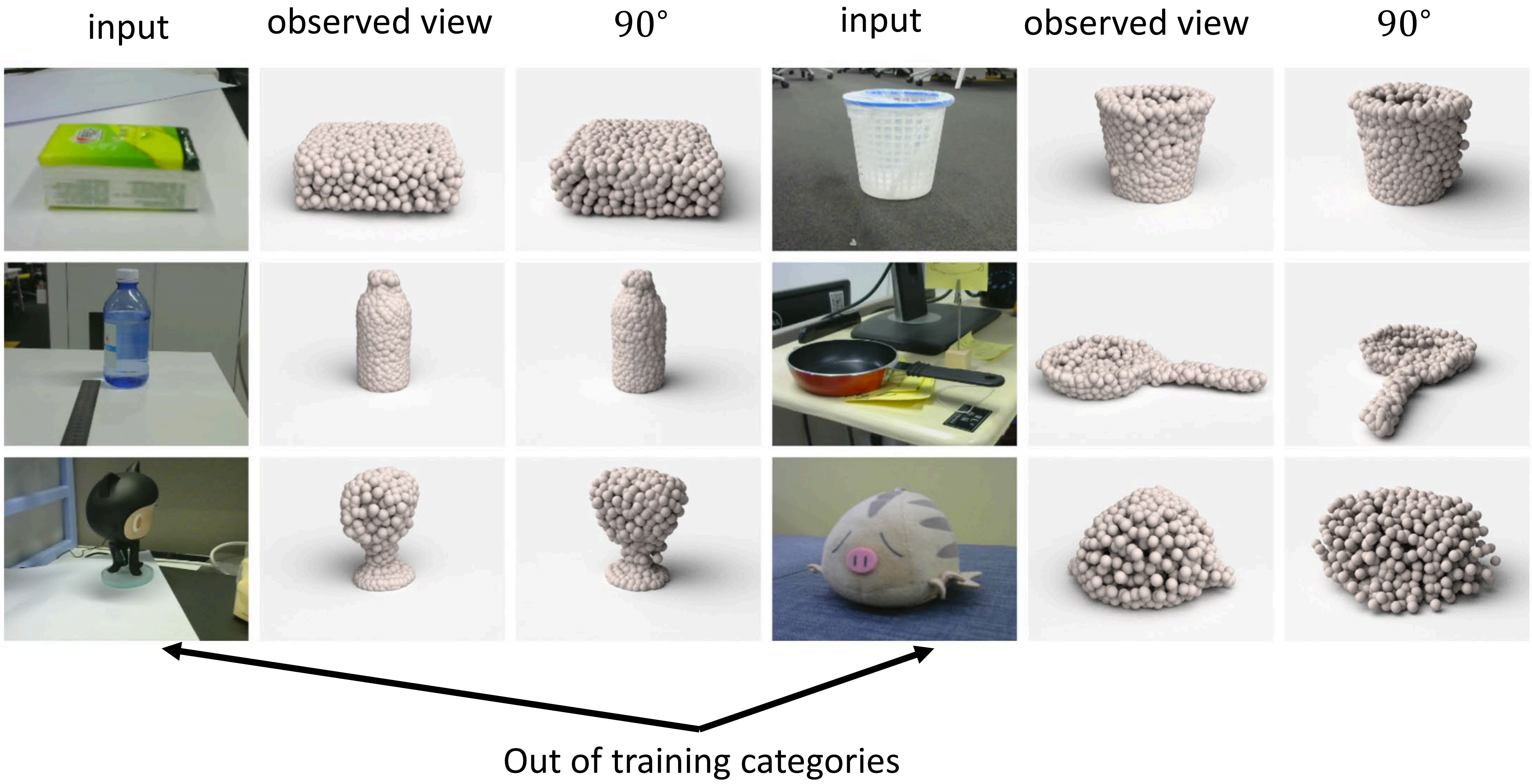


# Two-Branch Architecture



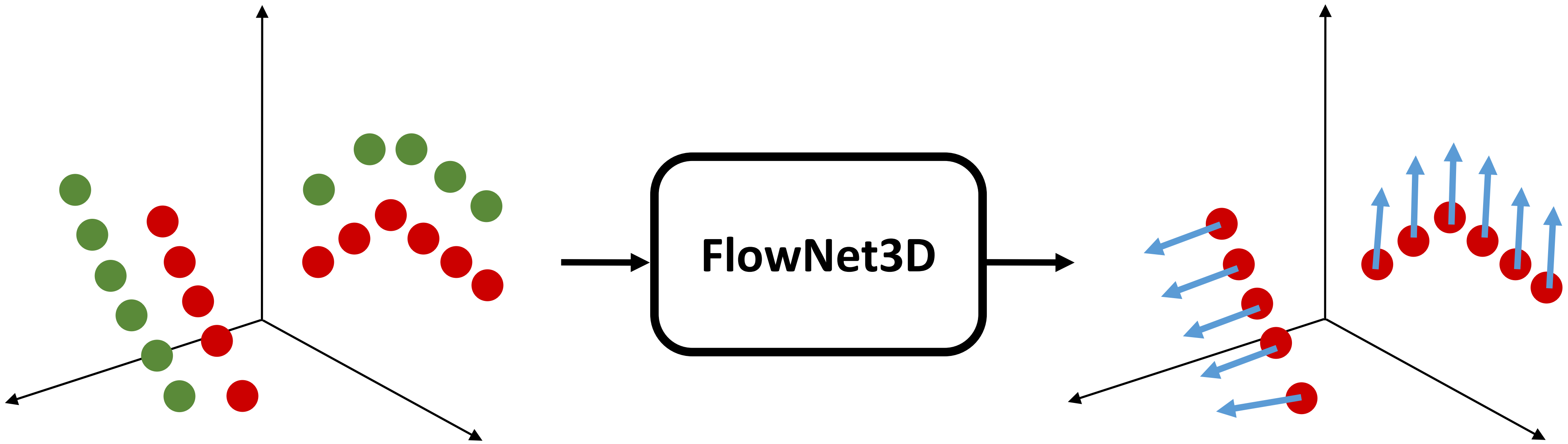

**Set union by array concatenation**

# From Real Images



# FlowNet3D

- Directly learning scene flow in 3D point clouds, with 3D deep learning architectures.

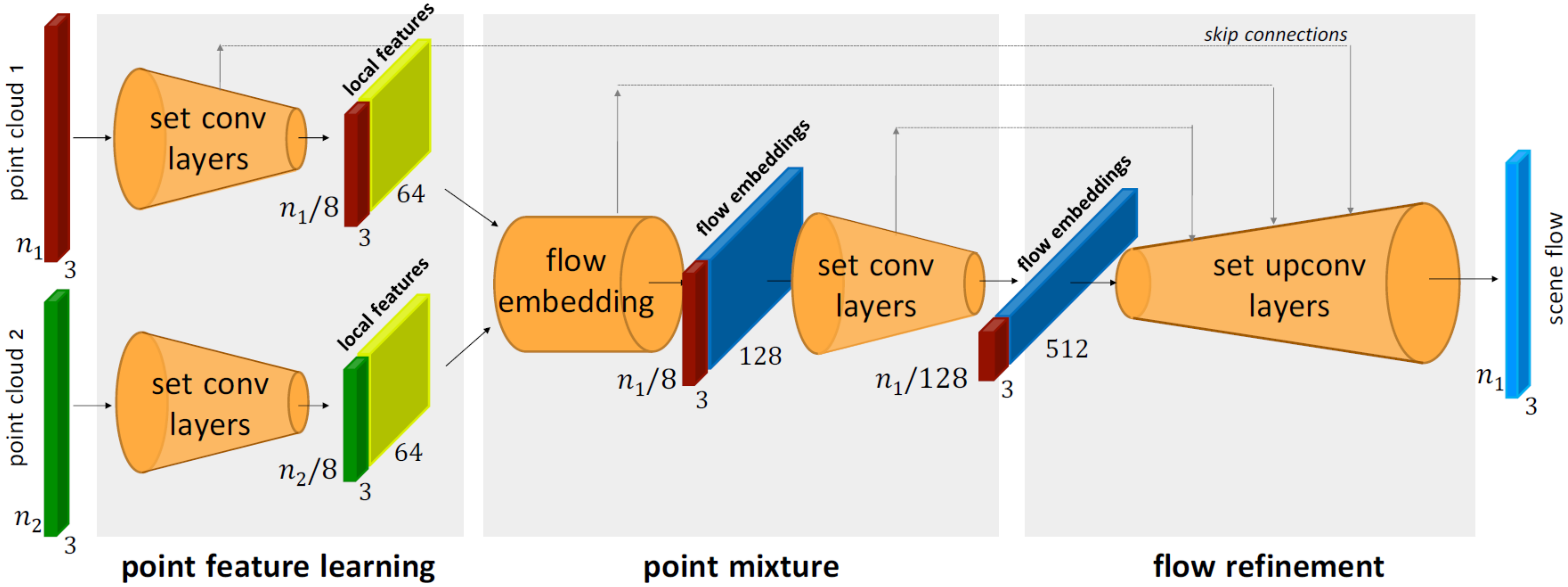



point cloud 1: N1x3
point cloud 2: N2x3

scene flow: N1x3

*Xingyu Liu, Charles R. Qi, Leonidas Guibas. Learning Scene Flow in 3D Point Clouds, (CVPR 2019).*

# FlowNet3D




set conv = set abstraction

Composed of many many mini-pointnet++ modules …

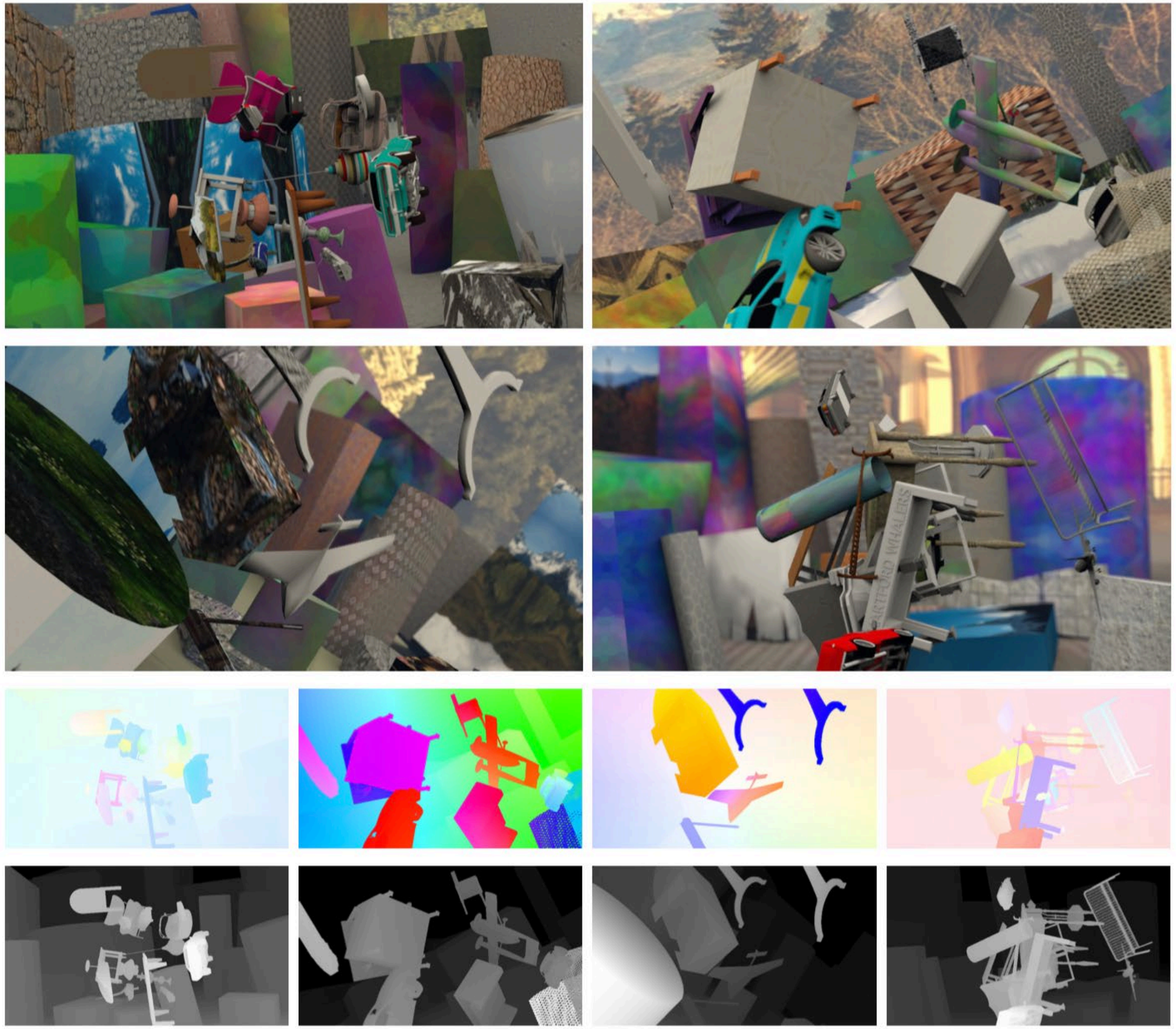
Pointnet++

# Training on Synthetic Data

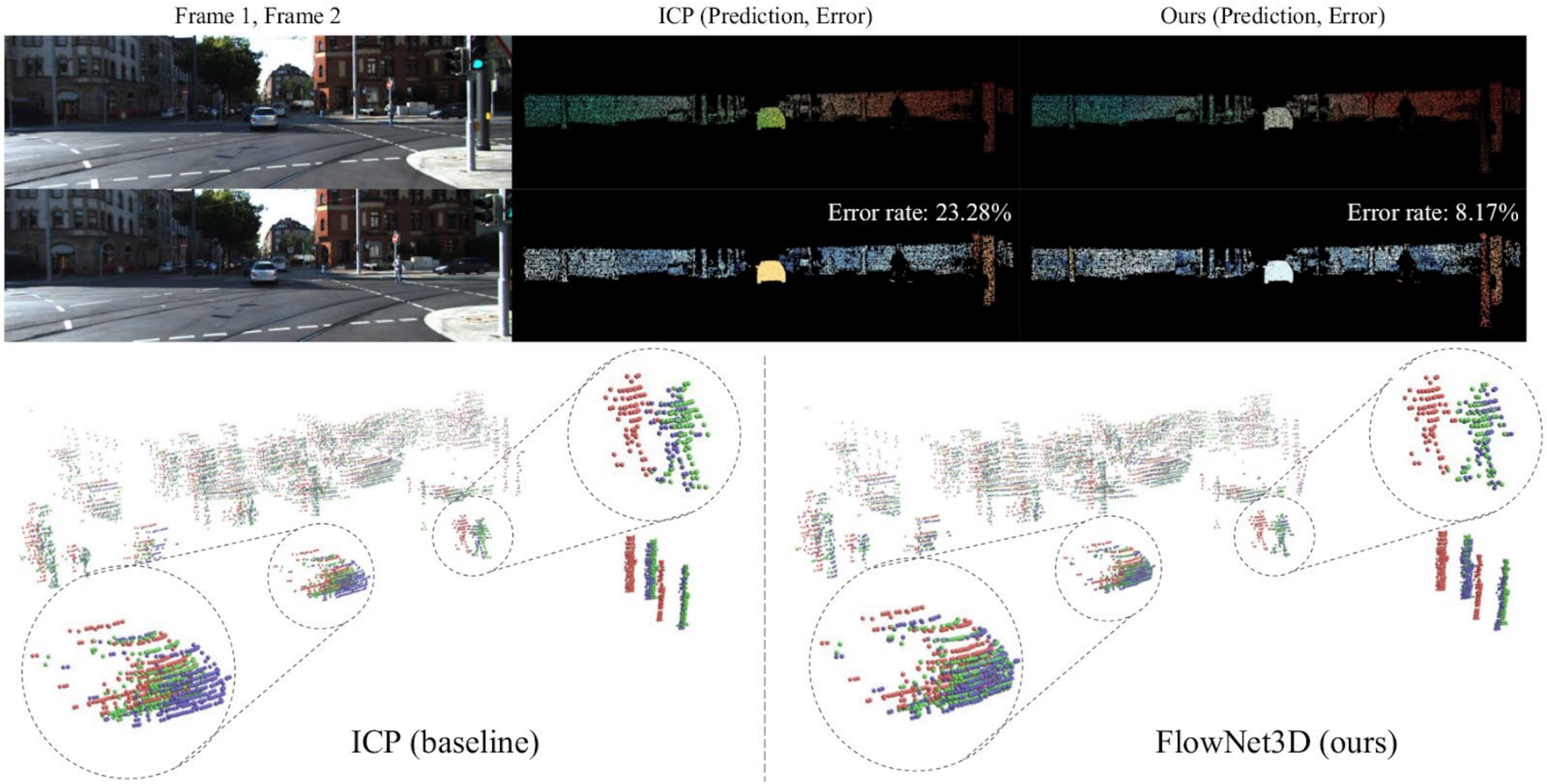
FlyingThings3D [Mayer et al. 2016] dataset from MPI

Random ShapeNet objects

Very challenging dataset with strong occlusions and large motions.



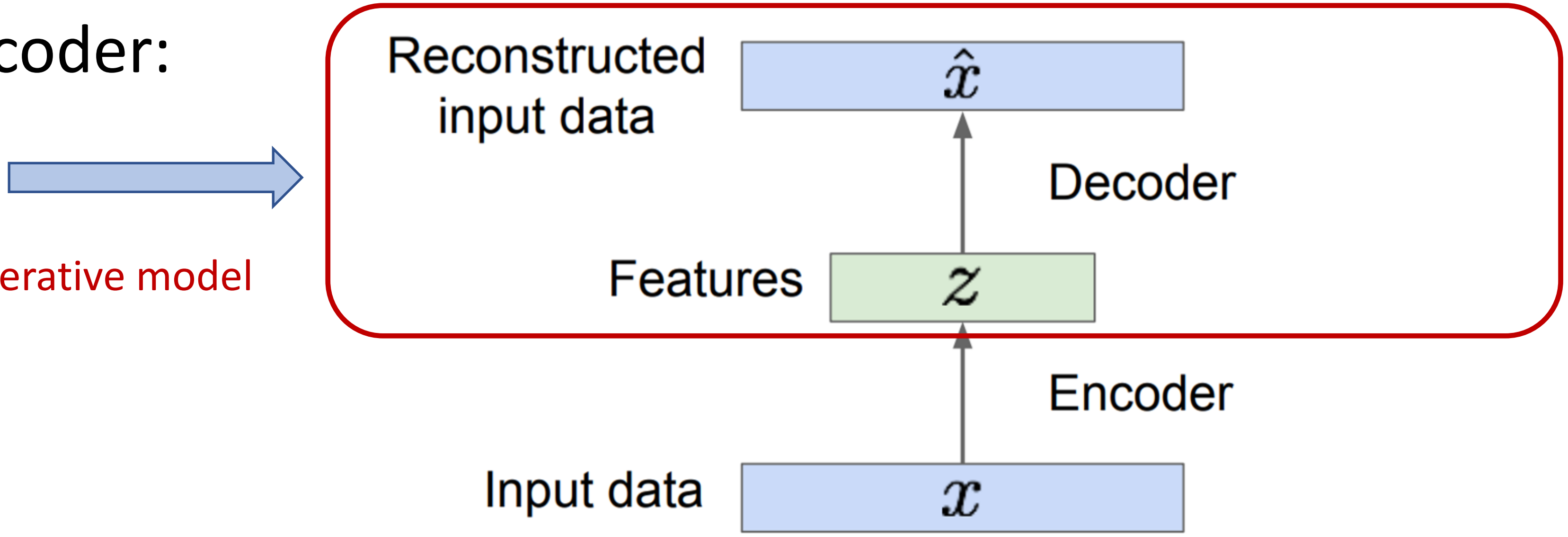
# KITTI Results



# Generative Models: Autoencoders and Variational Autoencoders

# Deep Generative Models: VAEs

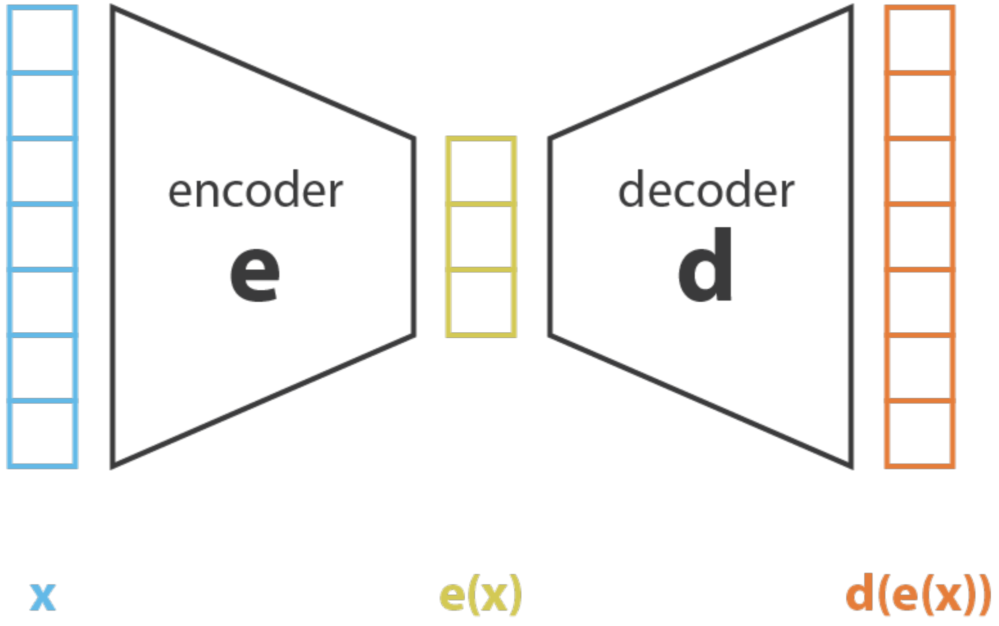
- Autoencoder:




Use as generative model

- Variational autoencoder (VAE): an autoencoder whose encoding distribution is regularized during training in order to ensure that its latent space has good properties, allowing us to generate new data
- Related to variational inference in Statistics

# Dimensionality Reduction




**initial data**
in space $R^n$

**encoded data**
in latent space $R^m$ (with $m<n$)

**encoded-decoded data**
back in the initial space $R^n$

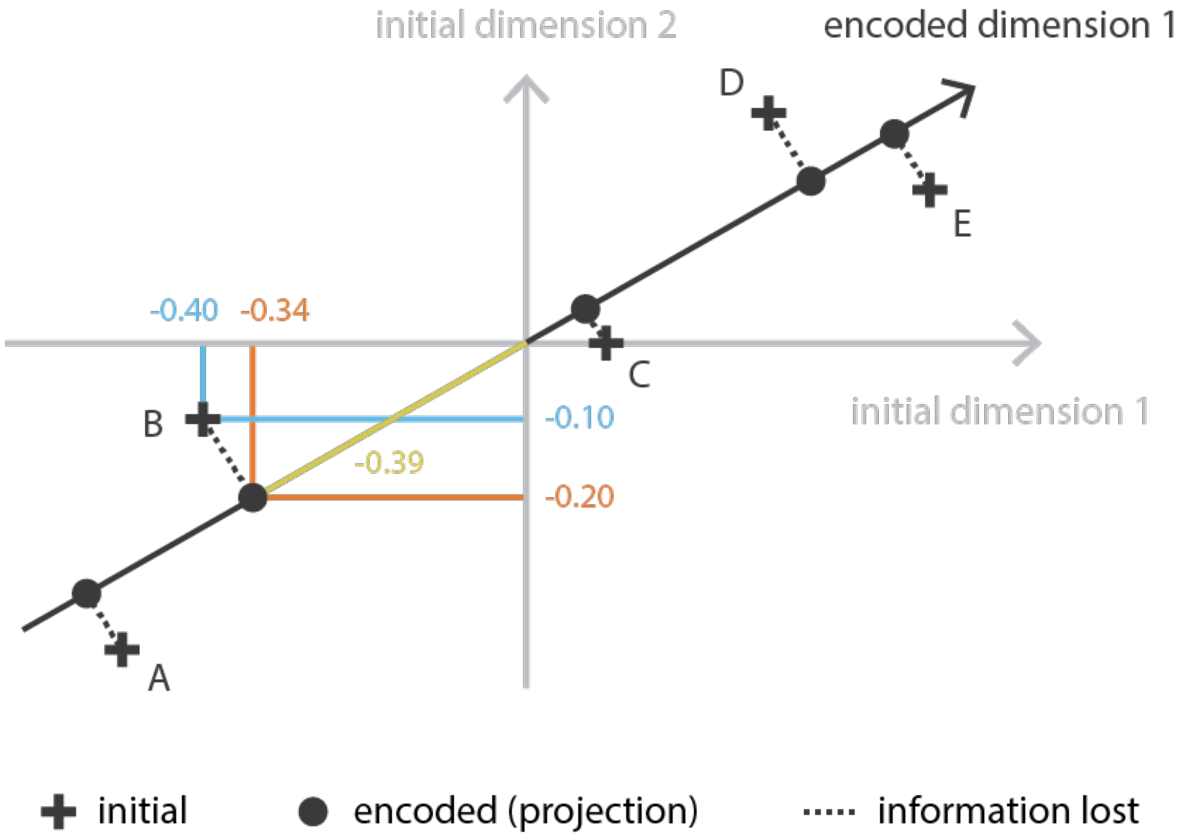
$x = d(e(x))$ ➡ **lossless encoding**
no information is lost when reducing the number of dimensions

$x \neq d(e(x))$ ➡ **lossy encoding**
some information is lost when reducing the number of dimensions and can't be recovered later

$$(e^*, d^*) = \underset{(e,d)\in E\times D}{\arg\min}\ \epsilon(x, d(e(x)))$$

# Detour: Principal Components Analysis (PCA)

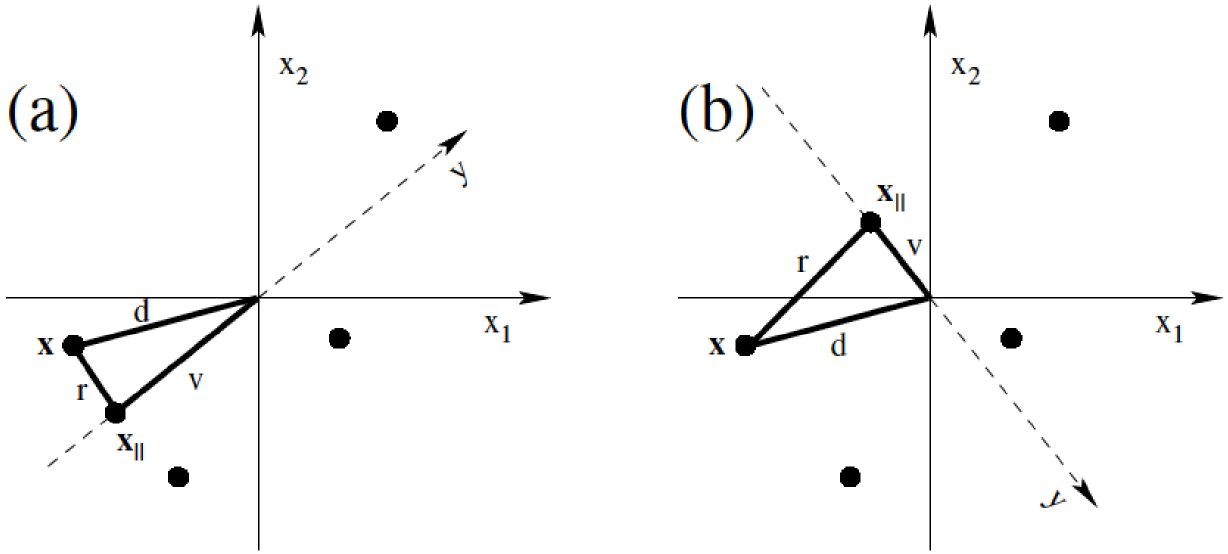
- Build new features that are linear combinations of old features




| Point | Initial | Encoded | Decoded |
|---|---|---|---|
| A | (-0.50, -0.40) | -0.63 | (-0.54, -0.33) |
| B | (-0.40, -0.10) | -0.39 | (-0.34, -0.20) |
| C | (0.10, 0.00) | 0.09 | (0.07 0.04) |
| D | (0.30, 0.30) | 0.41 | (0.35, 0.21) |
| E | (0.50, 0.20) | 0.53 | (0.46, 0.27) |

# Reconstruction Error and Variance

- For centred data, minimizing the reconstruction error is equivalent to maximizing the variance of the projected data.

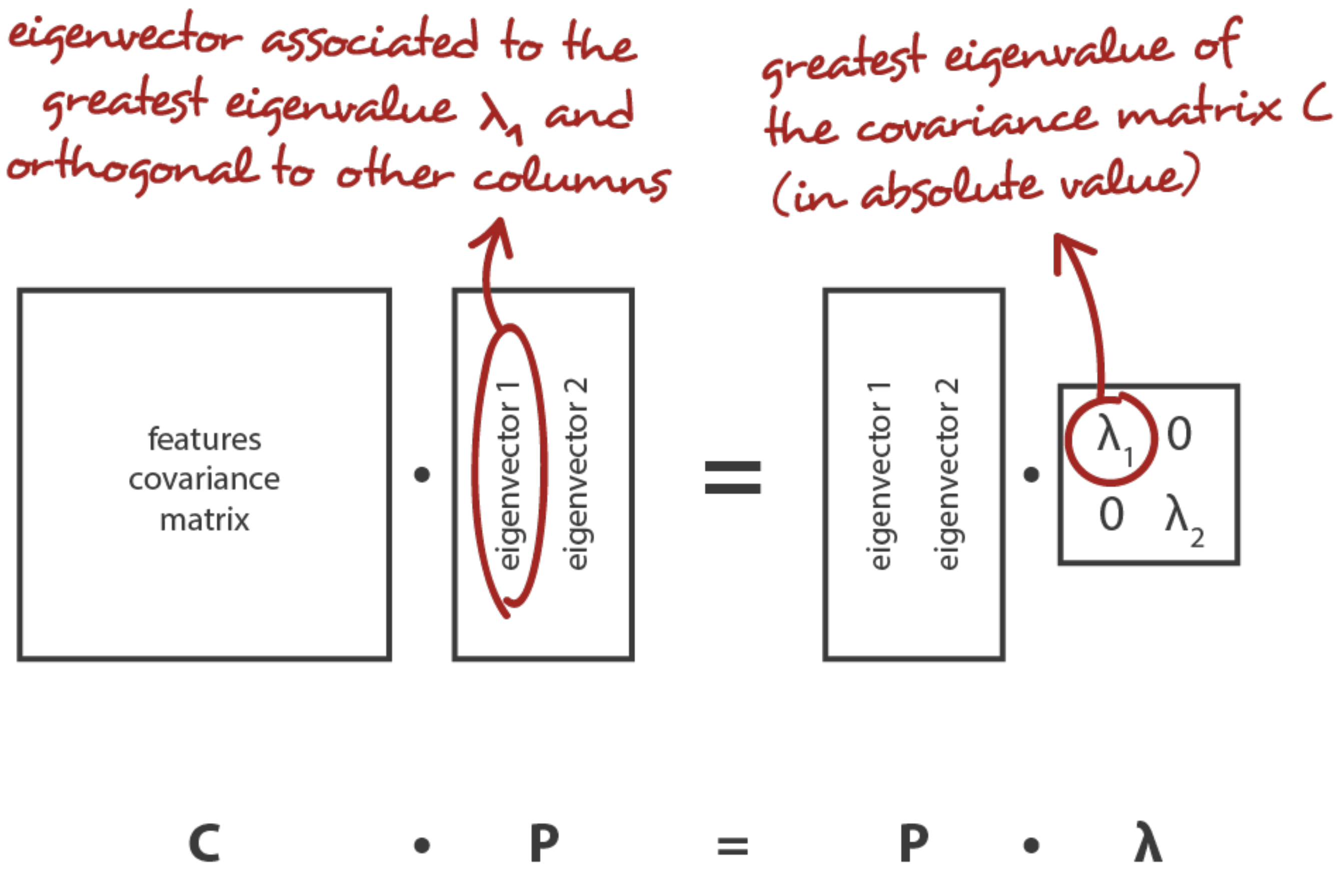


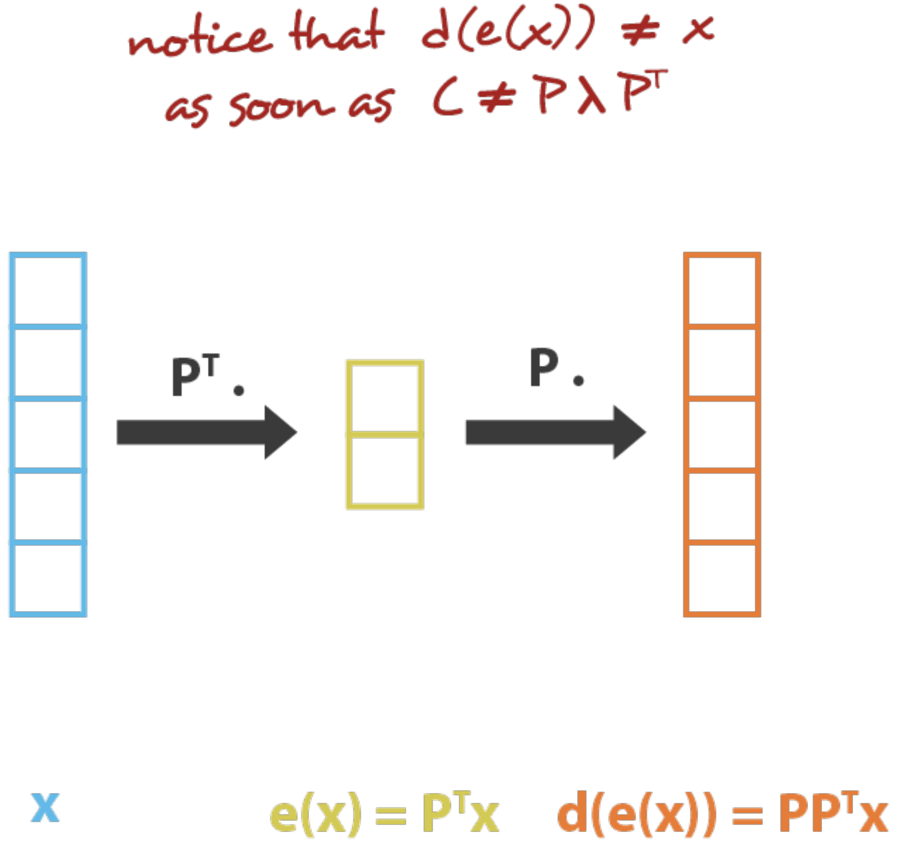

NB, projections of centered data are centered

CS233 material …

$$r^2 + v^2 = d^2$$

# Eigen-analysis of the Data Covariance Matrix



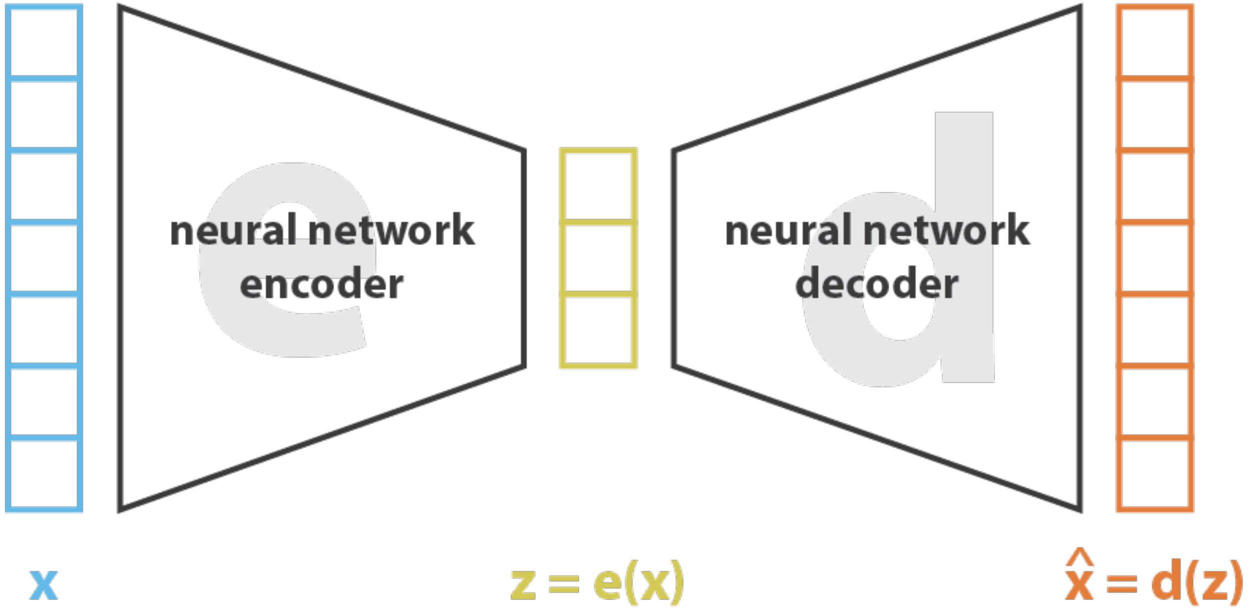
# The General PCA Problem

**Principal Component Analysis (PCA): Given a set** $\{\mathbf{x}^\mu : \mu = 1, ..., M\}$ **of** $I$**-dimensional data points** $\mathbf{x}^\mu = (x_1^\mu, x_2^\mu, ..., x_I^\mu)^T$ **with zero mean,** $\langle \mathbf{x}^\mu \rangle_\mu = \mathbf{0}_I$**, find an orthogonal matrix U with determinant** $|\mathbf{U}| = +1$ **generating the transformed data points** $\mathbf{x}'^\mu := \mathbf{U}^T \mathbf{x}^\mu$ **such that for any given dimensionality** $P$ **the data projected onto the first** $P$ **axes,** $\mathbf{x}'^\mu_{||} := (x'^\mu_1, x'^\mu_2, ..., x'^\mu_P, 0, ..., 0)^T$**, have the smallest**

$$\textbf{reconstruction error} \quad E := \langle \| \mathbf{x}'^\mu - \mathbf{x}'^\mu_{||} \|^2 \rangle_\mu \qquad (8)$$

**among all possible projections onto a** $P$**-dimensional subspace. The row vectors of matrix U define the new axes and are called the** ***principal components.***

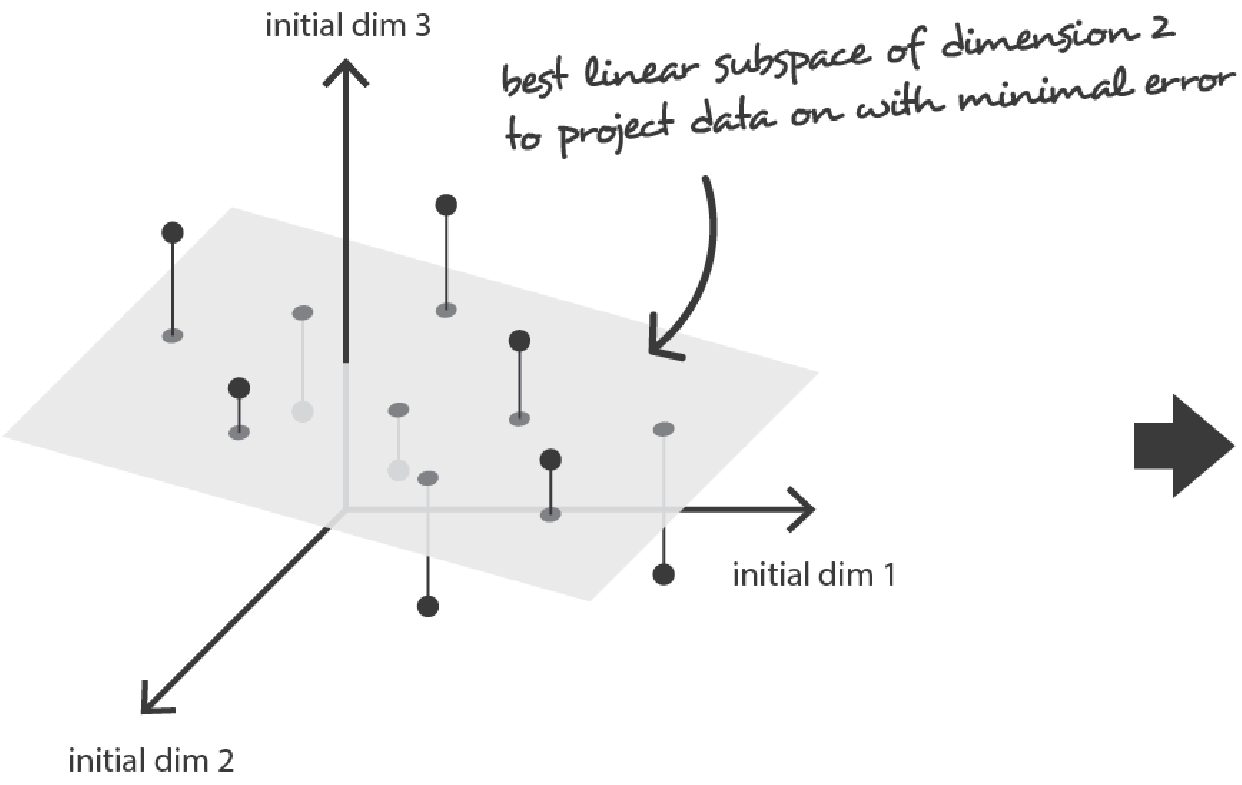
# Autoencoder: Use Neural Nets for E and D

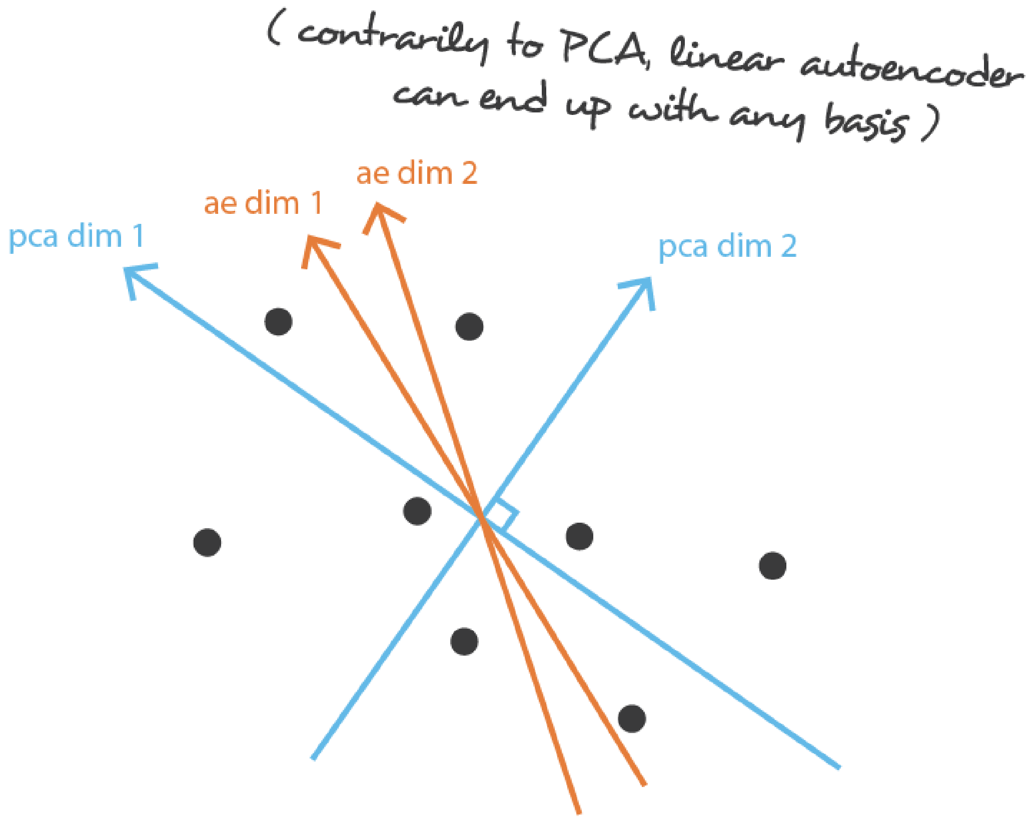



Use more powerful encoders and decoders

$$\text{loss} = || x - \hat{x} ||^2 = || x - d(z) ||^2 = || x - d(e(x)) ||^2$$

# Autoencoder vs PCA



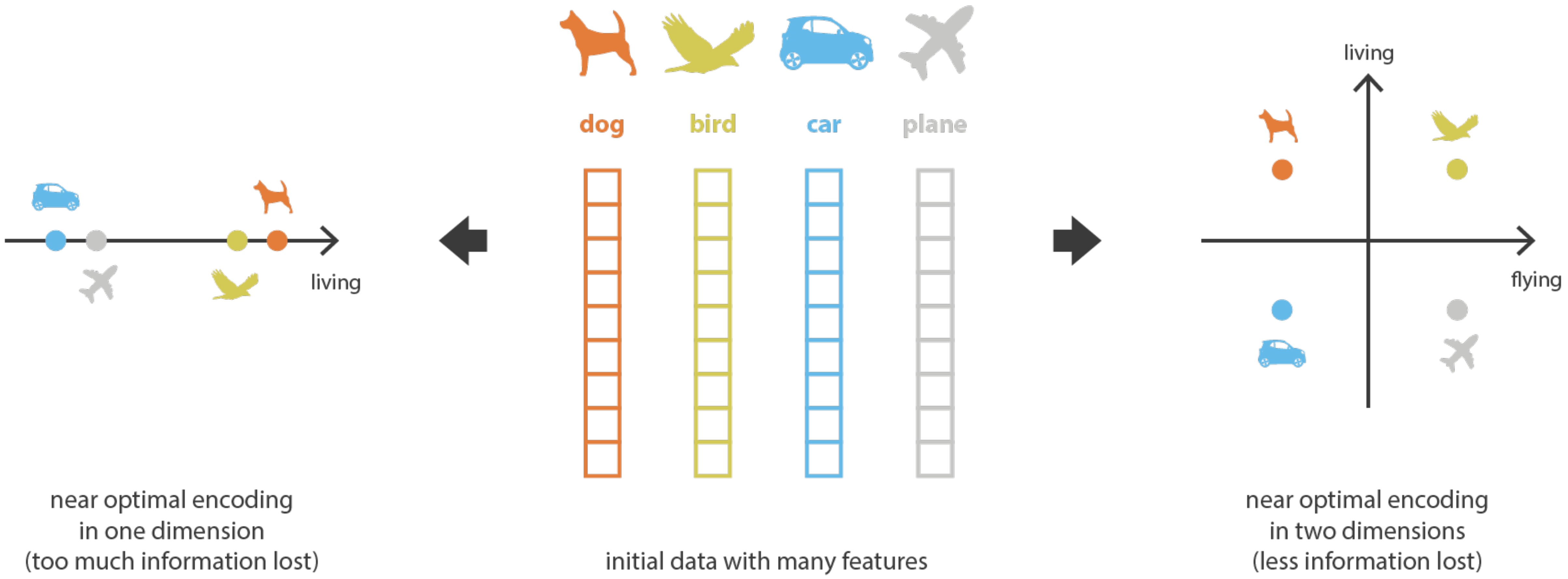

**Data in the full initial space**

In order to reduce dimensionality, PCA and linear autoencoder target, in theory, the same optimal subspace to project data on...

**Data projected on the best linear subspace**

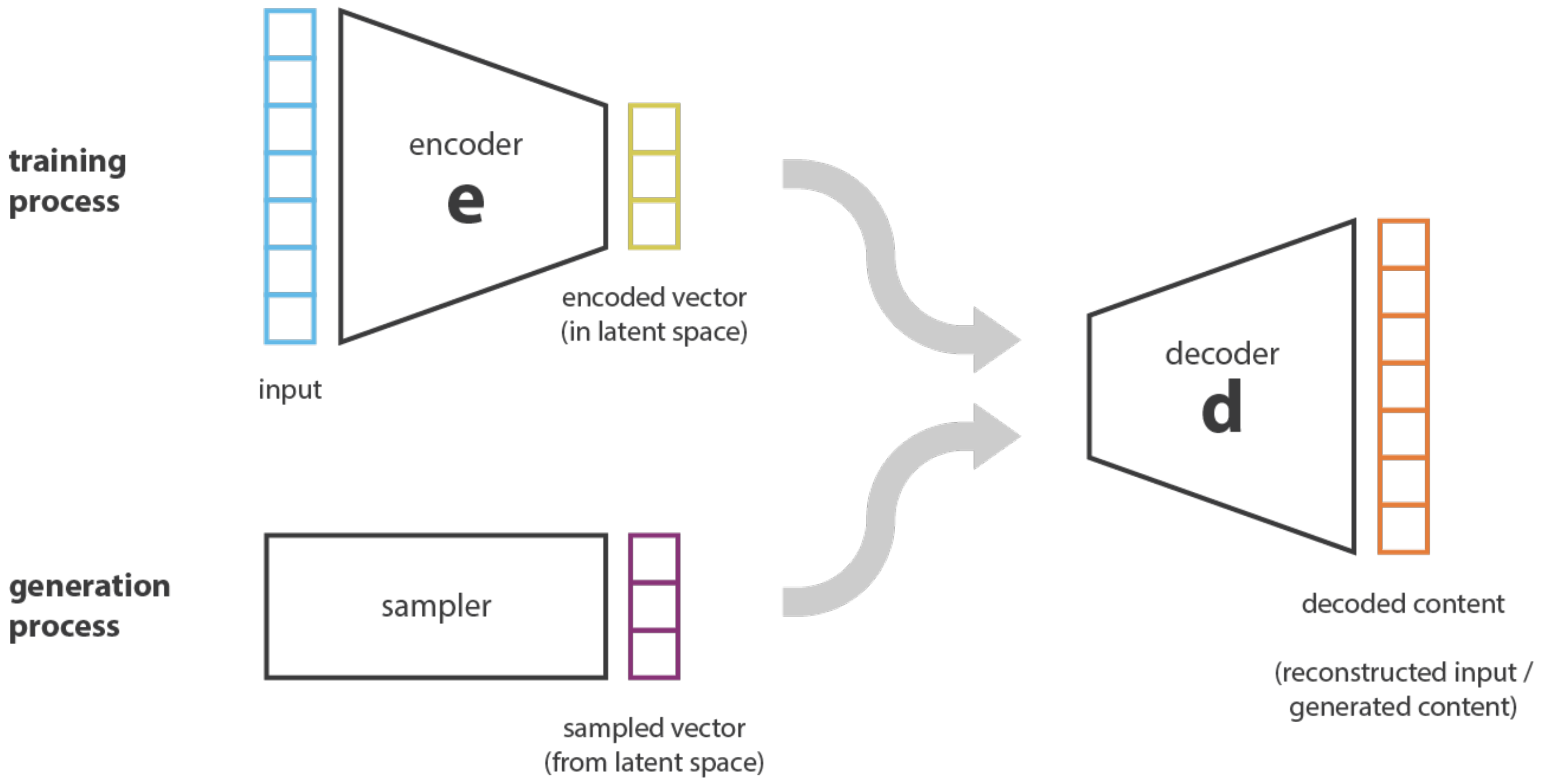
... but not necessarily with the same basis due to different constraints (in PCA the first component is the one that explains the maximum of variance and components are orthogonal)

# Don't Overencode!



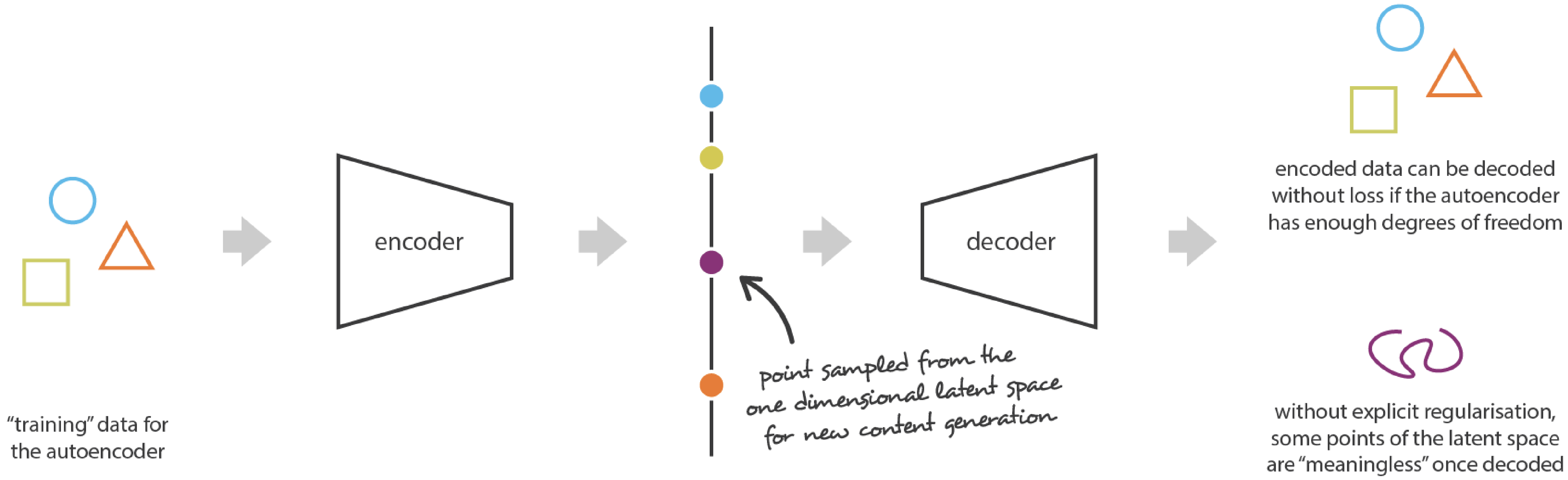

We want to structure of latent space to reflect the structure of the data – especially if we want to sample the latent space for generating new data

# Autoencoders for Content Generation?




Severe overfitting

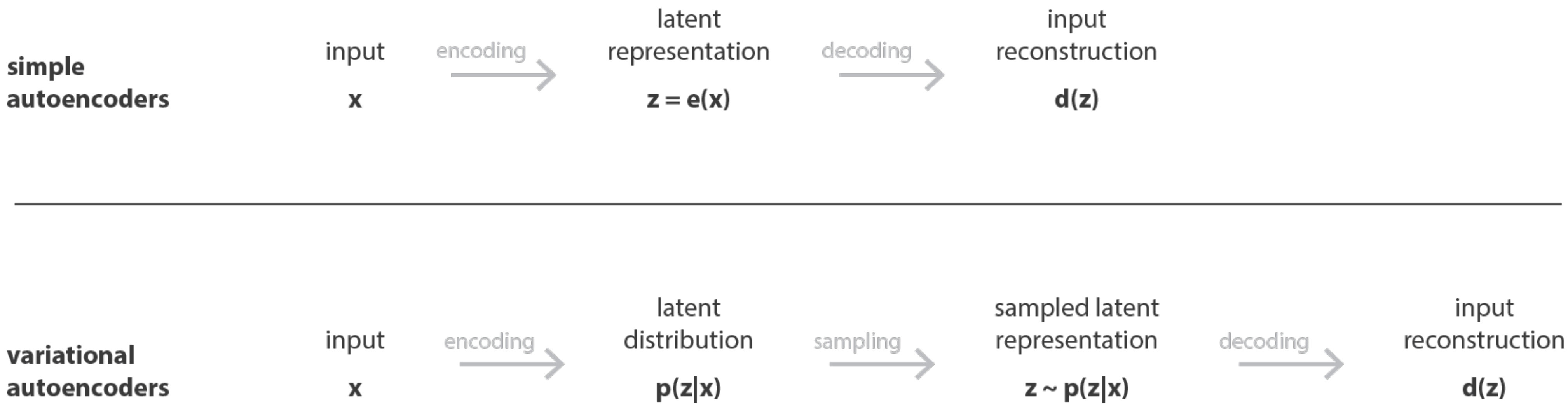
# Autoencoders for Content Generation?



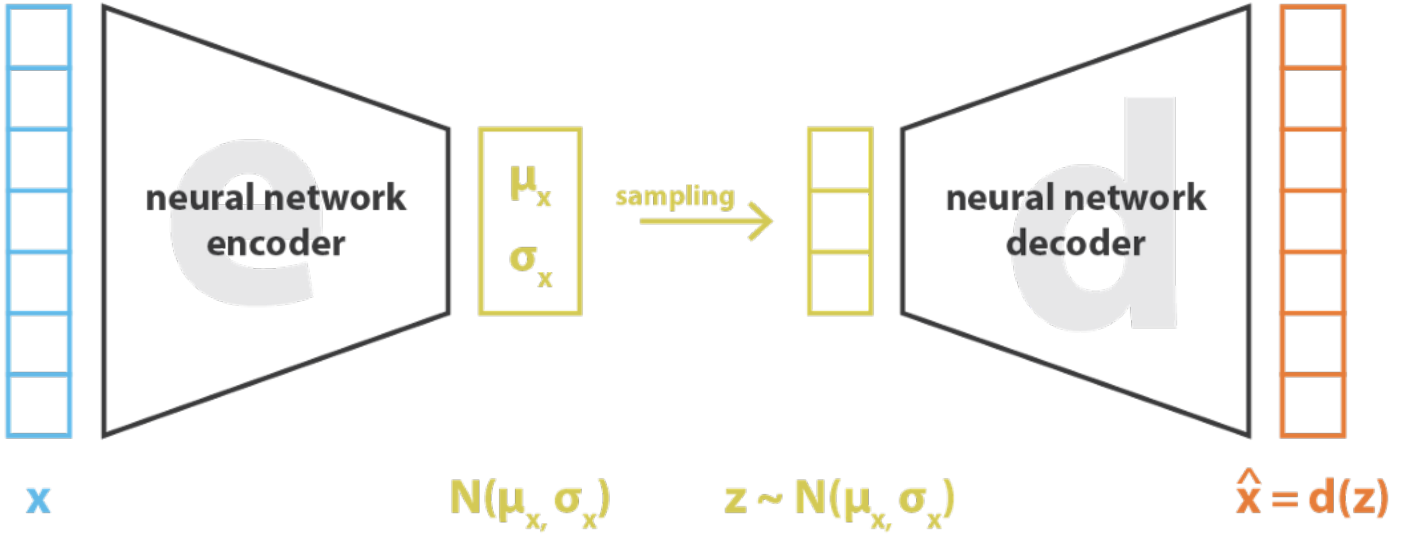

An autoencoder is solely trained to encode and decode with as small loss as possible, no matter how the latent space is organized

# Variational Autoencoder (VAE)

- A variational autoencoder is an autoencoder whose training is regularized
  - to avoid overfitting and
  - to ensure that the latent space has good properties that enable generative processes
- Instead of encoding an input as a single point, we encode it as a distribution over the latent space.



# Regularize the Distribution in Latent Space

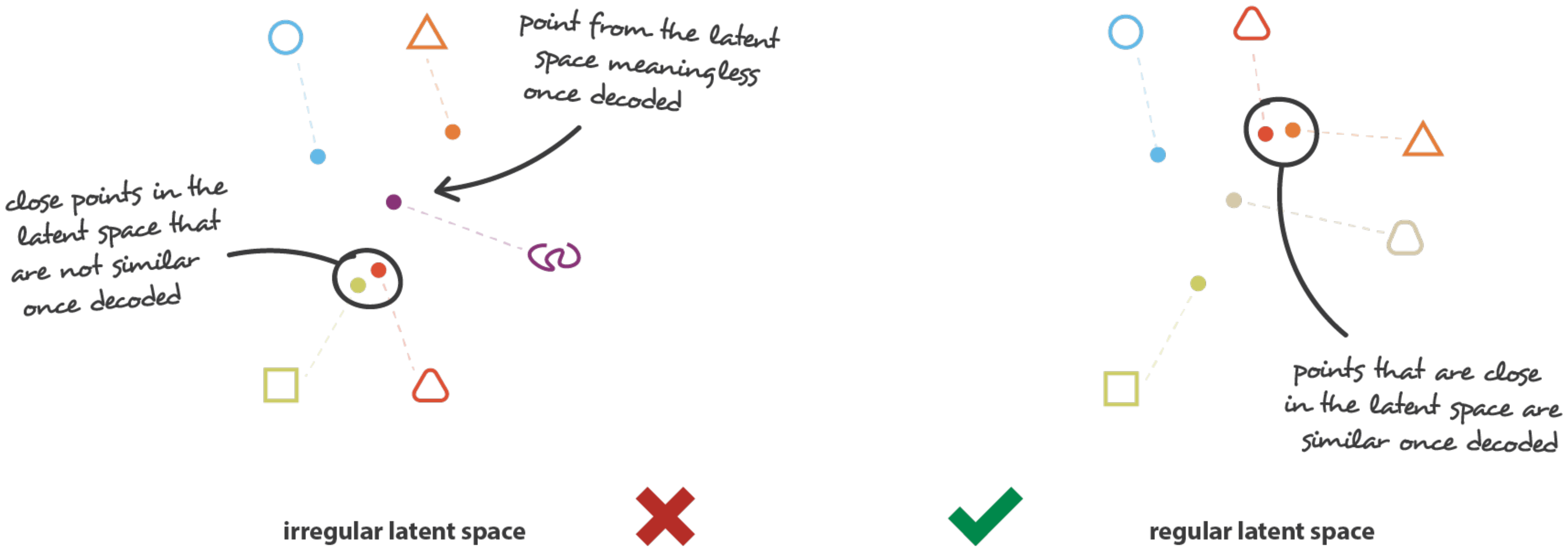



$$\text{loss} = \|x - \hat{x}\|^2 + \text{KL}[\, N(\mu_x, \sigma_x), N(0, I)\,] = \|x - d(z)\|^2 + \text{KL}[\, N(\mu_x, \sigma_x), N(0, I)\,]$$

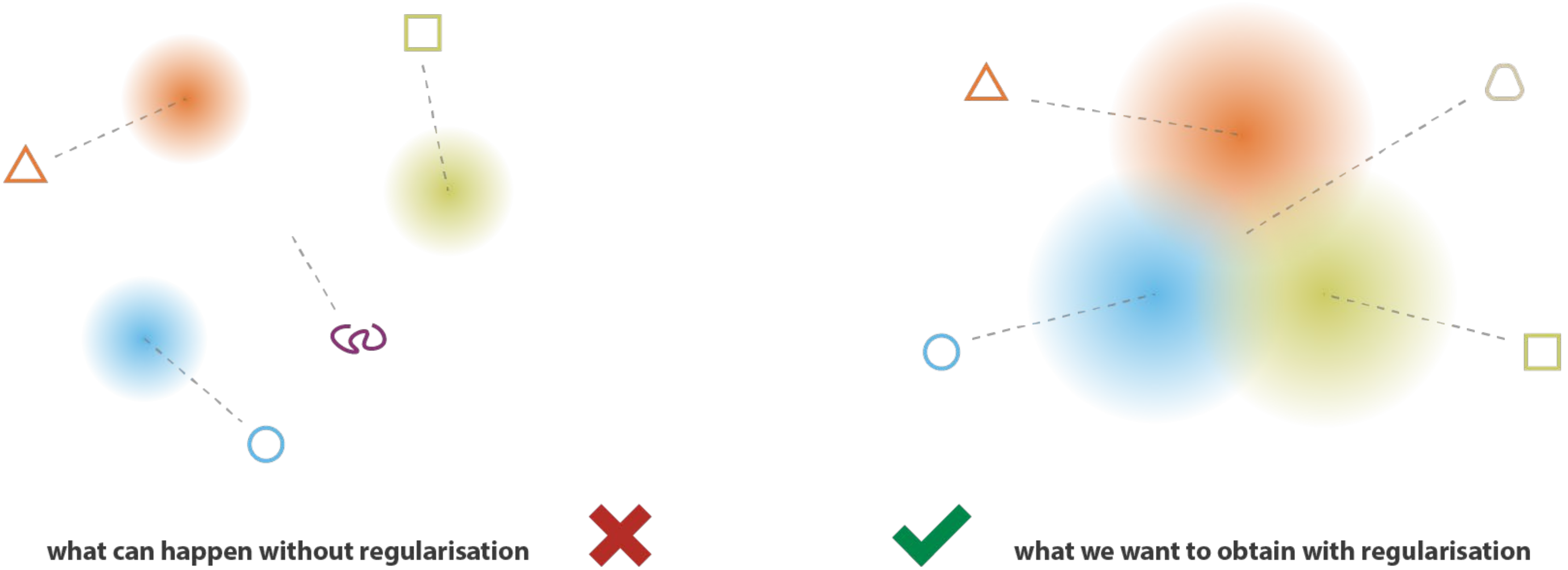
Make the latent space distribution look like a simple Gaussian

Add a second loss measuring distribution distance (via the Kulback-Leibler divergence)

# Continuity and Completeness in Latent Space

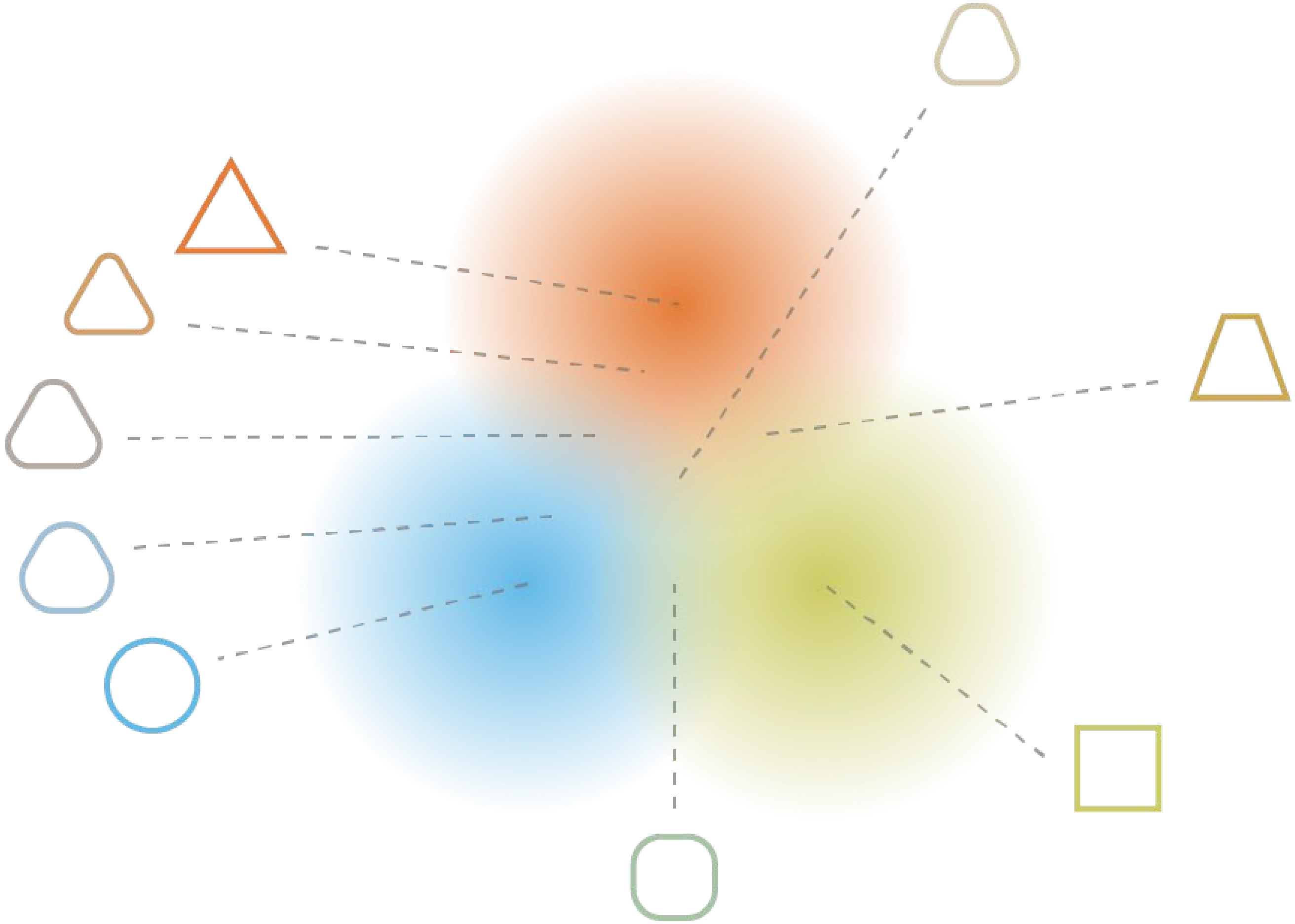


# The Effect of Regularization



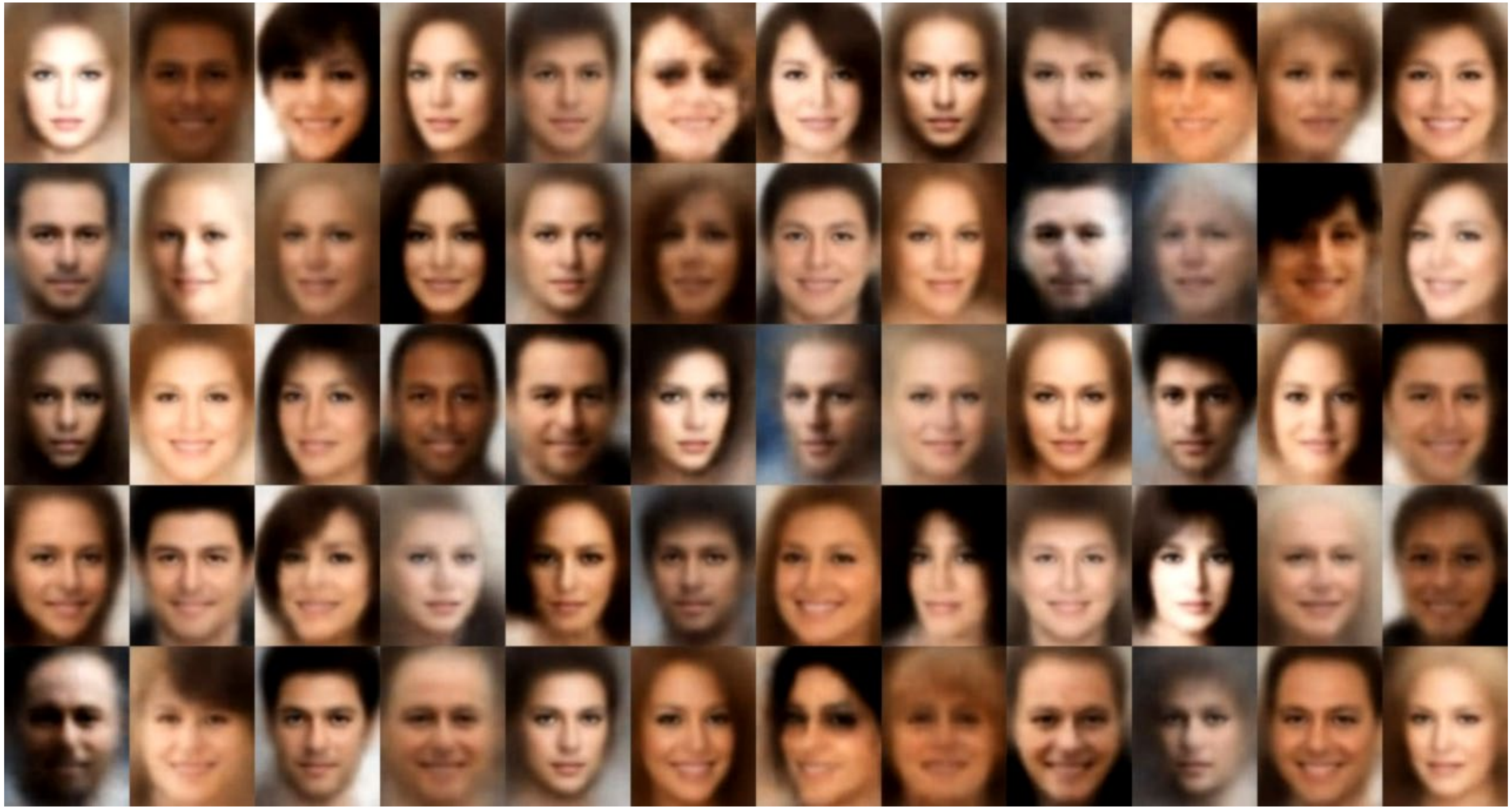

Overfitting with “punctual” distributions

# The Effect of Regularization



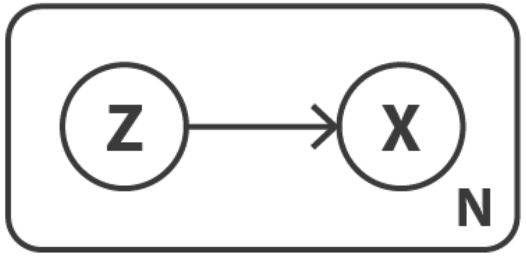
Create smooth gradients over the information encoded in the latent space

# But at the Expense of Reconstruction Quality



(source: Wojciech Mormul on Github)

# Variational Inference I: Probabilistic Synthesis




graphical model

$$p(z) \equiv \mathcal{N}(0, I)$$

$$p(x \mid z) \equiv \mathcal{N}(f(z), cI) \quad f \in F \quad c > 0$$

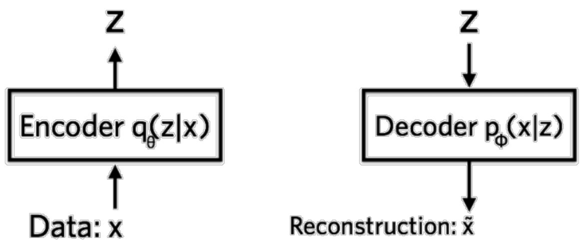
assume diagonal covariance

$$p(z \mid x) = \frac{p(x \mid z)p(z)}{p(x)} = \frac{p(x \mid z)p(z)}{\int p(x \mid u)p(u)du}$$

by Bayes, but intractable

VI: Approximate a complex target distribution by a simpler parametric distribution (e.g., a Gaussian)

$$q_x(z) \equiv \mathcal{N}(g(x), h(x)) \quad g \in G \quad h \in H$$




$$\begin{aligned}
(g^*, h^*) &= \underset{(g,h) \in G \times H}{\arg\min} \; KL\left(q_x(z), p(z \mid x)\right) \\
&= \underset{(g,h) \in G \times H}{\arg\min} \left(\mathbb{E}_{z \sim q_x}\left(\log q_x(z)\right) - \mathbb{E}_{z \sim q_x}\left(\log \frac{p(x \mid z)p(z)}{p(x)}\right)\right) \\
&= \underset{(g,h) \in G \times H}{\arg\min} \left(\mathbb{E}_{z \sim q_x}\left(\log q_x(z)\right) - \mathbb{E}_{z \sim q_x}\left(\log p(z)\right) - \mathbb{E}_{z \sim q_x}\left(\log p(x \mid z)\right) + \mathbb{E}_{z \sim q_x}\left(\log p(x)\right)\right) \\
&= \underset{(g,h) \in G \times H}{\arg\max} \left(\mathbb{E}_{z \sim q_x}\left(\log p(x \mid z)\right) - KL\left(q_x(z), p(z)\right)\right) \\
&= \underset{(g,h) \in G \times H}{\arg\max} \left(\mathbb{E}_{z \sim q_x}\left(-\frac{\|x - f(z)\|^2}{2c}\right) - KL\left(q_x(z), p(z)\right)\right)
\end{aligned}$$

# Variational Inference II

$$
\begin{aligned}
f^* &= \arg\max_{f \in F} \mathbb{E}_{z \sim q_x^*}(\log p(x \mid z)) \\
&= \arg\max_{f \in F} \mathbb{E}_{z \sim q_x^*}\left(-\frac{\|x - f(z)\|^2}{2c}\right)
\end{aligned}
$$

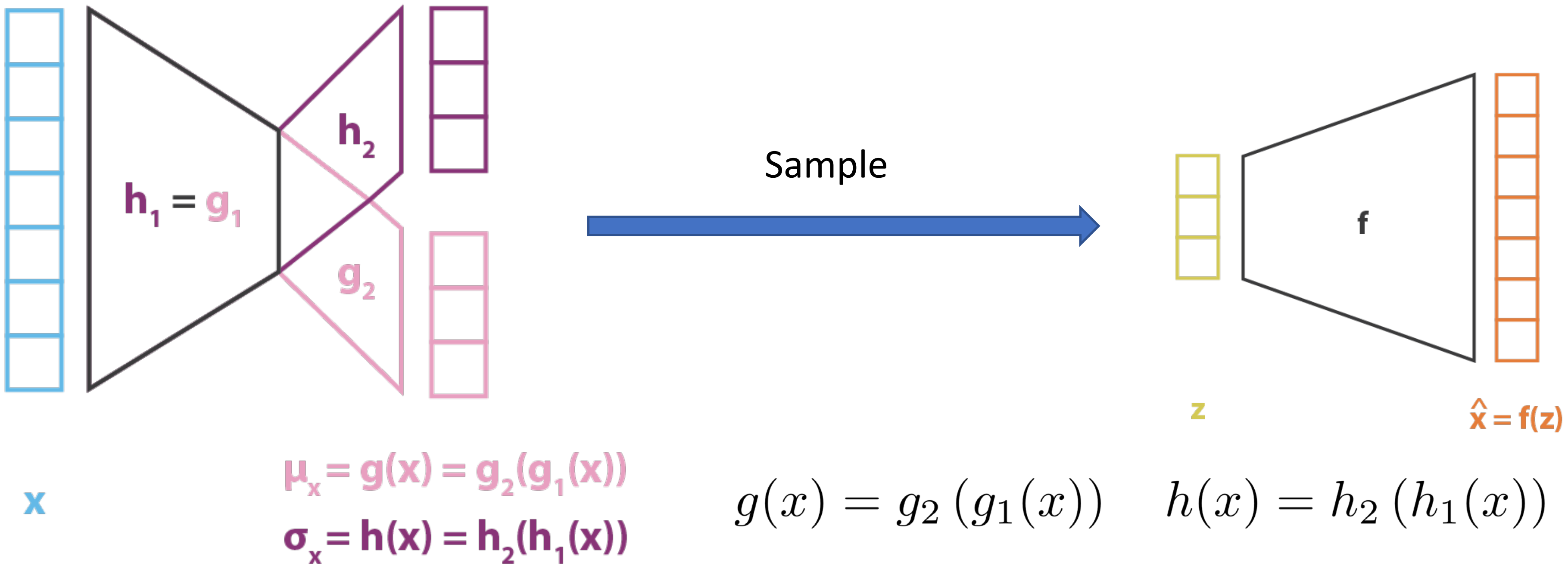
$$
(f^*, g^*, h^*) = \underset{(f,g,h) \in F \times G \times H}{\arg\max} \left(\mathbb{E}_{z \sim q_x}\left(-\frac{\|x - f(z)\|^2}{2c}\right) - KL\left(q_x(z), p(z)\right)\right)
$$

# ELBO: Evidence Lower BOund

$$
\begin{aligned}
&\log p(x) \\
=&\log \int_z p(x,z) \\
=&\log \int_z p(x,z)\frac{q_x(z)}{q_x(z)} \\
\geq&\mathbb{E}_{z\sim q_x}\left[\log \frac{p(x,z)}{q_x(z)}\right] \\
=&\mathbb{E}_{z\sim q_x}\left[\log \frac{p(x\mid z)p(z)}{q_x(z)}\right] \\
=&\mathbb{E}_{z\sim q_x}[\log p(x\mid z)] + \mathbb{E}_{z\sim q_x}\left[\log \frac{p(z)}{q_x(z)}\right] \\
=&\mathbb{E}_{z\sim q_x}[\log p(x\mid z)] + \int_z q_x(z)\log \frac{p(z)}{q_x(z)} \\
=&\mathbb{E}_{z\sim q_x}[\log p(x\mid z)] - D_{KL}[q_x(z)\|p(z)] \\
=&\,\text{likelihood} - KL
\end{aligned}
$$

By Jensen's inequality on a concave function (log)

# Variational Inference with NNs




$$g(x) = g_2\left(g_1(x)\right) \quad h(x) = h_2\left(h_1(x)\right) \quad g_1(x) = h_1(x)$$

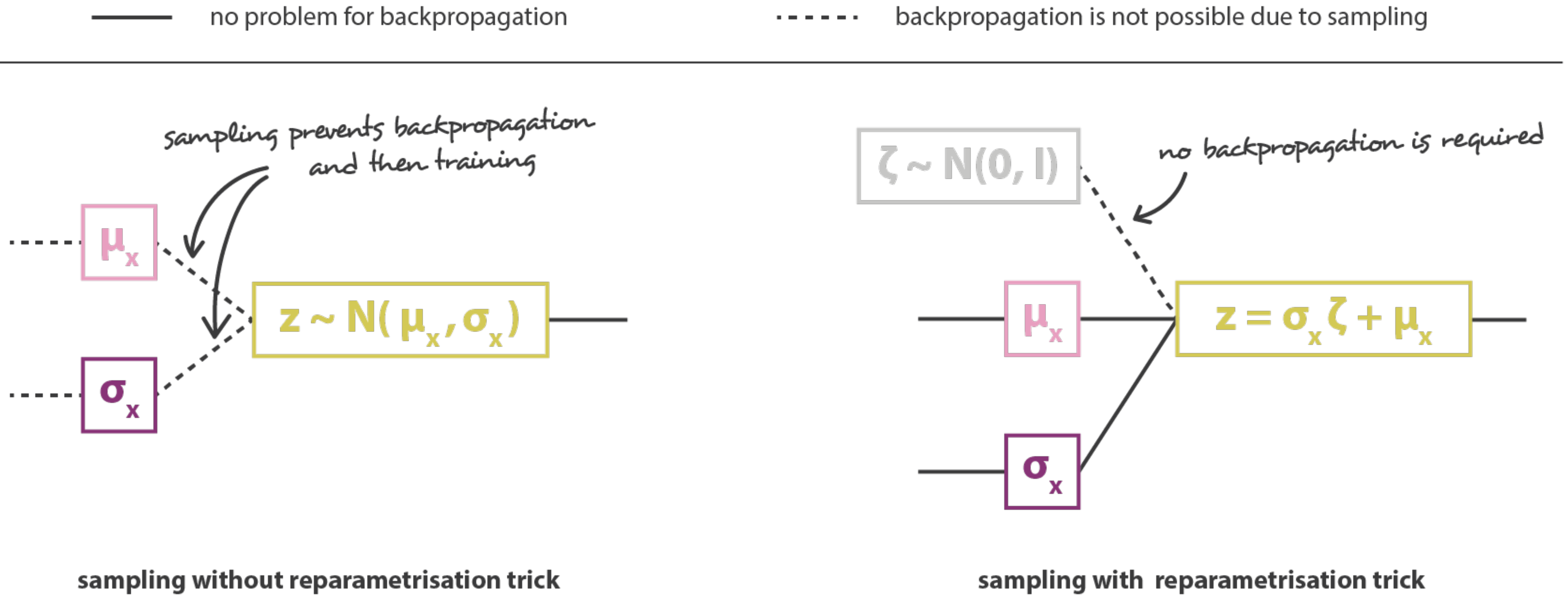
Sampling is a problem w. back propagation

# The Reparametrization Trick

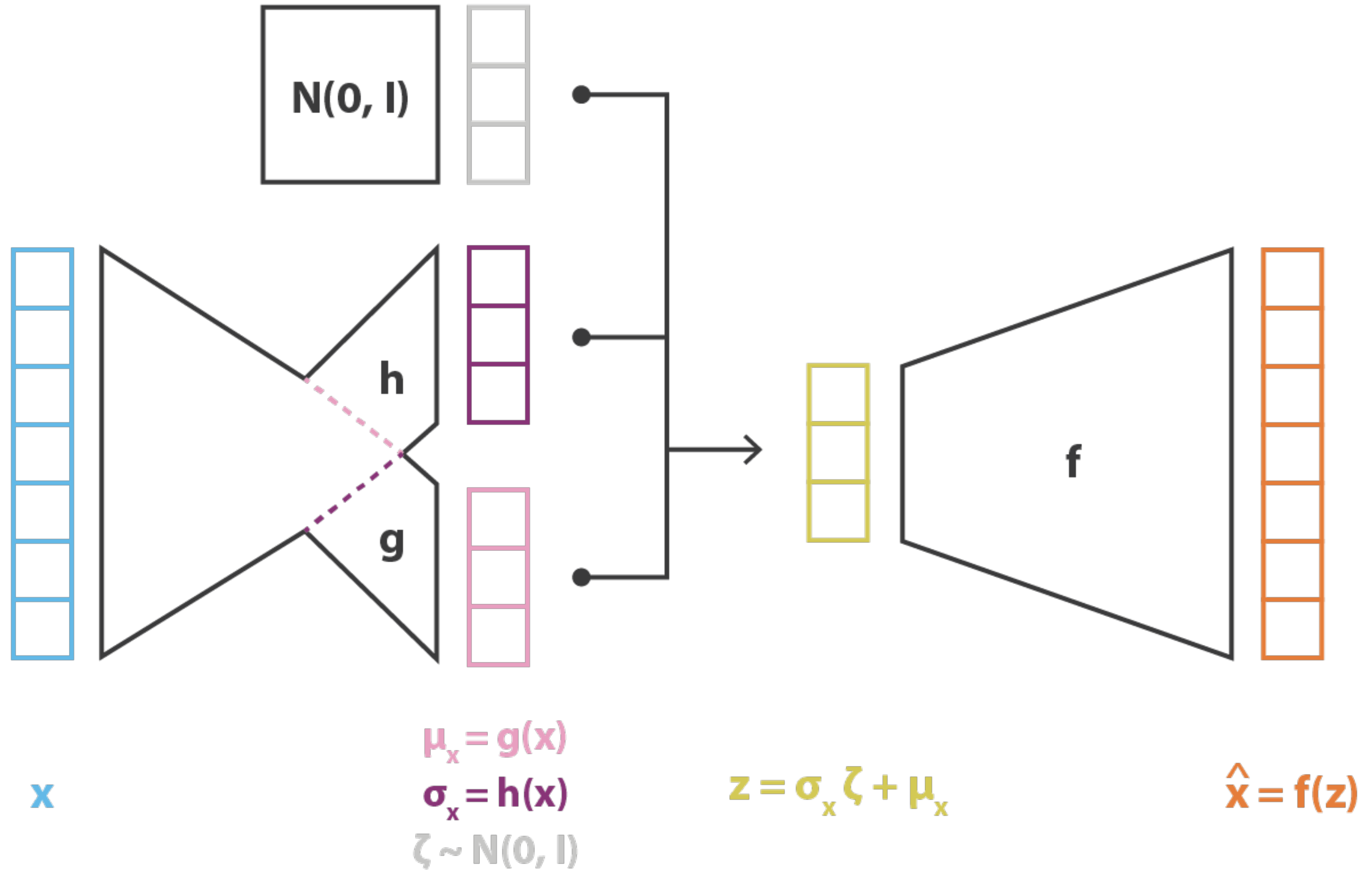
$z \sim \mathcal{N}(g(x), h(x))$

$$z = h(x)\zeta + g(x) \quad \zeta \sim \mathcal{N}(0, I)$$

for Gaussians with diagonal covariance

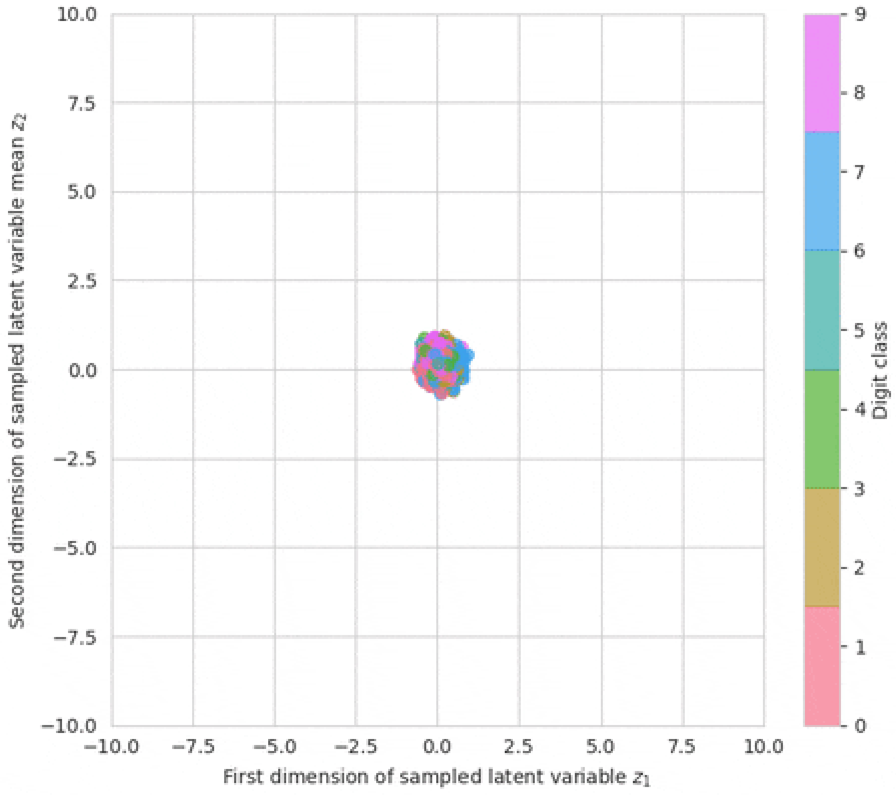


# The Final VAE

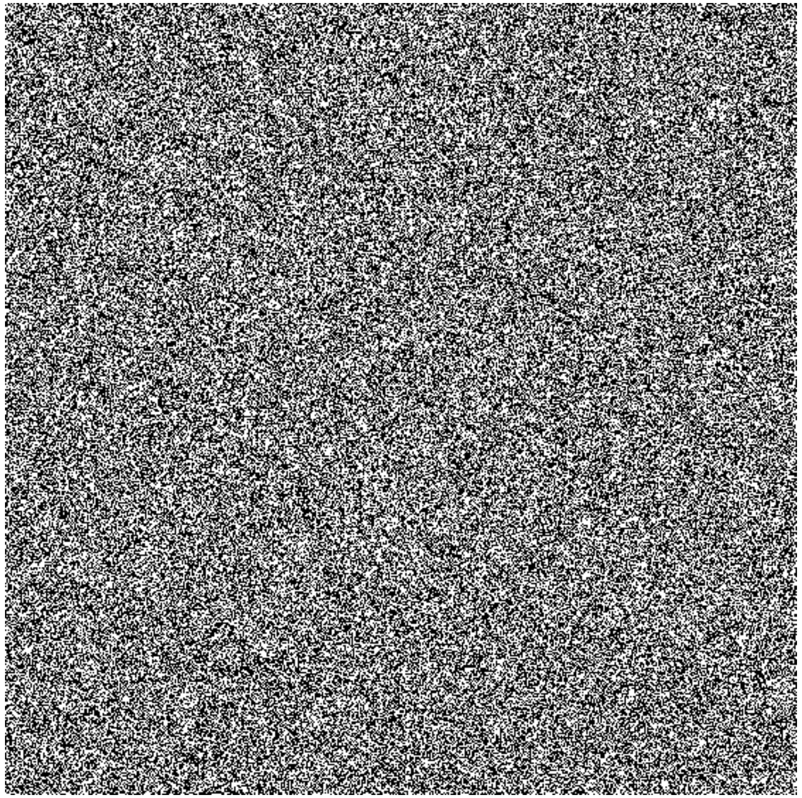



$$\text{loss} = C\,\|x - \hat{x}\|^2 + KL[\,N(\mu_x, \sigma_x), N(0, I)\,] = C\,\|x - f(z)\|^2 + KL[\,N(g(x), h(x)), N(0, I)\,]$$

# MNIST Example




Class differentiation



Sampling the likelihood

# Generative Models: Deep Neural Implicits

JJ (Jeong Joon) Park

# Example
# Student Presentation

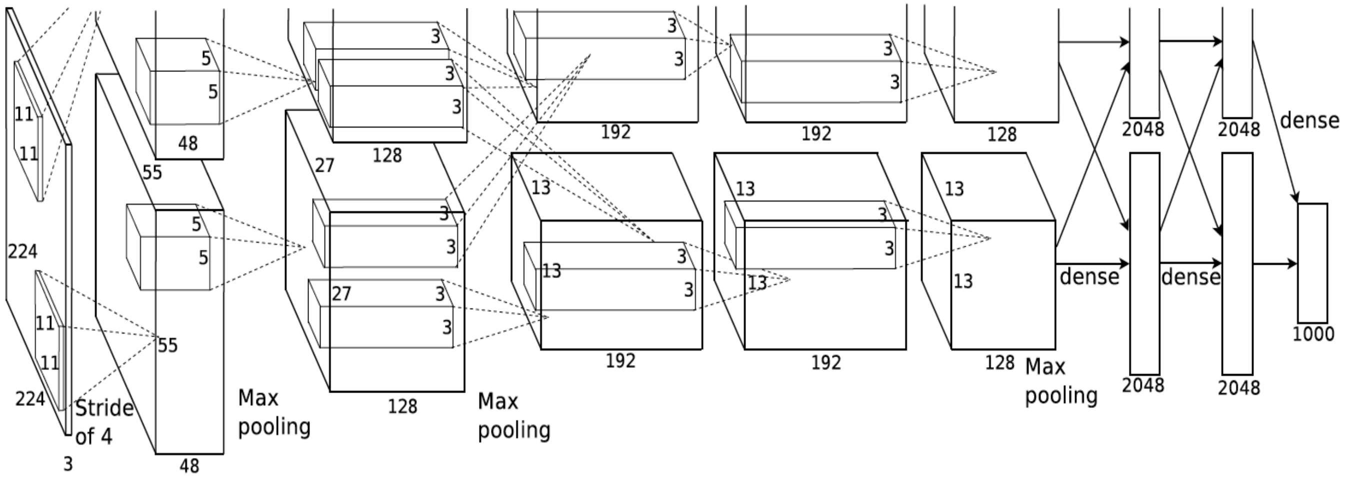
# DeepSDF: Learning Continuous SDFs for Shape Representation

**Jeong Joon Park**[1], Peter Florence[2], Julian Straub[3], Richard Newcombe[3], Steven Lovegrove[3]

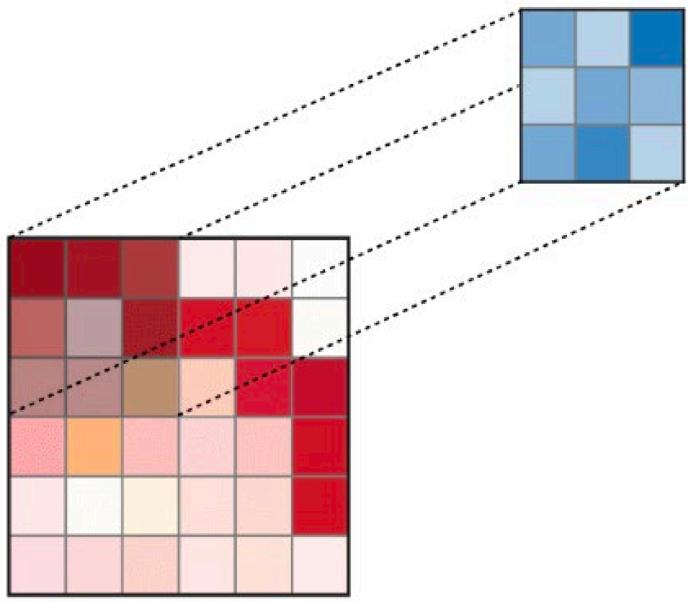
[1] University of Washington, [2] MIT, [3] Facebook Reality Labs

CVPR 2019

# Representation for 2D Deep Learning



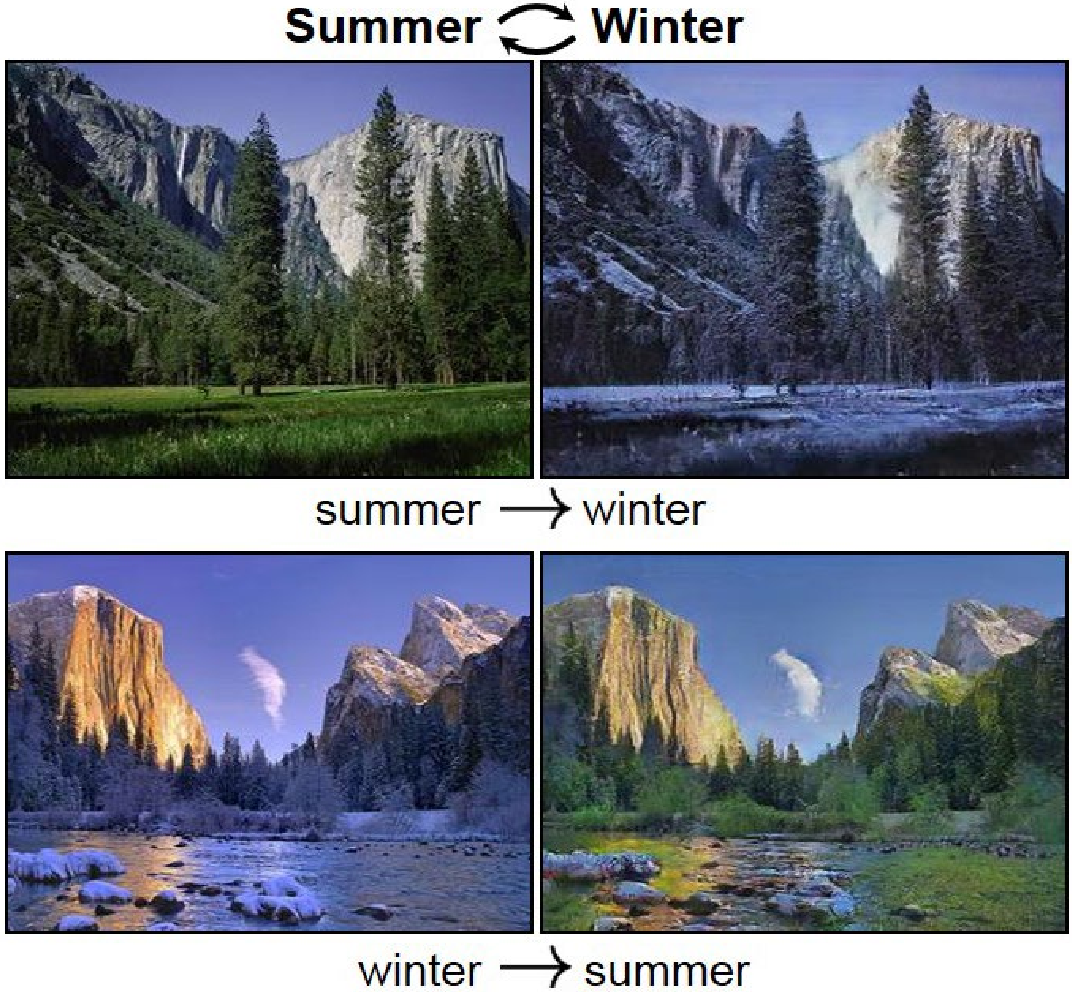

ImageNet. 2012



Convolution Layer

# Representation for 2D Deep Learning

Liu et al, 2018



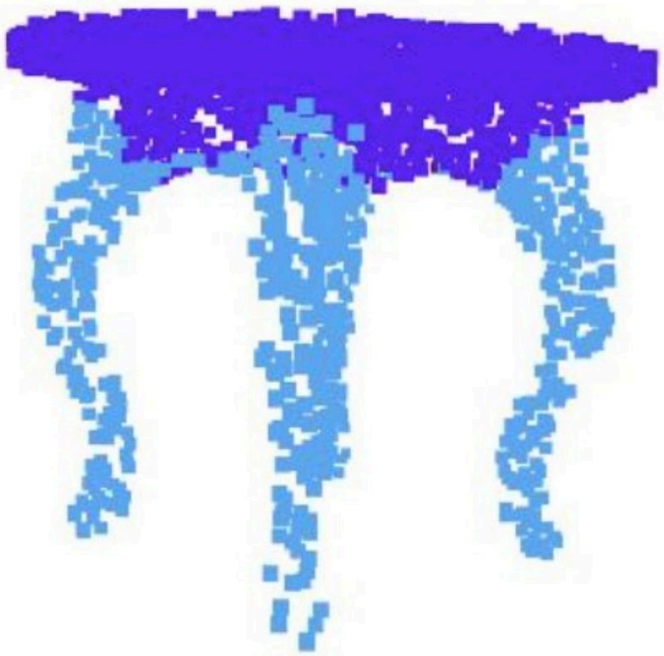

CycleGAN, 2017

# Representations for 3D Deep Learning

Voxel

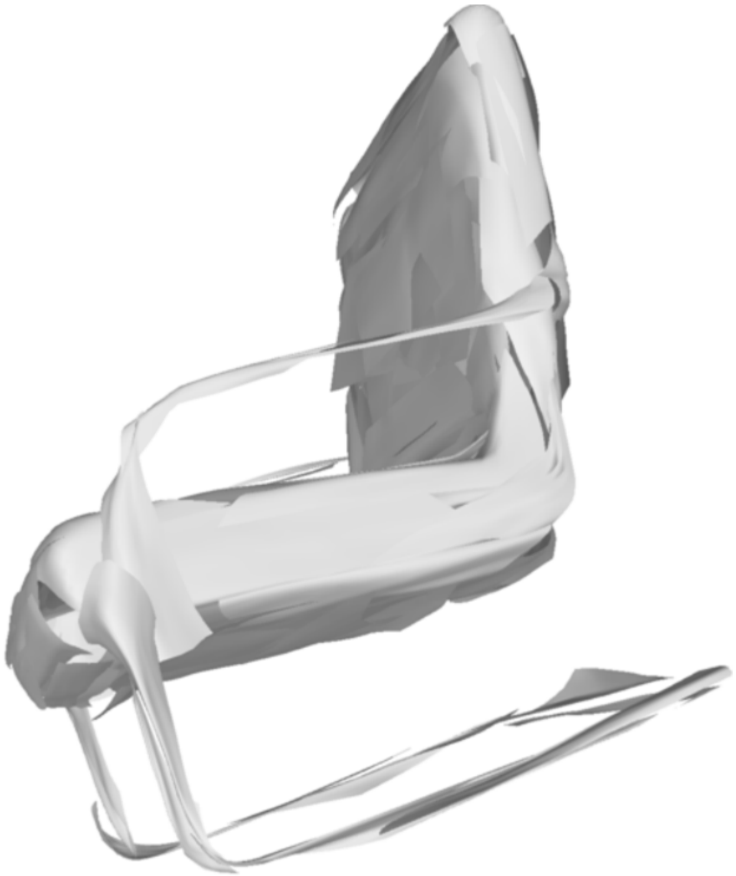
Wu et al. 2016

Points



Qi et al. 2017

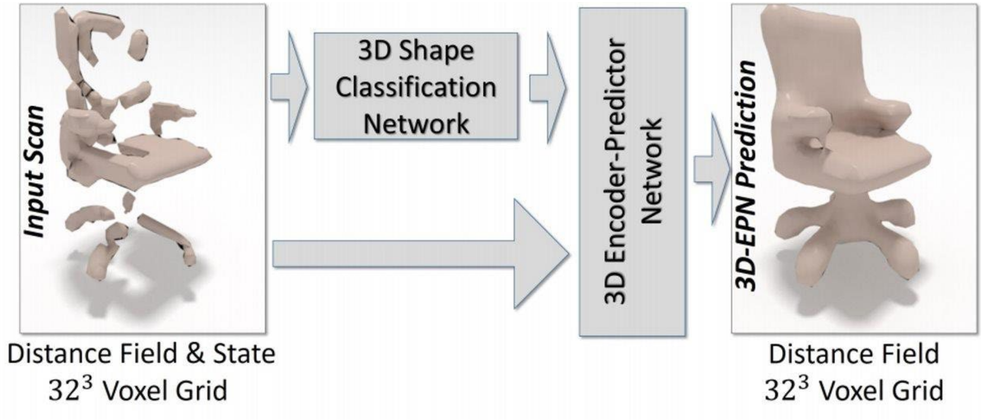
Meshes

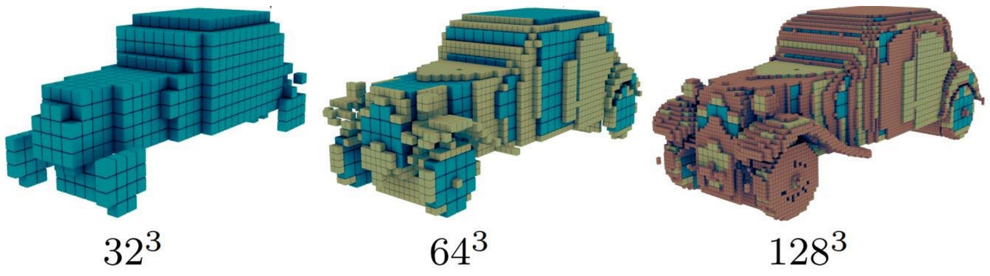


Groueix et al. 2018

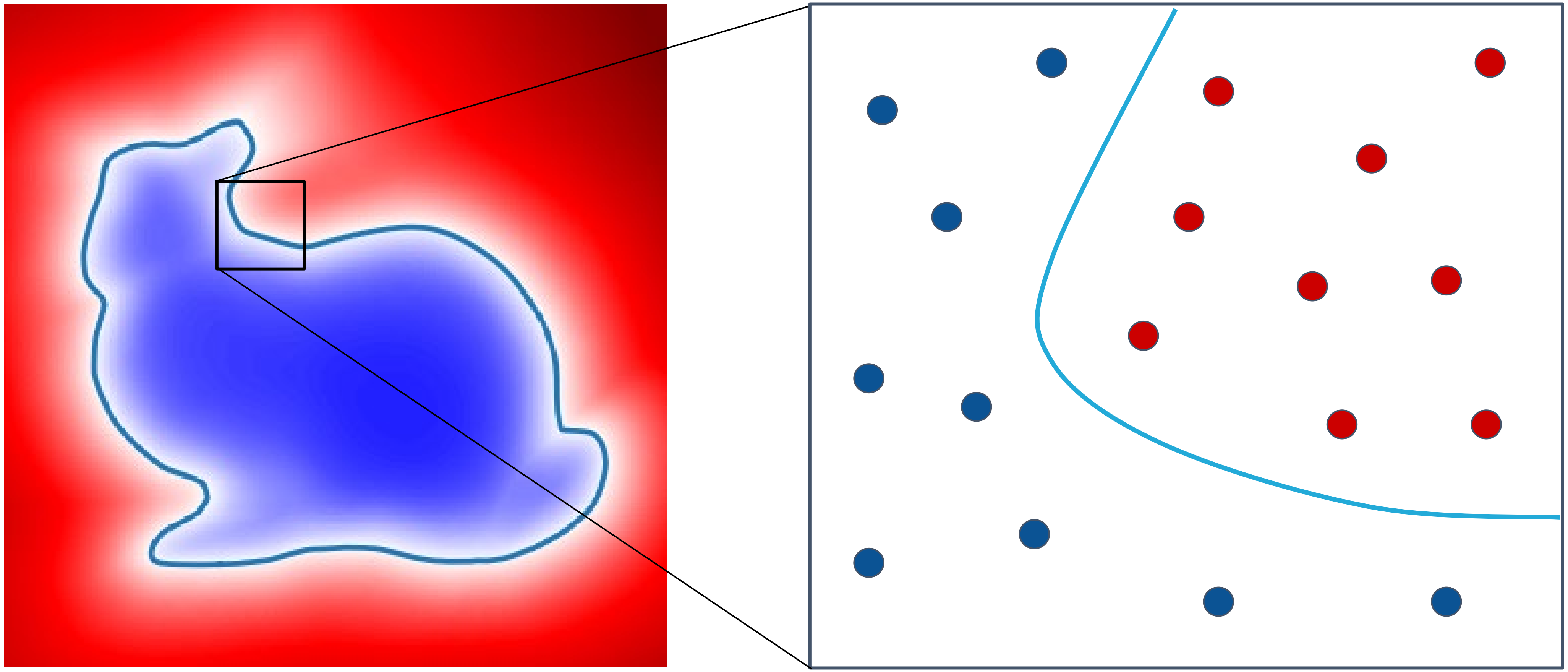
# Voxel Representation

- Memory Intensive, Computationally Expensive (N^3)

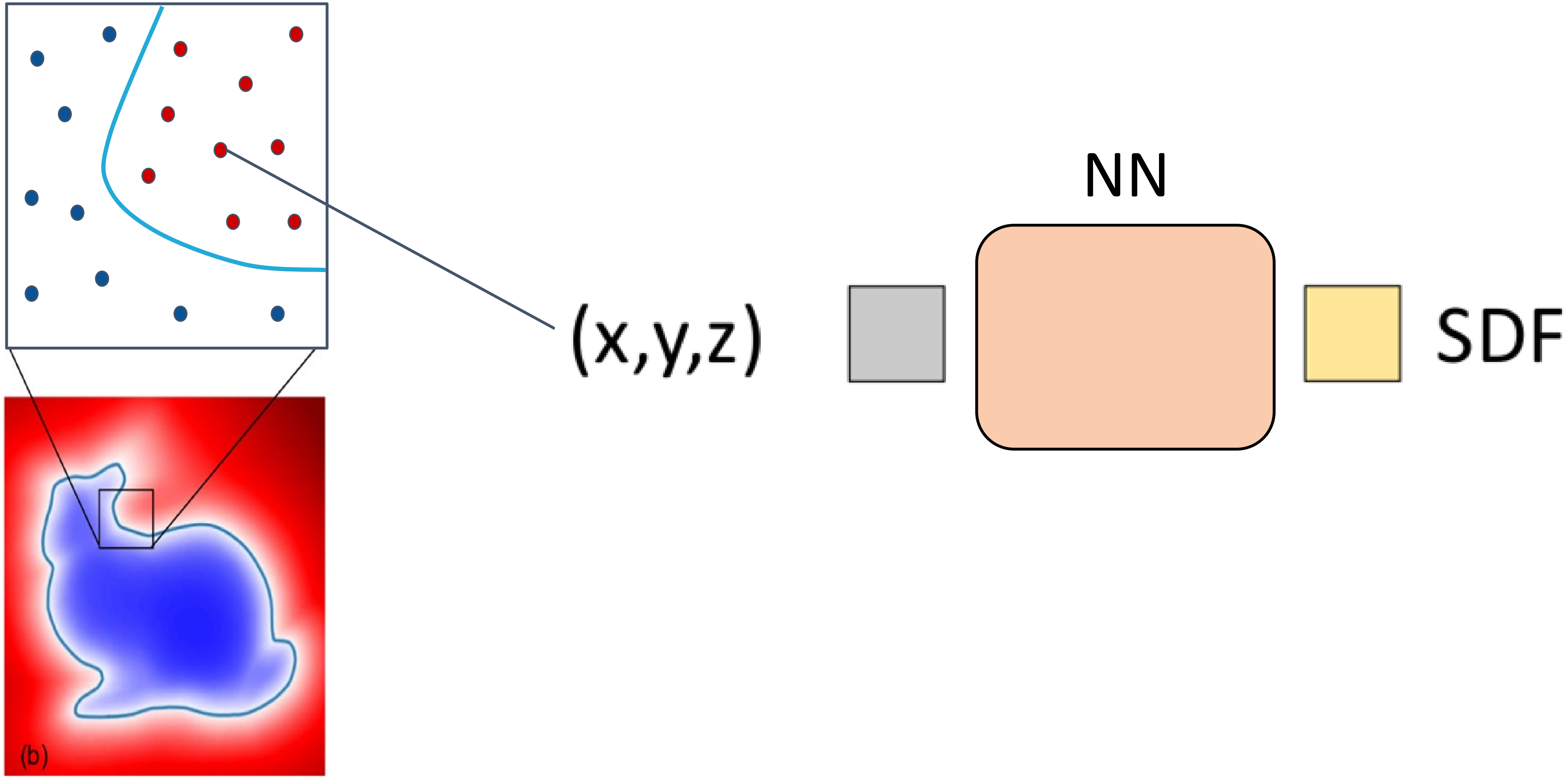


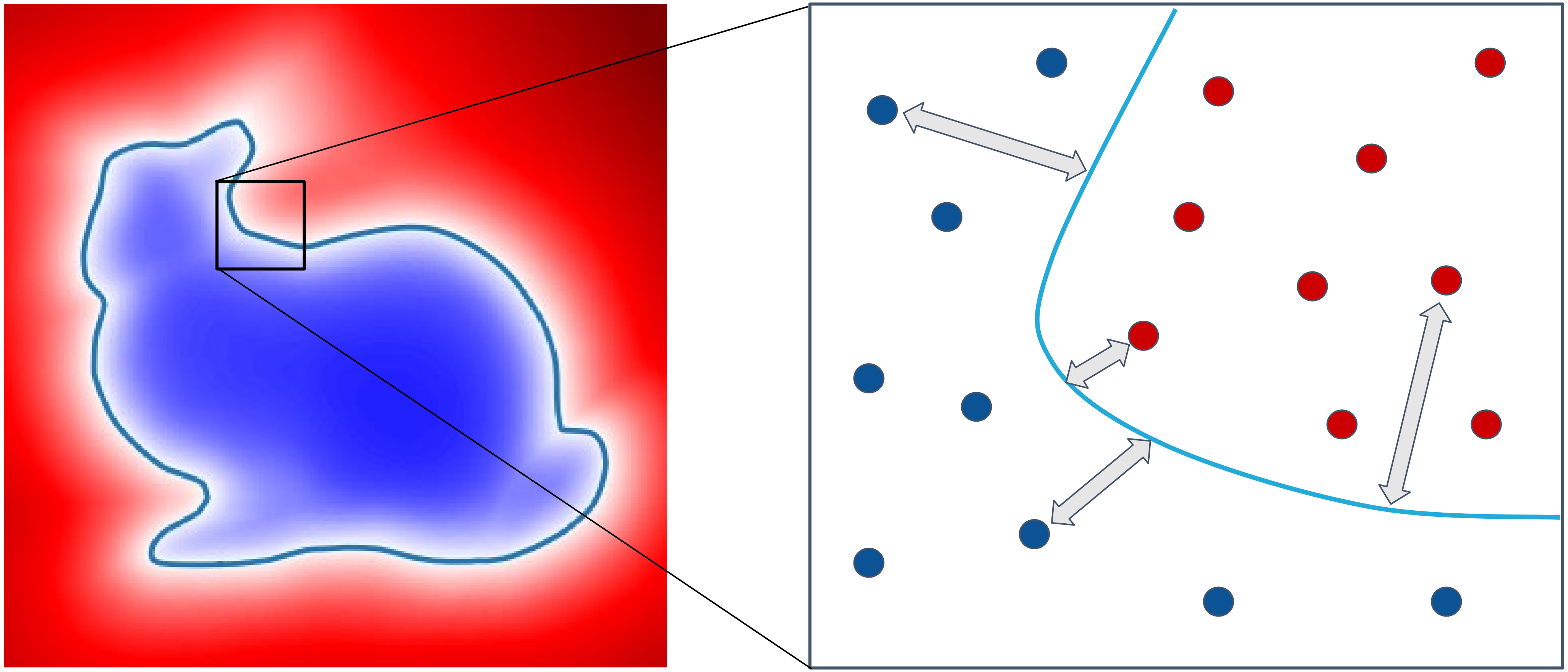
# Surface as Decision Boundary



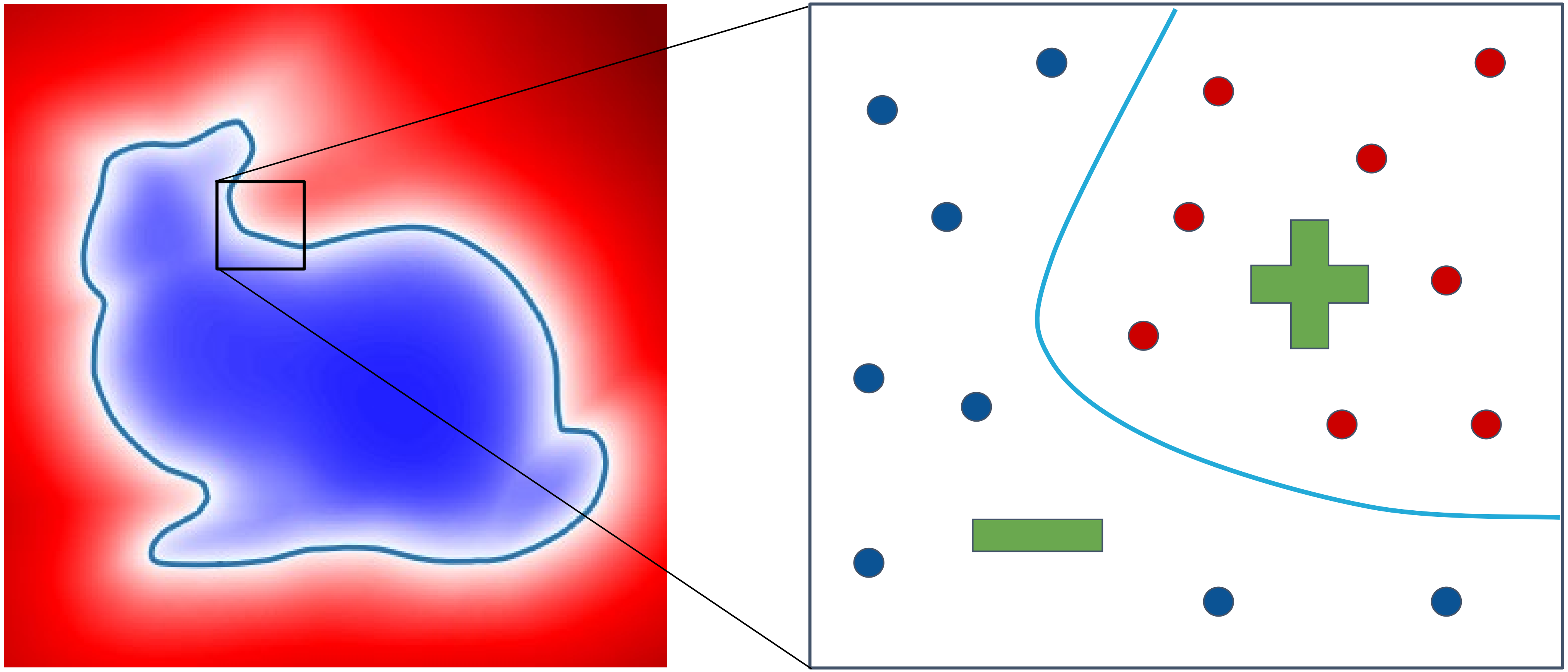
# Regression of Continuous SDF



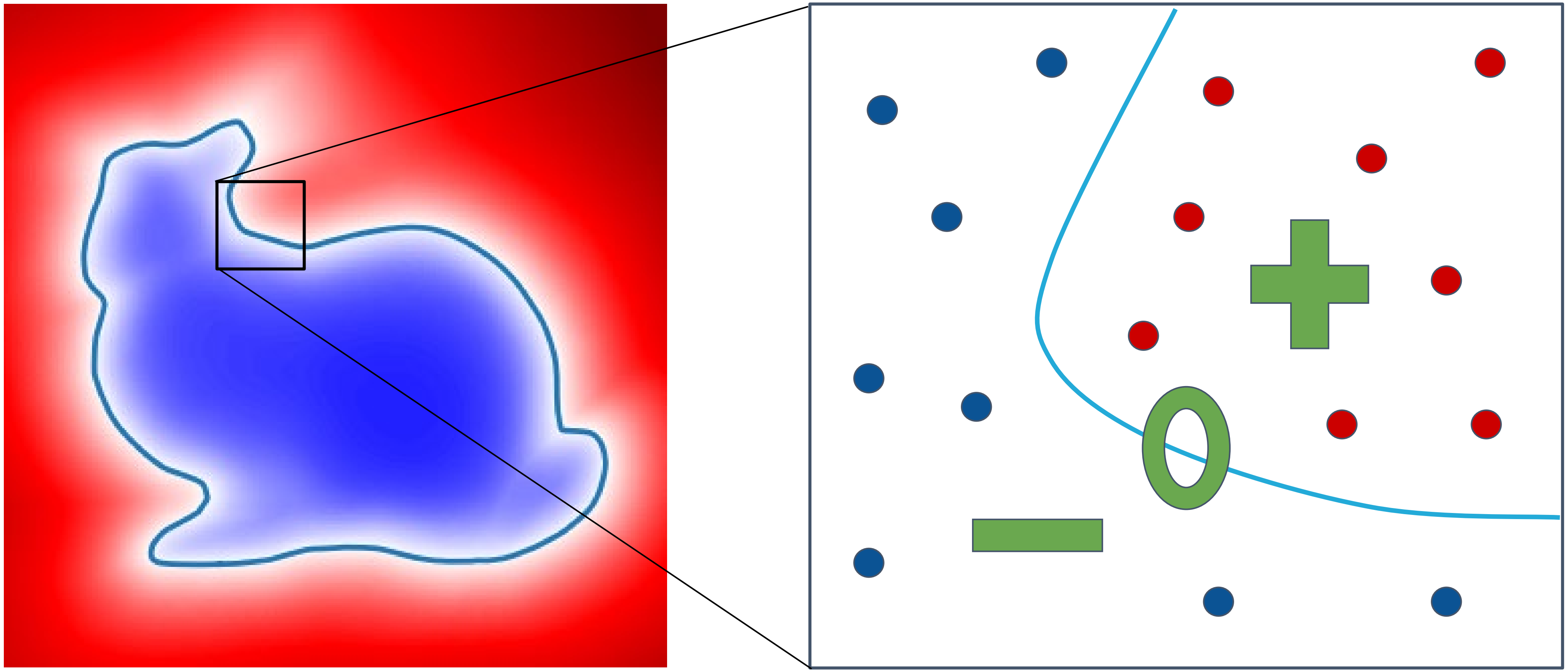
# Signed Distance Function



# Signed Distance Function



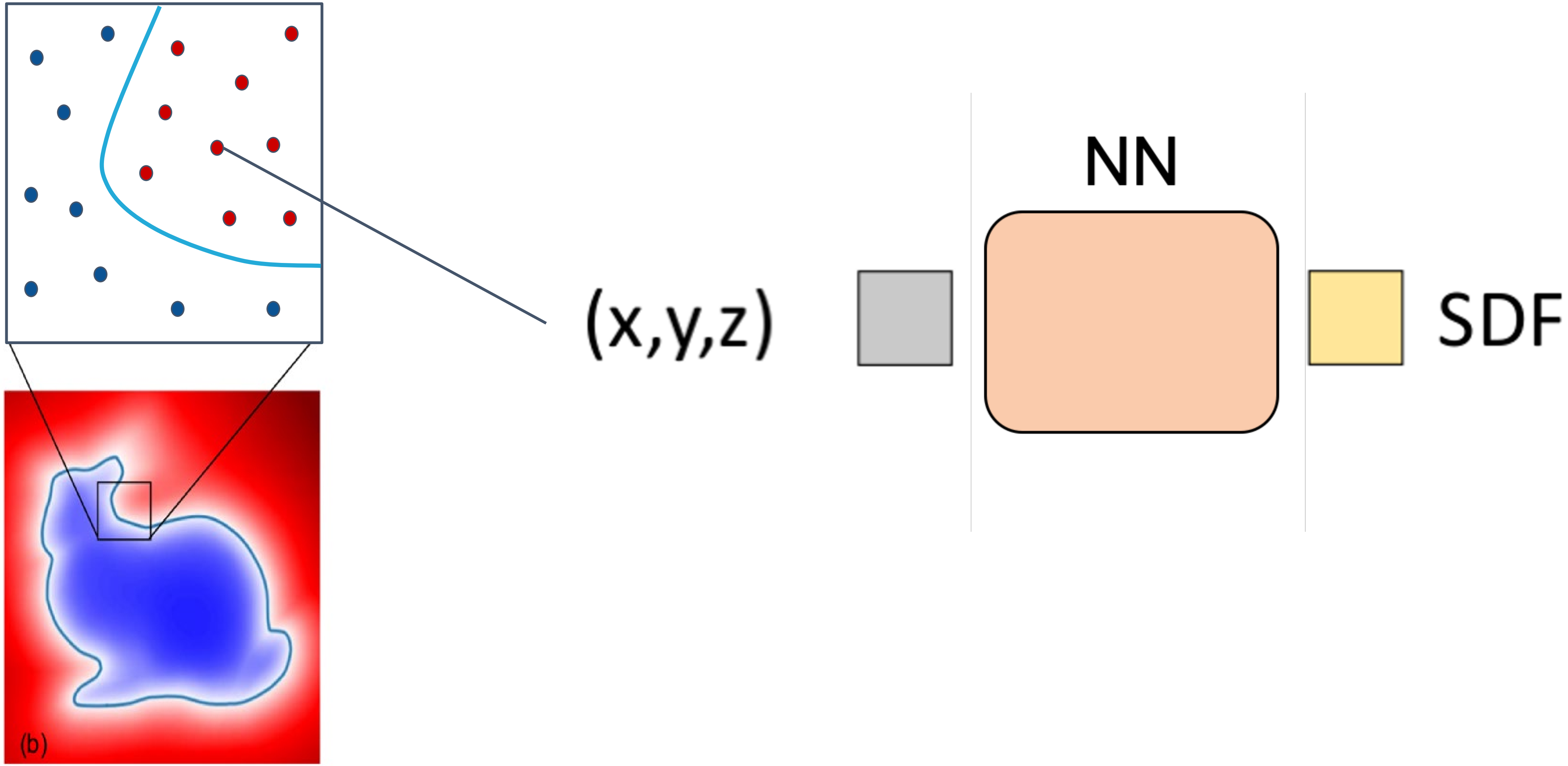
# Signed Distance Function



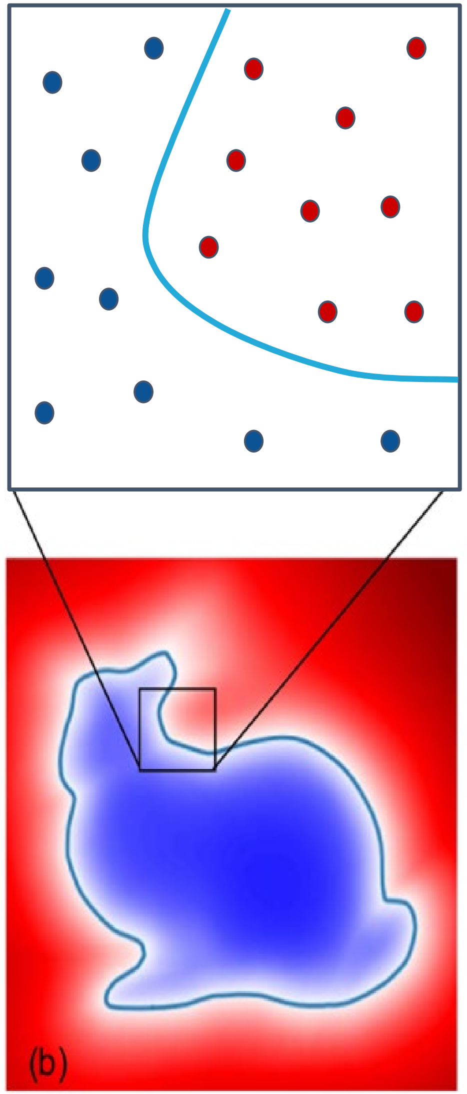
# Discrete SDF

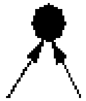
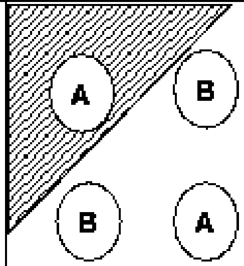
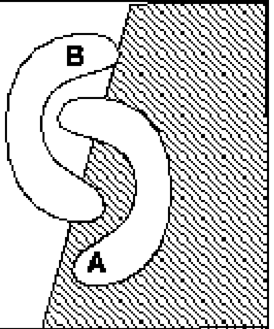
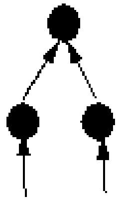
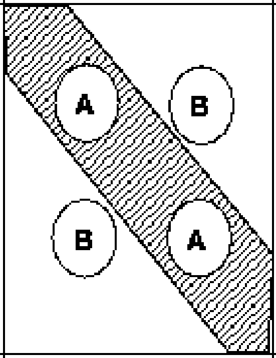
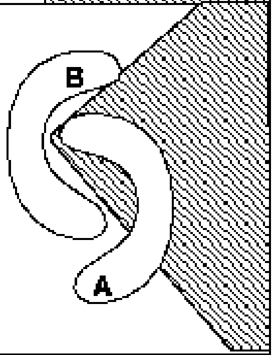
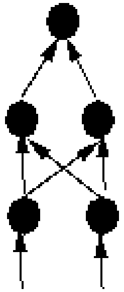
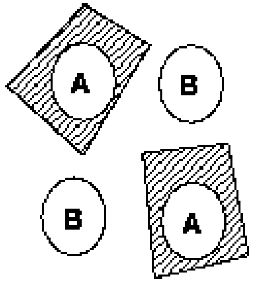
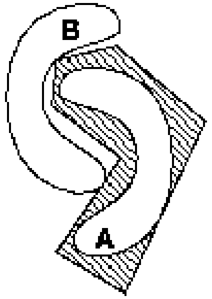
| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| -0.9 | -0.3 | 0.0 | 0.2 | 1 | 1 | 1 | 1 | 1 |
| -1 | -0.9 | -0.2 | 0.0 | 0.2 | 1 | 1 | 1 | 1 |
| -1 | -0.9 | -0.3 | 0.0 | 0.1 | 0.9 | 1 | 1 | 1 |
| -1 | -0.8 | -0.3 | 0.0 | 0.2 | 0.8 | 1 | 1 | 1 |
| -1 | -0.9 | -0.4 | -0.1 | 0.1 | 0.8 | 0.9 | 1 | 1 |
| -1 | -0.7 | -0.3 | 0.0 | 0.3 | 0.6 | 1 | 1 | 1 |
| -1 | -0.7 | -0.4 | 0.0 | 0.2 | 0.7 | 0.8 | 1 | 1 |
| -0.9 | -0.7 | -0.2 | 0.0 | 0.2 | 0.8 | 0.9 | 1 | 1 |
| -0.1 | 0.0 | 0.0 | 0.1 | 0.3 | 1 | 1 | 1 | 1 |
| 0.5 | 0.3 | 0.2 | 0.4 | 0.8 | 1 | 1 | 1 | 1 |

# Continuous SDF

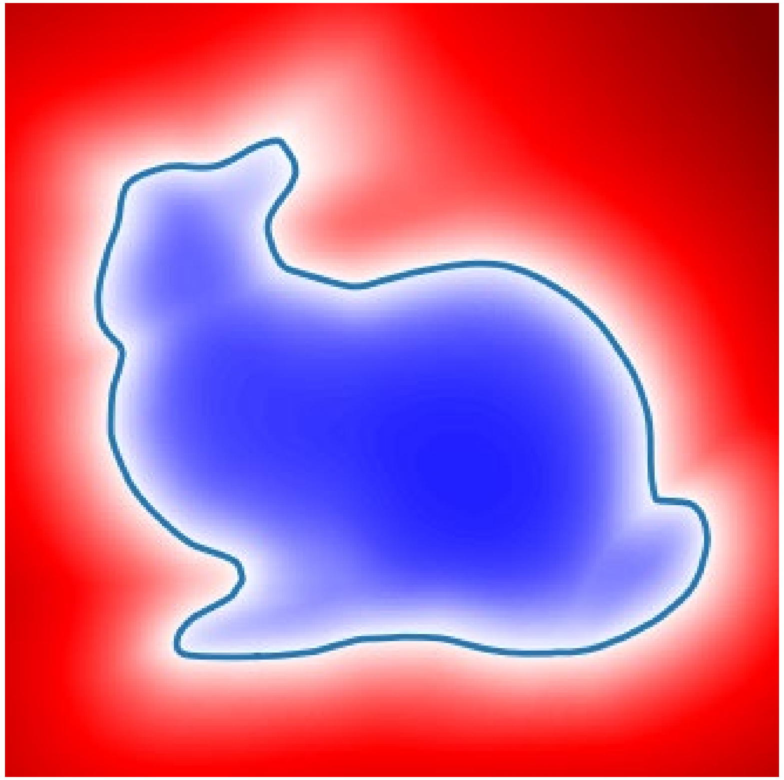


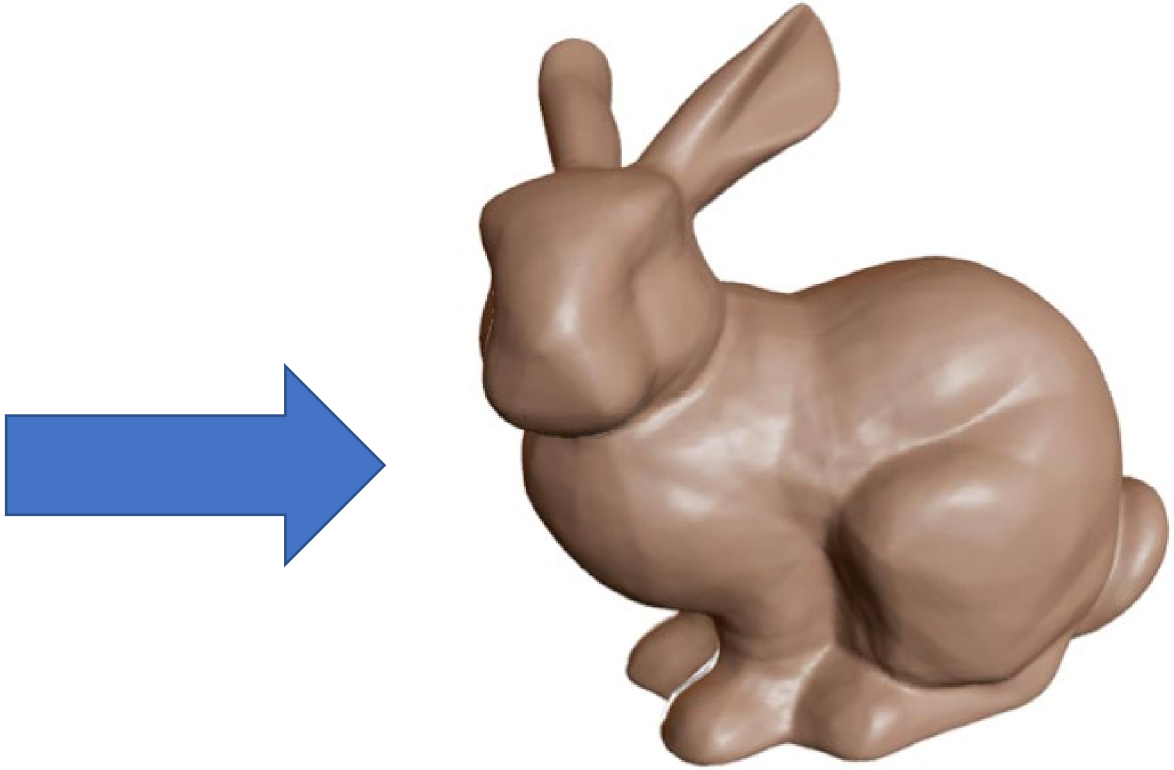
# Universal Approximation Theorem




| Structure | Regions | XOR | Meshed regions |
|---|---|---|---|
| single layer  | Half plane bounded by hyper-plane |  |  |
| two layer  | Convex open or closed regions |  |  |
| three layer  | Arbitrary (limited by # of nodes) |  |  |

# Implicit to Explicit

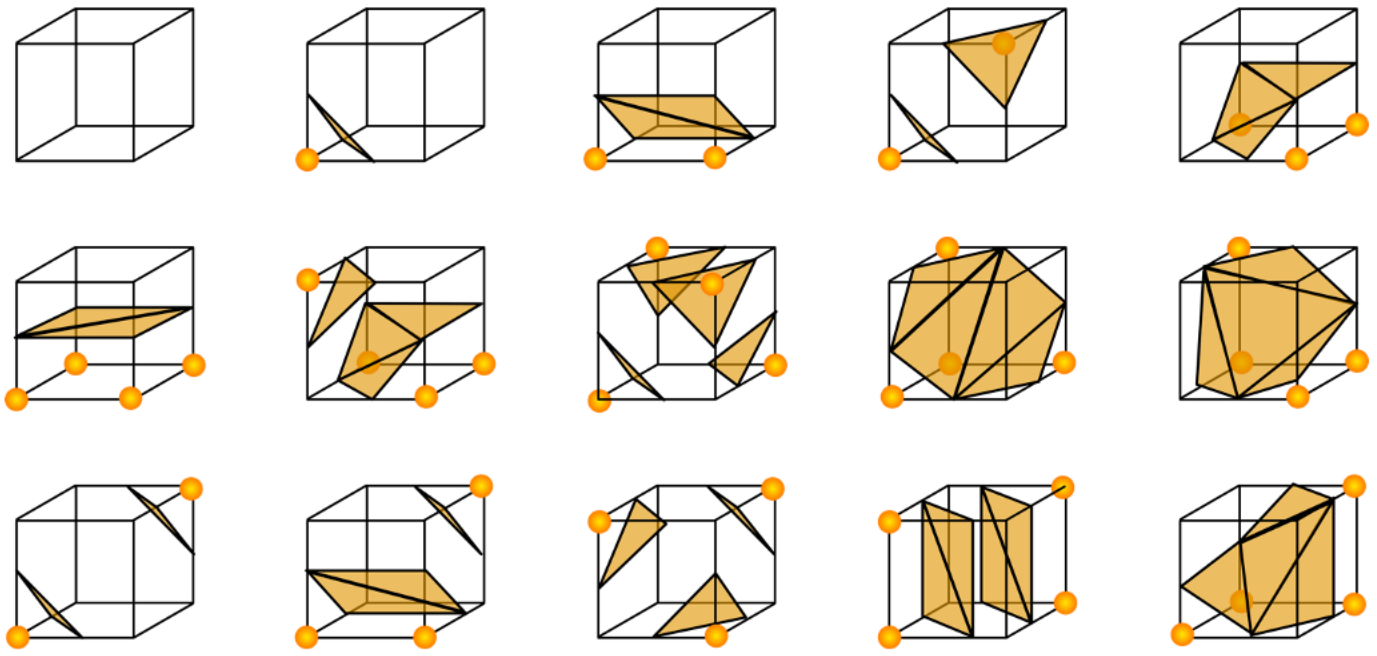




# Implicit to Explicit – Discrete sampling

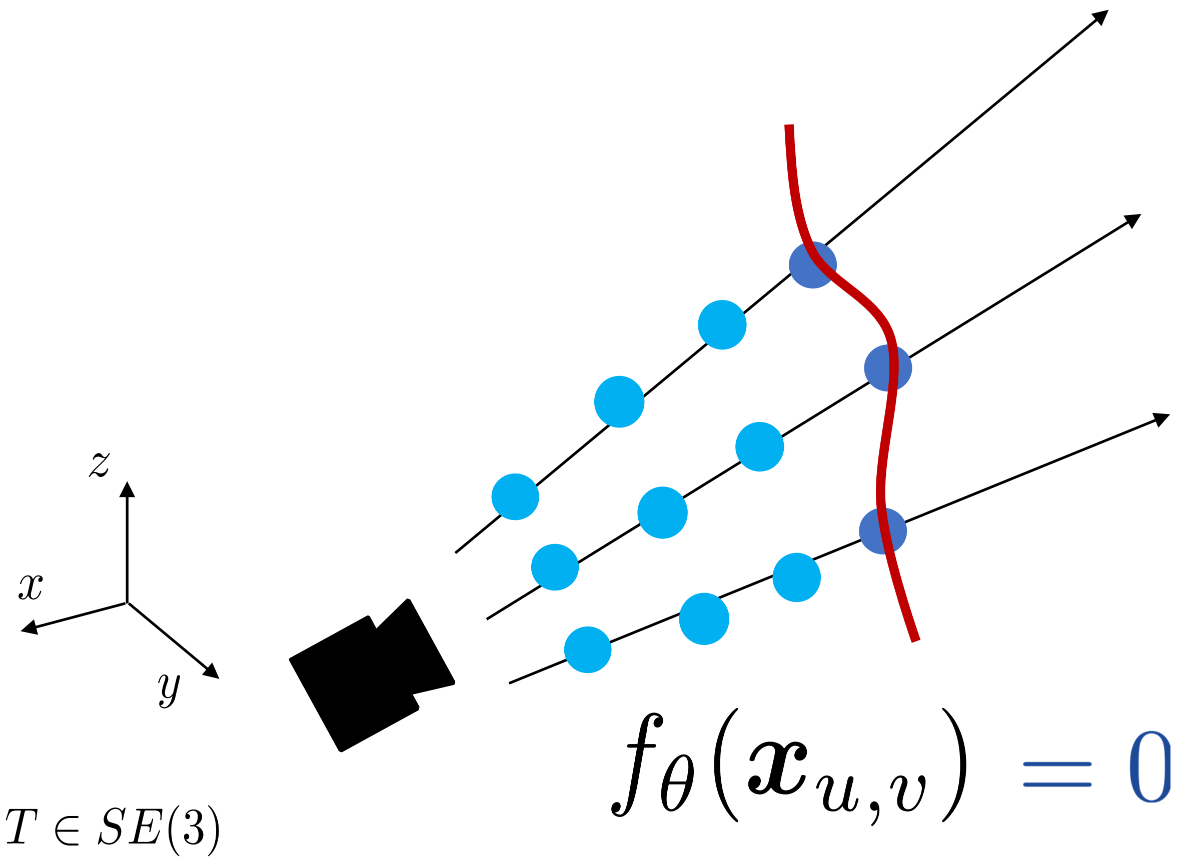
| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| -0.9 | -0.3 | 0.0 | 0.2 | 1 | 1 | 1 | 1 | 1 |
| -1 | -0.9 | -0.2 | 0.0 | 0.2 | 1 | 1 | 1 | 1 |
| -1 | -0.9 | -0.3 | 0.0 | 0.1 | 0.9 | 1 | 1 | 1 |
| -1 | -0.8 | -0.3 | 0.0 | 0.2 | 0.8 | 1 | 1 | 1 |
| -1 | -0.9 | -0.4 | -0.1 | 0.1 | 0.8 | 0.9 | 1 | 1 |
| -1 | -0.7 | -0.3 | 0.0 | 0.3 | 0.6 | 1 | 1 | 1 |
| -1 | -0.7 | -0.4 | 0.0 | 0.2 | 0.7 | 0.8 | 1 | 1 |
| -0.9 | -0.7 | -0.2 | 0.0 | 0.2 | 0.8 | 0.9 | 1 | 1 |
| -0.1 | 0.0 | 0.0 | 0.1 | 0.3 | 1 | 1 | 1 | 1 |
| 0.5 | 0.3 | 0.2 | 0.4 | 0.8 | 1 | 1 | 1 | 1 |

# Marching Cubes

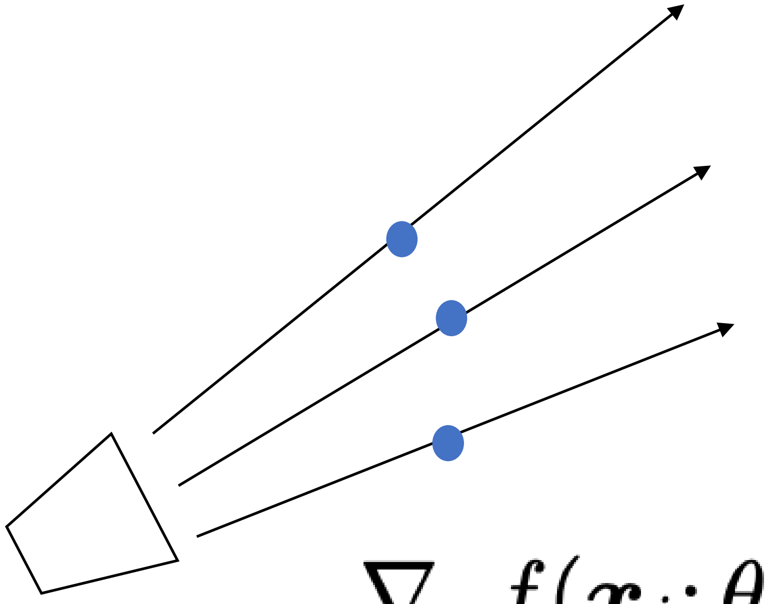


Lorensen et al., 1987

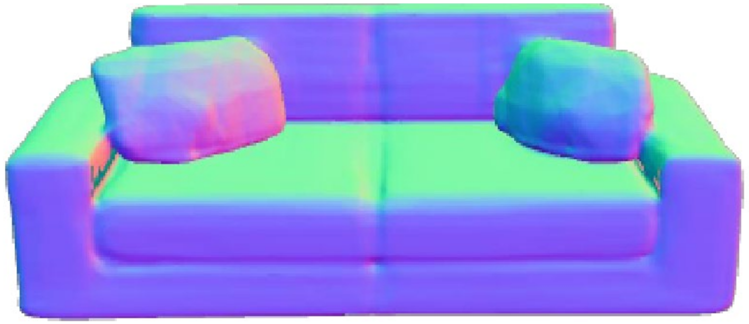
# 2. Raycasting

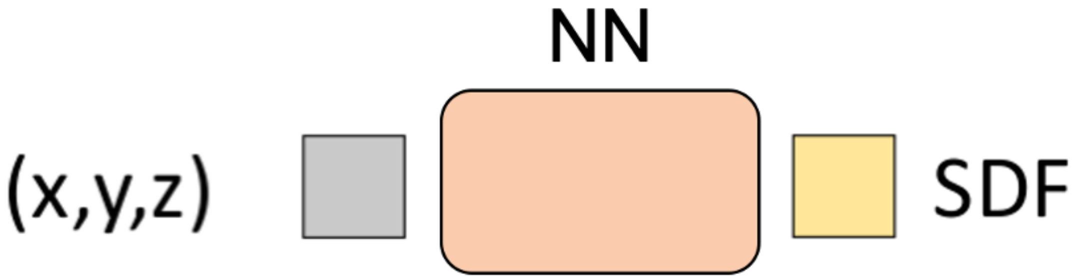


# 2. Raycasting



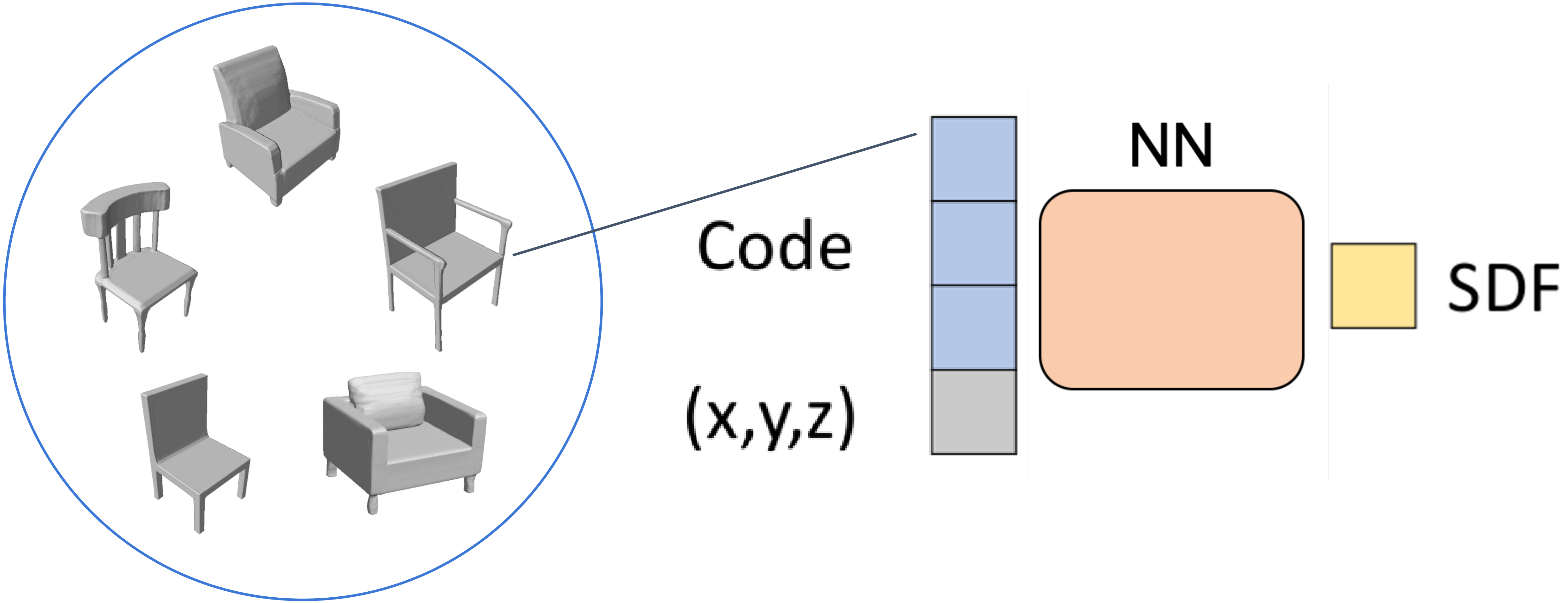
$$\nabla_{\boldsymbol{x}} f(\boldsymbol{x}_i; \theta)$$



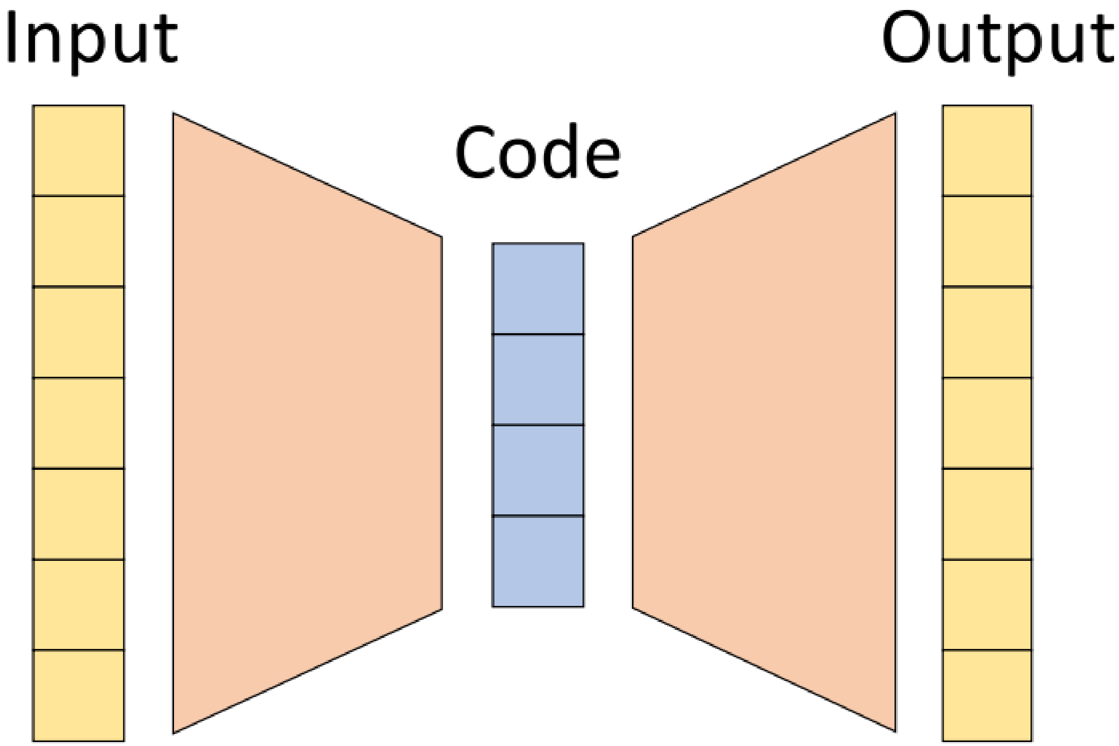



$x$ $f_\theta(x)$

# Coding Multiple Shapes

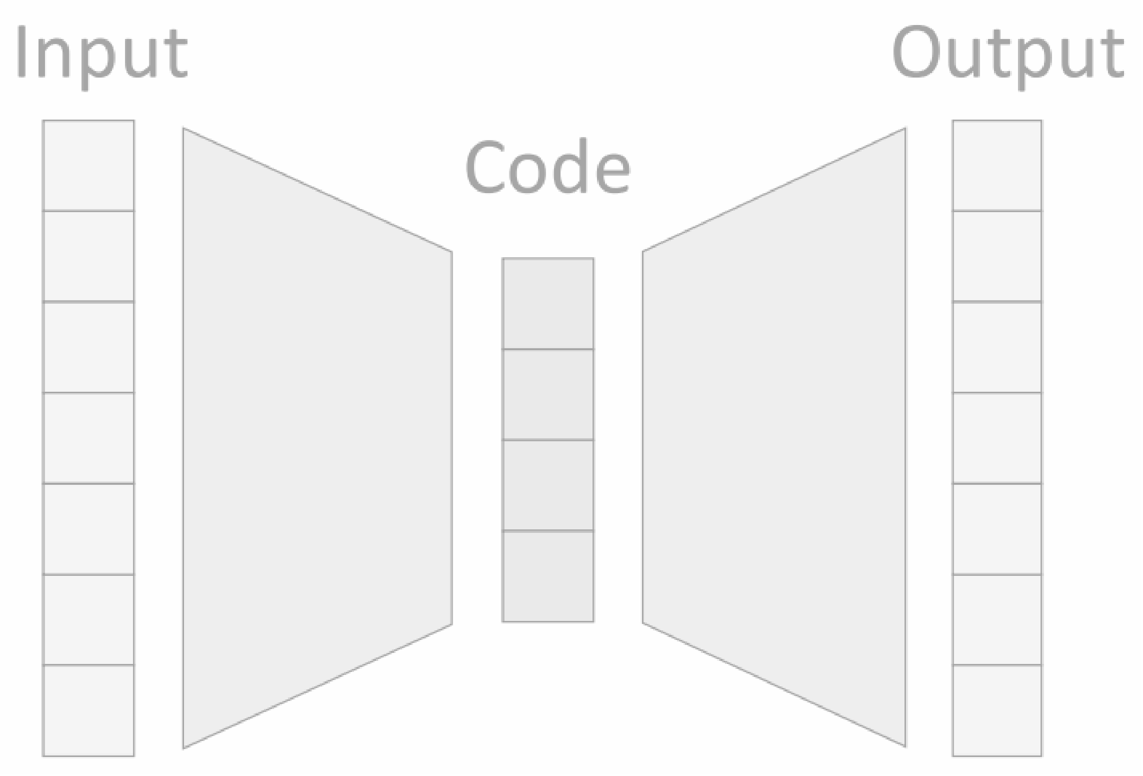


# Auto-Encoder

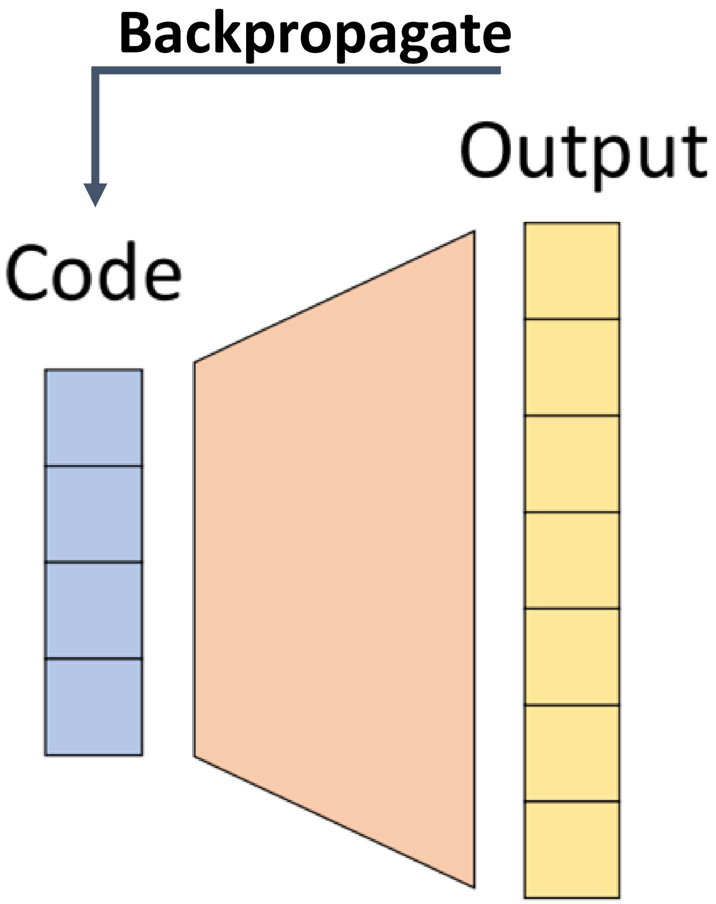



**Auto-Encoder**

# Auto-Decoder

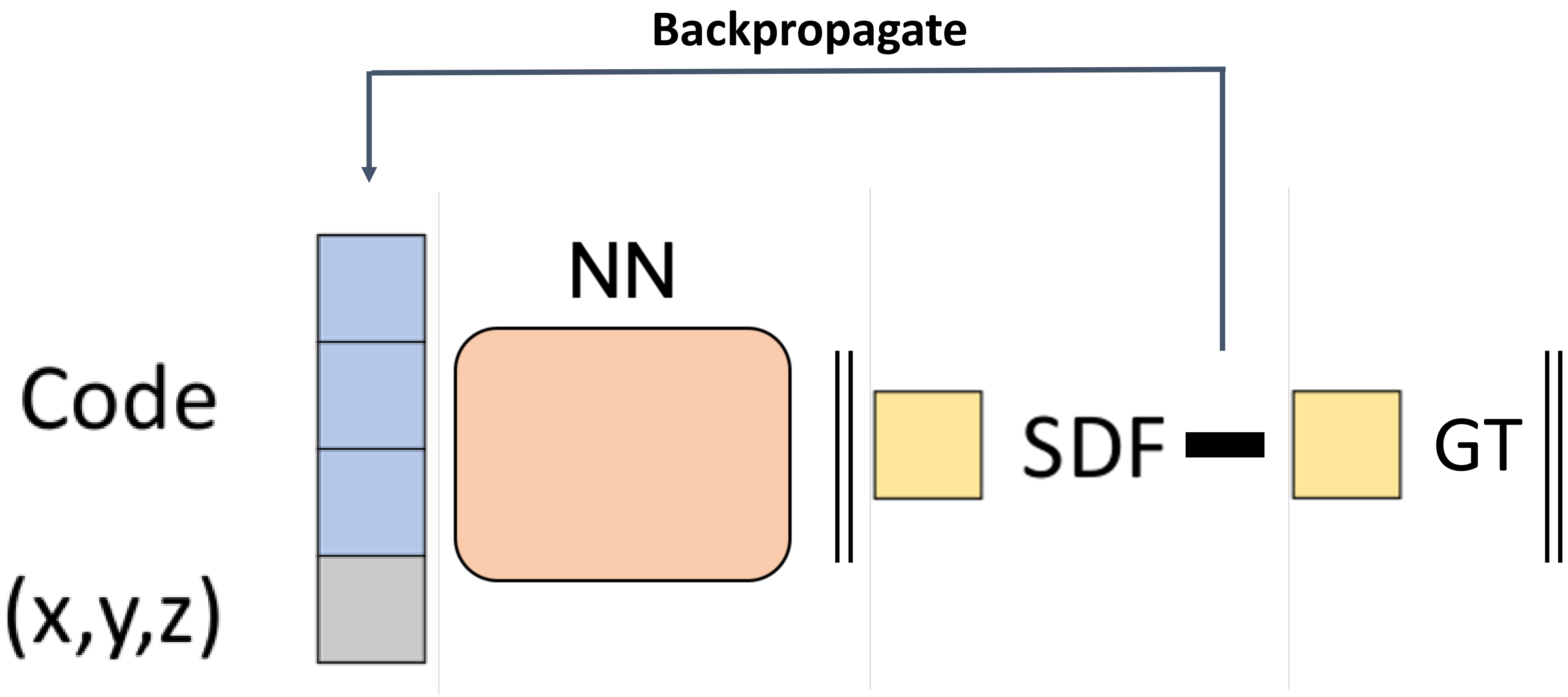



**Auto-Encoder**

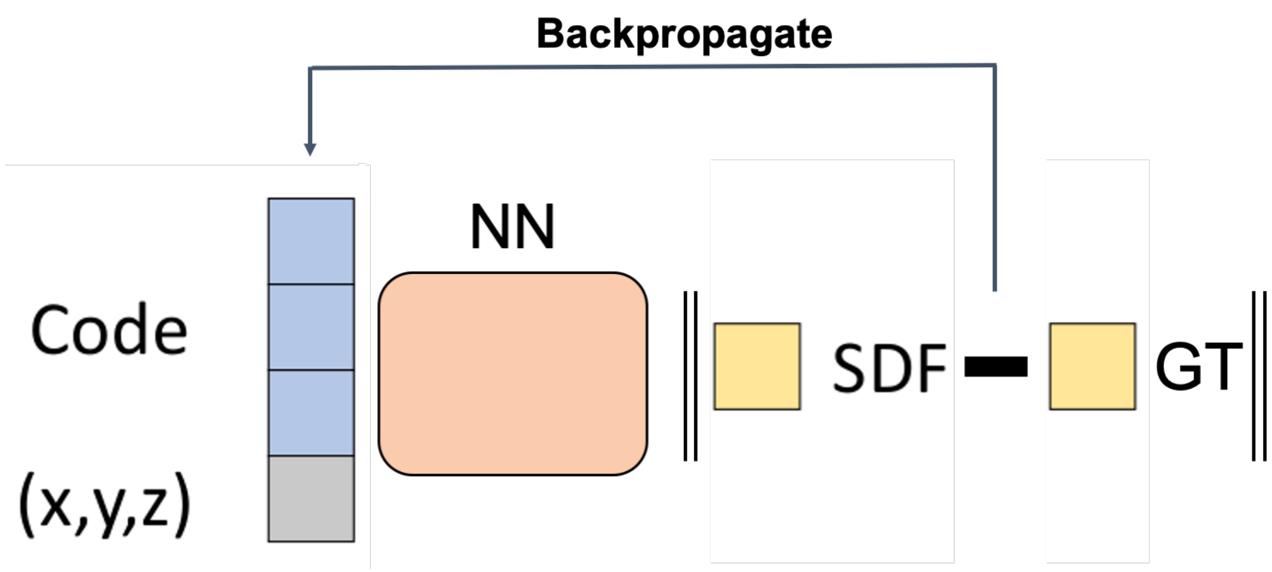



**Auto-Decoder**

# Auto-Decoder



# Benefits of Auto-Decoder

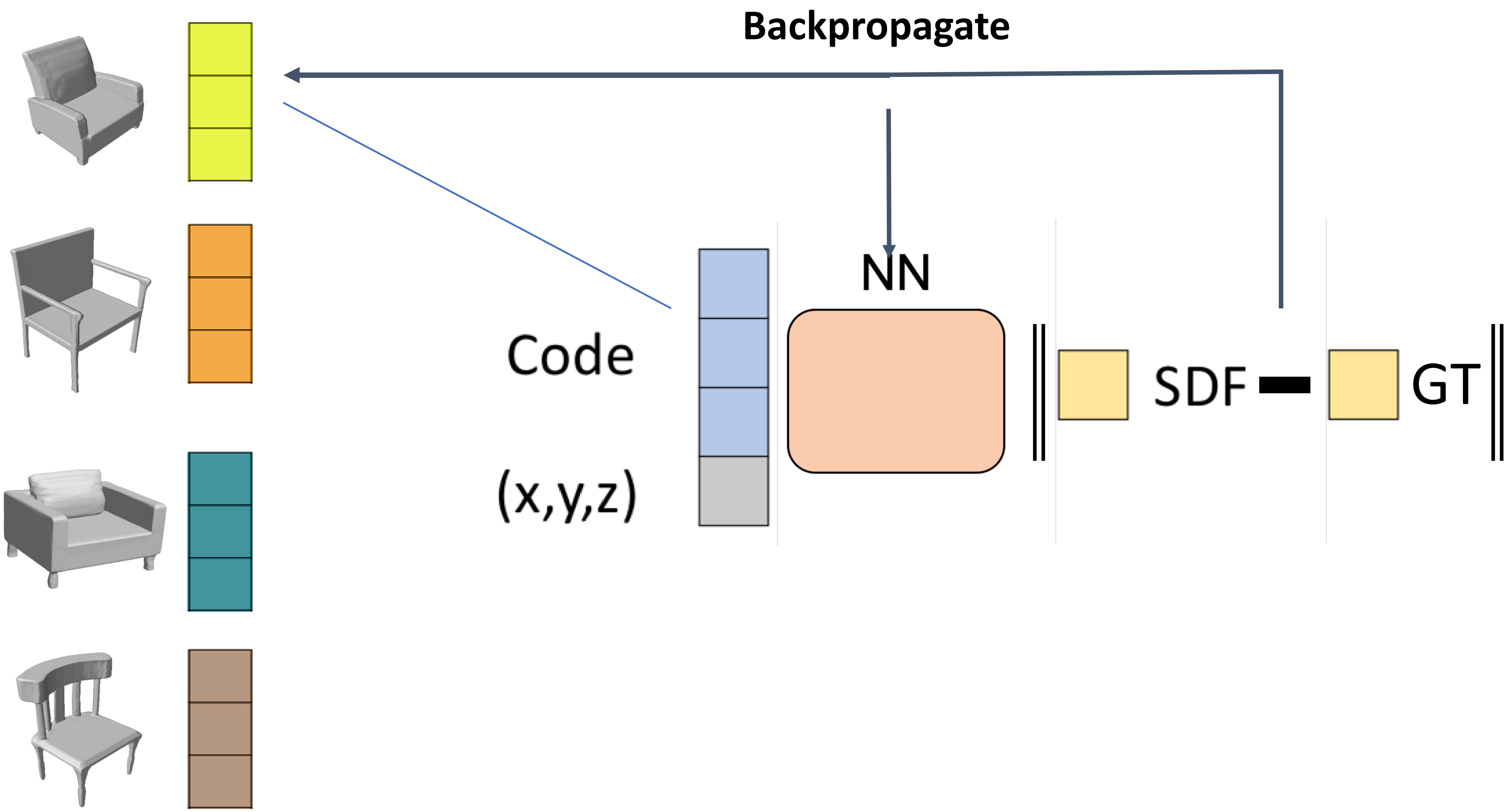



Benefits during Inference

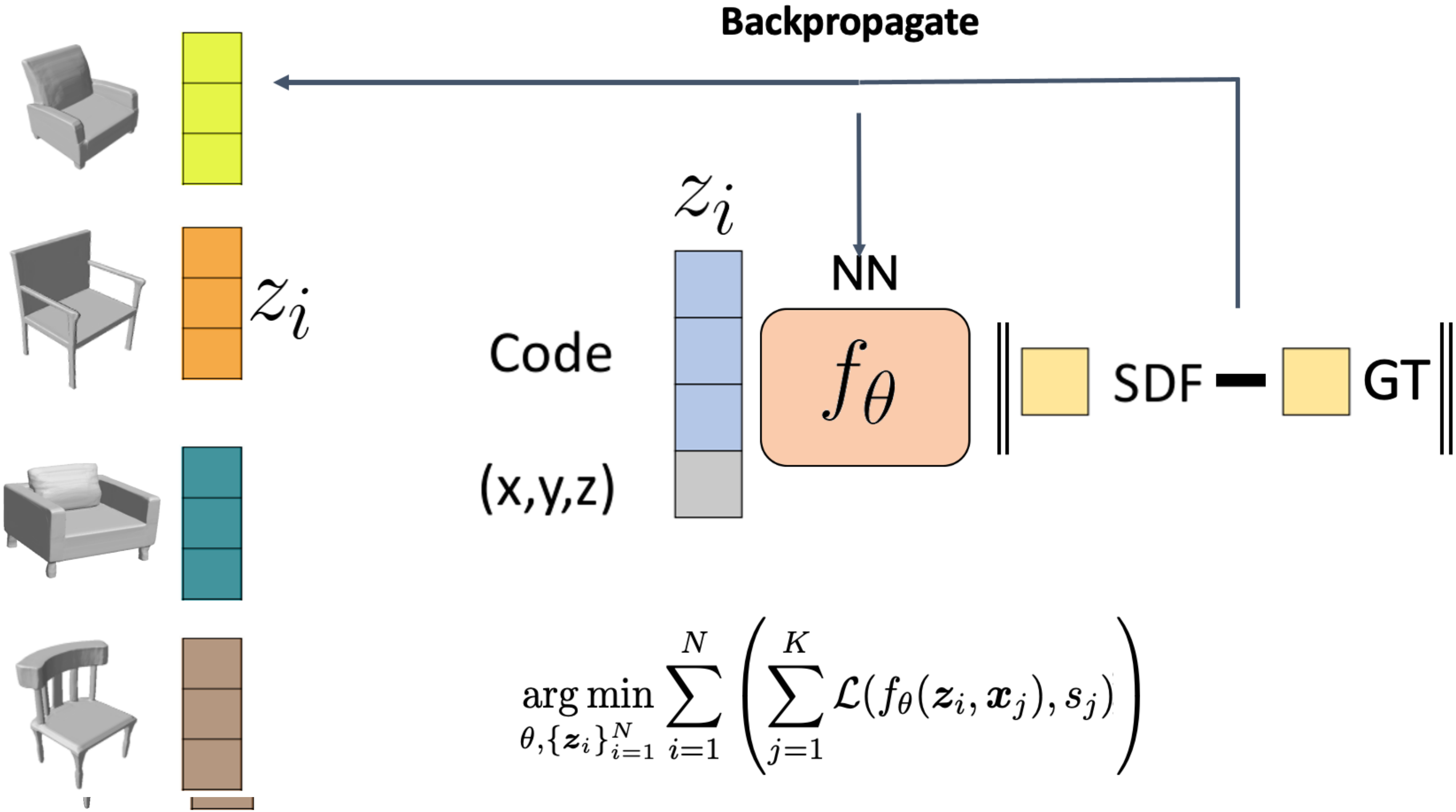
1. Any Number of Observations – Partial

2. More Controlled Inference – e.g. Accuracy, Priors

# Auto-Decoder Training

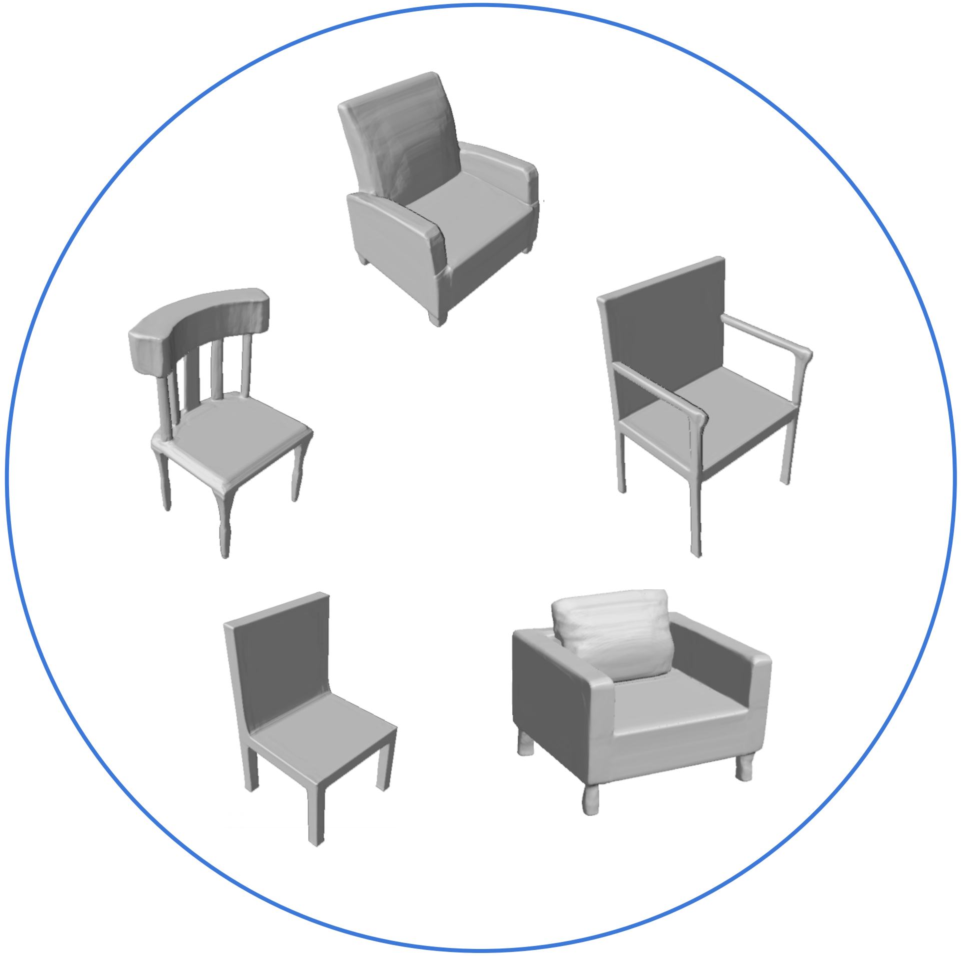


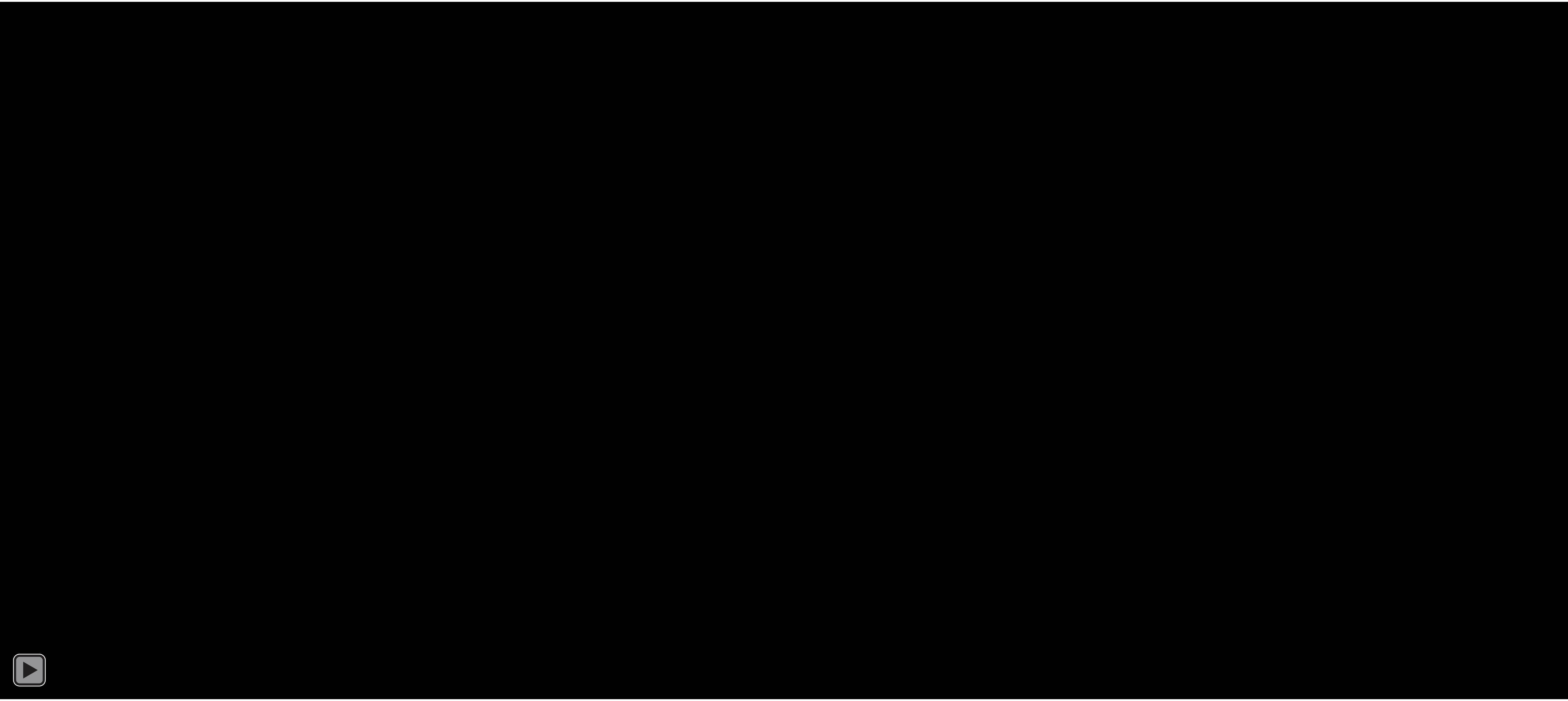
# Auto-Decoder Training



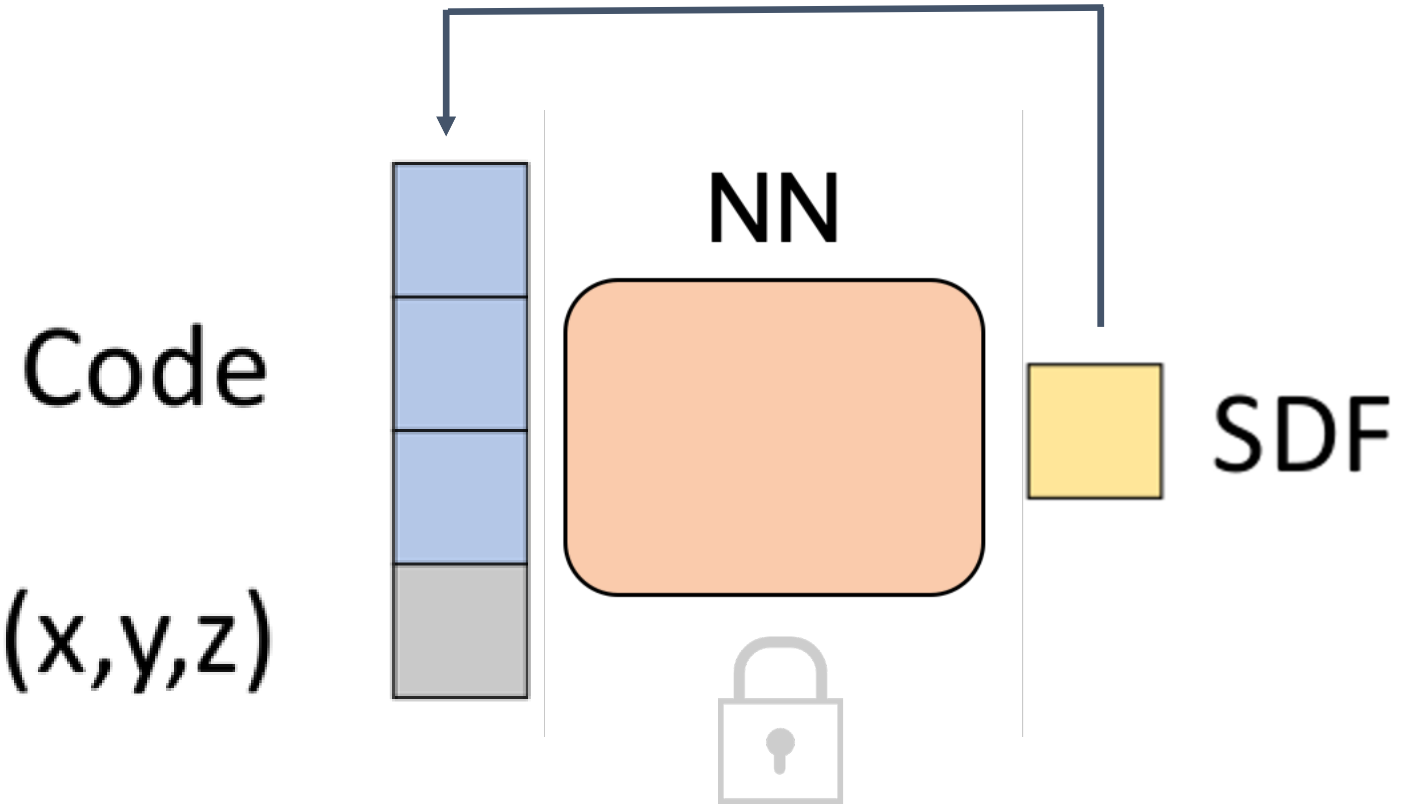

$$\underset{\theta, \{\boldsymbol{z}_i\}_{i=1}^{N}}{\arg\min} \sum_{i=1}^{N} \left( \sum_{j=1}^{K} \mathcal{L}(f_\theta(\boldsymbol{z}_i, \boldsymbol{x}_j), s_j) \right)$$

# Latent Space of Shapes



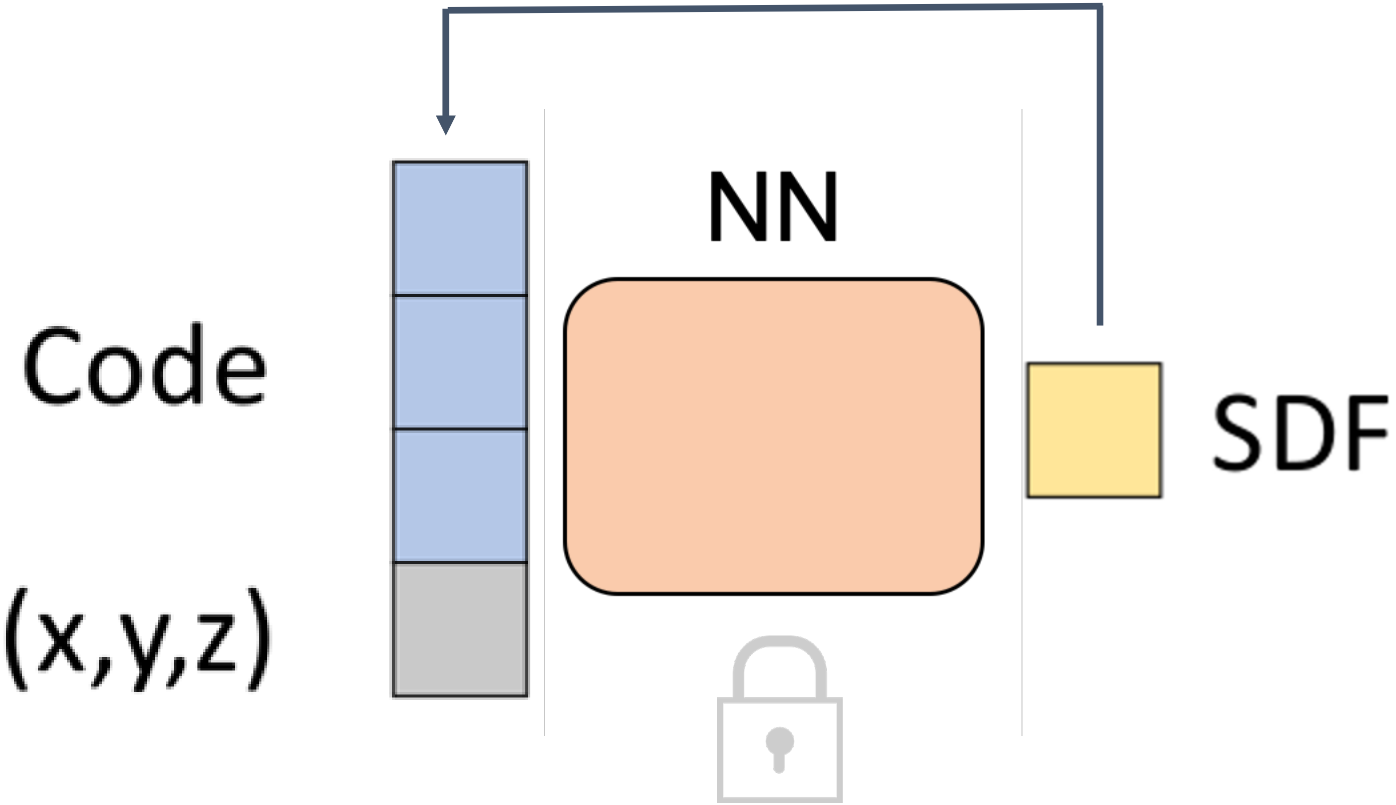


# Auto-Decoder Inference




Test Shape

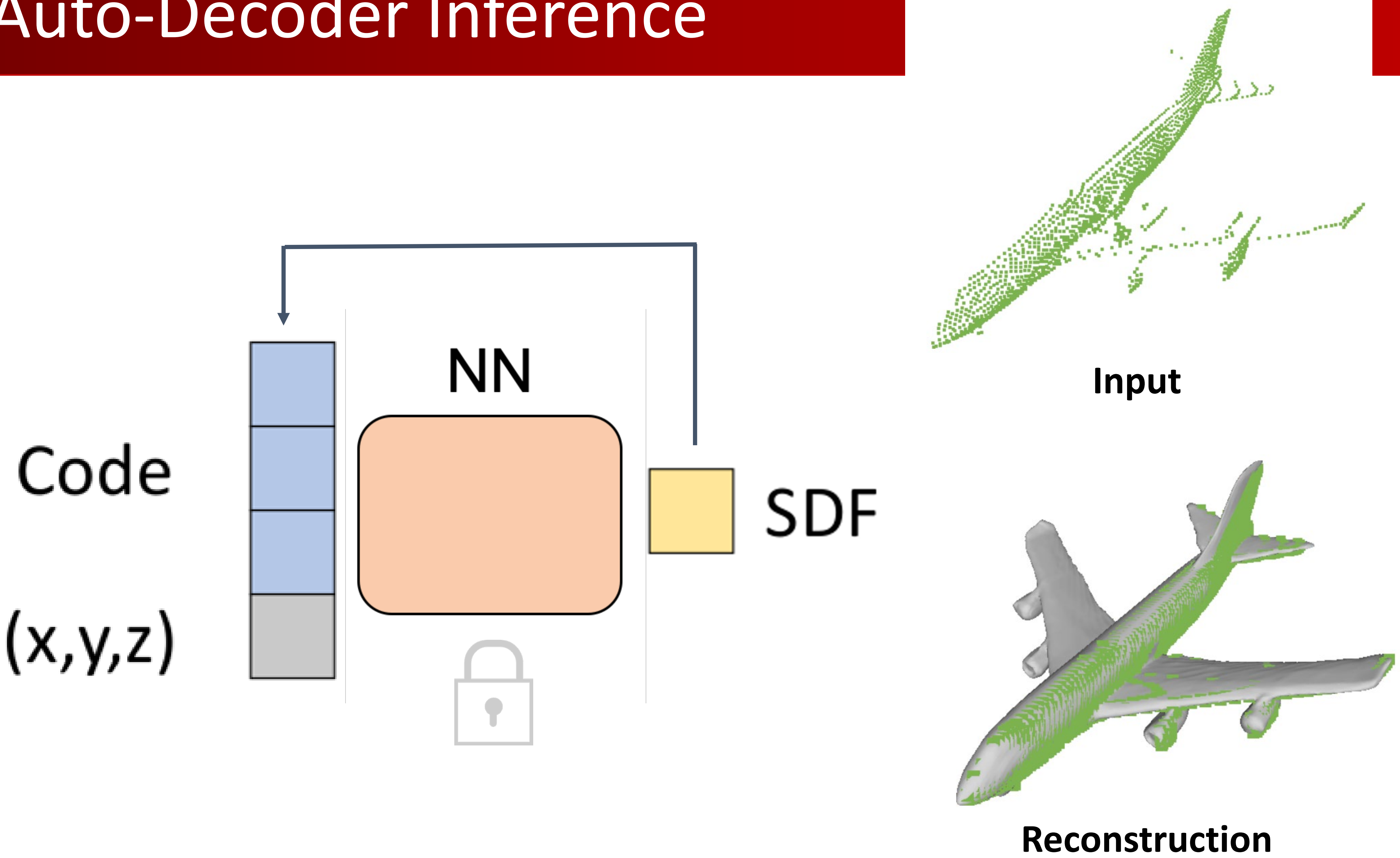
# Auto-Decoder Inference




Test Shape

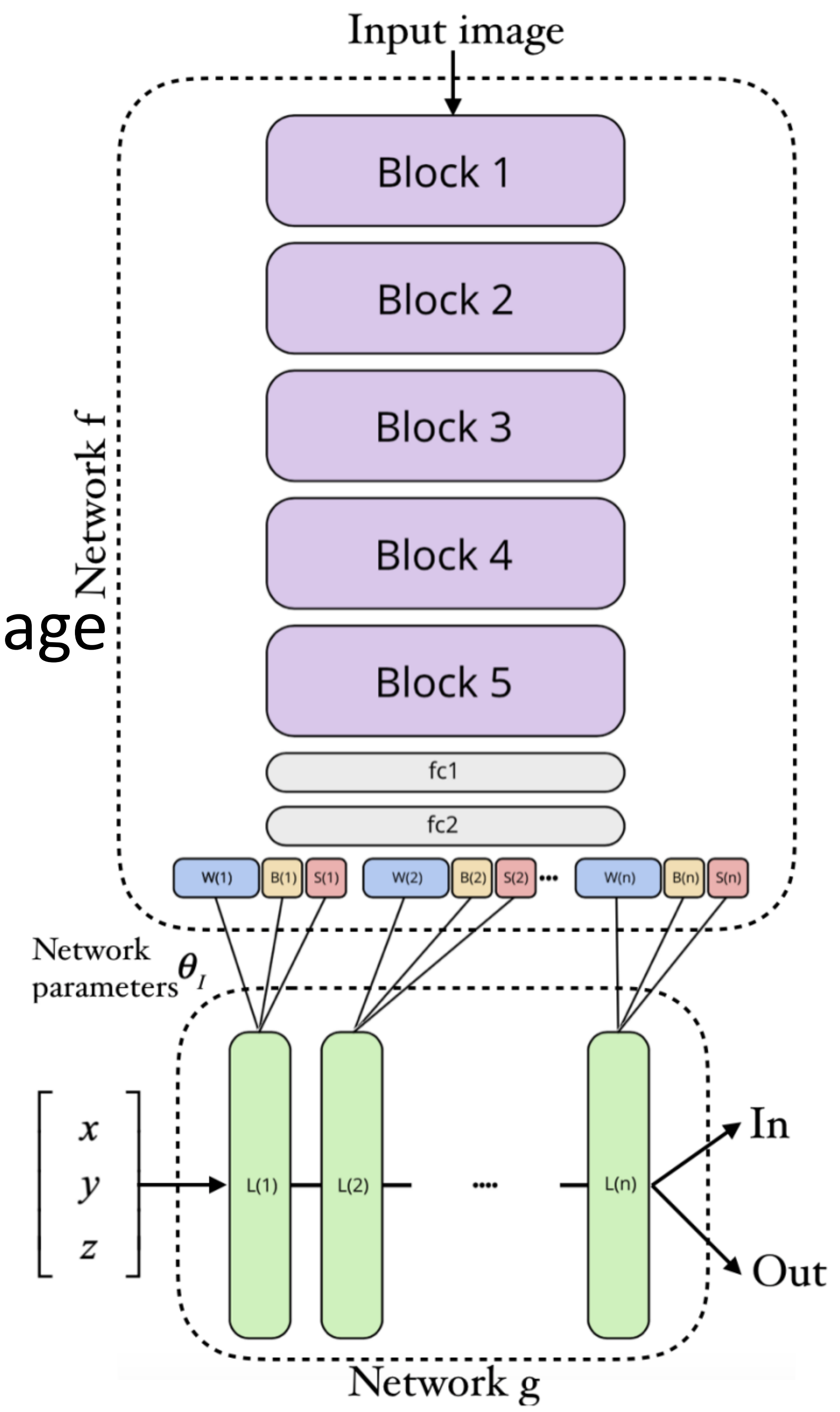
Reconstruction

# Auto-Decoder Inference



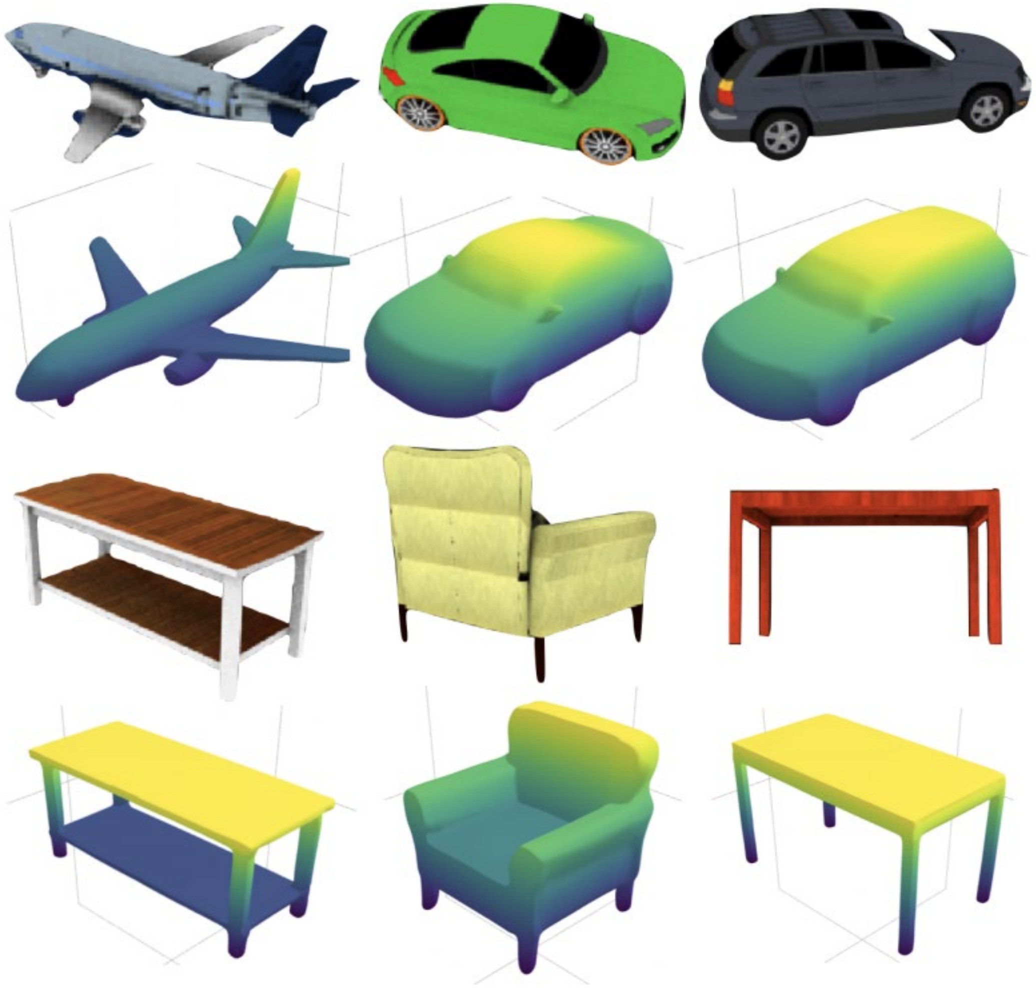
# Alternative: Image to SDF

- Instead of conditioning on code, predict to weights of the MLP itself
- Takes an image → outputs the weights
- The new network models the implicit field for the image

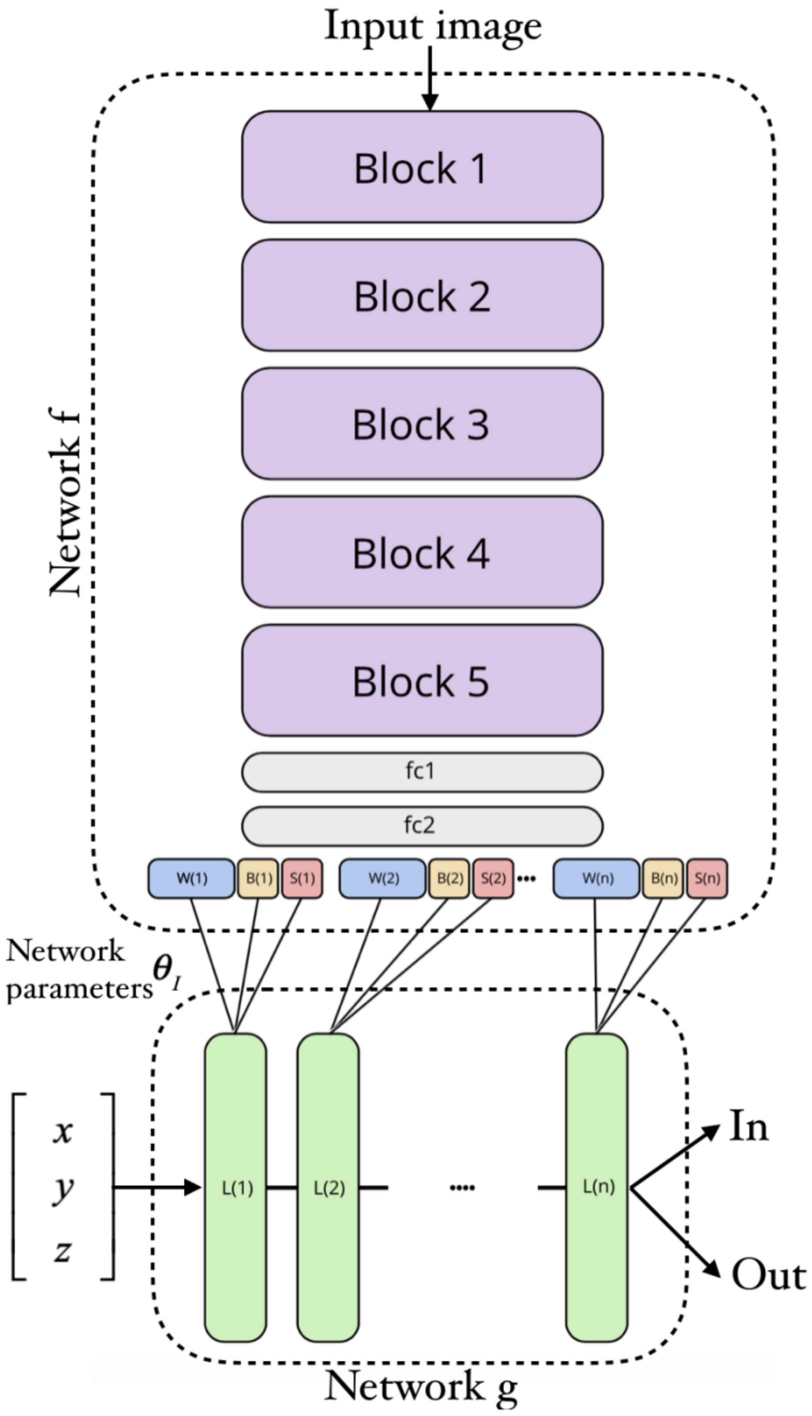



Littwin et al. 2019

# Alternative: Image to SDF

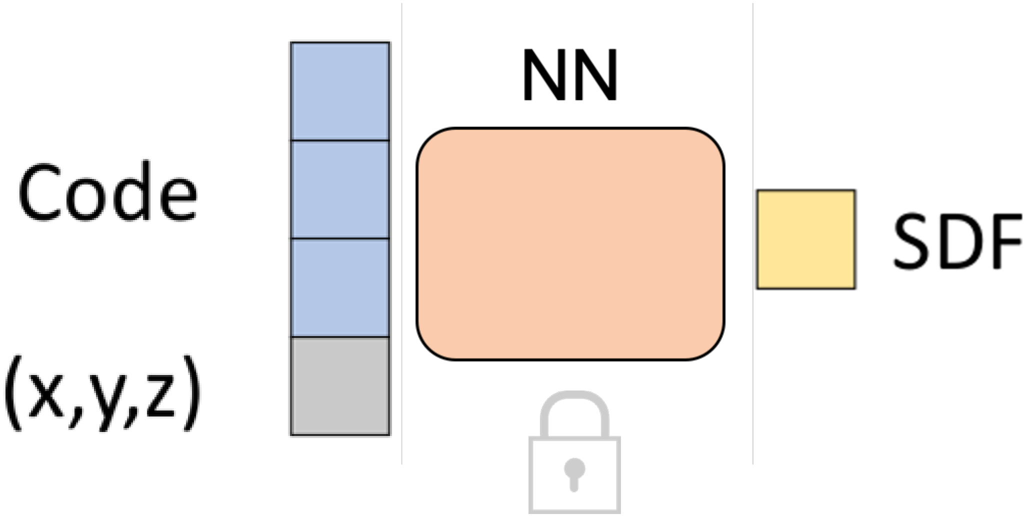


Littwin et al. 2019



# Adding Priors to Inference

$$\hat{\boldsymbol{z}} = \arg\min_{\boldsymbol{z}} \sum_{(\boldsymbol{x}_j, \boldsymbol{s}_j) \in X} \mathcal{L}(f_\theta(\boldsymbol{z}, \boldsymbol{x}_j), s_j)$$



# Adding Priors to Inference

$$\hat{\boldsymbol{z}} = \arg\min_{\boldsymbol{z}} \sum_{(\boldsymbol{x}_j, \boldsymbol{s}_j) \in X} \mathcal{L}(f_\theta(\boldsymbol{z}, \boldsymbol{x}_j), s_j)$$

Distribution Prior: $\frac{1}{\sigma^2} ||\boldsymbol{z}||_2^2$

SDF Regularization: $\left( \|\nabla_{\boldsymbol{x}} f(\boldsymbol{x}; \theta)\| - 1 \right)^2$ (Matan et al. 2020)

Normal Regularization: $\|\nabla_{\boldsymbol{x}} f(\boldsymbol{x}_i; \theta) - \boldsymbol{n}_i\|$

# Results

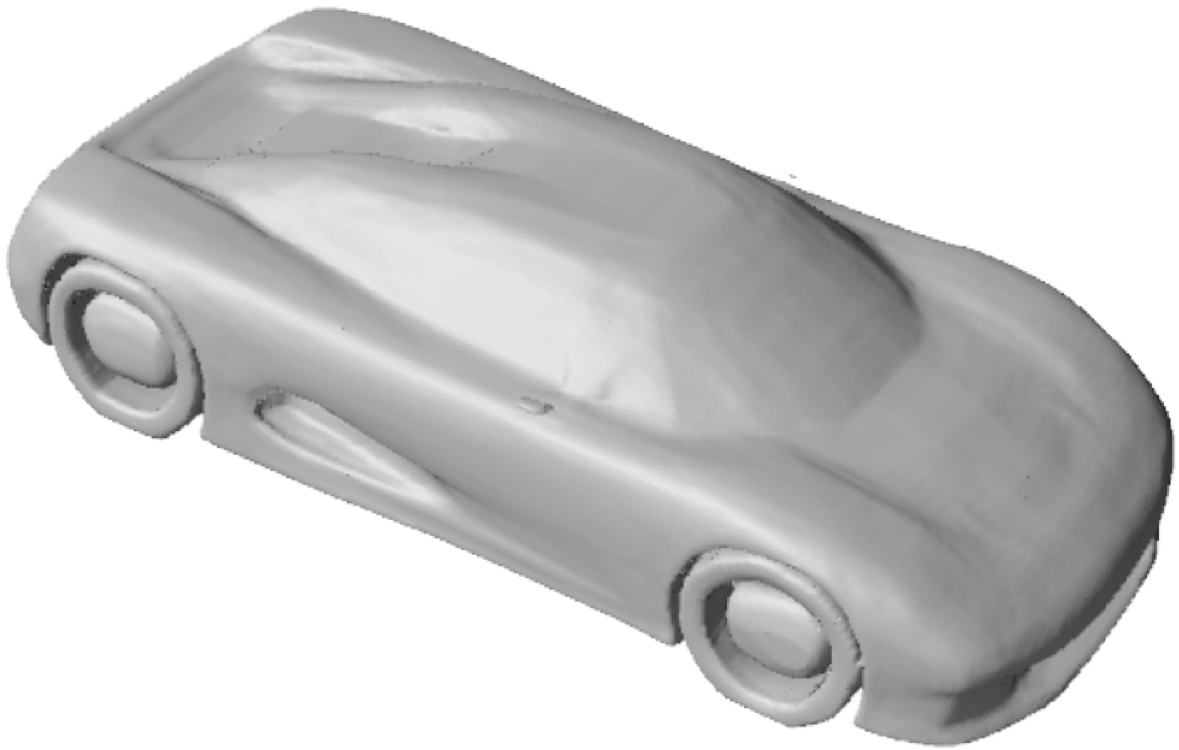
## Auto-encoding unknown shapes

| CD, median | | | | | |
|---|---|---|---|---|---|
| AtlasNet-Sph. | 0.511 | 0.079 | 0.389 | 2.180 | 0.330 |
| AtlasNet-25 | 0.276 | 0.065 | 0.195 | 0.993 | 0.311 |
| DeepSDF | **0.072** | **0.036** | **0.068** | **0.219** | **0.088** |

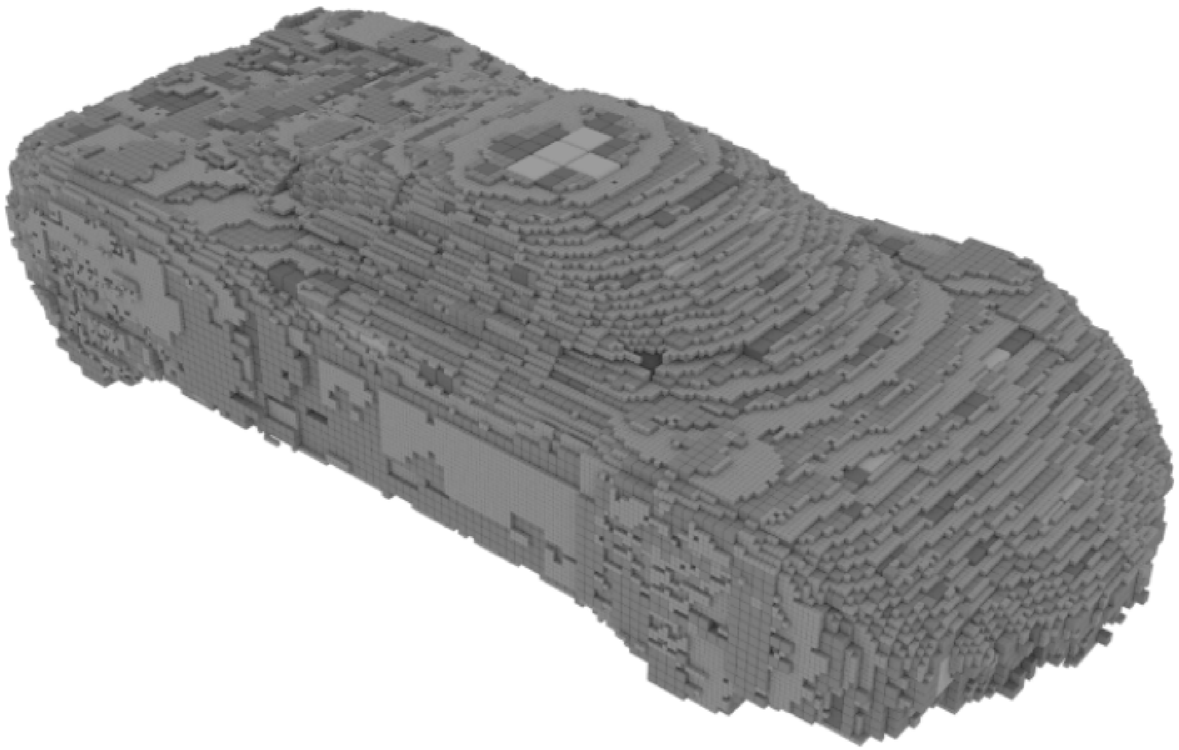
## Shape completion

| | *lower is better* | | | | *higher is better* | |
|---|---|---|---|---|---|---|
| Method \Metric | CD, med. | CD, mean | EMD | Mesh acc. | Mesh comp. | Cos sim. |
| chair | | | | | | |
| 3D-EPN | 2.25 | 2.83 | 0.084 | 0.059 | 0.209 | 0.752 |
| DeepSDF | **1.28** | **2.11** | **0.071** | **0.049** | **0.500** | **0.766** |
| plane | | | | | | |
| 3D-EPN | 1.63 | 2.19 | 0.063 | 0.040 | 0.165 | 0.710 |
| DeepSDF | **0.37** | **1.16** | **0.049** | **0.032** | **0.722** | **0.823** |

# Results: Comparison with Octree-Based



Our
Reconstruction

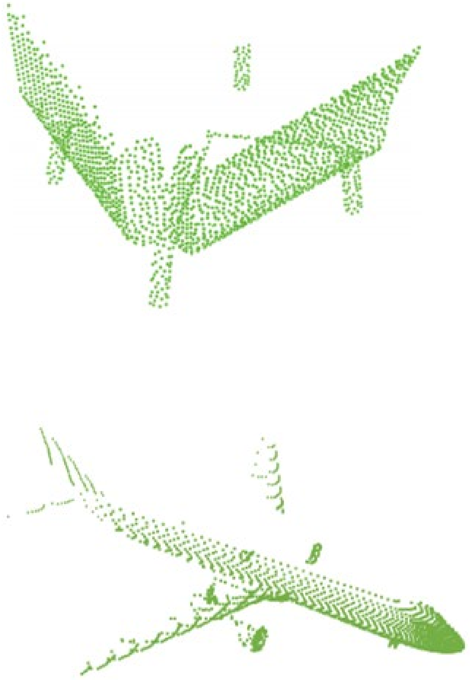


Octree Based

# Results: Comparisons with Mesh-Based

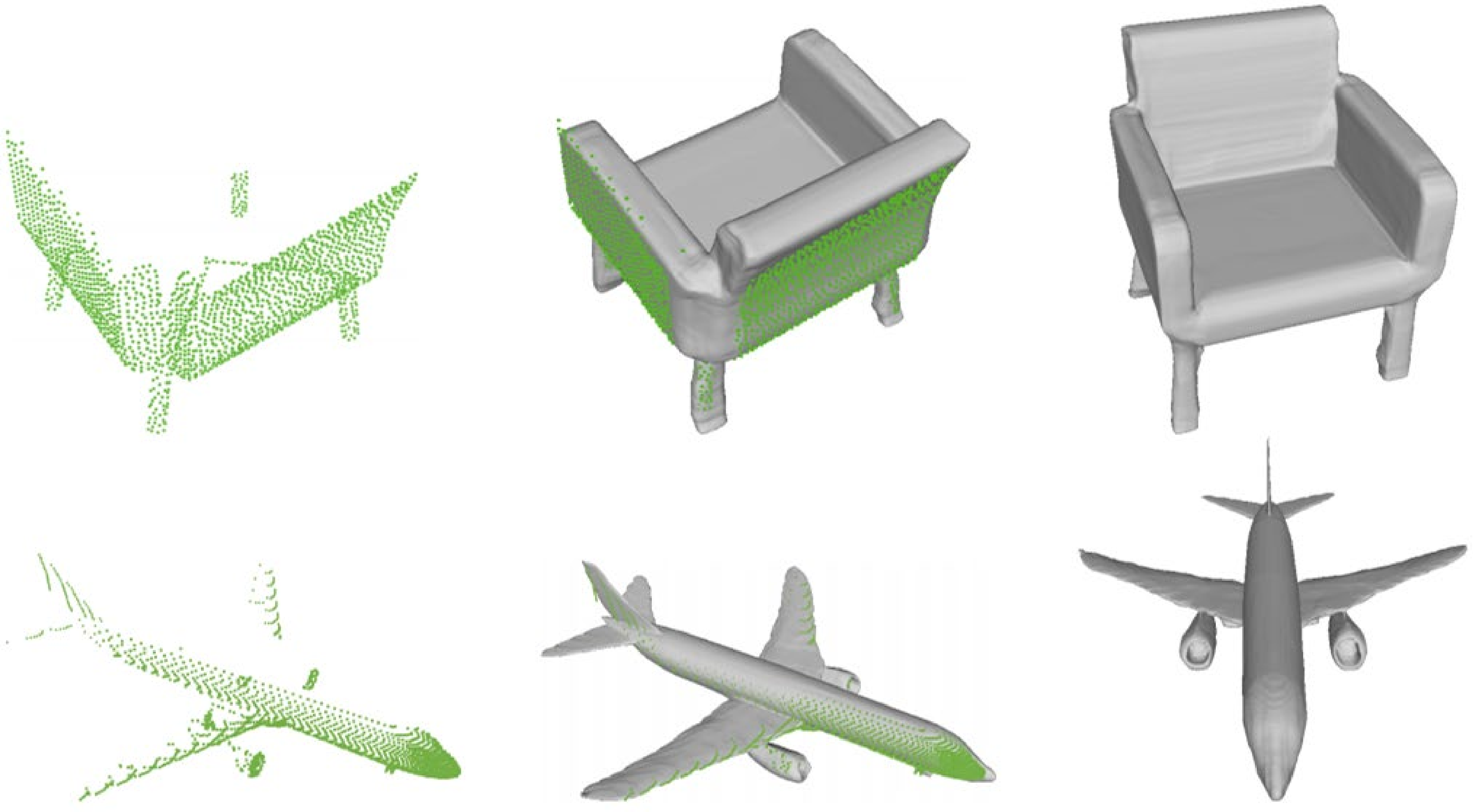
Ground Truth | Our Reconstruction | Atlasnet (25 Patches) | Atlasnet (1 Patch)

# Shape Completion



(a) Input Depth

# Shape Completion

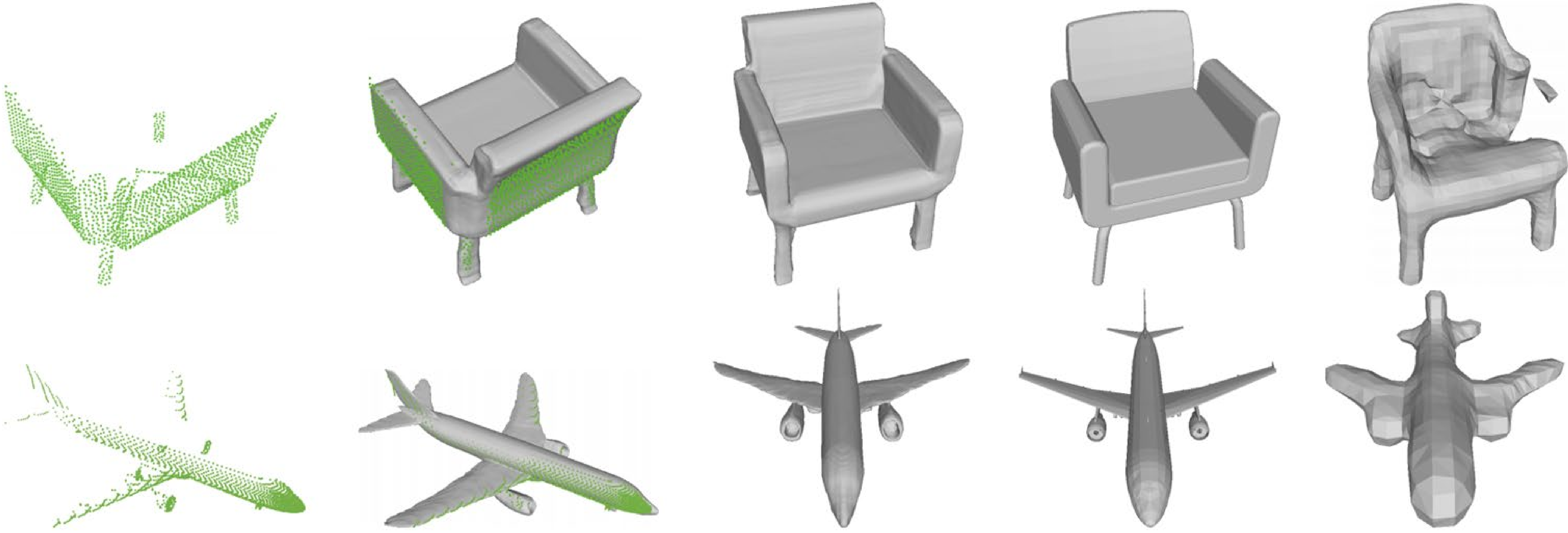


(a) Input Depth (b) Completion (ours) (c) Second View (ours)

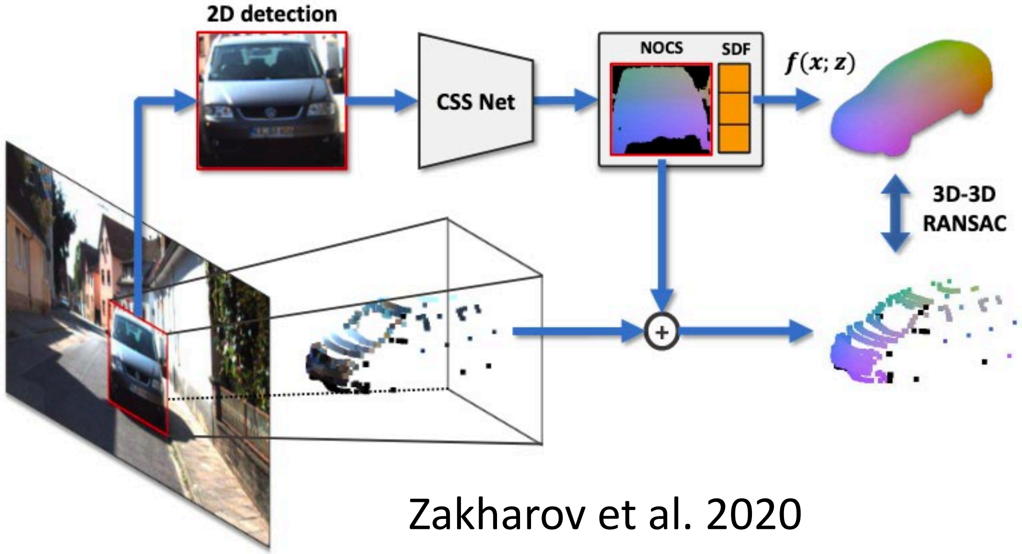
# Shape Completion

(a) Input Depth (b) Completion (ours) (c) Second View (ours) (d) Ground truth

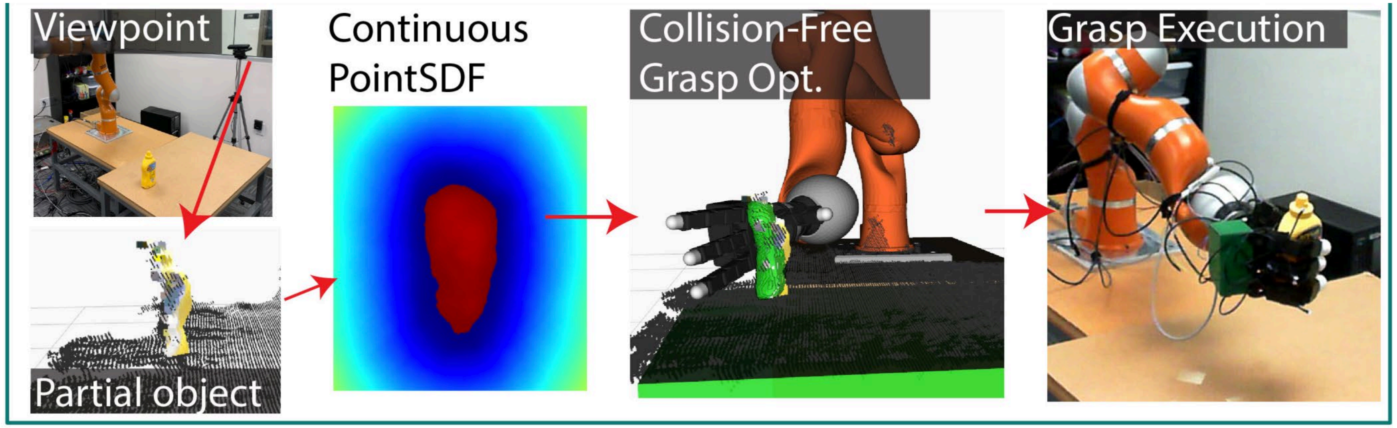
# Shape Completion



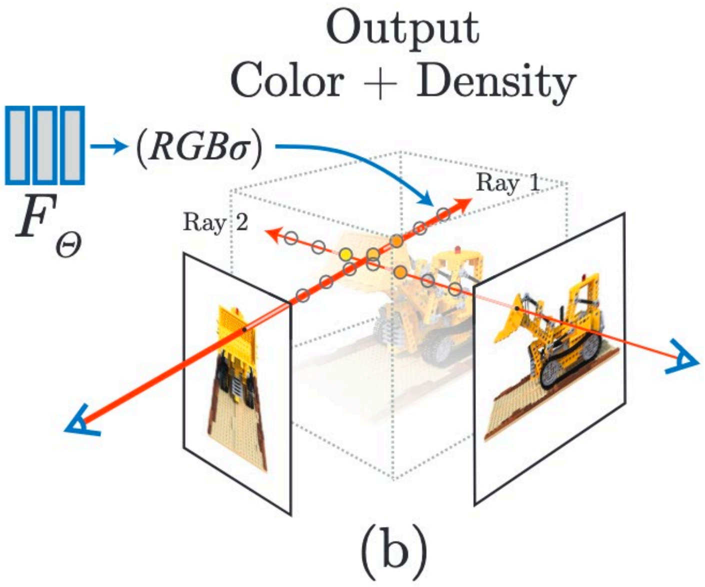
(a) Input Depth (b) Completion (ours) (c) Second View (ours) (d) Ground truth (e) 3D-EPN



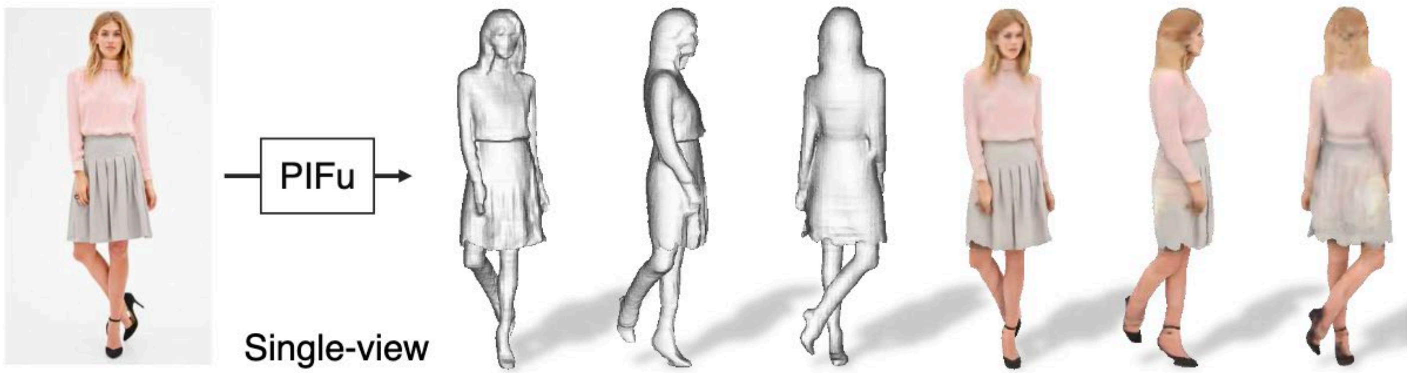

Zakharov et al. 2020




Merwe et al. 2020

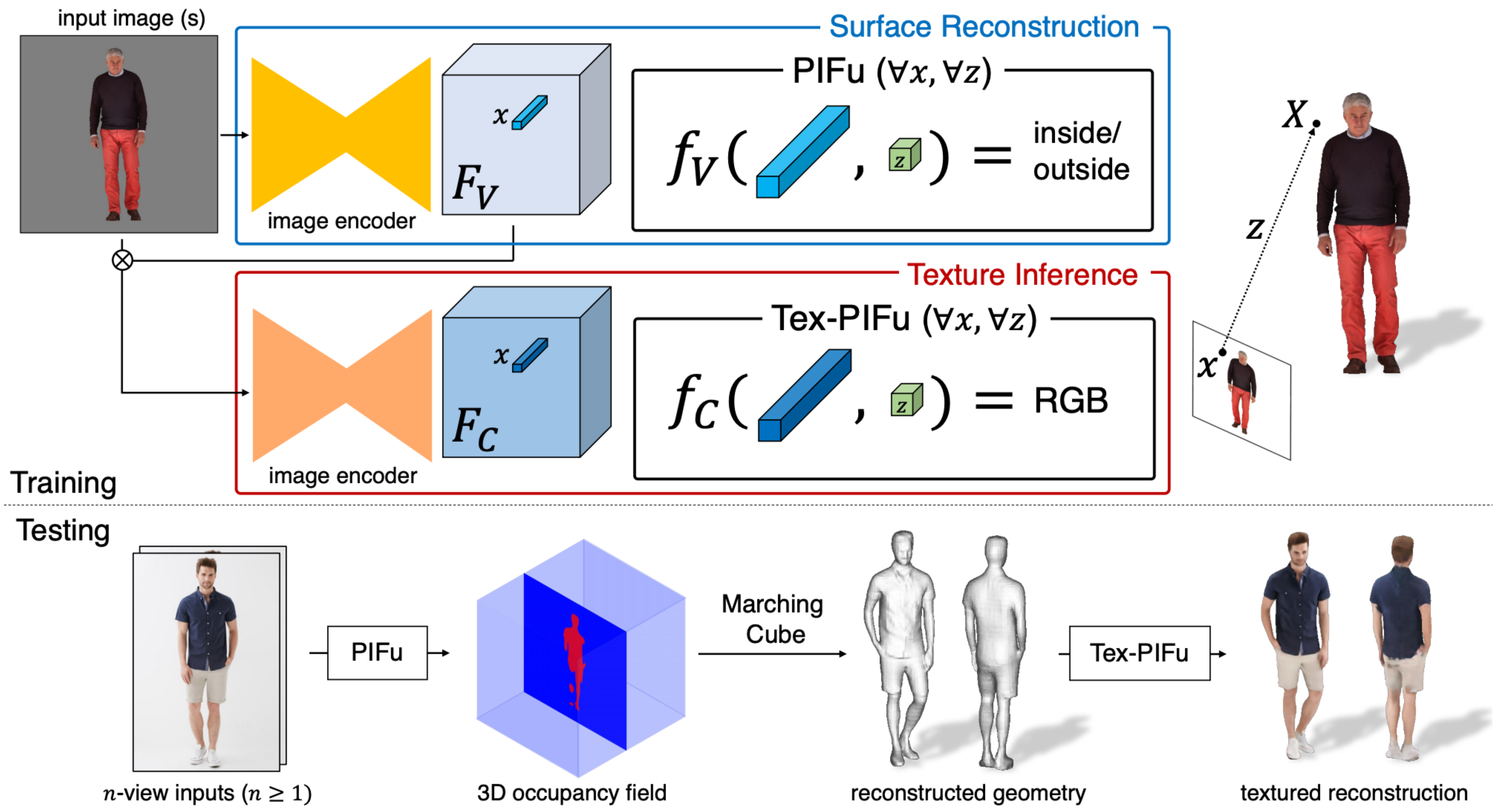



Mildenhall et al. 2020




Saito et al. 2019

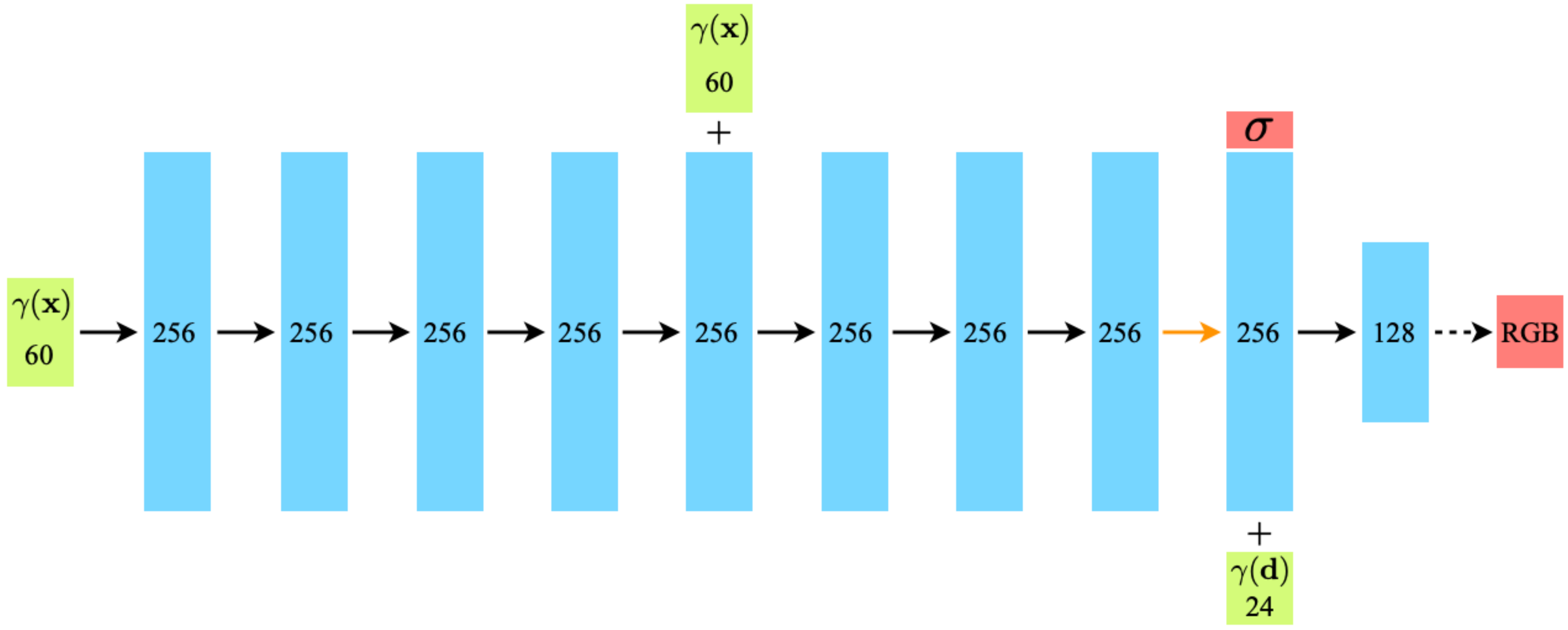
# DeepSDF Extensions: PiFU




Saito et al. 2019

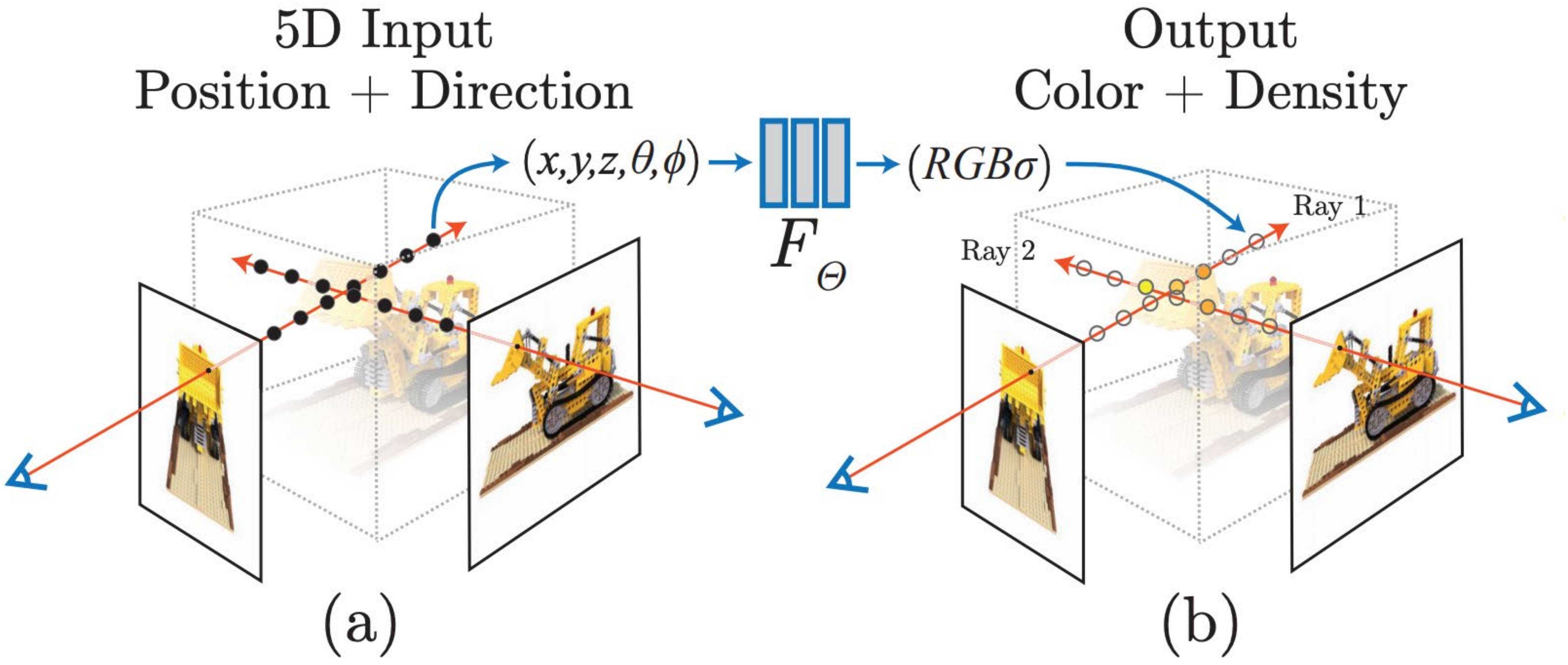
# DeepSDF Extensions: NeRF

- Coordinate-based modeling of RGB and Densities Instead of SDFs



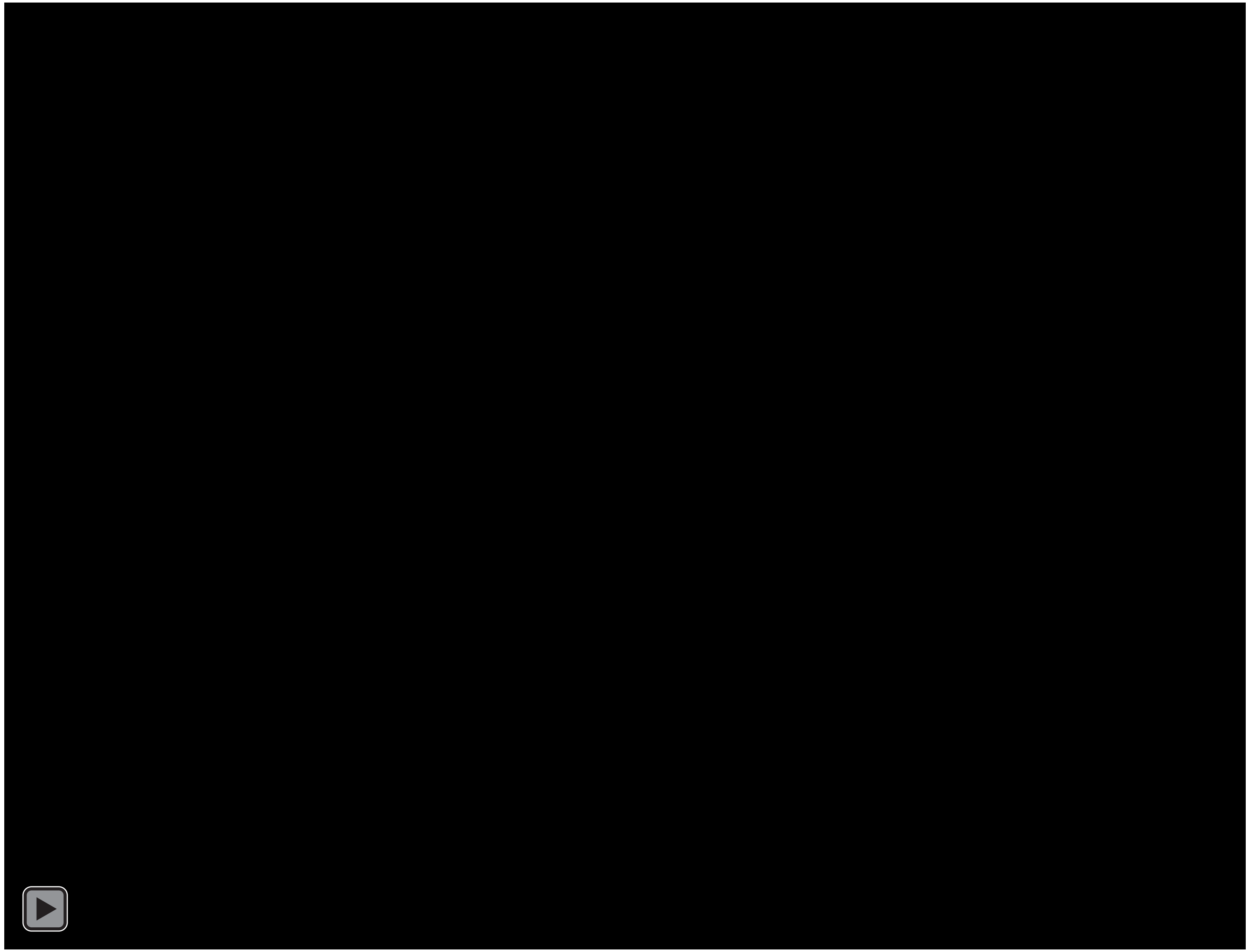

Mildenhall et al. 2020

# DeepSDF Extensions: NeRF




Mildenhall et al. 2020

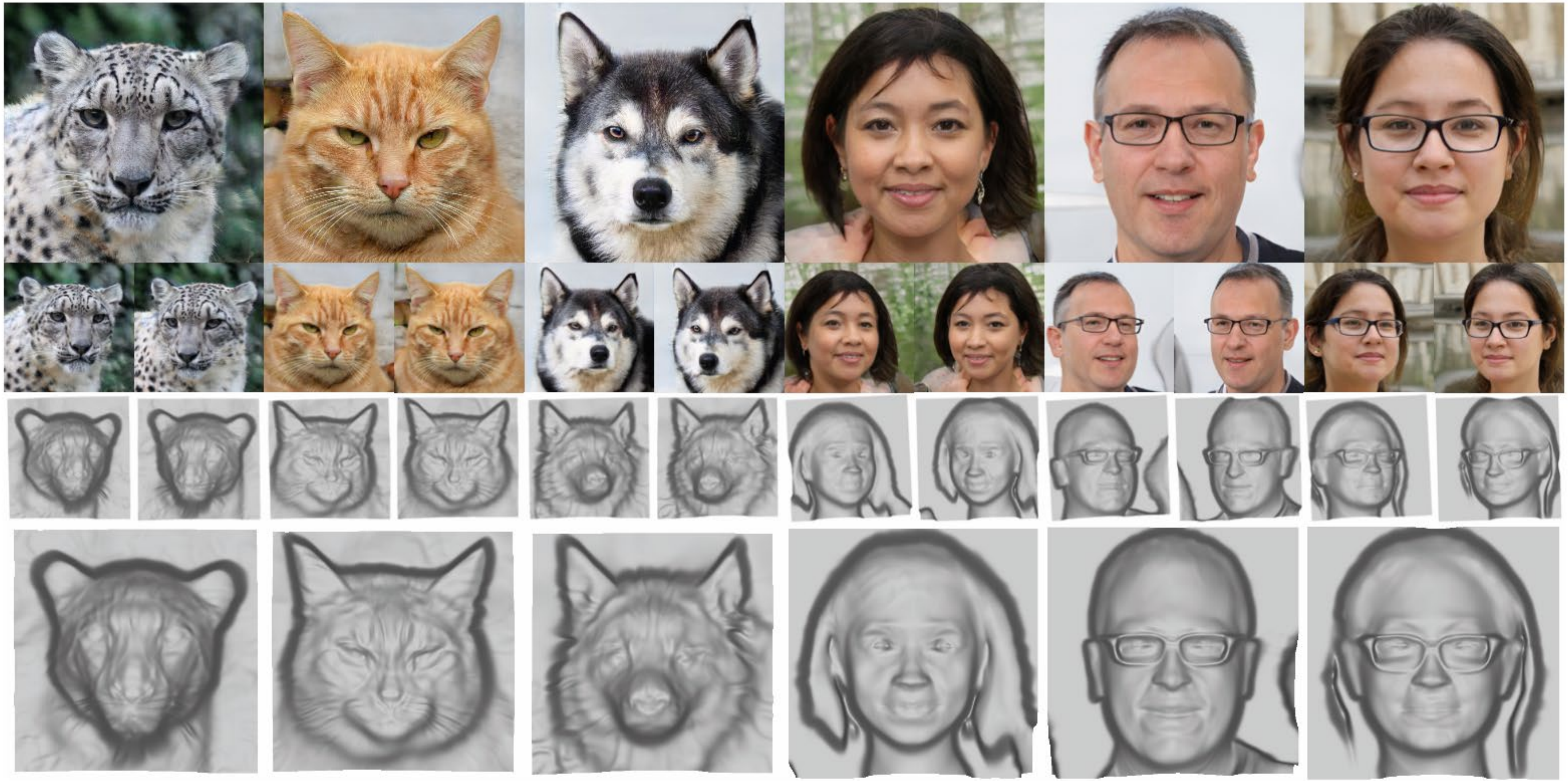
# DeepSDF Extensions: NeRF



Mildenhall et al. 2020

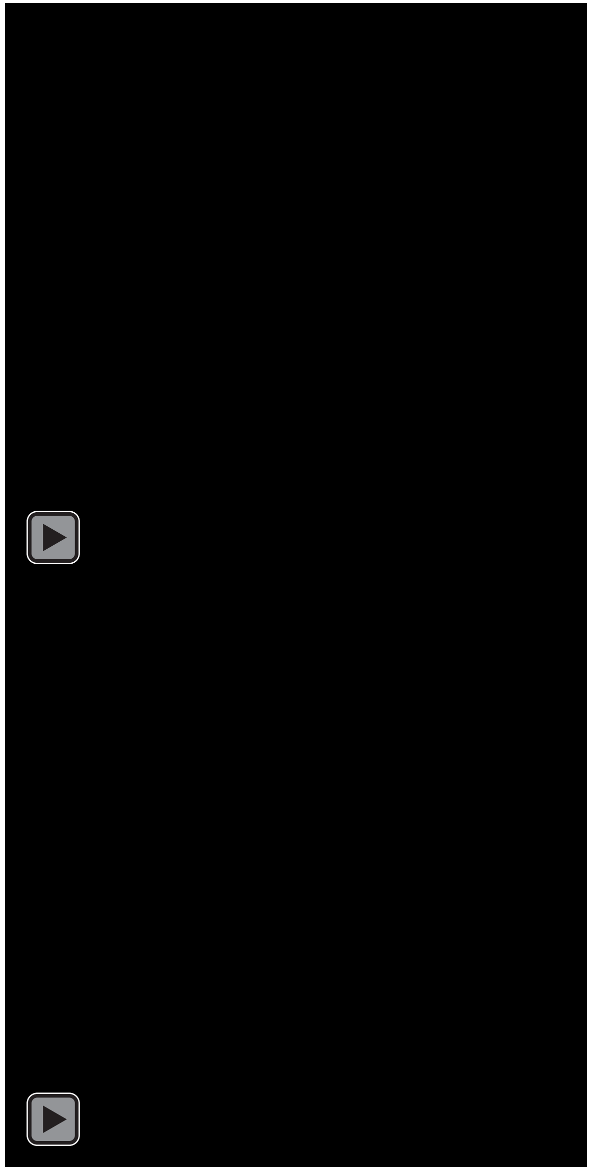
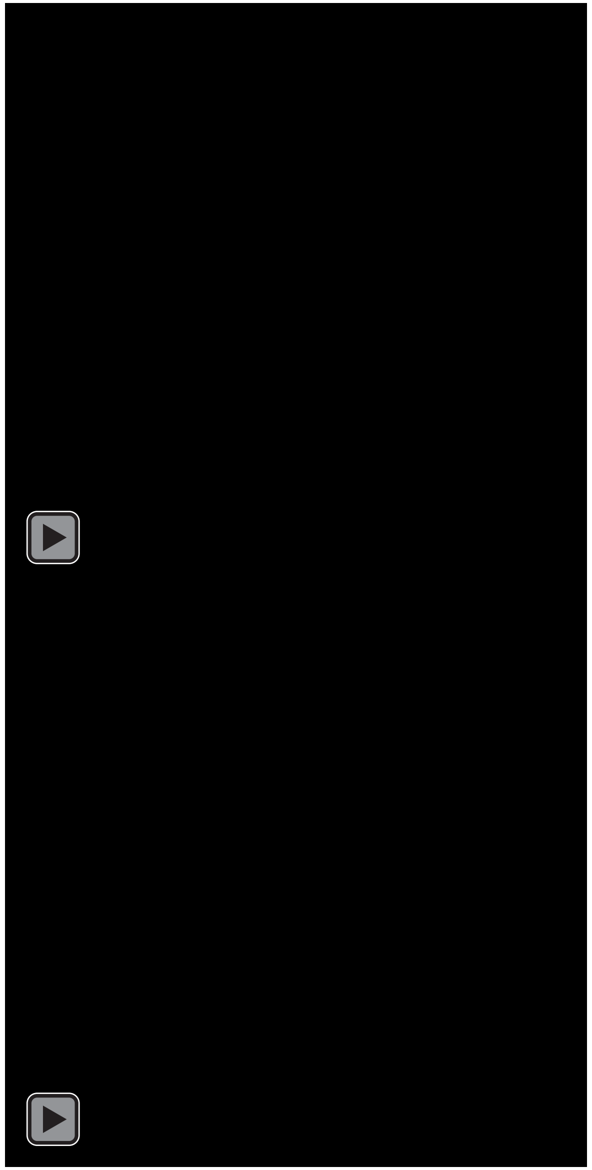
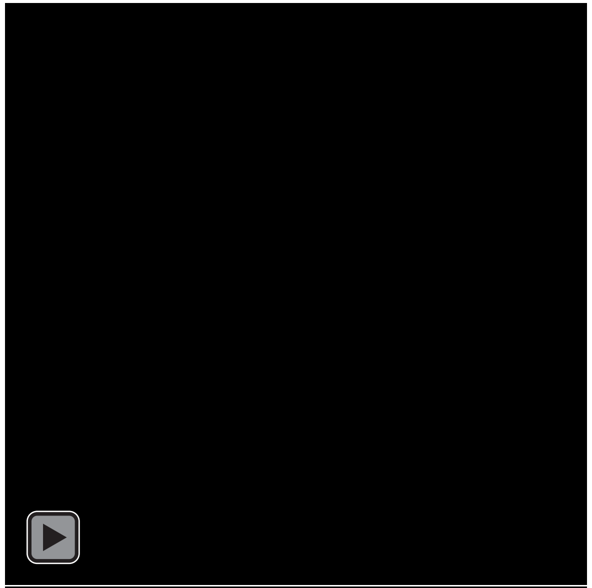
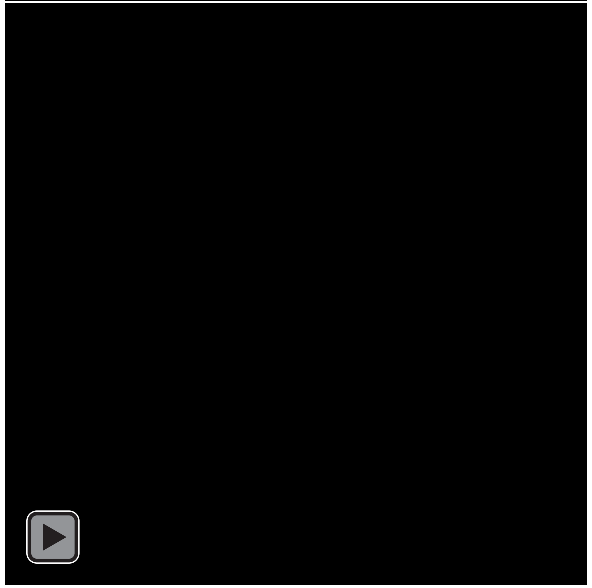
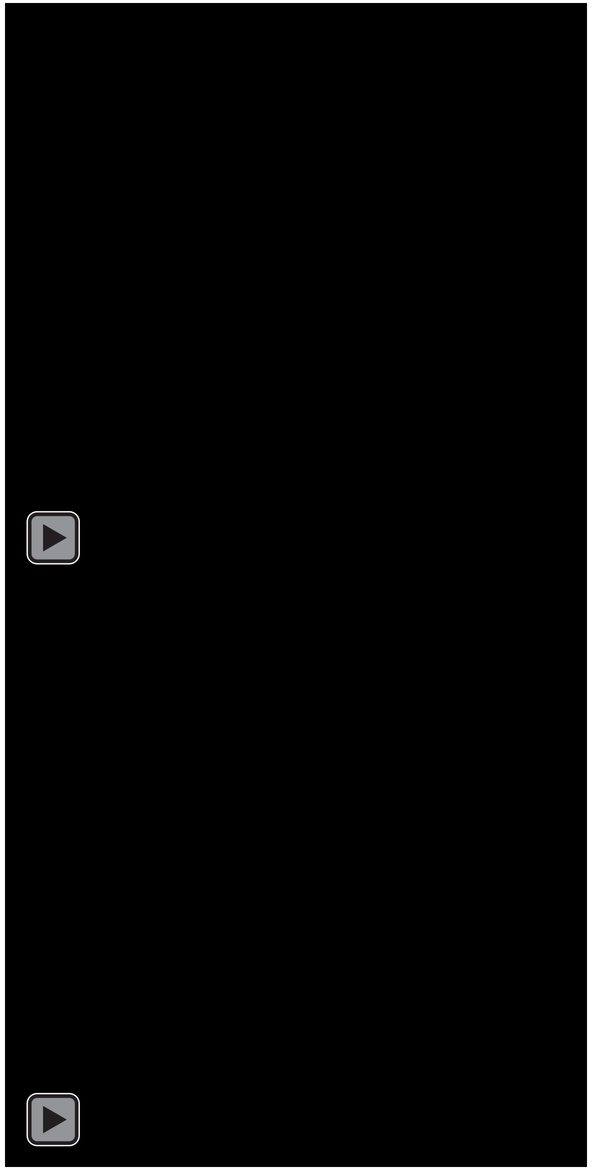
# DeepSDF Extension: StyleSDF

- A 3D GAN using DeepSDF + NeRF modeling



Or-El et al. 2021

# DeepSDF Extension: StyleSDF

# Thank you!

- Speaker: Jeong Joon Park

# That's All