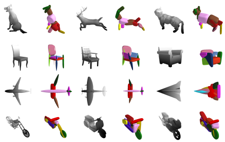
Learning Deep Models with Primitive-based Representations

Despoina Paschalidou

Autonomous Vision Group, Max Planck Institute for Intelligent Systems
Tübingen
Computer Vision Lab, ETH Zürich

Slides are available at

htpps://paschalidoud.github.io/talks/primitive-based-representations

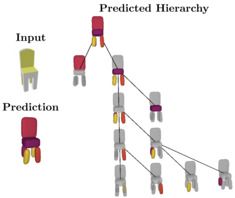


Superquadrics Revisited: Learning 3D Shape Parsing beyond Cuboids

Despoina Paschalidou, Ali Osman Ulusoy, Andreas Geiger

CVPR 2019

https://superquadrics.com/learnable-superquadrics.html



Learning Unsupervised Hierarchical Part Decomposition of 3D Objects from a Single RGB Image

Despoina Paschalidou, Luc van Gool, Andreas Geiger

CVPR 2020

https://superquadrics.com/hierarchical-primitives.html

Neural networks for 2D computer vison tasks

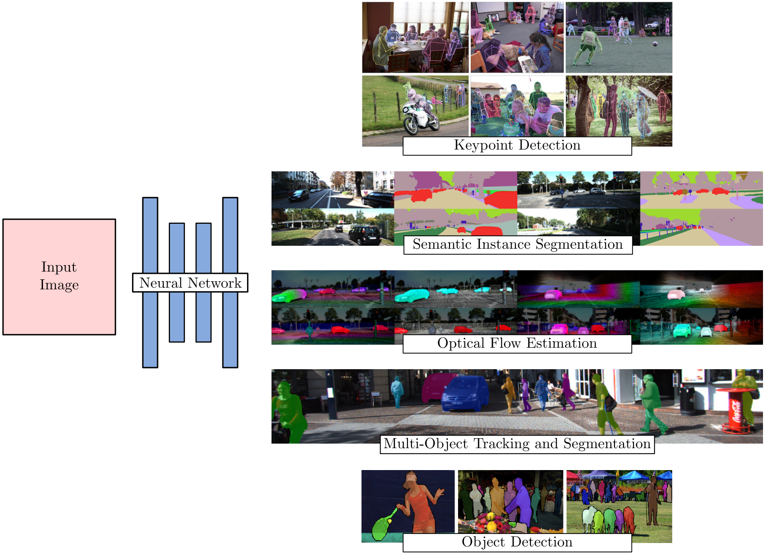
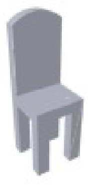
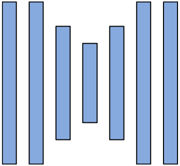


Image Source: KITTI Vision Benchmark and COCO Dataset

Can we learn to infer 3D from a 2D image?



Input Image



Neural Network

3D Reconstruction

What is the optimal 3D Representation?

Input Image

Depth

Voxel Grid

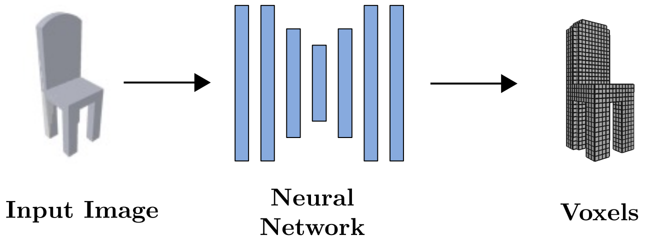
Pointcloud

Mesh

Primitives

Implicit Surface

3D Representations



Discretization of 3D shape into grid:

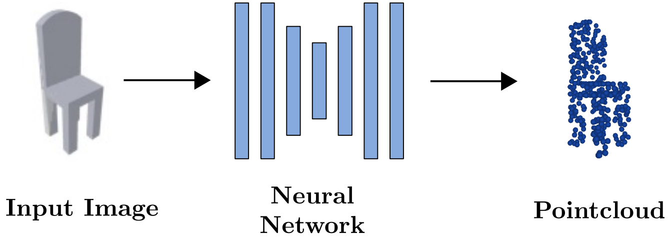
✓ Accurately captures the **shape details**

✗ **Parametrization size** proportional to **reconstruction quality**

✗ Unable to yield **smooth reconstructions**

✗ Do not convey **semantic information**

3D Representations



Discretization of surface with **3D points**:

✓ Accurately captures the **shape details**

✗ Lacks surface connectivity

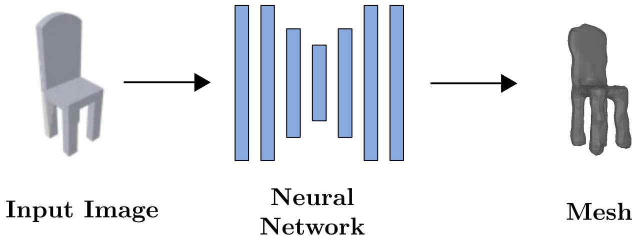
✗ Fixed number of points

✗ **Parametrization size** proportional to **reconstruction quality**

✗ Unable to yield **smooth reconstructions**

✗ Do not convey **semantic information**

3D Representations



Discretization of surface into **vertices and faces**:

✓ Accurately captures the **shape details**

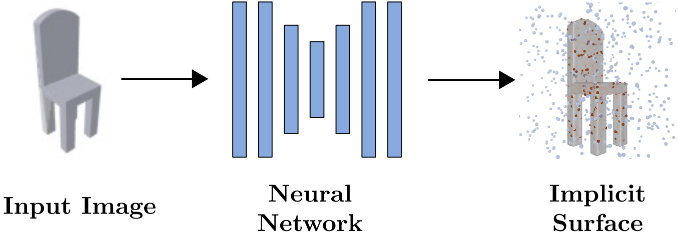
✓ Yields **smooth reconstructions**

✗ Requires class-specific template topology

✗ **Parametrization size**

✗ Do not convey **semantic information**

3D Representations



No discretization

✓ Accurately captures the **shape details**

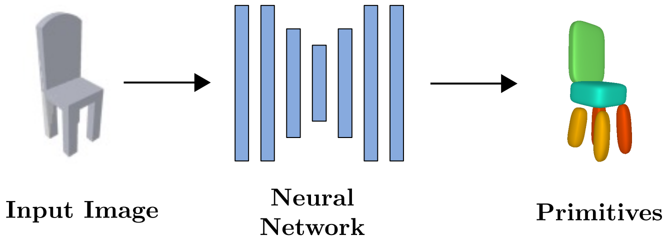
✓ Low **parametrization size**

✓ Yields **smooth reconstructions**

✗ Requires post-processing

✗ Do not convey **semantic information**

3D Representations



Discretization of 3D shape into **parts**:

✓ Low **parametrization size**

✓ Yields **smooth reconstructions**

✓ Yields **semantic reconstructions**

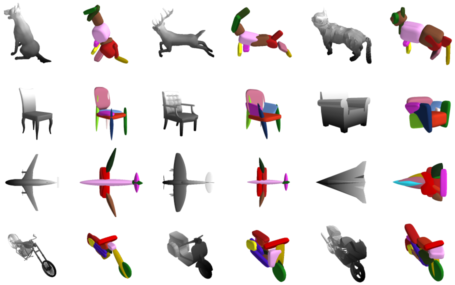
✓ **Inter-object coherence**

∼ Accurately captures the **shape details**

Superquadrics Revisited: Learning 3D Shape Parsing beyond Cuboids

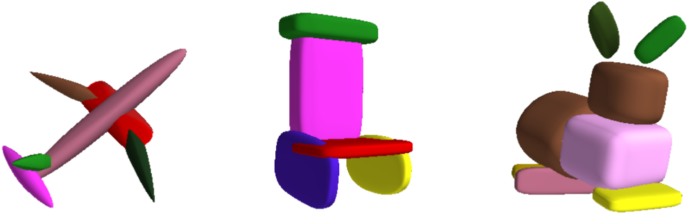
Despoina Paschalidou, Ali Osman Ulusoy, Andreas Geiger

CVPR 2019



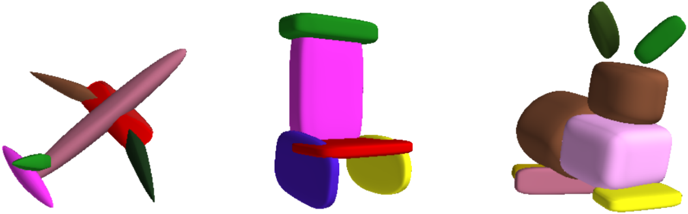
https://superquadrics.com/learnable-superquadrics.html

3D Geometric Primitives



Primitive-based 3D Representations:

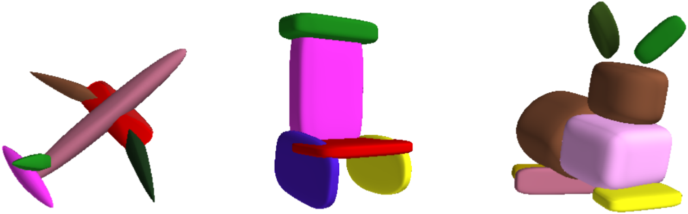
3D Geometric Primitives



Primitive-based 3D Representations:

- **Parsimonious Description**: Few primitives required to represent a 3D object

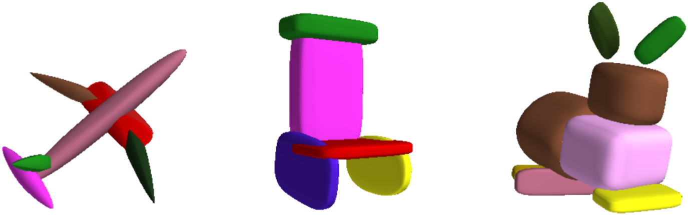
3D Geometric Primitives



Primitive-based 3D Representations:

- **Parsimonious Description**: Few primitives required to represent a 3D object
- Convey semantic information (parts, functionality, etc.)

3D Geometric Primitives



Primitive-based 3D Representations:

- **Parsimonious Description**: Few primitives required to represent a 3D object
- Convey semantic information (parts, functionality, etc.)
- **Main Challenge**: Variable number of primitives, little annotated datasets

3D Shape Abstraction

Goal of this work:

3D Shape Abstraction

Goal of this work:

- Learn 3D shape abstraction from raw 3D point clouds / meshes

3D Shape Abstraction

Goal of this work:

- Learn 3D shape abstraction from raw 3D point clouds / meshes
- Infer variable number of primitives

3D Shape Abstraction

Goal of this work:

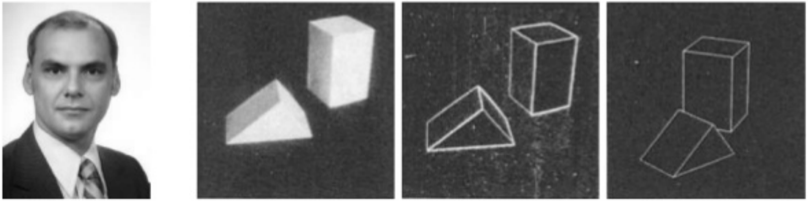
- Learn 3D shape abstraction from raw 3D point clouds / meshes
- Infer variable number of primitives
- No supervision at primitive level

3D Shape Abstraction

Goal of this work:

- Learn 3D shape abstraction from raw 3D point clouds / meshes
- Infer variable number of primitives
- No supervision at primitive level
- Infer from point clouds or images

1963: 3D Solids



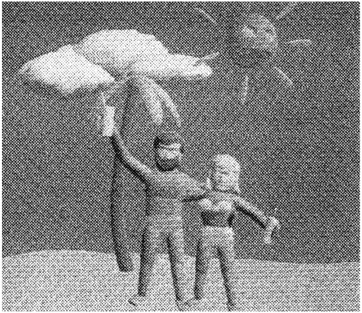
Larry Roberts
"Father of Computer Vision"

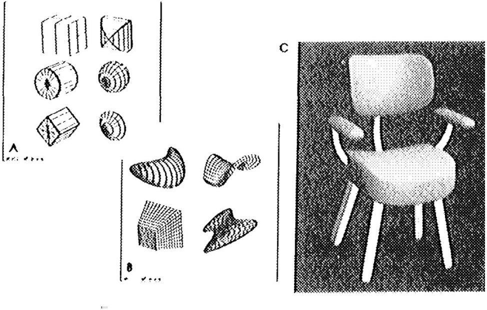
Input image

2x2 gradient operator

computed 3D model
rendered from new viewpoint

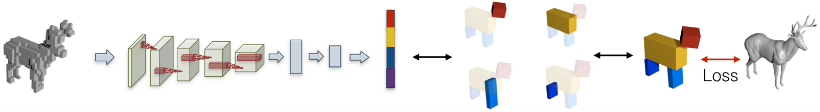
1986: Pentland's Superquadrics





- 1 superquadric can be represented with 11 parameters
- Scene on the left **contructed with 100 primitives** required less than 1000 bytes!
- Early fitting-based approaches did not work robustly

2017: 3D Reconstructions with Volumetric Primitives

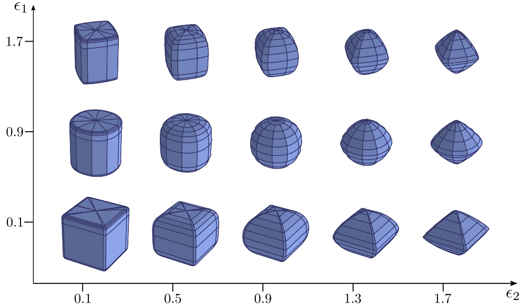


- **Unsupervised** method for learning **cuboidal primitives**
- **Variable number of primitives**
- While **cuboids are sufficient for capturing the structure** of an object they **do not lead to expressive abstractions**.
- Computational expensive reinforcement learning for learning the existence probabilities

Tulsiani: Learning Shape Abstractions by Assembling Volumetric Primitives. CVPR, 2017.

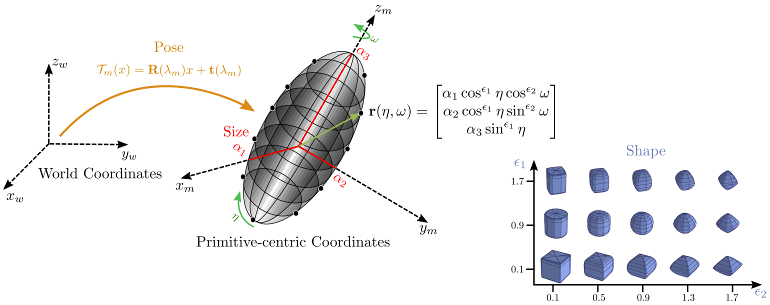
Can we train a network to output superquadrics?

*Everything in nature takes its form from the **sphere**, the **cone** and the **cylinder**.* - Paul Cezanne.



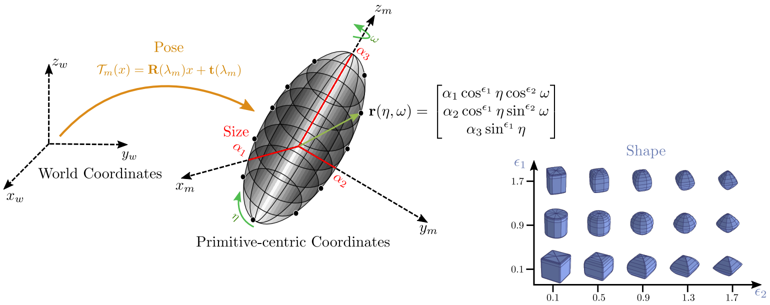
Superquadrics Space Shape

Superquadrics as geometric primitives



Their chief advantage is that they ***allow complex solids and surfaces to be constructed and altered easily from a few interactive parameters. [Barr 1981]***

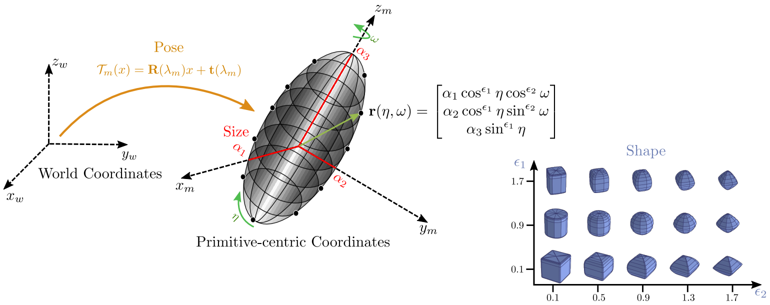
Superquadrics as geometric primitives



Their chief advantage is that they ***allow complex solids and surfaces to be constructed and altered easily from a few interactive parameters. [Barr 1981]***

- Fully described with just 11 parameters

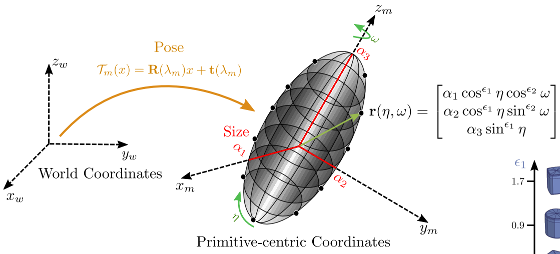
Superquadrics as geometric primitives

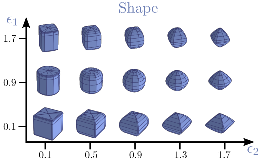


Their chief advantage is that they ***allow complex solids and surfaces to be constructed and altered easily from a few interactive parameters. [Barr 1981]***

- Fully described with just 11 parameters
- Represent a diverse class of shapes such as cylinders, spheres, cuboids, ellipsoids in a **single continuous parameter space**

Superquadrics as geometric primitives

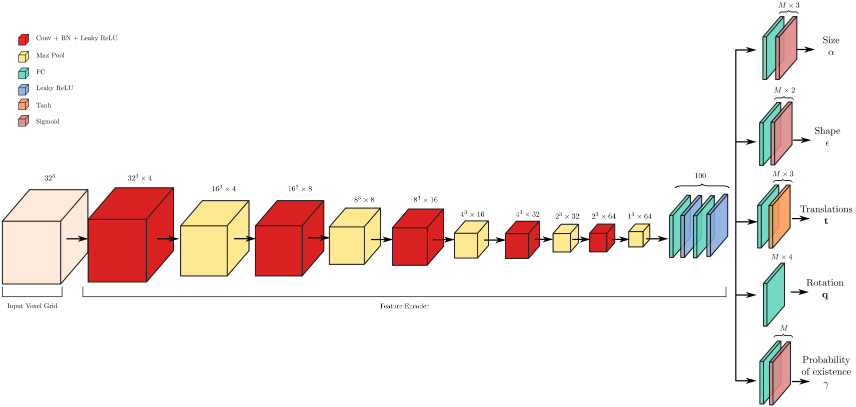




Their chief advantage is that they ***allow complex solids and surfaces to be constructed and altered easily from a few interactive parameters. [Barr 1981]***

- Fully described with just 11 parameters
- Represent a diverse class of shapes such as cylinders, spheres, cuboids, ellipsoids in a **single continuous parameter space**
- Their large shape vocabulary allows for **faster** and **smoother fitting** than cuboids

Learning 3D Shape Parsing



Neural network encodes input image/shape and **for each primitive** predicts:

- 11 parameters: 6 pose $(\mathbf{R}, \mathbf{t})$ + 3 scale $(\boldsymbol{\alpha})$ + 2 shape $(\boldsymbol{\epsilon})$
- Probability of existence: $\gamma \in [0, 1]$

Loss Function

Overall Loss:

$$\mathcal{L}(\mathbf{P}, \mathbf{X}) = \mathcal{L}_{P \to X}(\mathbf{P}, \mathbf{X}) + \mathcal{L}_{X \to P}(\mathbf{X}, \mathbf{P}) + \mathcal{L}_{\gamma}(\mathbf{P})$$

Composed of:

- $\mathcal{L}_{P \to X}(\mathbf{P}, \mathbf{X})$: Primitive-to-Pointcloud Loss
- $\mathcal{L}_{X \to P}(\mathbf{X}, \mathbf{P})$: Pointcloud-to-Primitive Loss
- $\mathcal{L}_{\gamma}(\mathbf{P})$: Existence and Parsimony Loss

Loss Function

Overall Loss:

$$\mathcal{L}(\mathbf{P}, \mathbf{X}) = \mathcal{L}_{P \to X}(\mathbf{P}, \mathbf{X}) + \mathcal{L}_{X \to P}(\mathbf{X}, \mathbf{P}) + \mathcal{L}_{\gamma}(\mathbf{P})$$

Composed of:

- $\mathcal{L}_{P \to X}(\mathbf{P}, \mathbf{X})$: Primitive-to-Pointcloud Loss
- $\mathcal{L}_{X \to P}(\mathbf{X}, \mathbf{P})$: Pointcloud-to-Primitive Loss
- $\mathcal{L}_{\gamma}(\mathbf{P})$: Existence and Parsimony Loss

Target and Predicted Shape:

- **Target:** $\mathbf{X} = \{x_i\}_{i=1}^{N}$

Loss Function

Overall Loss:

$$\mathcal{L}(\mathbf{P}, \mathbf{X}) = \mathcal{L}_{P \to X}(\mathbf{P}, \mathbf{X}) + \mathcal{L}_{X \to P}(\mathbf{X}, \mathbf{P}) + \mathcal{L}_{\gamma}(\mathbf{P})$$

Composed of:

- $\mathcal{L}_{P \to X}(\mathbf{P}, \mathbf{X})$: Primitive-to-Pointcloud Loss
- $\mathcal{L}_{X \to P}(\mathbf{X}, \mathbf{P})$: Pointcloud-to-Primitive Loss
- $\mathcal{L}_{\gamma}(\mathbf{P})$: Existence and Parsimony Loss

Target and Predicted Shape:

- **Target:** $\mathbf{X} = \{x_i\}_{i=1}^{N}$
- **Predicted:** $\mathbf{P} = \{(\lambda_m, \gamma_m)\}_{m=1}^{M}$

Loss Function

Overall Loss:

$$\mathcal{L}(\mathbf{P}, \mathbf{X}) = \mathcal{L}_{P \to X}(\mathbf{P}, \mathbf{X}) + \mathcal{L}_{X \to P}(\mathbf{X}, \mathbf{P}) + \mathcal{L}_{\gamma}(\mathbf{P})$$

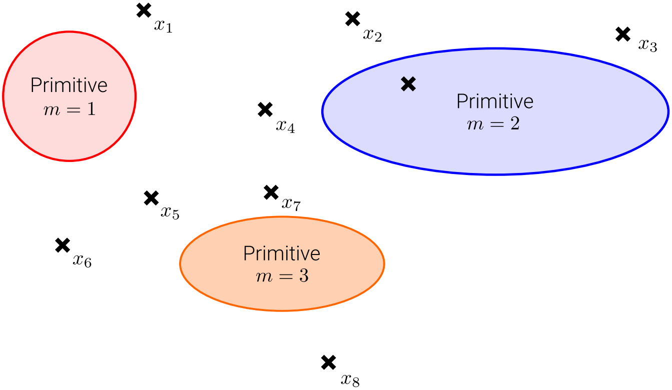
Composed of:

- $\mathcal{L}_{P \to X}(\mathbf{P}, \mathbf{X})$: Primitive-to-Pointcloud Loss
- $\mathcal{L}_{X \to P}(\mathbf{X}, \mathbf{P})$: Pointcloud-to-Primitive Loss
- $\mathcal{L}_{\gamma}(\mathbf{P})$: Existence and Parsimony Loss

Target and Predicted Shape:

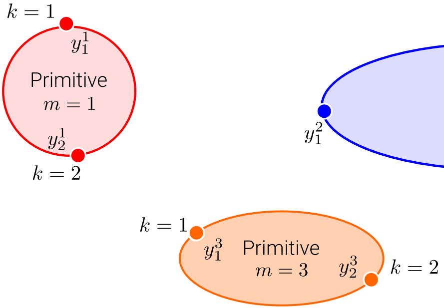
- **Target:** $\mathbf{X} = \{x_i\}_{i=1}^{N}$
- **Predicted:** $\mathbf{P} = \{(\lambda_m, \gamma_m)\}_{m=1}^{M}$
- **m-th primitive:** $\mathbf{Y}_m = \{y_k^m\}_{k=1}^{K}$

Loss Function



Target shape: $\mathbf{X} = \{x_i\}_{i=1}^N$

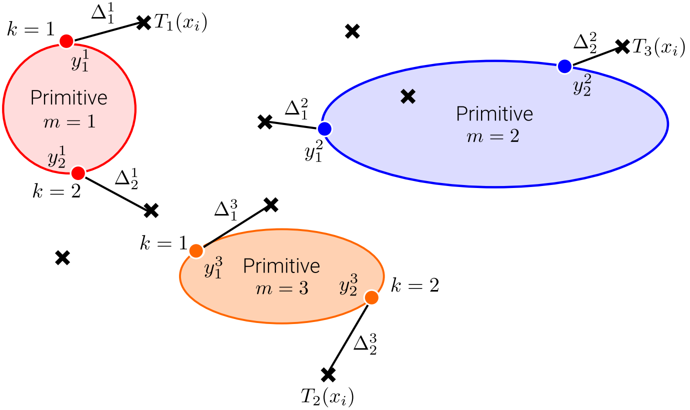
Loss Function



Target shape: $\mathbf{X} = \{x_i\}_{i=1}^N$

m-th primitive: $\mathbf{Y}_m = \{y_k^m\}_{k=1}^K$

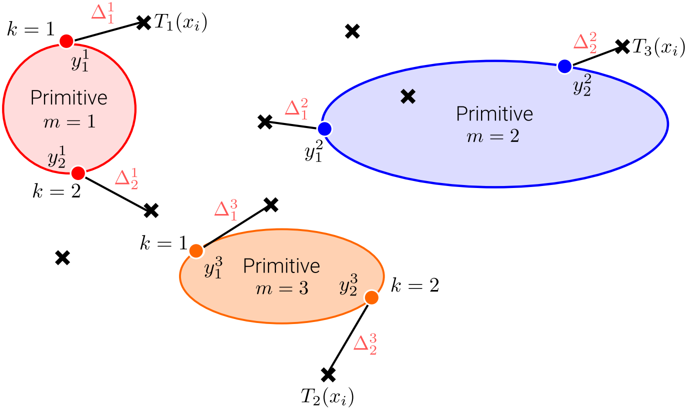
Primitive-to-Pointcloud Loss



$$\mathcal{L}_{P\rightarrow X}^{m}(\mathbf{P},\mathbf{X}) = \frac{1}{K}\sum_{k=1}^{K} \Delta_k^m$$

$$\Delta_k^m = \min_{i=1,..,N} \|\mathcal{T}_m(\mathbf{x}_i) - \mathbf{y}_k^m\|_2$$

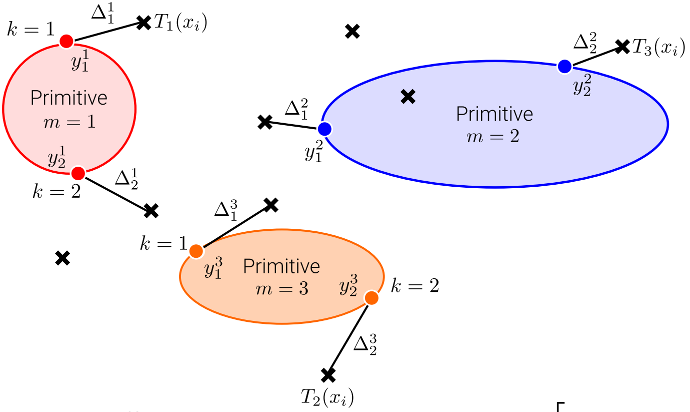
Primitive-to-Pointcloud Loss



$$\mathcal{L}^m_{P \to X}(\mathbf{P}, \mathbf{X}) = \frac{1}{K} \sum_{k=1}^{K} \Delta_k^m$$

$$\Delta_k^m = \min_{i=1,..,N} \|\mathcal{T}_m(\mathbf{x}_i) - \mathbf{y}_k^m\|_2$$

Primitive-to-Pointcloud Loss

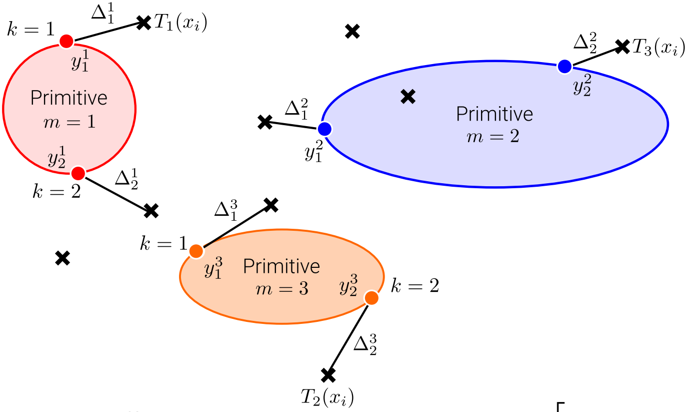


$$\mathcal{L}_{P\rightarrow X}^{m}(\mathbf{P},\mathbf{X}) = \frac{1}{K}\sum_{k=1}^{K} \Delta_k^m$$

$$\Delta_k^m = \min_{i=1,..,N} \|\mathcal{T}_m(\mathbf{x}_i) - \mathbf{y}_k^m\|_2$$

$$\mathcal{L}_{P\rightarrow X}(\mathbf{P},\mathbf{X}) = \mathbb{E}_{p(\mathbf{z})}\left[\sum_{m|z_m=1} \mathcal{L}_{P\rightarrow X}^{m}(\mathbf{P},\mathbf{X})\right] = \sum_{m=1}^{M} \gamma_m \,\mathcal{L}_{P\rightarrow X}^{m}(\mathbf{P},\mathbf{X})$$

Primitive-to-Pointcloud Loss



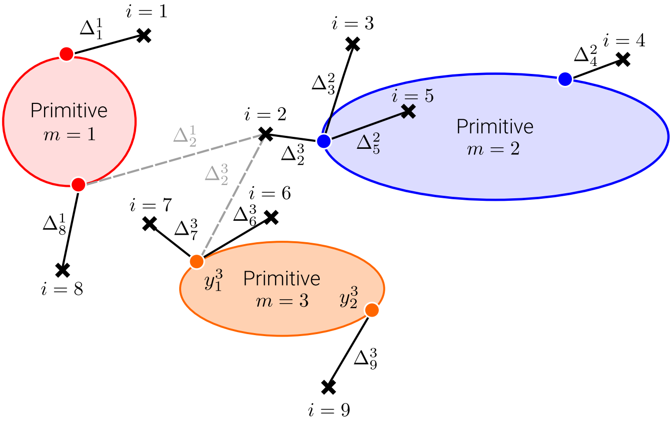
$$\mathcal{L}^m_{P\rightarrow X}(\mathbf{P},\mathbf{X}) = \frac{1}{K}\sum_{k=1}^{K} \Delta_k^m$$

$$\Delta_k^m = \min_{i=1,..,N} \|\mathcal{T}_m(\mathbf{x}_i) - \mathbf{y}_k^m\|_2$$

$$\mathcal{L}_{P\rightarrow X}(\mathbf{P},\mathbf{X}) = \mathbb{E}_{p(\mathbf{z})}\left[\sum_{m|z_m=1} \mathcal{L}^m_{P\rightarrow X}(\mathbf{P},\mathbf{X})\right]$$

$$= \sum_{m=1}^{M} \gamma_m \, \mathcal{L}^m_{P\rightarrow X}(\mathbf{P},\mathbf{X})$$

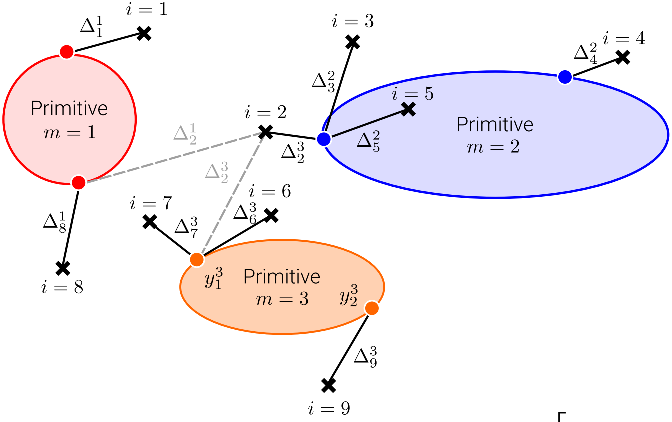
Pointcloud-to-Primitive Loss



$$\mathcal{L}^i_{X\to P}(\mathbf{X},\mathbf{P}) = \min_{m|z_m=1} \Delta_i^m$$

$$\Delta_i^m = \min_{k=1,..,K} \|\mathcal{T}_m(\mathbf{x}_i) - \mathbf{y}_k^m\|_2$$

Pointcloud-to-Primitive Loss

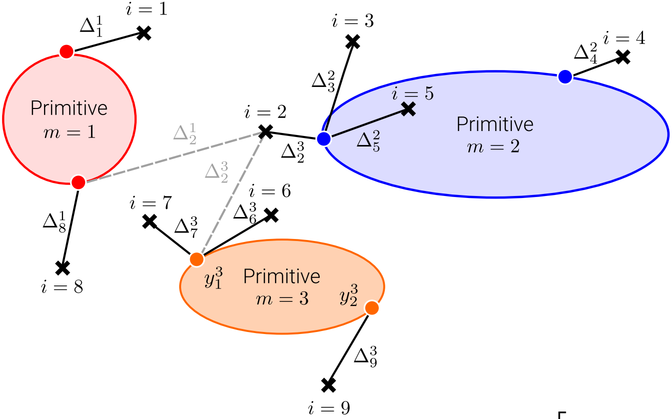


$$\mathcal{L}^i_{X\to P}(\mathbf{X},\mathbf{P}) = \min_{m|z_m=1} \Delta_i^m$$

$$\Delta_i^m = \min_{k=1,..,K} \|\mathcal{T}_m(\mathbf{x}_i) - \mathbf{y}_k^m\|_2$$

$$\mathcal{L}_{X\to P}(\mathbf{X},\mathbf{P}) = \mathbb{E}_{p(\mathbf{z})}\left[\sum_{\mathbf{x}_i\in\mathbf{X}} \mathcal{L}^i_{X\to P}(\mathbf{X},\mathbf{P})\right]$$

Pointcloud-to-Primitive Loss



$$\mathcal{L}^i_{X \to P}(\mathbf{X}, \mathbf{P}) = \min_{m | z_m = 1} \Delta_i^m$$

$$\Delta_i^m = \min_{k=1,..,K} \|\mathcal{T}_m(\mathbf{x}_i) - \mathbf{y}_k^m\|_2$$

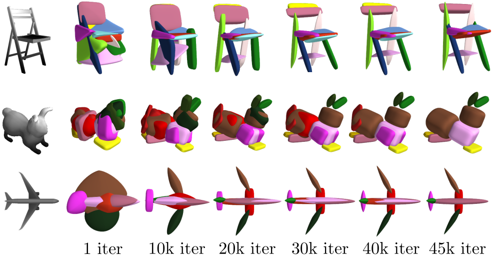
$$\mathcal{L}_{X \to P}(\mathbf{X}, \mathbf{P}) = \mathbb{E}_{p(\mathbf{z})} \left[\sum_{\mathbf{x}_i \in \mathbf{X}} \mathcal{L}^i_{X \to P}(\mathbf{X}, \mathbf{P}) \right]$$

$$= \sum_{\mathbf{x}_i \in \mathbf{X}} \sum_{m=1}^{M} \Delta_i^m \, \gamma_m \prod_{\bar{m}=1}^{m-1} (1 - \gamma_{\bar{m}})$$

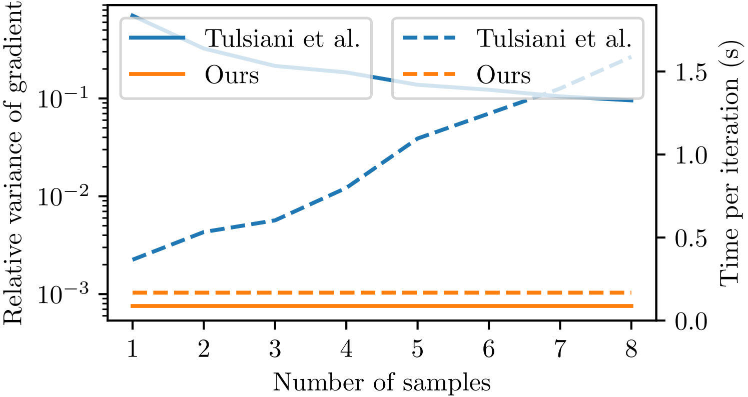
Existence and Parsimony Loss

$$\mathcal{L}_\gamma(\mathbf{P}) = \max\left(1 - \sum_{m=1}^{M} \gamma_m, 0\right) + \beta \sqrt{\sum_{m=1}^{M} \gamma_m}$$

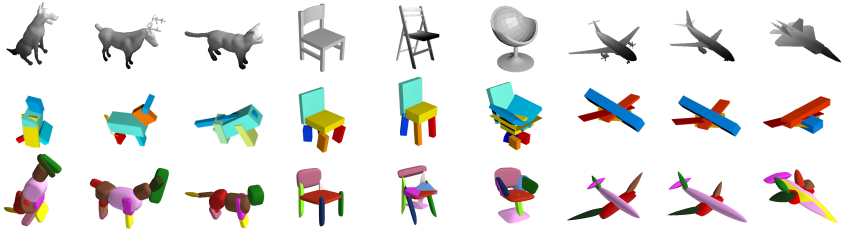
- **First term**: Enforces at least one primitive to exist
- **Second term**: Encourages parsimony
- Two-stage training



Comparison to Tulsiani et. al. / REINFORCE



Single view 3D Reconstruction on ShapeNet



	Chamfer Distance			Volumetric IoU		
	Chairs	Aeroplanes	Animals	Chairs	Aeroplanes	Animals
Cuboids	0.0121	0.0153	0.0110	0.1288	0.0650	0.3339
Superquadrics	**0.0006**	**0.0003**	**0.0003**	**0.1408**	**0.1808**	**0.7506**

Single view 3D Reconstruction on SURREAL

3D Shape Abstractions with Superquadrics

Limitations:

3D Shape Abstractions with Superquadrics

Limitations:

- Trade-off between number of primitives and representation accuaracy

3D Shape Abstractions with Superquadrics

Limitations:

- Trade-off between number of primitives and representation accuaracy
- Two-stage training to get a variable number of primitives

3D Shape Abstractions with Superquadrics

Limitations:

- Trade-off between number of primitives and representation accuaracy
- Two-stage training to get a variable number of primitives
- Bidirectional reconstruction loss suffers from various local minima

3D Shape Abstractions with Superquadrics

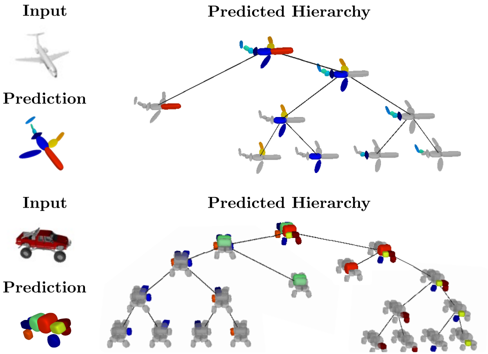
Limitations:

- Trade-off between number of primitives and representation accuaracy
- Two-stage training to get a variable number of primitives
- Bidirectional reconstruction loss suffers from various local minima
- Superquadrics :-)

Learning Unsupervised Hierarchical Part Decomposition of 3D Objects from a Single RGB Image

Despoina Paschalidou, Luc van Gool, Andreas Geiger

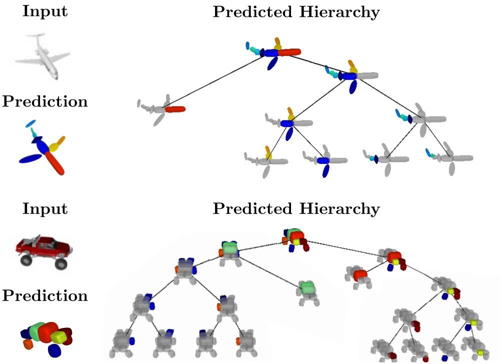
CVPR 2020



https://superquadrics.com/hierarchical-primitives.html

Hierarchical Part Decomposition

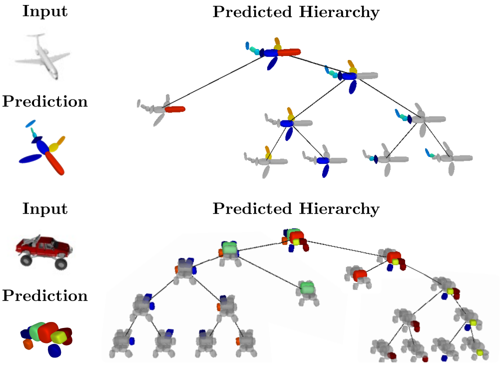
Goal of this work:



Hierarchical Part Decomposition

Goal of this work:

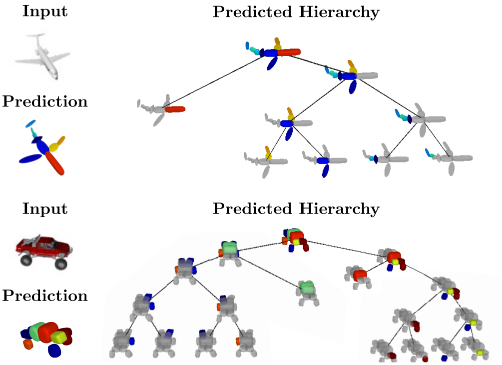
- Model relationships between parts



Hierarchical Part Decomposition

Goal of this work:

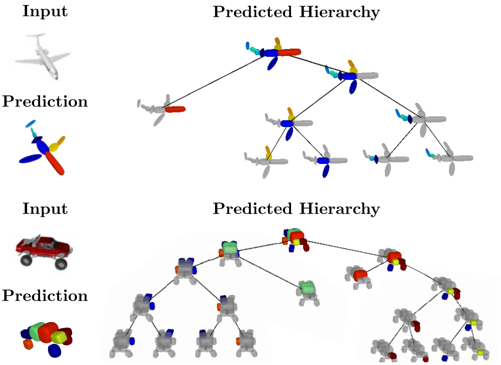
- Model relationships between parts
- Model objects with multiple levels of abstraction



Hierarchical Part Decomposition

Goal of this work:

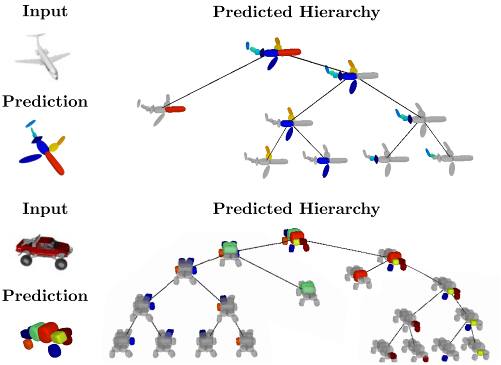
- Model relationships between parts
- Model objects with multiple levels of abstraction
- Infer variable number of primitives



Hierarchical Part Decomposition

Goal of this work:

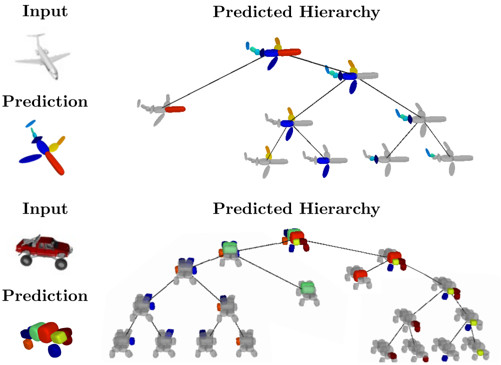
- Model relationships between parts
- Model objects with multiple levels of abstraction
- Infer variable number of primitives
- No supervision at primitive level and part relations



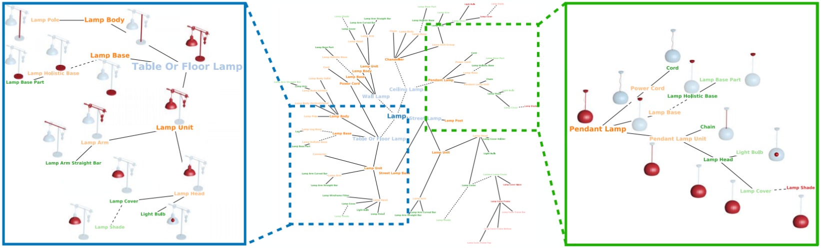
Hierarchical Part Decomposition

Goal of this work:

- Model relationships between parts
- Model objects with multiple levels of abstraction
- Infer variable number of primitives
- No supervision at primitive level and part relations
- Improve reconstruction quality while retaining semanticness

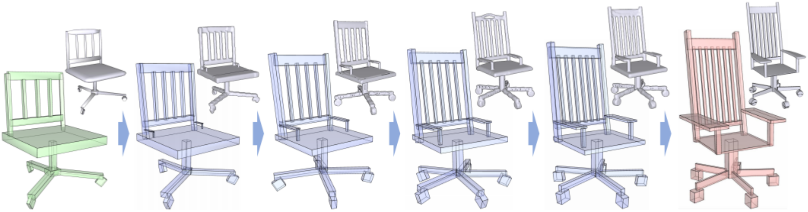


Supervised Structure-Aware Representations



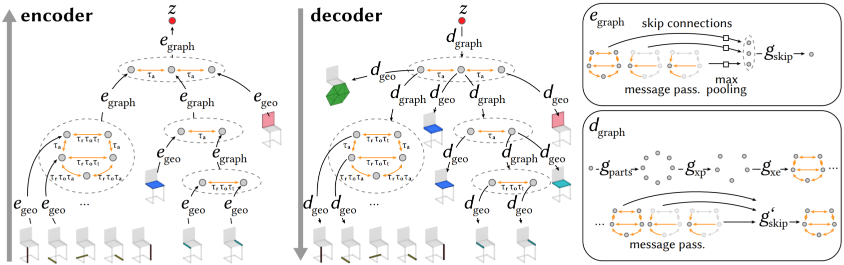
- Large-scale dataset of 3D objects annotated with fine-grained, instance-level, and hierarchical 3D part information

Supervised Structure-Aware Representations



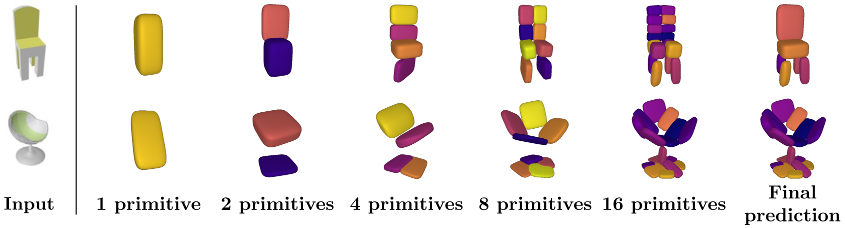
- Represent shapes using a symmetry hierarchy
- Learn a hierarchical organization of bounding boxes and then fills them with voxelized parts.

Supervised Structure-Aware Representations



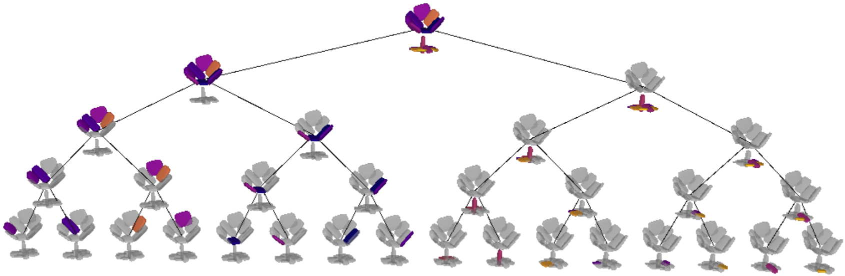
- Represent shapes as a hierarchy of n-ary graphs
- Requires supervision in terms of the primitive parameters and the hierarchies

Representation with multiple levels of abstraction



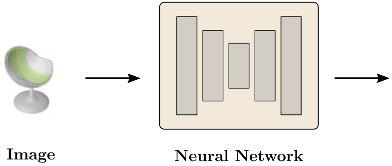
- Represent a 3D shape as a **binary tree of primitives**
- At each depth level, each node is **recursively** split into two until reaching the maximum depth
- Reconstructions from deeper depth levels are more detailed

Representation with multiple levels of abstraction



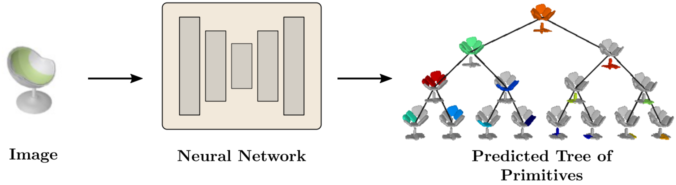
- Represent a 3D shape as a **binary tree of primitives**
- At each depth level, each node is **recursively** split into two until reaching the maximum depth
- Reconstructions from deeper depth levels are more detailed

Learning Hierarchical Part Decomposition of 3D Objects



Target and Predicted Shape:

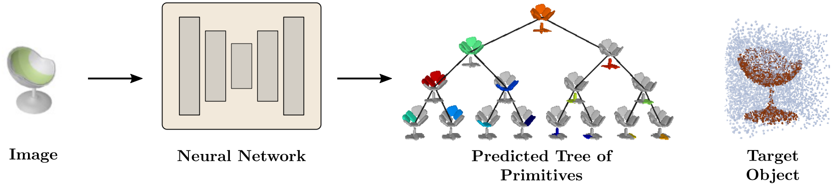
Learning Hierarchical Part Decomposition of 3D Objects



Target and Predicted Shape:

- **Binary Tree of Primitives:** $\mathcal{P} = \{\{p_k^d\}_{k=0}^{2^d-1} \mid d = \{0 \ldots D\}\}$

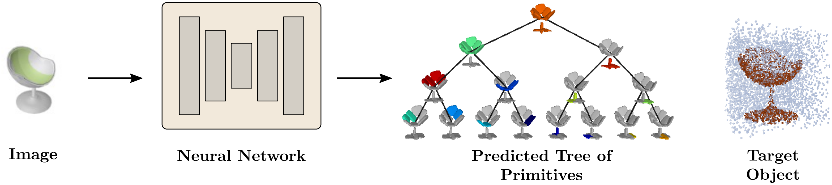
Learning Hierarchical Part Decomposition of 3D Objects



Target and Predicted Shape:

- **Binary Tree of Primitives:** $\mathcal{P} = \{\{p_k^d\}_{k=0}^{2^d-1} \mid d = \{0 \dots D\}\}$
- **Target:** Set of occupancy pairs $\mathcal{X} = \{(\mathbf{x}_i, o_i)\}_{i=1}^N$

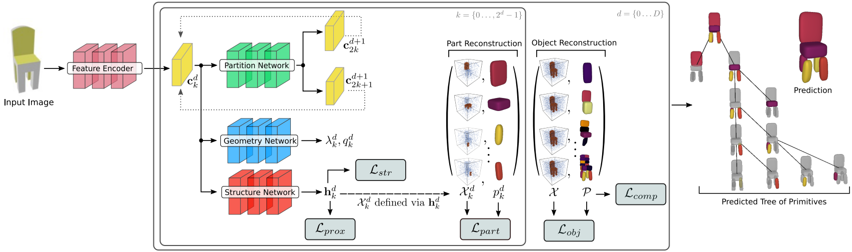
Learning Hierarchical Part Decomposition of 3D Objects



Target and Predicted Shape:

- **Binary Tree of Primitives:** $\mathcal{P} = \{\{p_k^d\}_{k=0}^{2^d-1} \mid d = \{0 \dots D\}\}$
- **Target:** Set of occupancy pairs $\mathcal{X} = \{(\mathbf{x}_i, o_i)\}_{i=1}^N$
- **Occupancy function of predicted at depth d :** $G^d(\mathbf{x}) = \max_{k \in 0 \dots 2^d-1} g_k^d\left(\mathbf{x}; \lambda_k^d\right)$

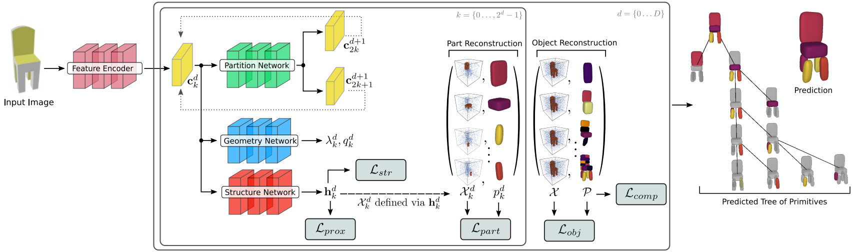
Learning Hierarchical Part Decomposition of 3D Objects



Neural network encodes input image/shape and **for each primitive** predicts:

- 11 parameters: 6 pose $(\mathbf{R}, \mathbf{t})$ + 3 scale $(\boldsymbol{\alpha})$ + 2 shape $(\boldsymbol{\epsilon})$
- Reconstruction quality: $q_k^d \in [0, 1]$

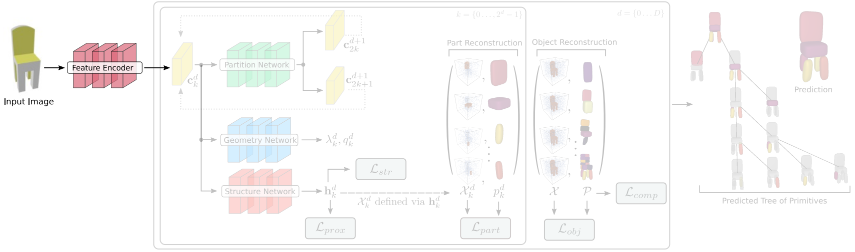
Learning Hierarchical Part Decomposition of 3D Objects



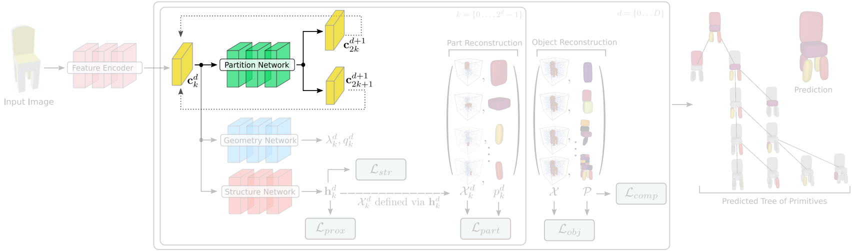
Components:

- Feature Encoder
- Partition Network
- Geometry Network
- Structure Network

Learning Hierarchical Part Decomposition of 3D Objects



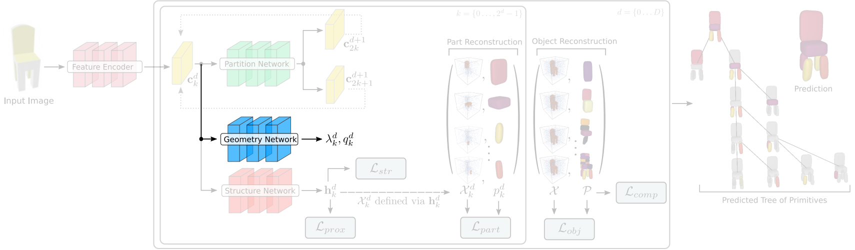
Learning Hierarchical Part Decomposition of 3D Objects



Partition Network: Recursively partition the **feature representation**

$$p_\theta(\mathbf{c}_k^d) = \{\mathbf{c}_{2k}^{d+1}, \mathbf{c}_{2k+1}^{d+1}\}$$

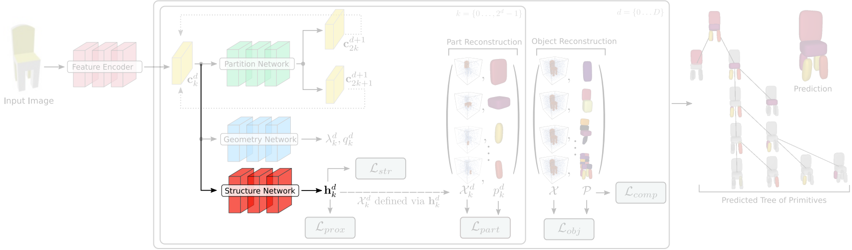
Learning Hierarchical Part Decomposition of 3D Objects



Geometry Network: Regress the primitive parameters

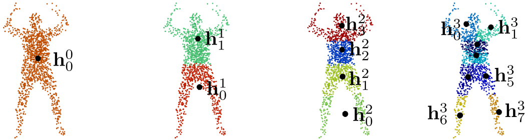
$$r_\theta(\mathbf{c}_k^d) = \{\lambda_k^d, q_k^d\}.$$

Learning Hierarchical Part Decomposition of 3D Objects



Structure Network: Assign object parts to primitives

$$\mathcal{H} = \{\{\mathbf{h}_k^d\}_{k=0}^{2^d-1} \mid d = \{0 \dots D\}\}$$



Loss Function

Overall Loss:

$$\mathcal{L}(\mathcal{P}, \mathcal{H}; \mathcal{X}) = \mathcal{L}_{str}(\mathcal{H}; \mathcal{X}) + \mathcal{L}_{rec}(\mathcal{P}; \mathcal{X}) + \mathcal{L}_{comp}(\mathcal{P}; \mathcal{X}) + \mathcal{L}_{prox}(\mathcal{P})$$

Composed of:

- $\mathcal{L}_{str}(\mathcal{H}, \mathcal{X})$: Structure Loss
- $\mathcal{L}_{rec}(\mathcal{P}, \mathcal{X})$: Reconstruction Loss
- $\mathcal{L}_{comp}(\mathcal{P}, \mathcal{X})$: Combatibility Loss
- $\mathcal{L}_{prox}(\mathcal{P})$: Proximity Loss

Loss Function

Overall Loss:

$$\mathcal{L}(\mathcal{P}, \mathcal{H}; \mathcal{X}) = \mathcal{L}_{str}(\mathcal{H}; \mathcal{X}) + \mathcal{L}_{rec}(\mathcal{P}; \mathcal{X}) + \mathcal{L}_{comp}(\mathcal{P}; \mathcal{X}) + \mathcal{L}_{prox}(\mathcal{P})$$

Composed of:

- $\mathcal{L}_{str}(\mathcal{H}, \mathcal{X})$: Structure Loss
- $\mathcal{L}_{rec}(\mathcal{P}, \mathcal{X})$: Reconstruction Loss
- $\mathcal{L}_{comp}(\mathcal{P}, \mathcal{X})$: Combatibility Loss
- $\mathcal{L}_{prox}(\mathcal{P})$: Proximity Loss

Target and Predicted Shape:

- **Target:** $\mathcal{X} = \{(\mathbf{x}_i, o_i)\}_{i=1}^{N}$

Loss Function

Overall Loss:

$$\mathcal{L}(\mathcal{P}, \mathcal{H}; \mathcal{X}) = \mathcal{L}_{str}(\mathcal{H}; \mathcal{X}) + \mathcal{L}_{rec}(\mathcal{P}; \mathcal{X}) + \mathcal{L}_{comp}(\mathcal{P}; \mathcal{X}) + \mathcal{L}_{prox}(\mathcal{P})$$

Composed of:

- $\mathcal{L}_{str}(\mathcal{H}, \mathcal{X})$: Structure Loss
- $\mathcal{L}_{rec}(\mathcal{P}, \mathcal{X})$: Reconstruction Loss
- $\mathcal{L}_{comp}(\mathcal{P}, \mathcal{X})$: Combatibility Loss
- $\mathcal{L}_{prox}(\mathcal{P})$: Proximity Loss

Target and Predicted Shape:

- **Target:** $\mathcal{X} = \{(\mathbf{x}_i, o_i)\}_{i=1}^{N}$
- **Binary Tree of Primitives:** $\mathcal{P} = \{\{p_k^d\}_{k=0}^{2^d-1} \mid d = \{0 \dots D\}\}$

Loss Function

Overall Loss:

$$\mathcal{L}(\mathcal{P}, \mathcal{H}; \mathcal{X}) = \mathcal{L}_{str}(\mathcal{H}; \mathcal{X}) + \mathcal{L}_{rec}(\mathcal{P}; \mathcal{X}) + \mathcal{L}_{comp}(\mathcal{P}; \mathcal{X}) + \mathcal{L}_{prox}(\mathcal{P})$$

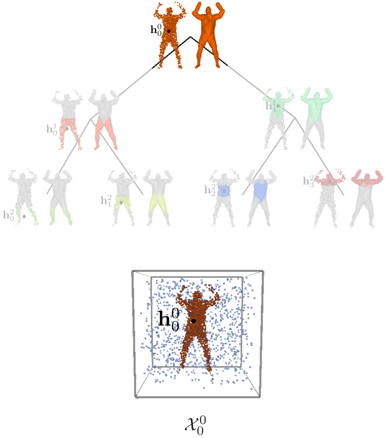
Composed of:

- $\mathcal{L}_{str}(\mathcal{H}, \mathcal{X})$: Structure Loss
- $\mathcal{L}_{rec}(\mathcal{P}, \mathcal{X})$: Reconstruction Loss
- $\mathcal{L}_{comp}(\mathcal{P}, \mathcal{X})$: Combatibility Loss
- $\mathcal{L}_{prox}(\mathcal{P})$: Proximity Loss

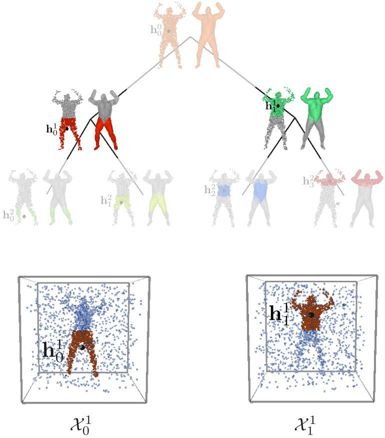
Target and Predicted Shape:

- **Target:** $\mathcal{X} = \{(\mathbf{x}_i, o_i)\}_{i=1}^{N}$
- **Binary Tree of Primitives:** $\mathcal{P} = \{\{p_k^d\}_{k=0}^{2^d-1} \mid d = \{0 \dots D\}\}$
- **Geometric Centroids:** $\mathcal{H} = \{\{\mathbf{h}_k^d\}_{k=0}^{2^d-1} \mid d = \{0 \dots D\}\}$

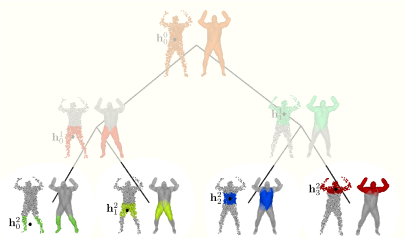
Structure Loss

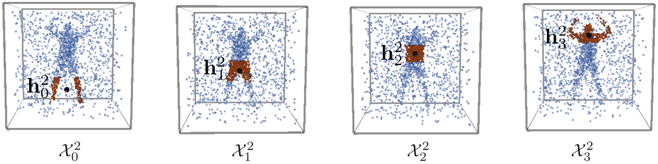


Structure Loss

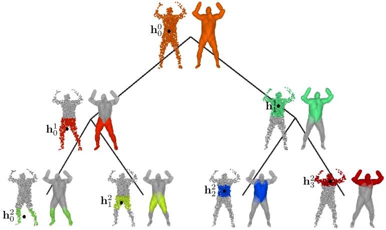


Structure Loss





Structure Loss



$$\mathcal{L}_{str}(\mathcal{H};\mathcal{X}) = \sum_{h_k^d \in \mathcal{H}} \frac{1}{2^d - 1} \sum_{(\mathbf{x}, o) \in \mathcal{X}_k^d} o \, \|\mathbf{x} - \mathbf{h}_k^d\|_2$$

Reconstruction Loss

$$\mathcal{L}_{rec}(\mathcal{P};\mathcal{X}) = \underbrace{\sum_{(\mathbf{x},o)\in\mathcal{X}} \sum_{d=0}^{D} L\left(G^d(\mathbf{x}), o\right)}_{\text{Object Reconstruction}} + \underbrace{\sum_{d=0}^{D} \sum_{k=0}^{2^d-1} \sum_{(\mathbf{x},o)\in\mathcal{X}_k^d} L\left(g_k^d\left(\mathbf{x};\lambda_k^d\right), o\right)}_{\text{Part Reconstruction}}$$

Reconstruction Loss

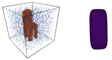
$$\mathcal{L}_{rec}(\mathcal{P};\mathcal{X}) = \underbrace{\sum_{(\mathbf{x},o)\in\mathcal{X}} \sum_{d=0}^{D} L\left(G^d(\mathbf{x}), o\right)}_{\text{Object Reconstruction}} + \underbrace{\sum_{d=0}^{D} \sum_{k=0}^{2^d-1} \sum_{(\mathbf{x},o)\in\mathcal{X}_k^d} L\left(g_k^d\left(\mathbf{x};\lambda_k^d\right), o\right)}_{\text{Part Reconstruction}}$$

Reconstruction Loss

$$\mathcal{L}_{rec}(\mathcal{P};\mathcal{X}) = \underbrace{\sum_{(\mathbf{x},o)\in\mathcal{X}} \sum_{d=0}^{D} L\left(G^d(\mathbf{x}), o\right)}_{\text{Object Reconstruction}} + \underbrace{\sum_{d=0}^{D}\sum_{k=0}^{2^d-1}\sum_{(\mathbf{x},o)\in\mathcal{X}_k^d} L\left(g_k^d\left(\mathbf{x};\lambda_k^d\right), o\right)}_{\text{Part Reconstruction}}$$

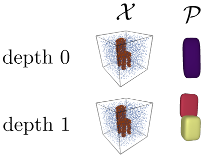
$\mathcal{X}$ $\mathcal{P}$

depth 0



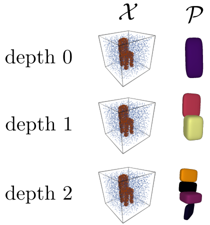
Reconstruction Loss

$$\mathcal{L}_{rec}(\mathcal{P};\mathcal{X}) = \underbrace{\sum_{(\mathbf{x},o)\in\mathcal{X}} \sum_{d=0}^{D} L\left(G^d(\mathbf{x}), o\right)}_{\text{Object Reconstruction}} + \underbrace{\sum_{d=0}^{D} \sum_{k=0}^{2^d-1} \sum_{(\mathbf{x},o)\in\mathcal{X}_k^d} L\left(g_k^d\left(\mathbf{x};\lambda_k^d\right), o\right)}_{\text{Part Reconstruction}}$$



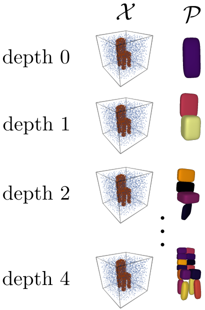
Reconstruction Loss

$$\mathcal{L}_{rec}(\mathcal{P};\mathcal{X}) = \underbrace{\sum_{(\mathbf{x},o)\in\mathcal{X}} \sum_{d=0}^{D} L\left(G^d(\mathbf{x}), o\right)}_{\text{Object Reconstruction}} + \underbrace{\sum_{d=0}^{D} \sum_{k=0}^{2^d-1} \sum_{(\mathbf{x},o)\in\mathcal{X}_k^d} L\left(g_k^d\left(\mathbf{x};\lambda_k^d\right), o\right)}_{\text{Part Reconstruction}}$$



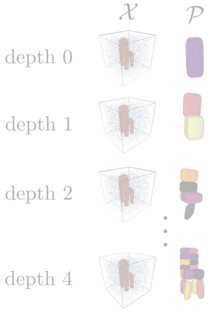
Reconstruction Loss

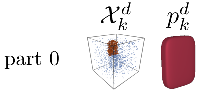
$$\mathcal{L}_{rec}(\mathcal{P};\mathcal{X}) = \underbrace{\sum_{(\mathbf{x},o)\in\mathcal{X}} \sum_{d=0}^{D} L\left(G^d(\mathbf{x}), o\right)}_{\text{Object Reconstruction}} + \underbrace{\sum_{d=0}^{D} \sum_{k=0}^{2^d-1} \sum_{(\mathbf{x},o)\in\mathcal{X}_k^d} L\left(g_k^d\left(\mathbf{x};\lambda_k^d\right), o\right)}_{\text{Part Reconstruction}}$$



Reconstruction Loss

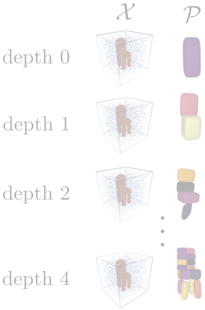
$$\mathcal{L}_{rec}(\mathcal{P};\mathcal{X}) = \underbrace{\sum_{(\mathbf{x},o)\in\mathcal{X}} \sum_{d=0}^{D} L\left(G^d(\mathbf{x}), o\right)}_{\text{Object Reconstruction}} + \underbrace{\sum_{d=0}^{D}\sum_{k=0}^{2^d-1}\sum_{(\mathbf{x},o)\in\mathcal{X}_k^d} L\left(g_k^d\left(\mathbf{x};\lambda_k^d\right), o\right)}_{\text{Part Reconstruction}}$$

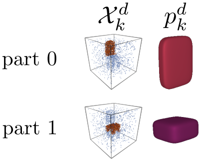




Reconstruction Loss

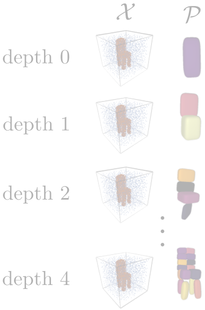
$$\mathcal{L}_{rec}(\mathcal{P};\mathcal{X}) = \underbrace{\sum_{(\mathbf{x},o)\in\mathcal{X}} \sum_{d=0}^{D} L\left(G^d(\mathbf{x}), o\right)}_{\text{Object Reconstruction}} + \underbrace{\sum_{d=0}^{D} \sum_{k=0}^{2^d-1} \sum_{(\mathbf{x},o)\in\mathcal{X}_k^d} L\left(g_k^d\left(\mathbf{x};\lambda_k^d\right), o\right)}_{\text{Part Reconstruction}}$$

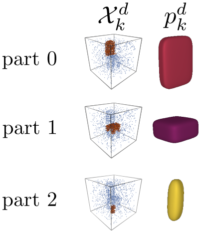




Reconstruction Loss

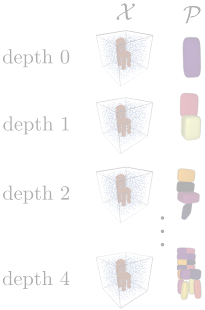
$$\mathcal{L}_{rec}(\mathcal{P};\mathcal{X}) = \underbrace{\sum_{(\mathbf{x},o)\in\mathcal{X}} \sum_{d=0}^{D} L\left(G^d(\mathbf{x}), o\right)}_{\text{Object Reconstruction}} + \underbrace{\sum_{d=0}^{D} \sum_{k=0}^{2^d-1} \sum_{(\mathbf{x},o)\in\mathcal{X}_k^d} L\left(g_k^d\left(\mathbf{x};\lambda_k^d\right), o\right)}_{\text{Part Reconstruction}}$$

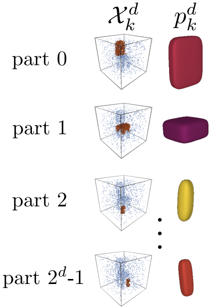




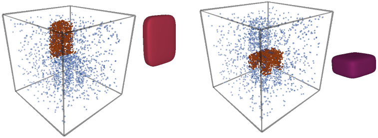
Reconstruction Loss

$$\mathcal{L}_{rec}(\mathcal{P};\mathcal{X}) = \underbrace{\sum_{(\mathbf{x},o)\in\mathcal{X}} \sum_{d=0}^{D} L\left(G^d(\mathbf{x}), o\right)}_{\text{Object Reconstruction}} + \underbrace{\sum_{d=0}^{D}\sum_{k=0}^{2^d-1}\sum_{(\mathbf{x},o)\in\mathcal{X}_k^d} L\left(g_k^d\left(\mathbf{x};\lambda_k^d\right), o\right)}_{\text{Part Reconstruction}}$$



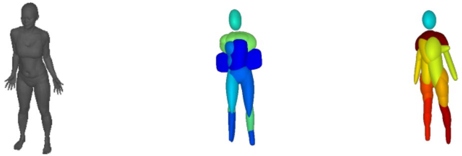


Compatibility Loss



$$\mathcal{L}_{comp}(\mathcal{P}) = \sum_{d=0}^{\mathcal{D}} \sum_{k=0}^{2^d-1} \left(q_k^d - \text{IoU}(p_k^d, \mathcal{X}_k^d) \right)^2$$

Proximity Loss



(a) **Input** (b) **without** (c) **Ours**

$$\mathcal{L}_{prox}(\mathcal{P}) = \sum_{d=0}^{D} \sum_{k=0}^{2^d-1} \|\mathbf{t}(\lambda_k^d) - \mathbf{h}_k^d\|_2$$

Loss Function

Overall Loss:

$$\mathcal{L}(\mathcal{P},\mathcal{H};\mathcal{X}) = \mathcal{L}_{str}(\mathcal{H};\mathcal{X}) + \mathcal{L}_{rec}(\mathcal{P};\mathcal{X}) + \mathcal{L}_{comp}(\mathcal{P};\mathcal{X}) + \mathcal{L}_{prox}(\mathcal{P})$$

Composed of:

Loss Function

Overall Loss:

$$\mathcal{L}(\mathcal{P}, \mathcal{H}; \mathcal{X}) = \mathcal{L}_{str}(\mathcal{H}; \mathcal{X}) + \mathcal{L}_{rec}(\mathcal{P}; \mathcal{X}) + \mathcal{L}_{comp}(\mathcal{P}; \mathcal{X}) + \mathcal{L}_{prox}(\mathcal{P})$$

Composed of:

- $\mathcal{L}_{str}(\mathcal{H}, \mathcal{X})$: Decomposes shape into parts

Loss Function

Overall Loss:

$$\mathcal{L}(\mathcal{P}, \mathcal{H}; \mathcal{X}) = \mathcal{L}_{str}(\mathcal{H}; \mathcal{X}) + \mathcal{L}_{rec}(\mathcal{P}; \mathcal{X}) + \mathcal{L}_{comp}(\mathcal{P}; \mathcal{X}) + \mathcal{L}_{prox}(\mathcal{P})$$

Composed of:

- $\mathcal{L}_{str}(\mathcal{H}, \mathcal{X})$: Decomposes shape into parts
- $\mathcal{L}_{rec}(\mathcal{P}, \mathcal{X})$: Predicted primitives match the shape

Loss Function

Overall Loss:

$$\mathcal{L}(\mathcal{P}, \mathcal{H}; \mathcal{X}) = \mathcal{L}_{str}(\mathcal{H}; \mathcal{X}) + \mathcal{L}_{rec}(\mathcal{P}; \mathcal{X}) + \mathcal{L}_{comp}(\mathcal{P}; \mathcal{X}) + \mathcal{L}_{prox}(\mathcal{P})$$

Composed of:

- $\mathcal{L}_{str}(\mathcal{H}, \mathcal{X})$: Decomposes shape into parts
- $\mathcal{L}_{rec}(\mathcal{P}, \mathcal{X})$: Predicted primitives match the shape
- $\mathcal{L}_{comp}(\mathcal{P}, \mathcal{X})$: Allows for variable number of primitives

Loss Function

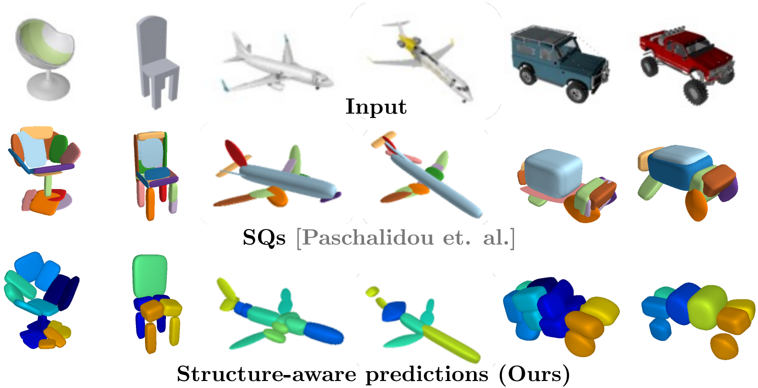
Overall Loss:

$$\mathcal{L}(\mathcal{P}, \mathcal{H}; \mathcal{X}) = \mathcal{L}_{str}(\mathcal{H}; \mathcal{X}) + \mathcal{L}_{rec}(\mathcal{P}; \mathcal{X}) + \mathcal{L}_{comp}(\mathcal{P}; \mathcal{X}) + \mathcal{L}_{prox}(\mathcal{P})$$

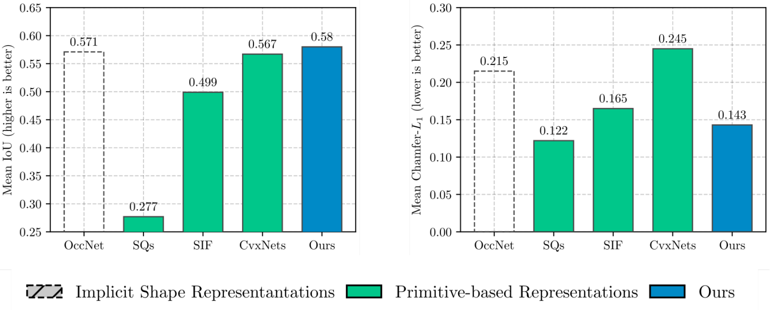
Composed of:

- $\mathcal{L}_{str}(\mathcal{H}, \mathcal{X})$: Decomposes shape into parts
- $\mathcal{L}_{rec}(\mathcal{P}, \mathcal{X})$: Predicted primitives match the shape
- $\mathcal{L}_{comp}(\mathcal{P}, \mathcal{X})$: Allows for variable number of primitives
- $\mathcal{L}_{prox}(\mathcal{P})$: Prevents vanishing gradients

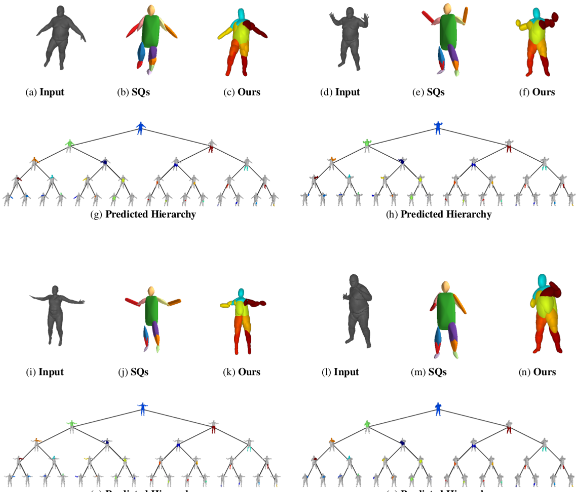
Single-view 3D Reconstruction on ShapeNet



Single-view 3D Reconstruction on ShapeNet



Single-view 3D Reconstruction on Dynamic FAUST



(a) **Input** (b) **SQs** (c) **Ours** (d) **Input** (e) **SQs** (f) **Ours**

(g) **Predicted Hierarchy** (h) **Predicted Hierarchy**

(i) **Input** (j) **SQs** (k) **Ours** (l) **Input** (m) **SQs** (n) **Ours**

(o) **Predicted Hierarchy** (p) **Predicted Hierarchy**

Learning Hierarchical Part Decomposition of 3D Objects

Limitations:

Learning Hierarchical Part Decomposition of 3D Objects

Limitations:

- Part decomposition does not guarantee semantic parts

Learning Hierarchical Part Decomposition of 3D Objects

Limitations:

- Part decomposition does not guarantee semantic parts
- Fixed maximum tree depth

Learning Hierarchical Part Decomposition of 3D Objects

Limitations:

- Part decomposition does not guarantee semantic parts
- Fixed maximum tree depth
- Occupancy loss (IoU) focuses less on fine details

Learning Hierarchical Part Decomposition of 3D Objects

Limitations:

- Part decomposition does not guarantee semantic parts
- Fixed maximum tree depth
- Occupancy loss (IoU) focuses less on fine details
- Superquadrics :-)

What comes next?

What comes next?

- Learning semantic parts
  - semanticness should not be enforced through geometry
  - consistency across pose and instances

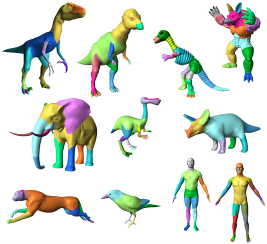


Image Source: Shapira 2008

What comes next?

- Learning semantic parts
  - semanticness should not be enforced through geometry
  - consistency across pose and instances
- Recovering higher level semantics
  - predict object dynamics, skeletons, joints, etc.
  - single RGB image is not sufficient

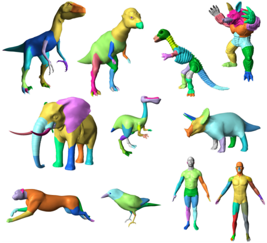
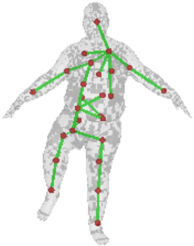


Image Source: Shapira 2008



What comes next?

- Learning semantic parts
  - semanticness should not be enforced through geometry
  - consistency across pose and instances
- Recovering higher level semantics
  - predict object dynamics, skeletons, joints, etc.
  - single RGB image is not sufficient
- More expressive primitives
  - trade-off between parsimony and geometrically accurate reconstruction

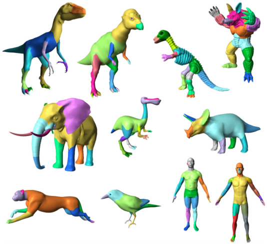
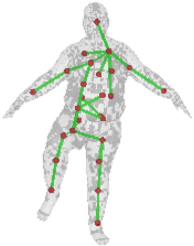


Image Source: Shapira 2008



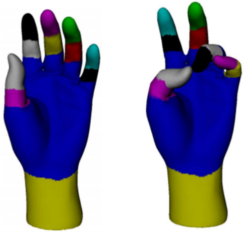


Image Source: Tierny 2007

Thank you for your attention!

https://superquadrics.com/