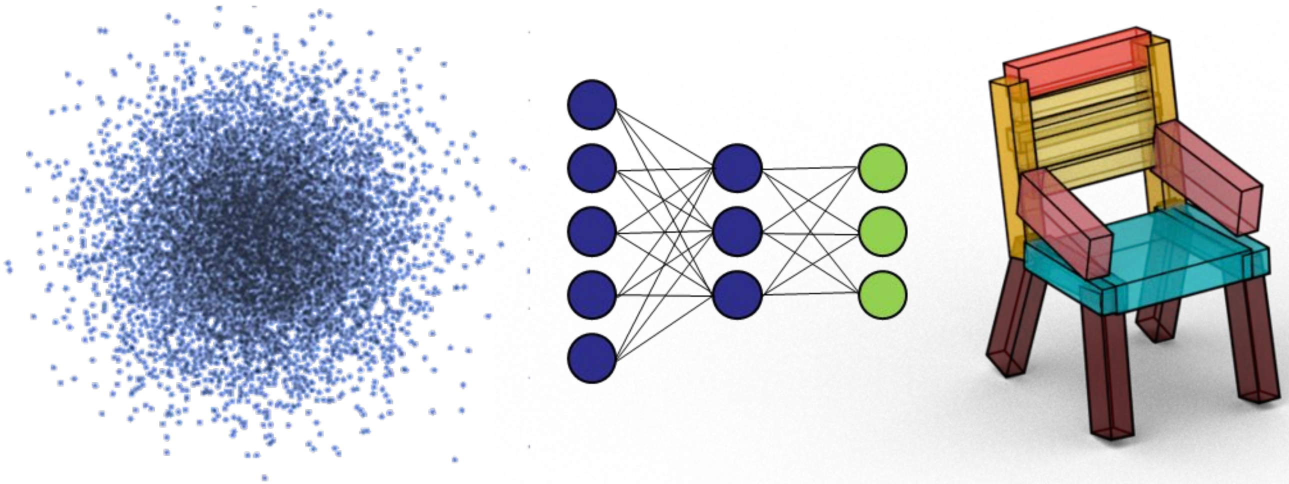
CS348n: Neural Representations and Generative Models for 3D Geometry



Leonidas Guibas
Computer Science Department
Stanford University

Leonidas Guibas Laboratory
Geometric Computing

Some Class Logistics

- Homework 2 is due today
- Homework 3 is out, due in two weeks
- Solutions to homework 1 will be sent out today

- Please take the questionnaire below to provide us with feedback on the class:
  - https://forms.gle/igFFpmnWaWL11Tfw9

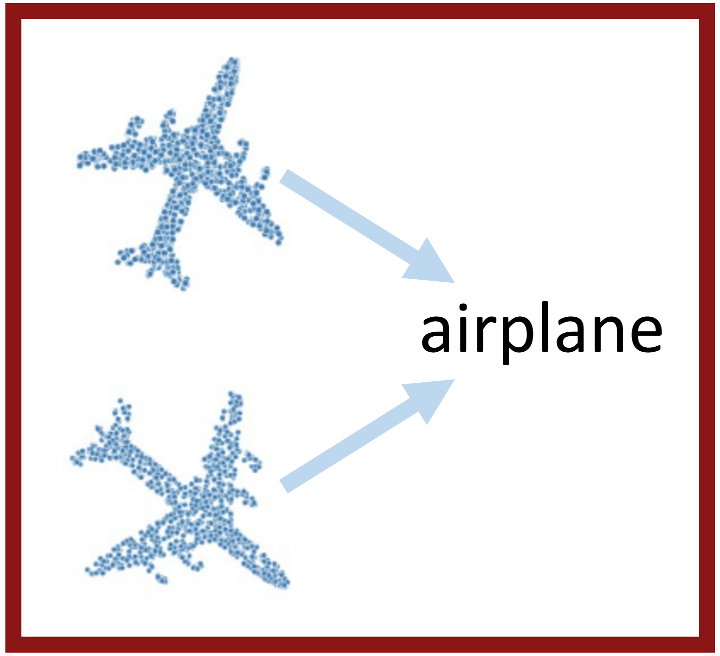
- Project proposals (1 page) are due next Wed

- The class will continue on Zoom next week

Last Time: Equivariance and Invariance

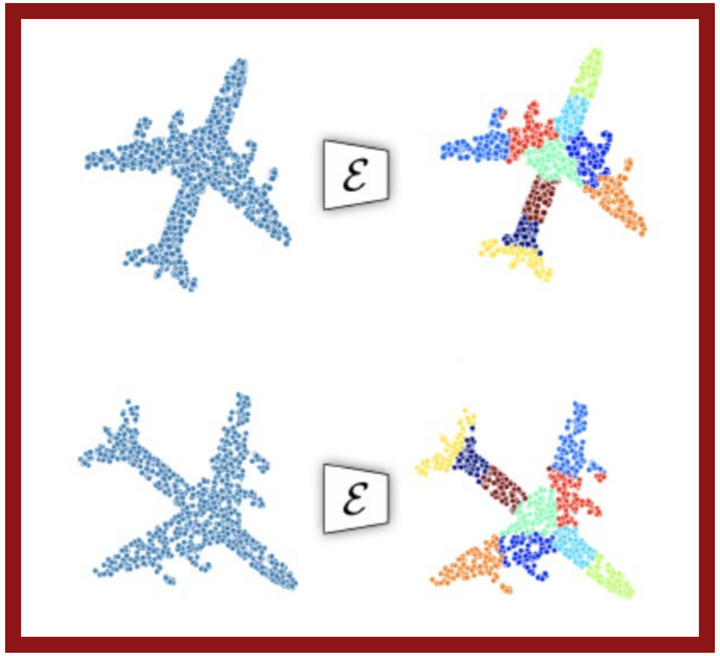
The Effect of Transformations on 3D Data

classification



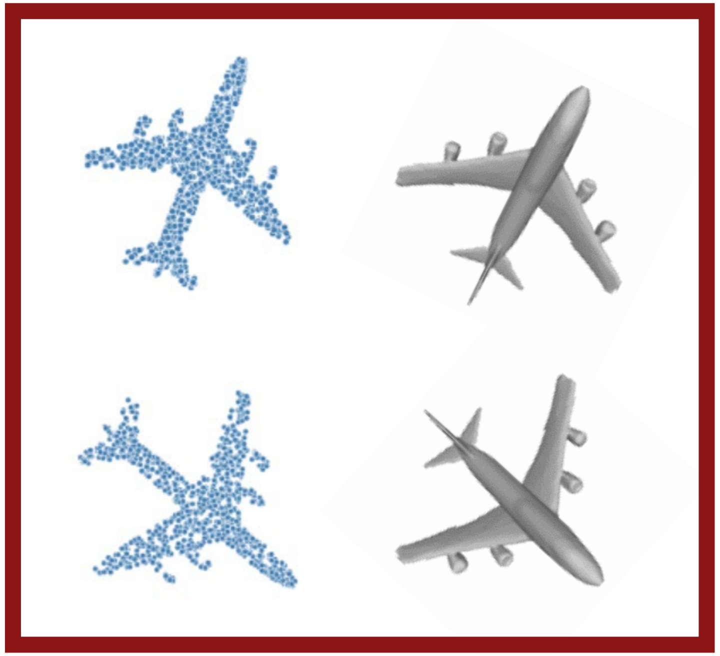
invariant

segmentation



equivariant
(or **invariant**, depending on data representation)

reconstruction



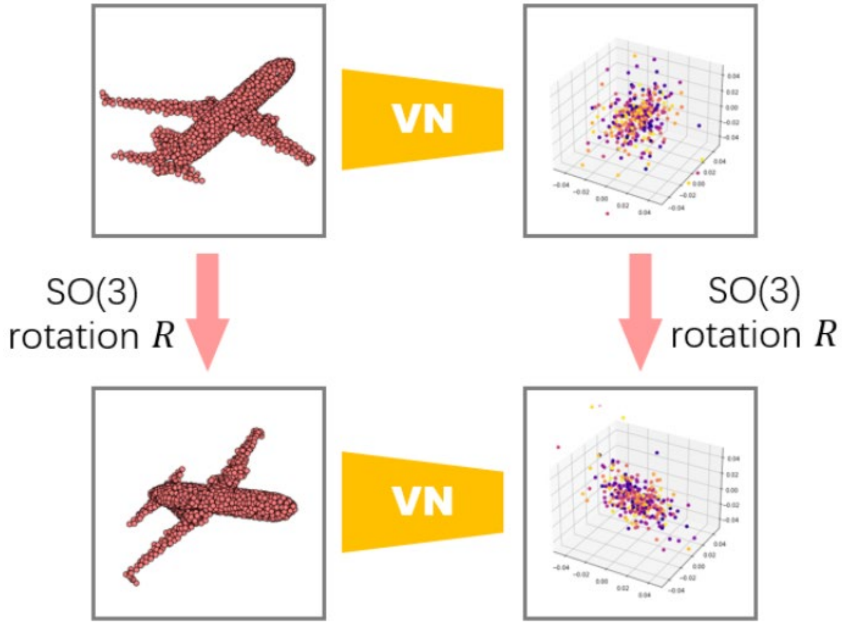
equivariant encoder
invariant decoder

[W. Sun, A. Tagliasacchi, B. Deng, S. Sabour, S. Yazdani, G. Hinton, K. M. Yi, arXiv:2012.04718 (2020)]
[J. J. Park, P. Florence, J. Straub, R. Newcombe, S. Lovegrove, CVPR 2019]

Equivariance

We say a neural network $f(\cdot\,;\theta)$ is rotation equivariant, if for any 3D rotation $R \in \mathrm{SO}(3)$ applied to its input $\mathbf{x}$, it is explicitly related to a transformation $D(R)$ on the network output satisfying

$$f(\mathbf{x}R;\theta) = f(\mathbf{x};\theta)D(R)$$



- $D(R)$ should be independent of $\mathbf{x}$
- **Special case:** when $D(R) = R$ is the identity mapping, it is the common-sense "equivariance"
- **Special case:** when $D(R) = \mathrm{I}$ is the constant mapping, it is invariance

Last Time: Vector Neurons (VNs)

Vector Neurons

Classical (scalar) feature $\boldsymbol{z} = [z_1, z_2, \cdots, z_C]^\top \in \mathbb{R}^C$, with $z_i \in \mathbb{R}$

Vector-list feature $\boldsymbol{V} = [\boldsymbol{v}_1, \boldsymbol{v}_2, \cdots, \boldsymbol{v}_C]^\top \in \mathbb{R}^{C\times 3}$, with $\boldsymbol{v}_i \in \mathbb{R}^3$

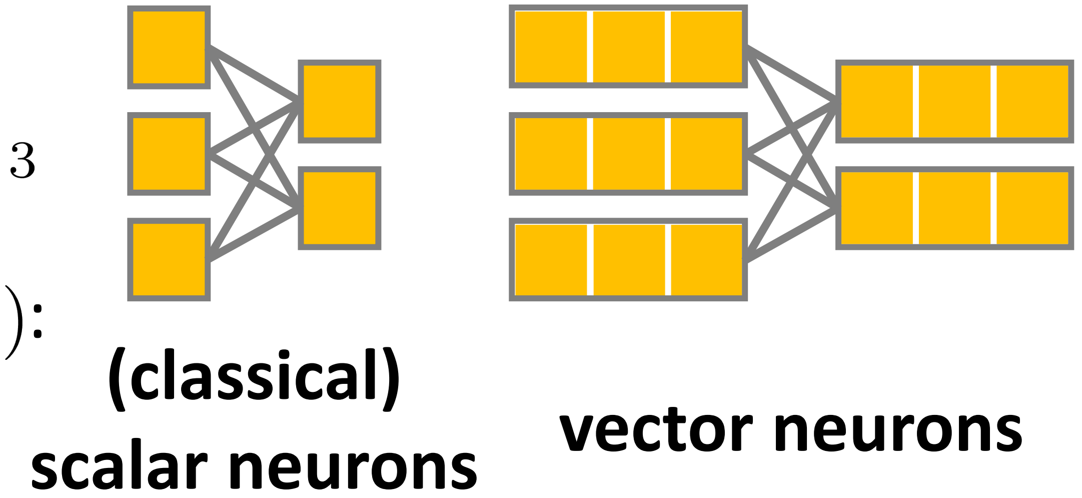
- For pointcloud with N points $\mathcal{V} = \{\boldsymbol{V}_1, \boldsymbol{V}_2, \cdots, \boldsymbol{V}_N\} \in \mathbb{R}^{N\times C\times 3}$

Mapping between network layers:

$$f(\cdot;\theta) : \mathbb{R}^{N\times C^{(d)}\times 3} \to \mathbb{R}^{N\times C^{(d+1)}\times 3}$$

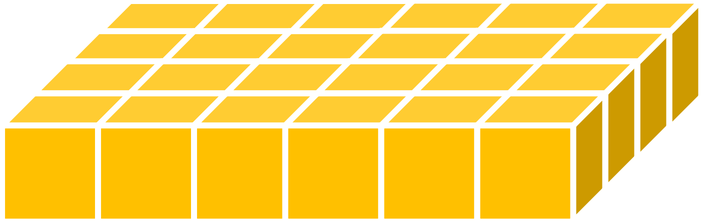
? Equivariance to rotation $R \in \mathrm{SO}(3)$:

$$f(\mathcal{V}R;\theta) = f(\mathcal{V};\theta)R$$



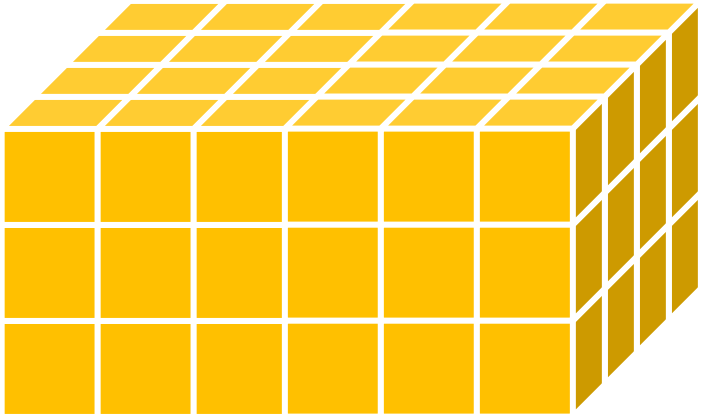
VN Features (for Point Cloud)

Classical:

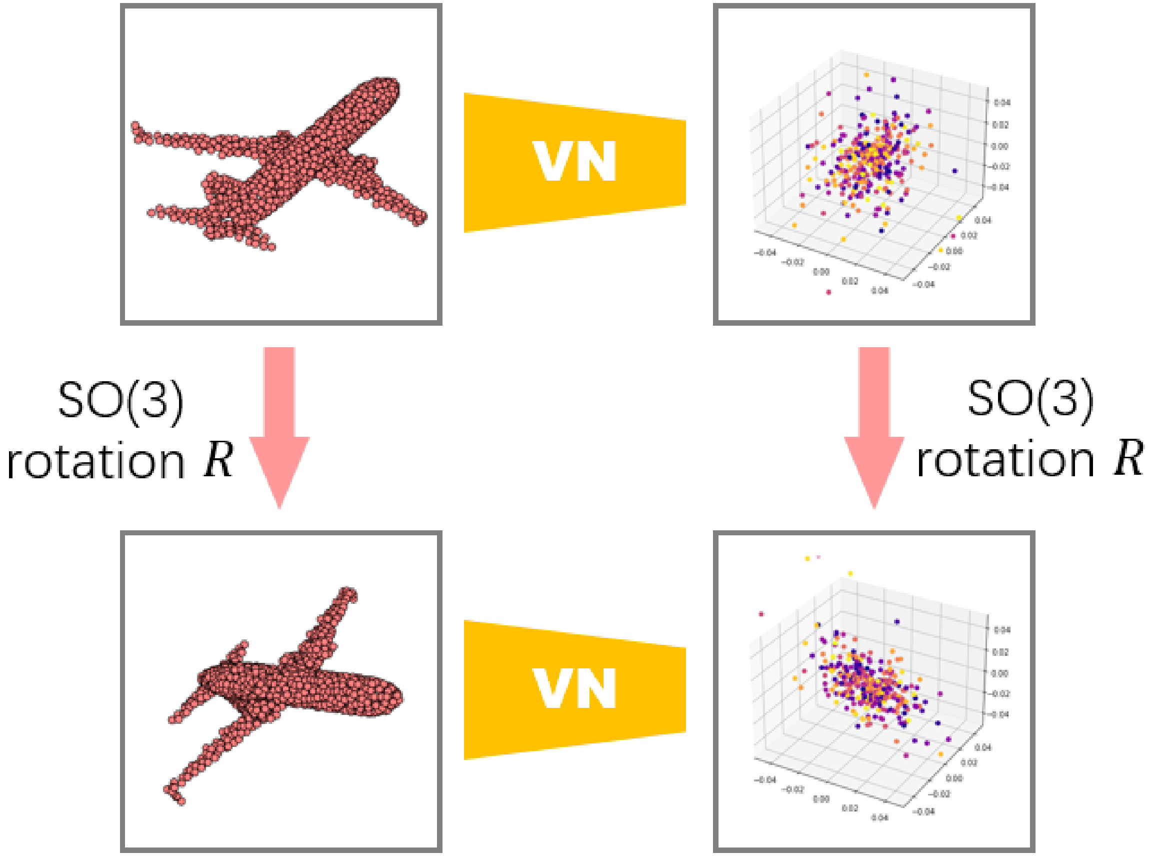


$N \times C \times 1$ feature

VN:

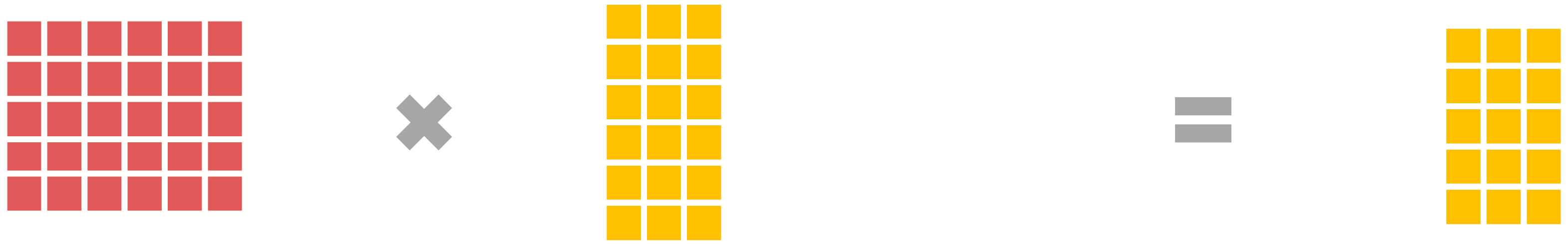


$N \times C \times 3$ feature



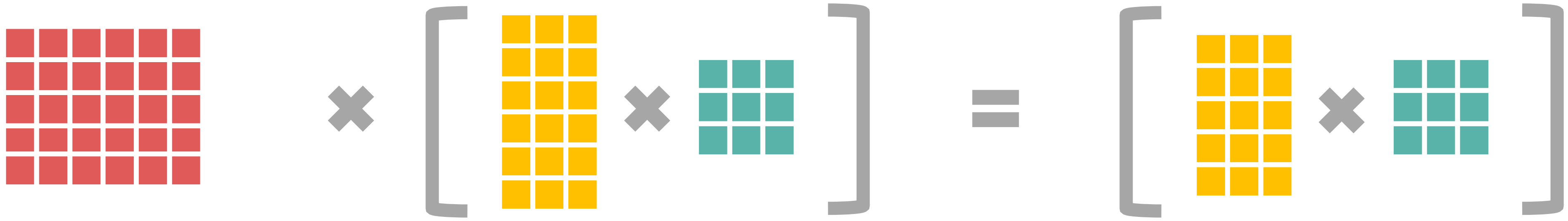
VN Linear Layer

Linear operator: left multiply by the learnable weight matrix



$C' \times C$ weight $\quad$ $C \times 3$ feature $\quad$ $C' \times 3$ feature

Equivariance: right multiply by the SO(3) rotation matrix



$C' \times C$ weight $\quad$ $C \times 3$ feature $\quad$ $C' \times 3$ feature

VN Non-Linearity

ReLU Non-Linearity

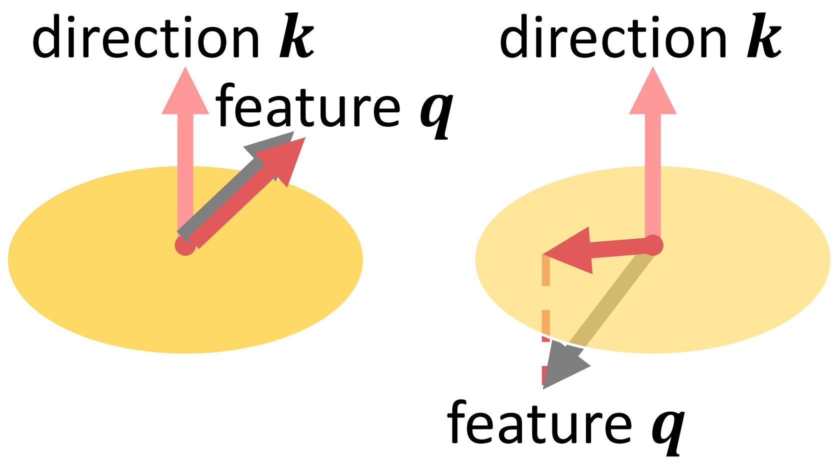
Weights $\mathbf{W} \in \mathbb{R}^{1\times C}$ and $\mathbf{U} \in \mathbb{R}^{1\times C}$

Learn a feature $\boldsymbol{q} = \mathbf{W}\boldsymbol{V} \in \mathbb{R}^{1\times 3}$

Learn a direction $\boldsymbol{k} = \mathbf{U}\boldsymbol{V} \in \mathbb{R}^{1\times 3}$

For each output vector neuron $\boldsymbol{v}' \in \boldsymbol{V}'$

$$\boldsymbol{v}' = \begin{cases} \boldsymbol{q} & \text{if } \langle \boldsymbol{q}, \boldsymbol{k} \rangle \geqslant 0 \\ \boldsymbol{q} - \langle \boldsymbol{q}, \frac{\boldsymbol{k}}{\|\boldsymbol{k}\|} \rangle \frac{\boldsymbol{k}}{\|\boldsymbol{k}\|} & \text{otherwise} \end{cases}$$



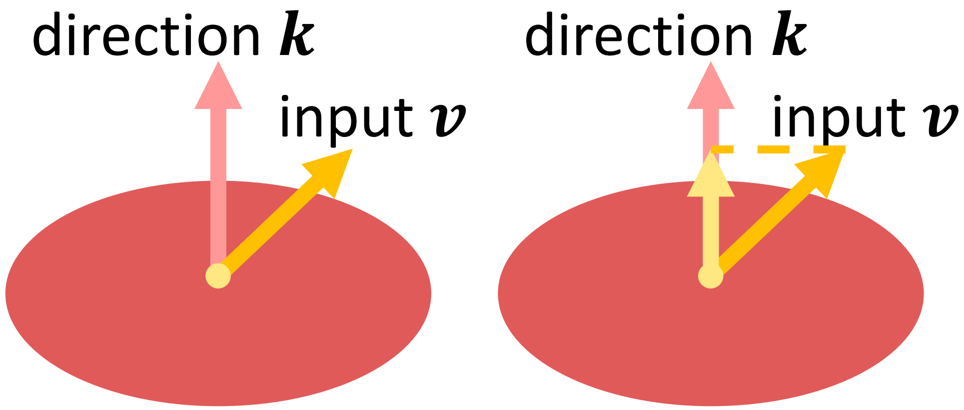
unchanged **clip!**

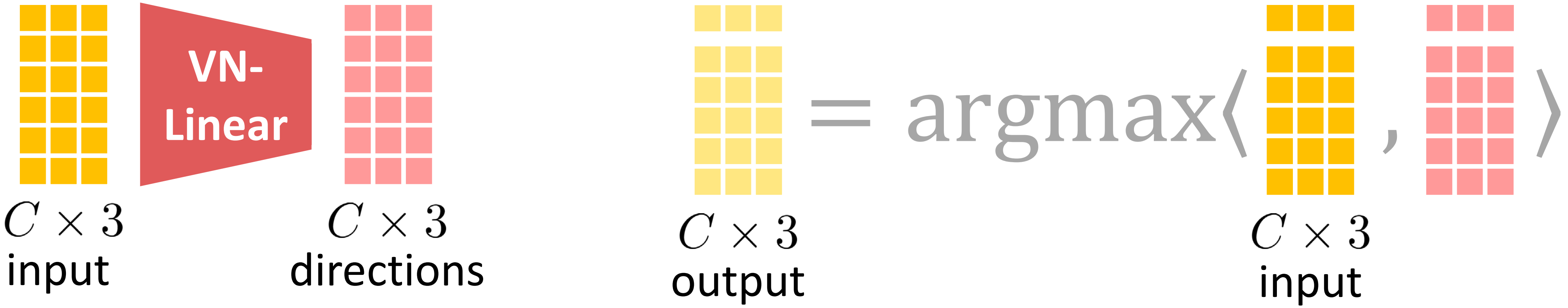
VN Pooling

✓ **Mean pooling**

? Max pooling

- (Similar to non-linearity)
- argmax alone learned directions



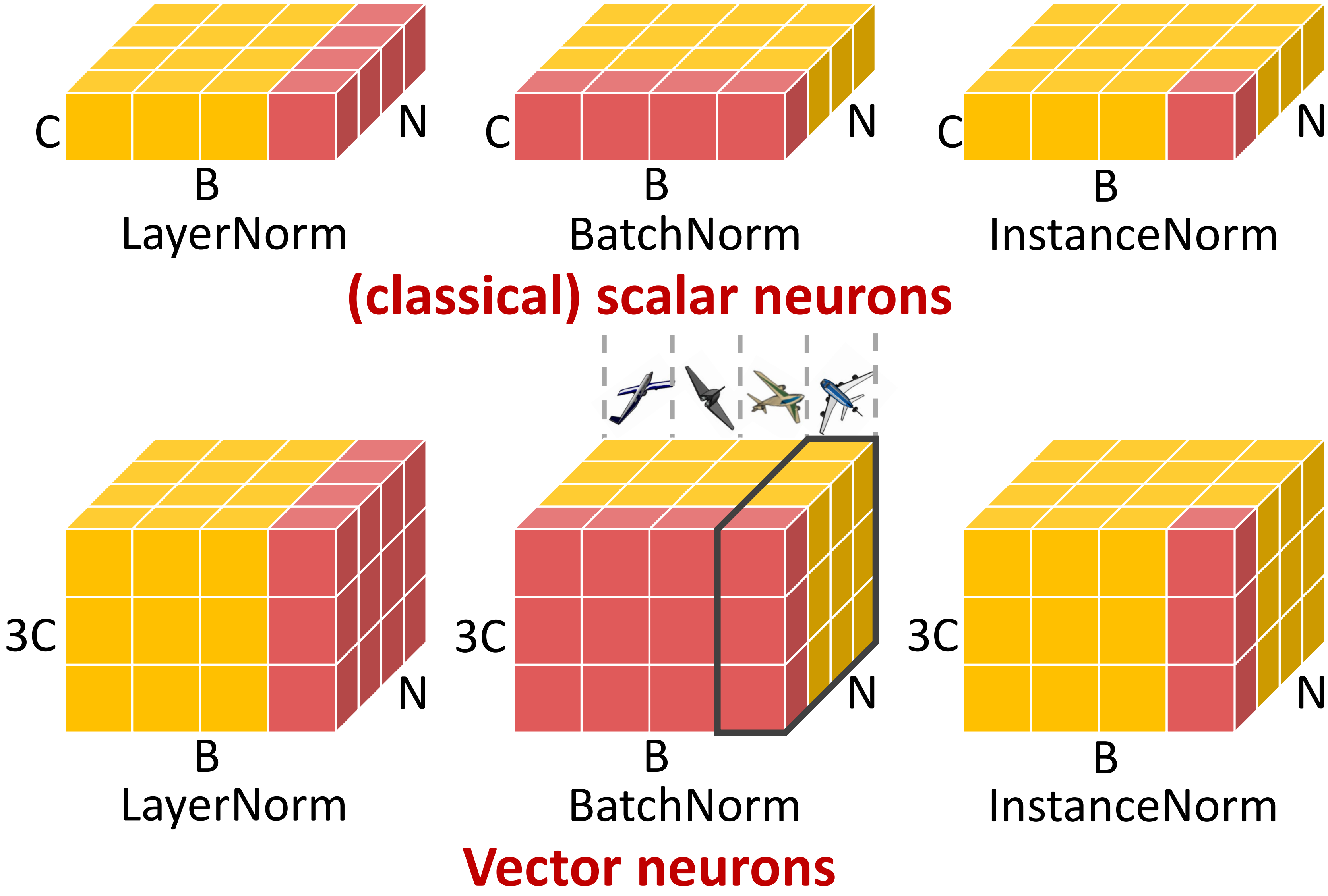


VN Normalizations

✓**LayerNorm**

✓**InstanceNorm**

✓**Dropout**

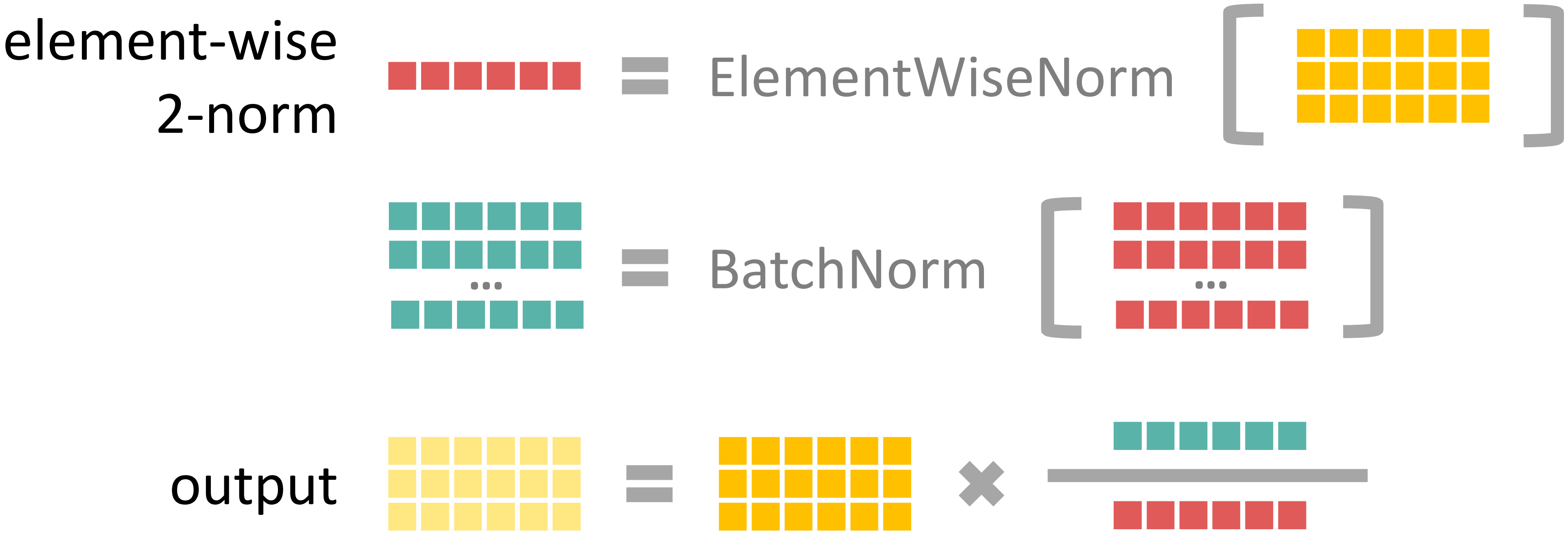


? BatchNorm

averaging across arbitrarily rotated inputs would not necessarily be meaningful

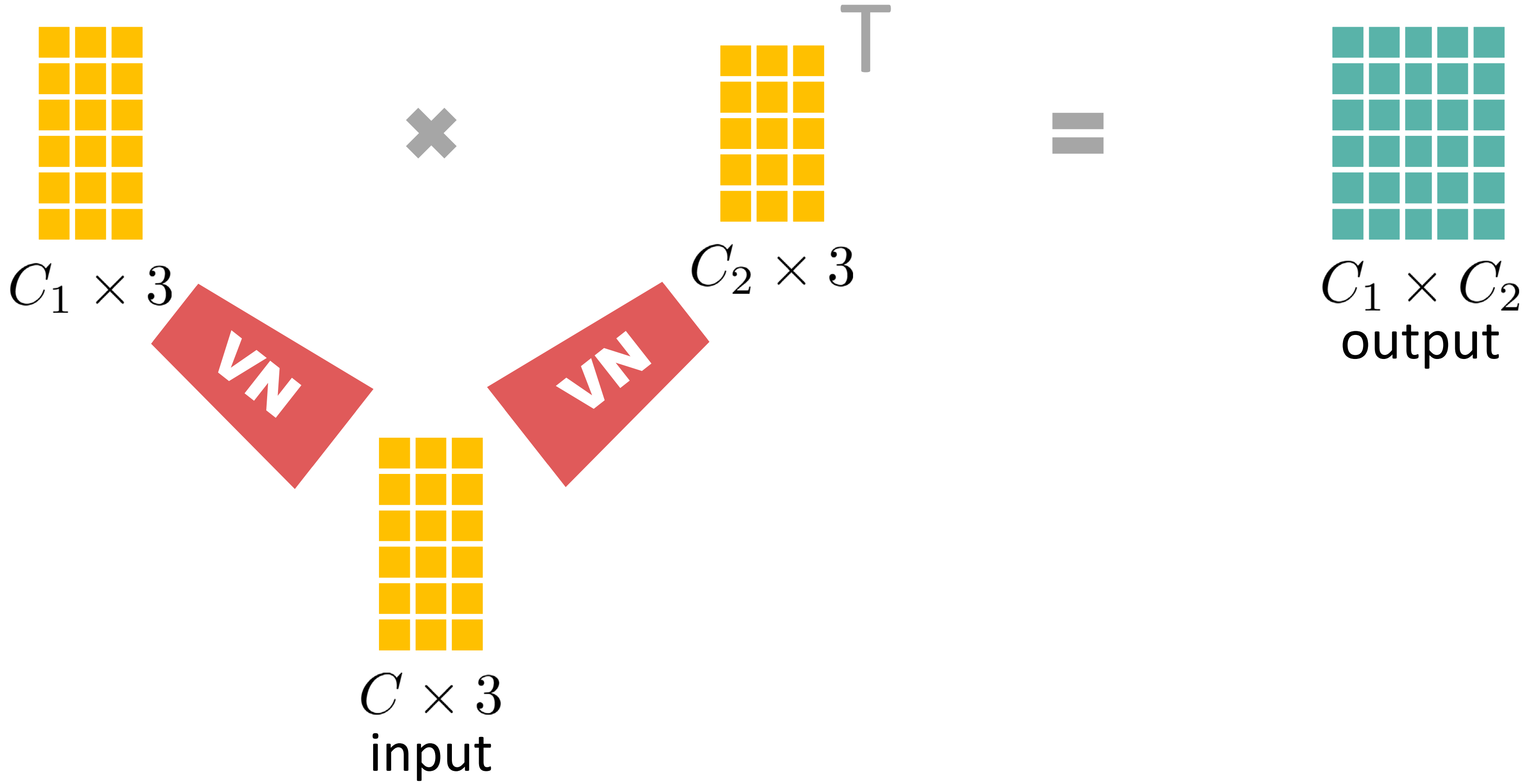
VN Normalizations

BatchNorm



VN Invariant Layer

(equivariant feature) × (equivariant feature)^T = (invariant feature)



Build VN Networks: VN-DGCNN

DGCNN

Edge feature: $'_{nm} = \mathrm{ReLU}(\Theta(\ _m - \ _n) + \Phi\ _n)$

Aggregation: $'_{n} = \mathrm{Pool}_{m:(n,m)\in\mathcal{E}}(\ '_{nm})$

VN-DGCNN

Edge feature: $'_{nm} = \text{VN-ReLU}(\Theta(\ _m - \ _n) + \Phi\ _n)$

Aggregation $'_{n} = \text{VN-Pool}_{m:(n,m)\in\mathcal{E}}(\ '_{nm})$

[Y. Wang, Y. Sun, Z. Liu, S. E. Sarma, M. M. Bronstein, J. M. Solomon, TOG 2019]

Build VN Networks: VN-PointNet

PointNet

$$\blacksquare\blacksquare\blacksquare\blacksquare\blacksquare' = \text{Pool}_{\boldsymbol{x}_n \in \mathcal{X}}\left(h(\blacksquare\blacksquare\blacksquare\blacksquare\blacksquare_1), h(\blacksquare\blacksquare\blacksquare\blacksquare\blacksquare_2) \cdots, h(\blacksquare\blacksquare\blacksquare\blacksquare\blacksquare_N)\right)$$

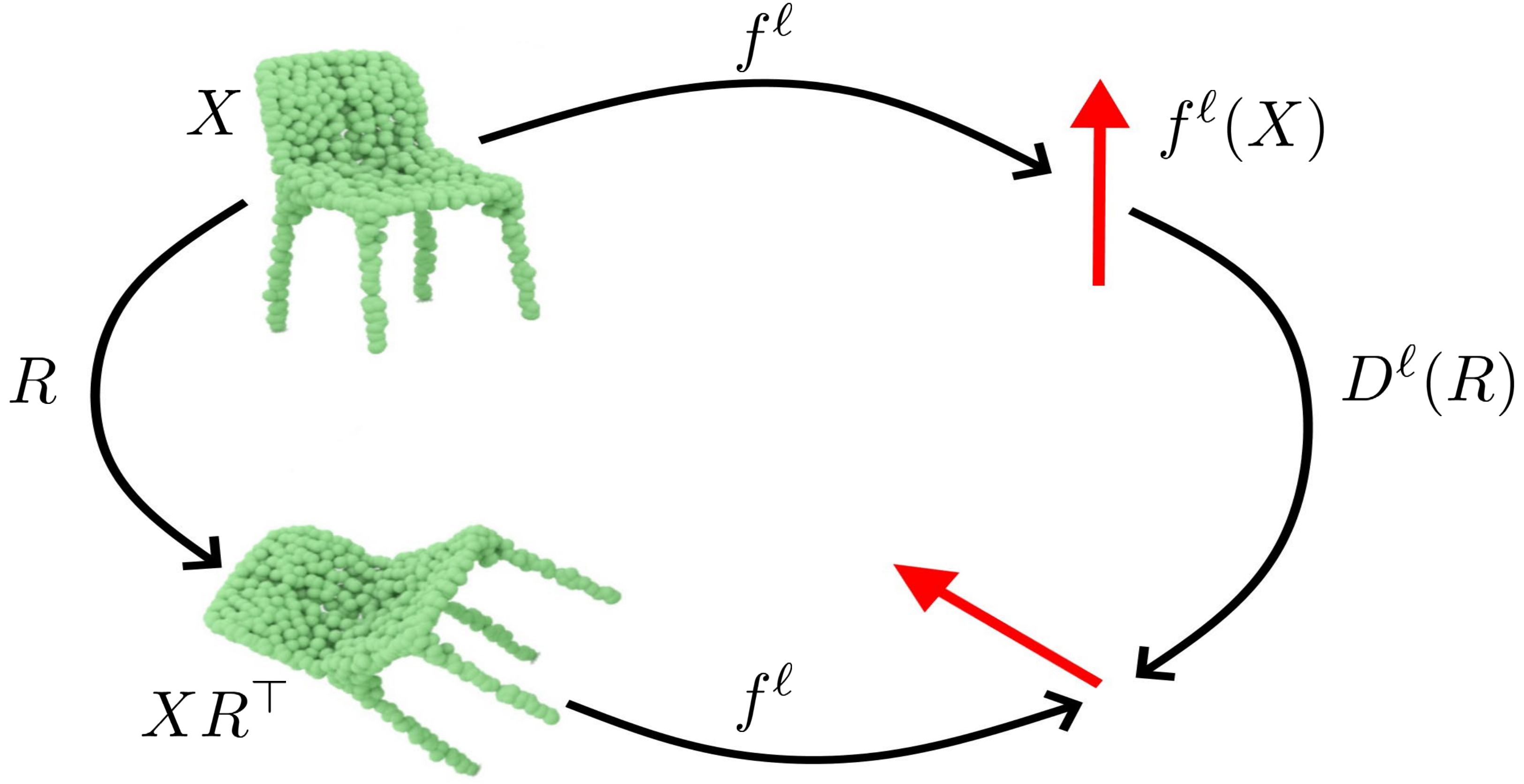
VN-PointNet

$$\begin{smallmatrix}\blacksquare\blacksquare\blacksquare\blacksquare\blacksquare\\\blacksquare\blacksquare\blacksquare\blacksquare\blacksquare\\\blacksquare\blacksquare\blacksquare\blacksquare\blacksquare\end{smallmatrix}' = \text{VN-Pool}_{\boldsymbol{V}_n \in \mathcal{V}}\left(f(\begin{smallmatrix}\blacksquare\blacksquare\blacksquare\blacksquare\blacksquare\\\blacksquare\blacksquare\blacksquare\blacksquare\blacksquare\\\blacksquare\blacksquare\blacksquare\blacksquare\blacksquare\end{smallmatrix}_1), f(\begin{smallmatrix}\blacksquare\blacksquare\blacksquare\blacksquare\blacksquare\\\blacksquare\blacksquare\blacksquare\blacksquare\blacksquare\\\blacksquare\blacksquare\blacksquare\blacksquare\blacksquare\end{smallmatrix}_2), \cdots, f(\begin{smallmatrix}\blacksquare\blacksquare\blacksquare\blacksquare\blacksquare\\\blacksquare\blacksquare\blacksquare\blacksquare\blacksquare\\\blacksquare\blacksquare\blacksquare\blacksquare\blacksquare\end{smallmatrix}_N)\right)$$

[C. R. Qi, H. Su, K. Mo, L. J. Guibas, CVPR 2017]

Last Time: Tensor Field Networks (TFNs)

SO(3) Equivariant Features



Wigner matrices

$$f^\ell : \mathbb{R}^{n\times 3} \to \mathbb{R}^{2\ell+1}$$

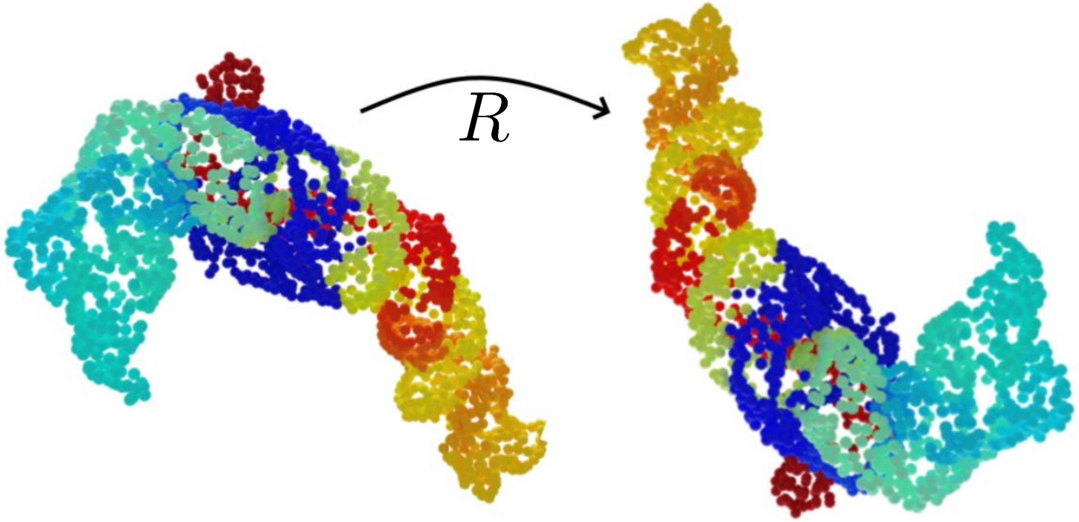
$$f^\ell(XR^\top) = D^\ell(R) f^\ell(X)$$

Examples of Type 0 Features

- Type 0 features are rotation invariant as $D^0(R) = 1$:

Segmentation

$$f^0(X)_{i,c,:} = f^0(XR^\top)$$



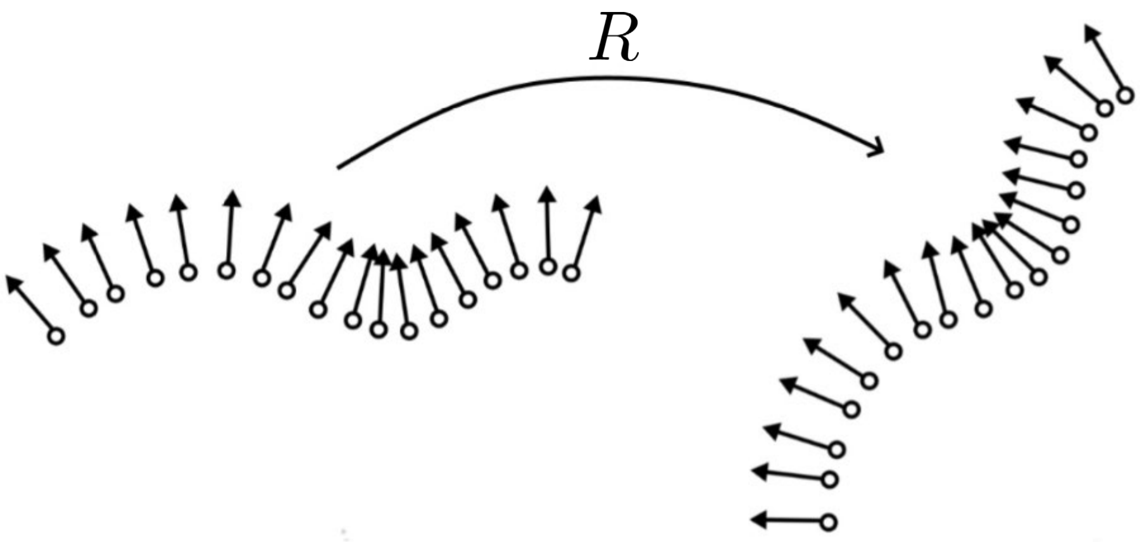
Classification

$g^0($  $) =$ Chair

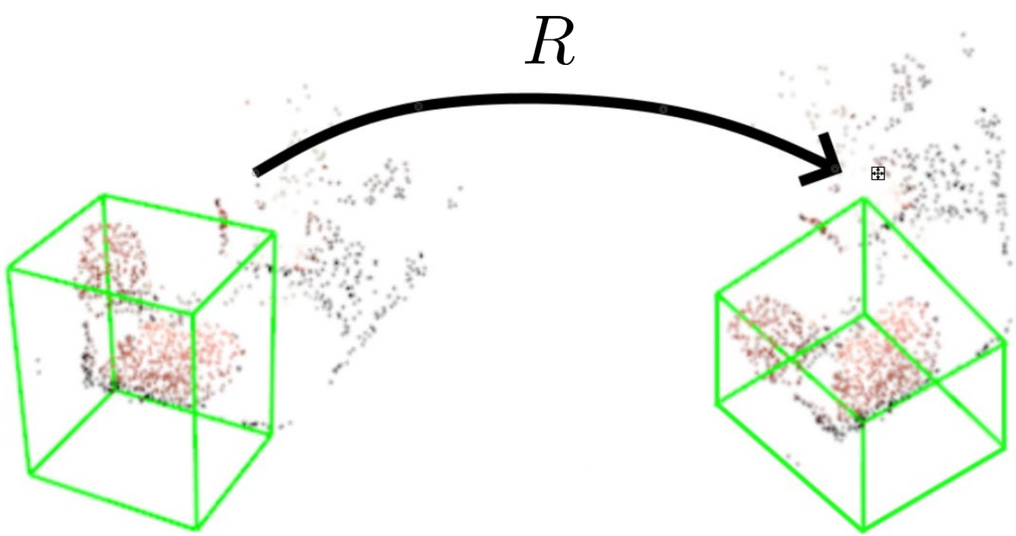
$g^0($  $) =$ Chair

Examples of type 1 Features

- We have $D^1(R) = R$, thefore type 1 features are 3D vectors rotating with the pointcloud X.



Pointcloud normals are type 1 features.

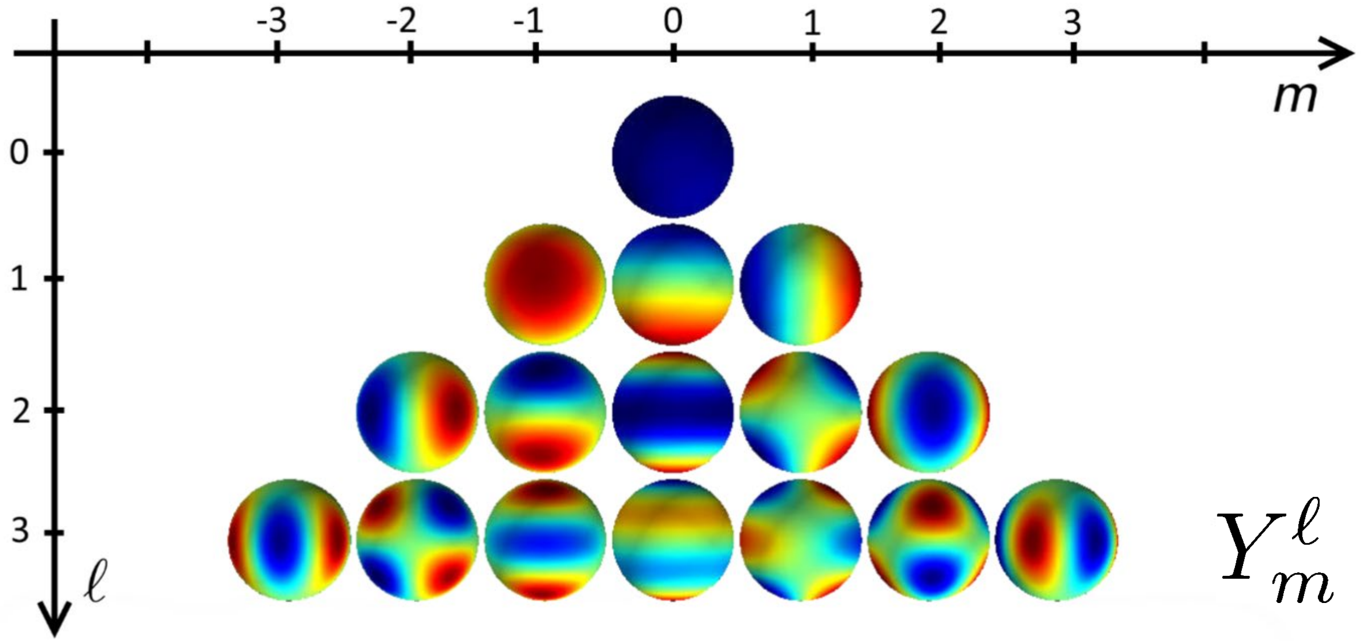


Bounding box center and principal directions are type 1 features, lengths are type 0 features

Spherical Harmonics & Higher Degree Features

- Spherical harmonics are homogeneous polynomials on $\mathbb{R}^3$, their restriction to $\mathcal{S}_2$ form an orthonormal basis of $L^2(\mathcal{S}_2)$.

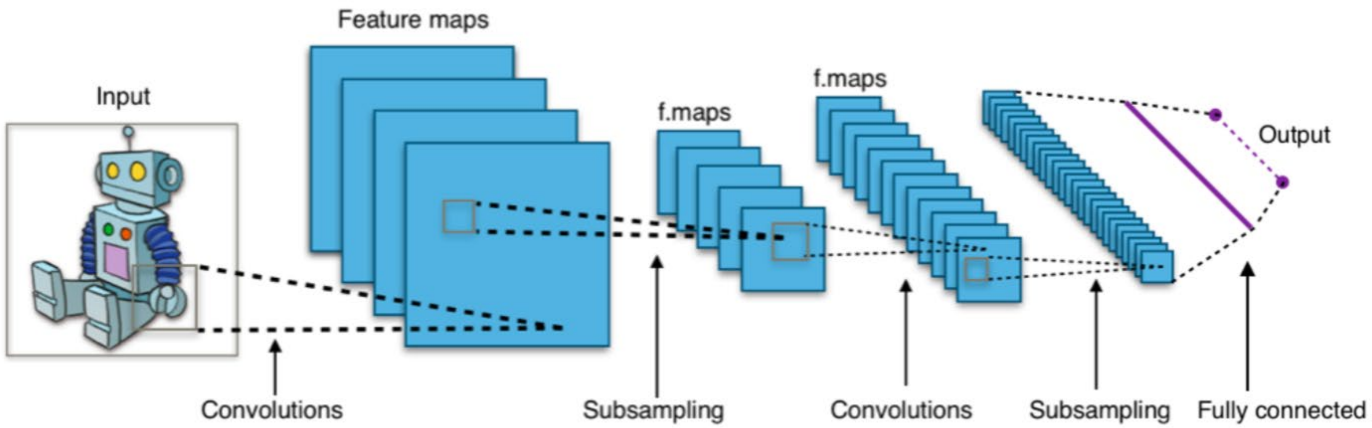
- Just like type ℓ equivariant features the vector of degree ℓ spherical harmonics $Y^\ell(x) \in \mathbb{R}^{2\ell+1}$, satisfies $Y^\ell(Rx) = D^\ell(R)Y^\ell(x)$.



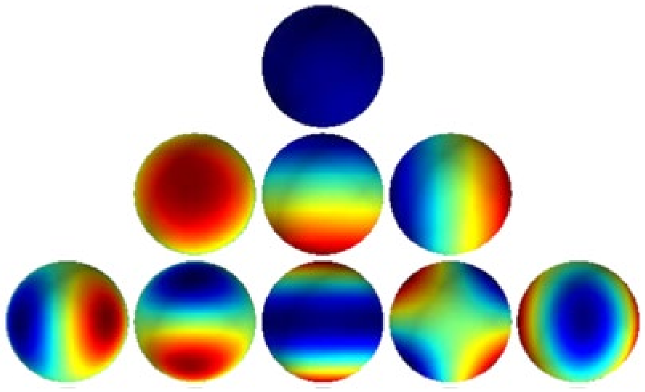
How does TFN work ?

- TFN is a convolutional architecture.
- It inherits its equivariance properties from SH kernels.

$$\mathrm{TFN} =$$



$+$



Steerable Kernel Basis

- A steerable kernel basis, is a kernel basis $(\kappa_k)_k$ such that, the rotation of any kernel κ_j linearly decomposes onto the basis by a rotation matrix $D(R)$:

$$\kappa_j(Rx) = \sum_k D(R)_{jk}\kappa_k(x).$$

- Using a steerable basis we can relate convolutions on X and on $XR^\top$ by a simple linear relation:

$$f *_{XR^\top} \kappa_j(x) = \sum_j D(R)_{jk} f *_X \kappa_k(x)$$

3D Steerable Basis

- For each ℓ we have a sterrable basis of the form:

$$\kappa_{rm}^{\ell}(x) := \varphi_r(\|x\|_2) Y_m^{\ell}\left(\frac{x}{\|x\|_2}\right)$$

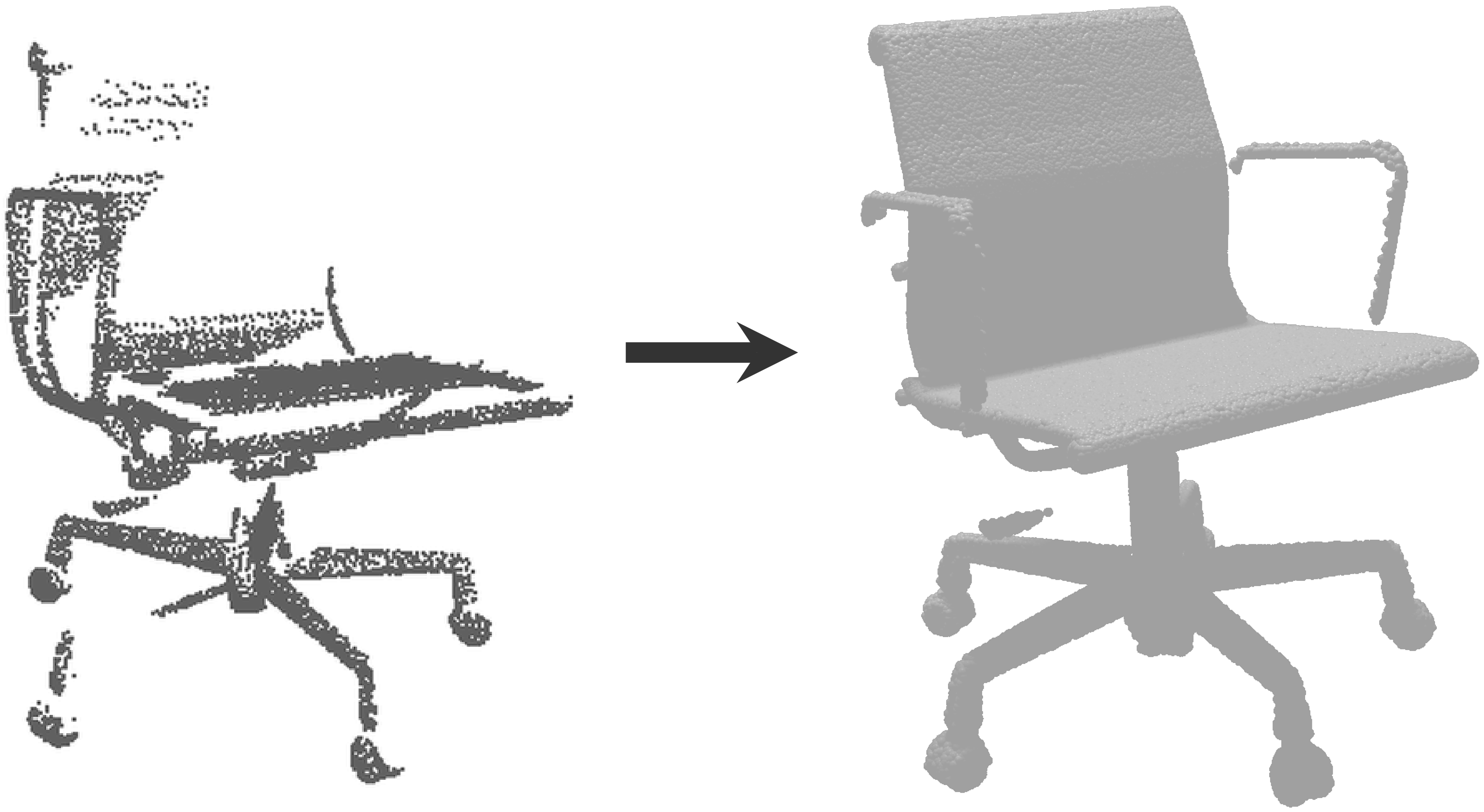
- Where φ_r is any radial function e.g. a gaussian shell:

$$\varphi_r(y) := \exp\left(\frac{-(y-\rho_r)^2}{2\sigma^2}\right)$$

Conditional Shape Generation Based on 3D Data

Goal: Scan Completion

- Complete or re-generate shape from a single view scan

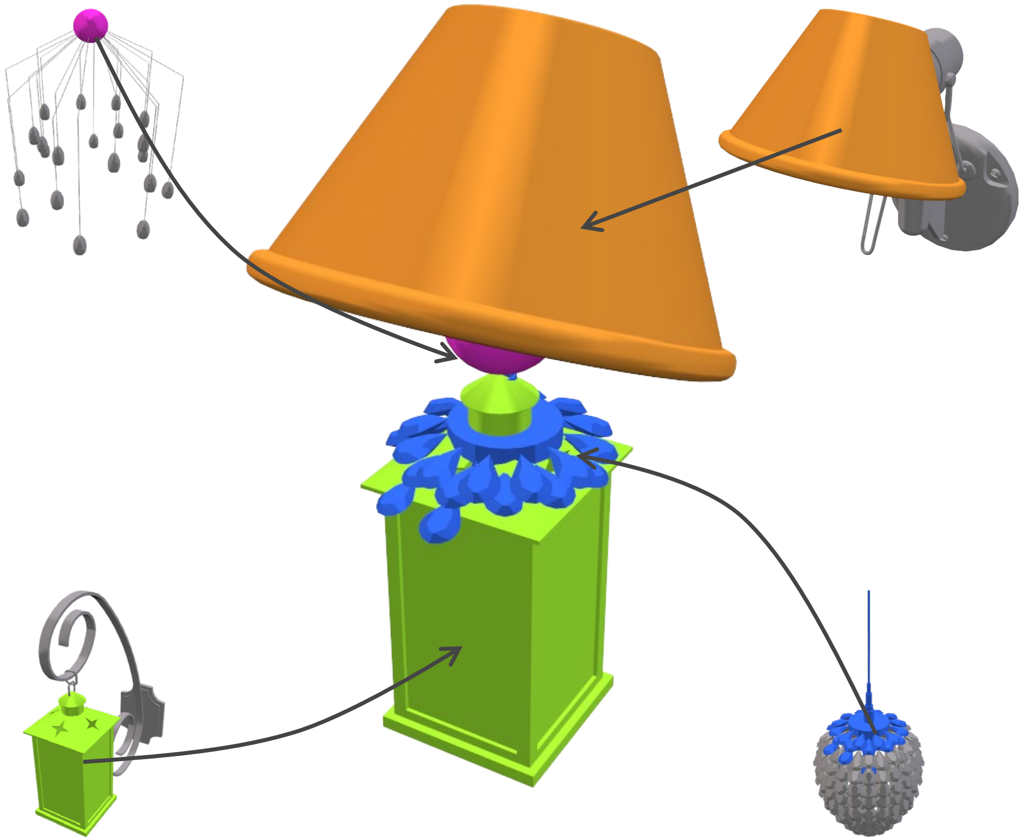


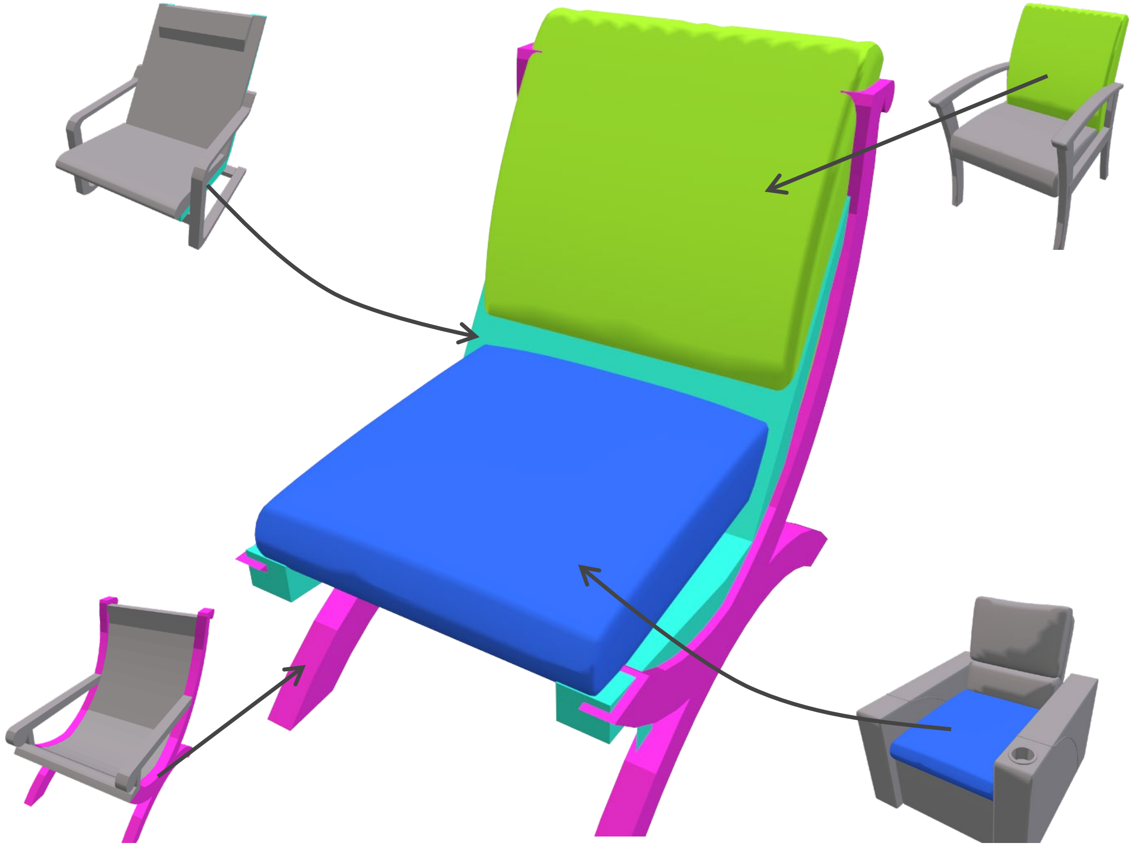
Motivation

- 3D scanning is laborious.

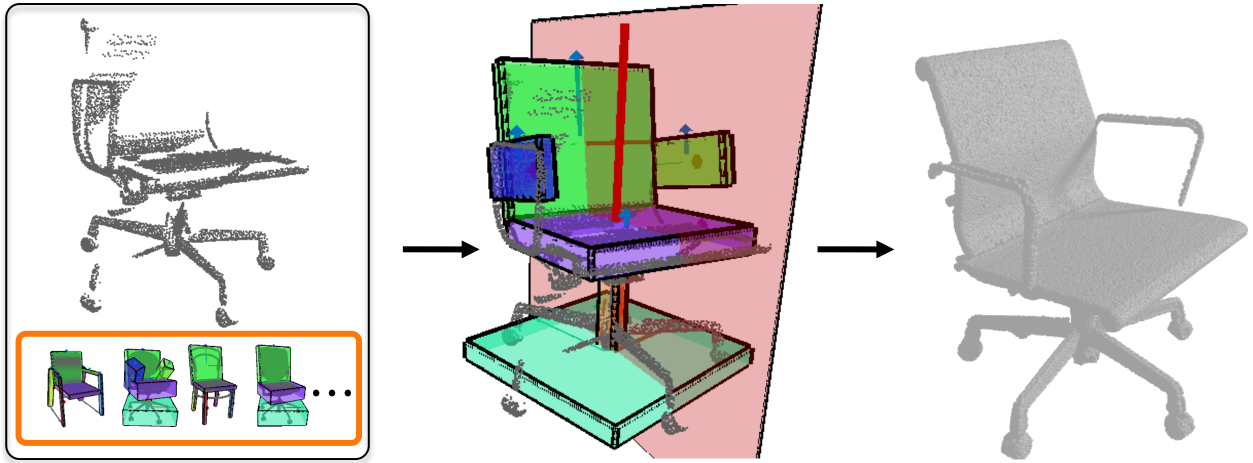
Goal: Composition-Based Modeling

Create a shape by assembling components of 3D models in a large-scale repository.





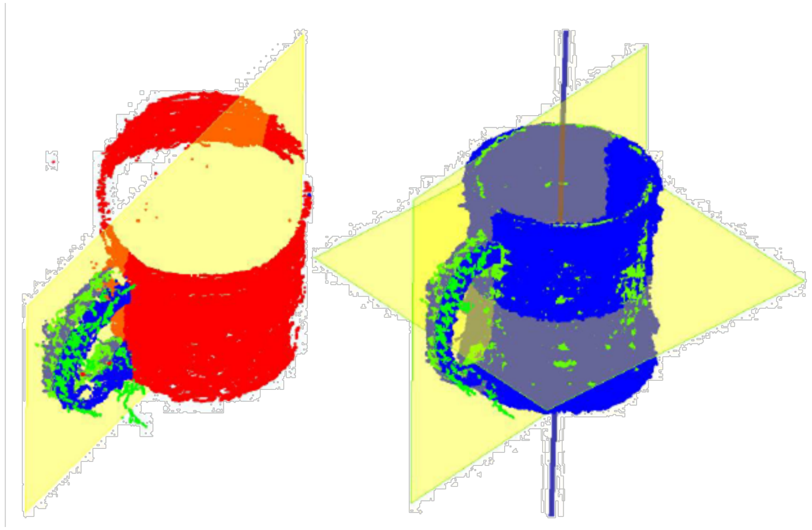
Data-Driven Structural Priors for Shape Completion



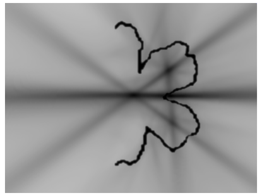
Minhyuk Sung[1] Vladimir G. Kim[1,2] Roland Angst[1,3] Leonidas Guibas[1]
[1]Stanford University [2]Adobe Research [3]Max Planck Institute for Informatics

Filling in What is Missing …

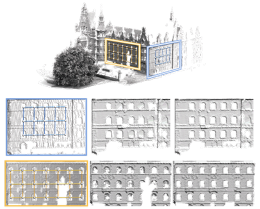
- Symmetry-based



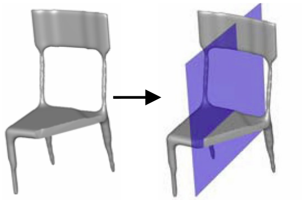
[Thrun el. al. 2005]



[Podolak et. al. 2006]

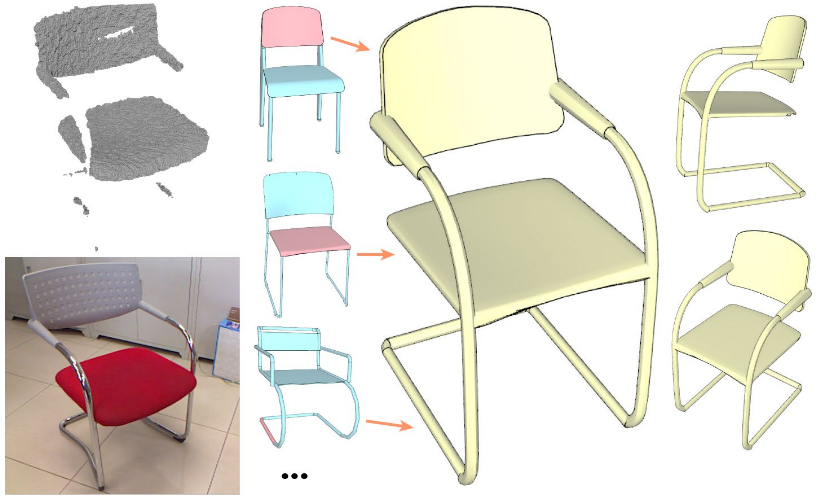


[Pauly el. al. 2008]

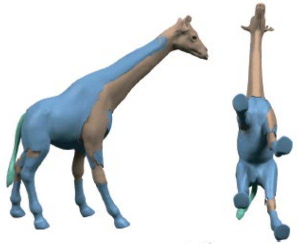


[Sipiran et. al., 2014]

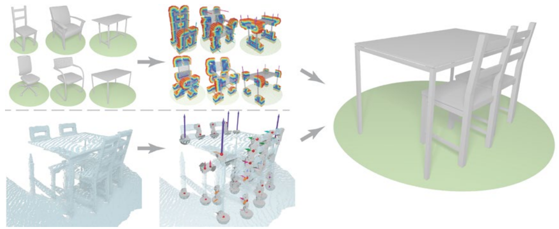
- Data-based



[Shen et. al. 2012]



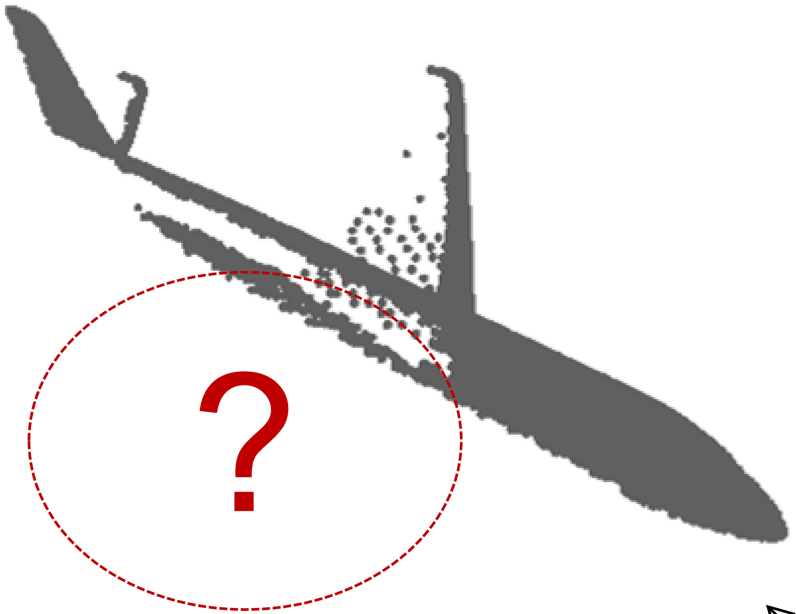
[Pauly el. al. 2005]



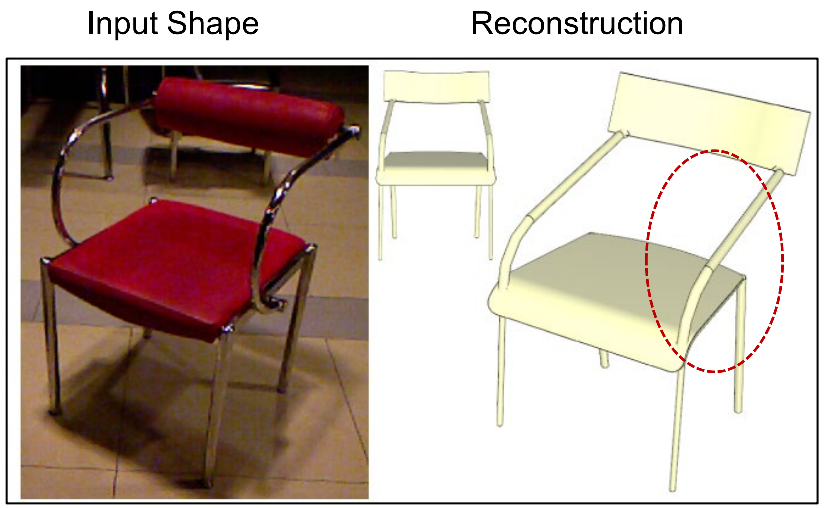
[Li et. al. 2015]

However …

- Symmetry-based
  - Hard to predict from *partial* data.



- Data-based (Priors)
  - Hard to recover the *exact* shape.

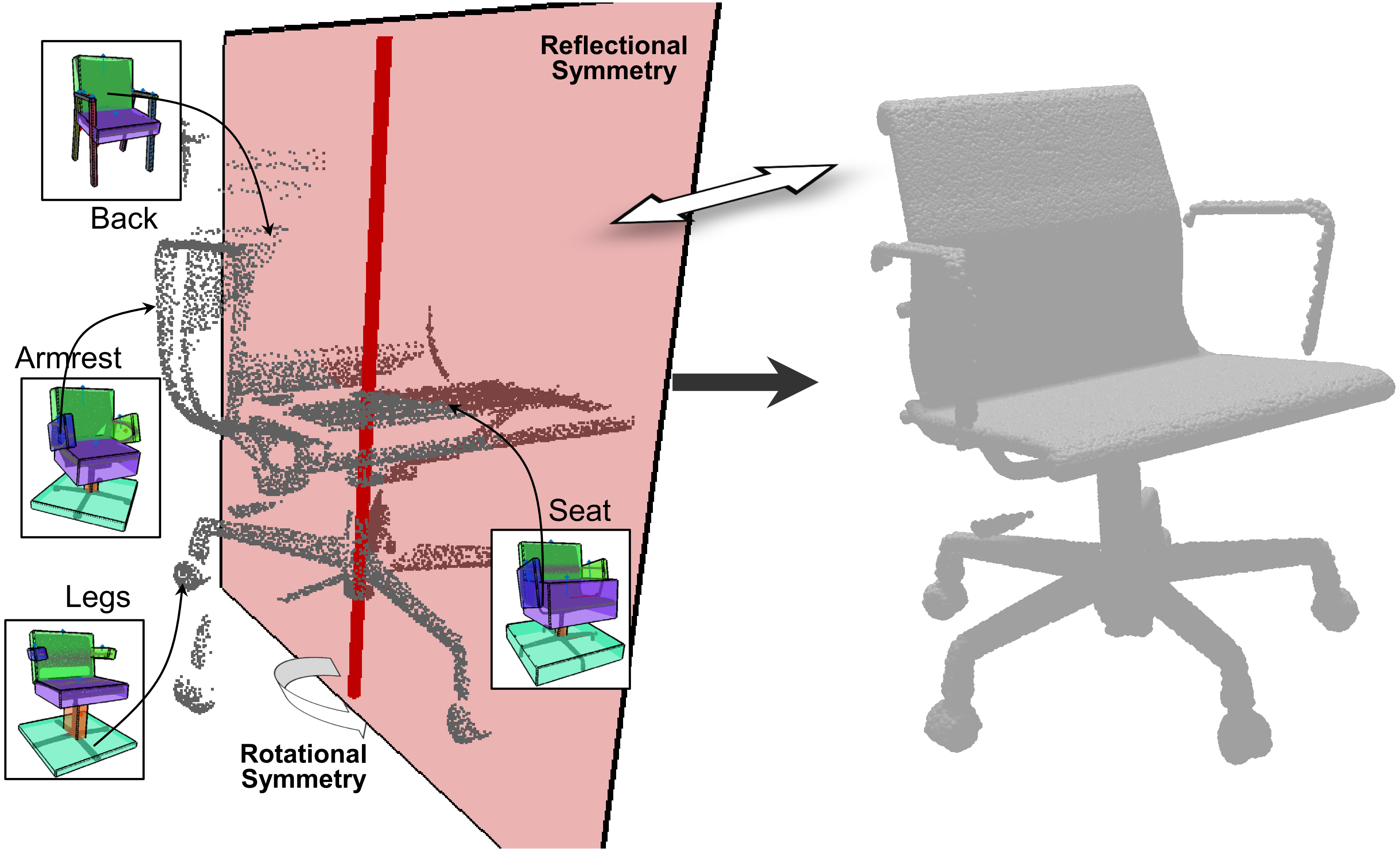


[Shen et. al. 2012]

Complementary!

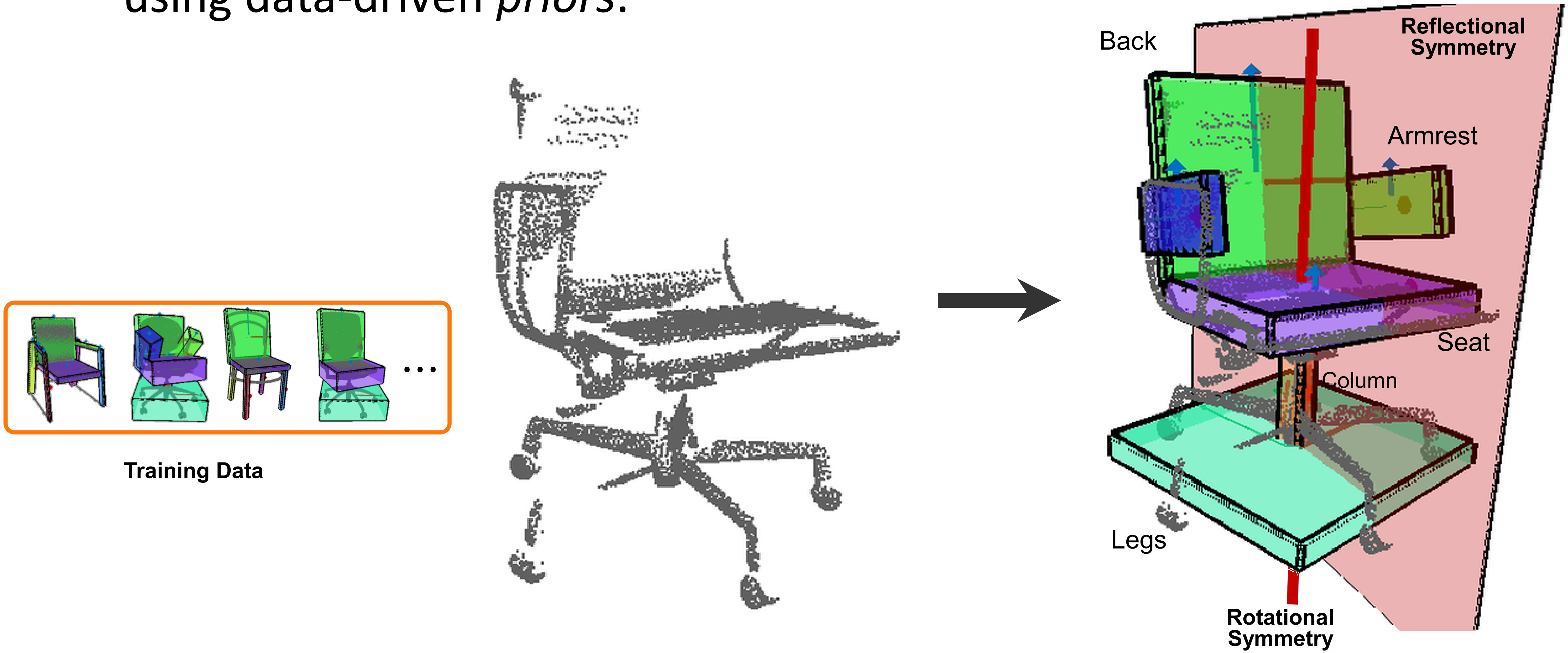
Get Best of Both Worlds

- Combine both symmetry and database sources.



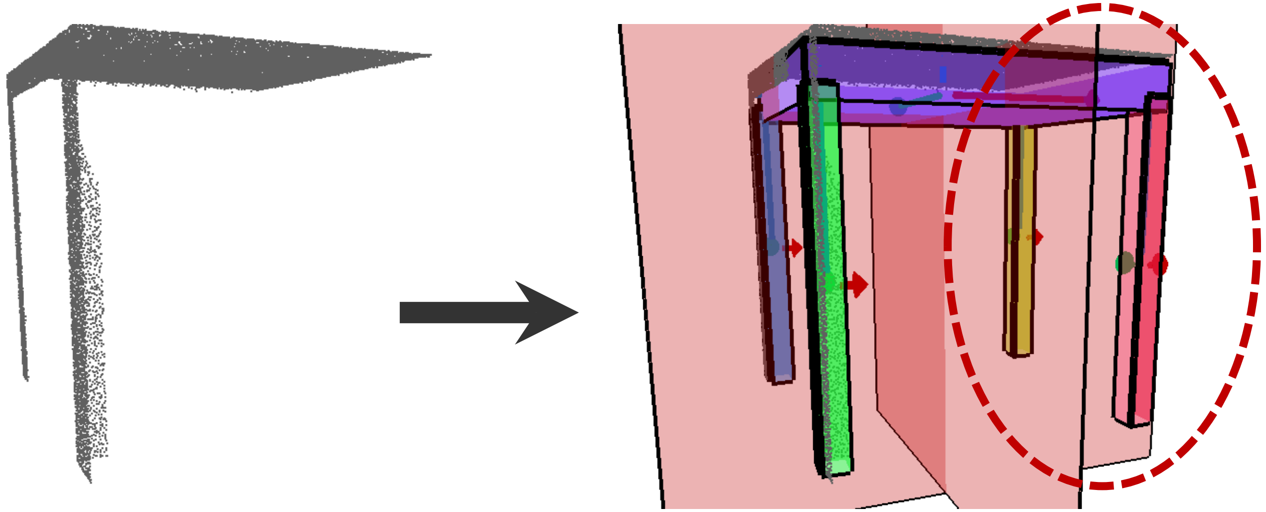
Approach

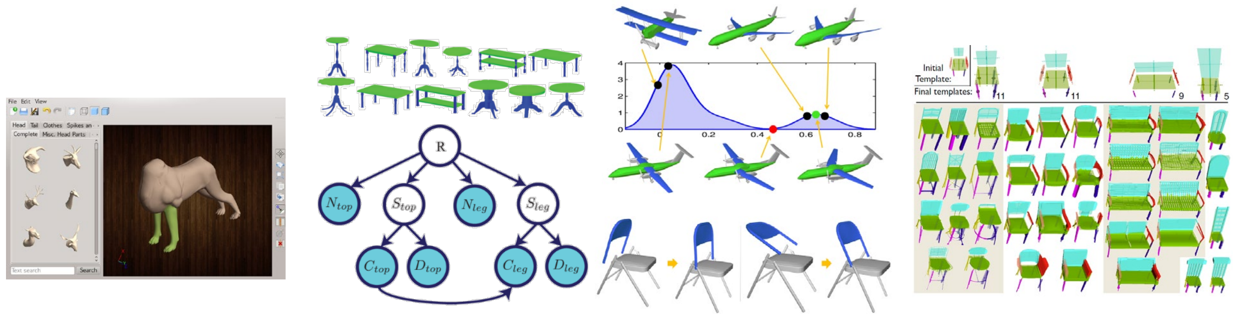
- Estimate part and symmetry structure from the *partial* scan data using data-driven *priors*.



Approach

- Predict *missing* parts based on *part relations*.



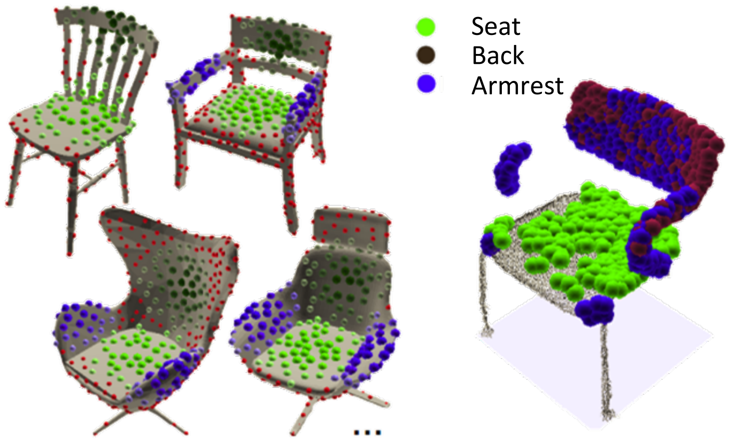


[Chaudhuri et. al. 2011] [Kalogerakis et. al. 2012] [Fish et. al. 2014] [Kim et. al. 2013]

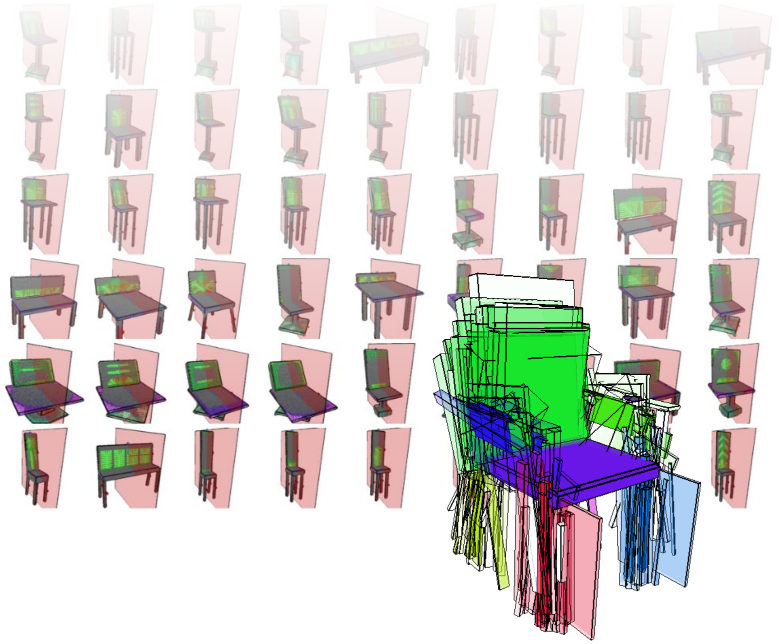
*Earlier efforts analyze **complete** shapes only*

Training

- Probabilistic shape model
  - Per-point classifiers
  - Pairwise part relations

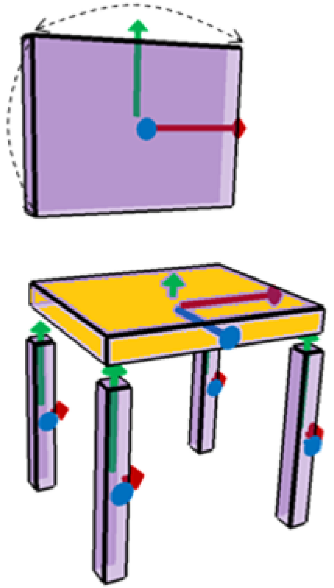


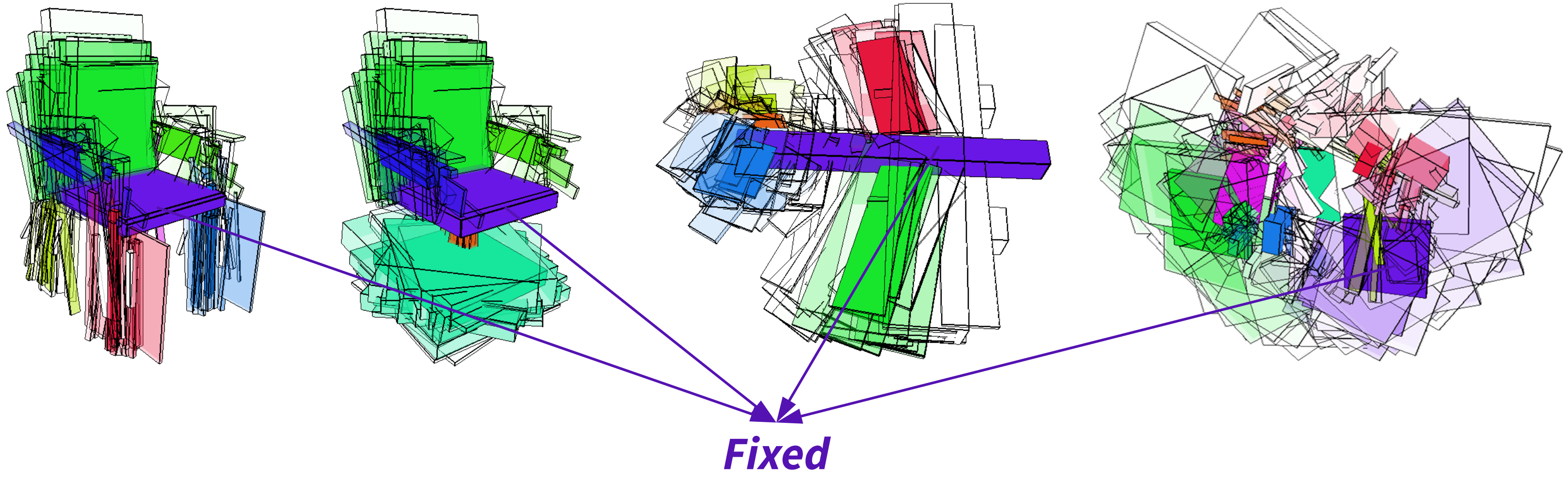
[Kim et. al. 2014]



Probabilistic Part Relations

- Part parameters
  - Local coordinates + Scale
- Pairwise relations
  - Gaussian distributions of ***relative*** pose, height, and scale

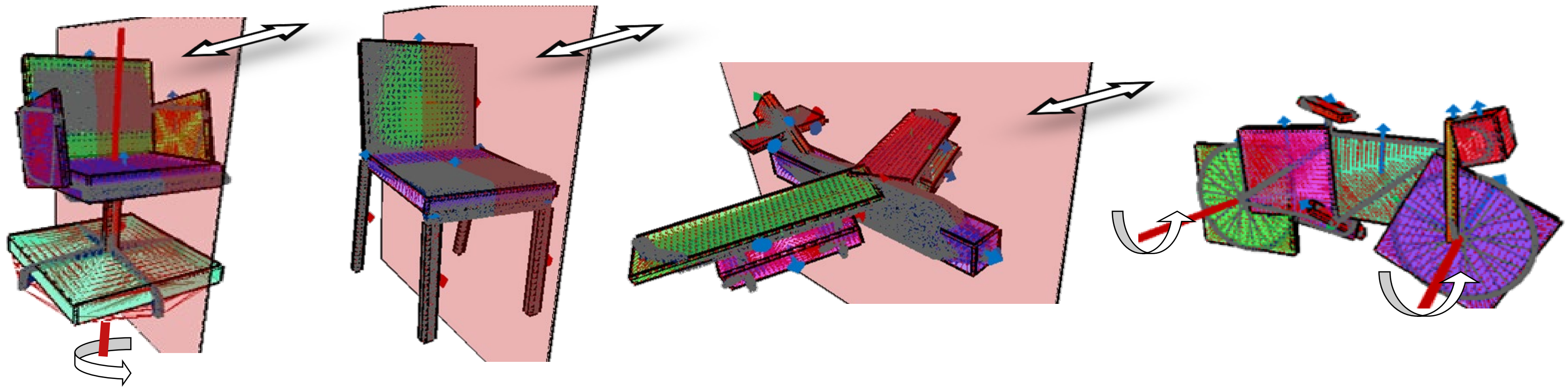




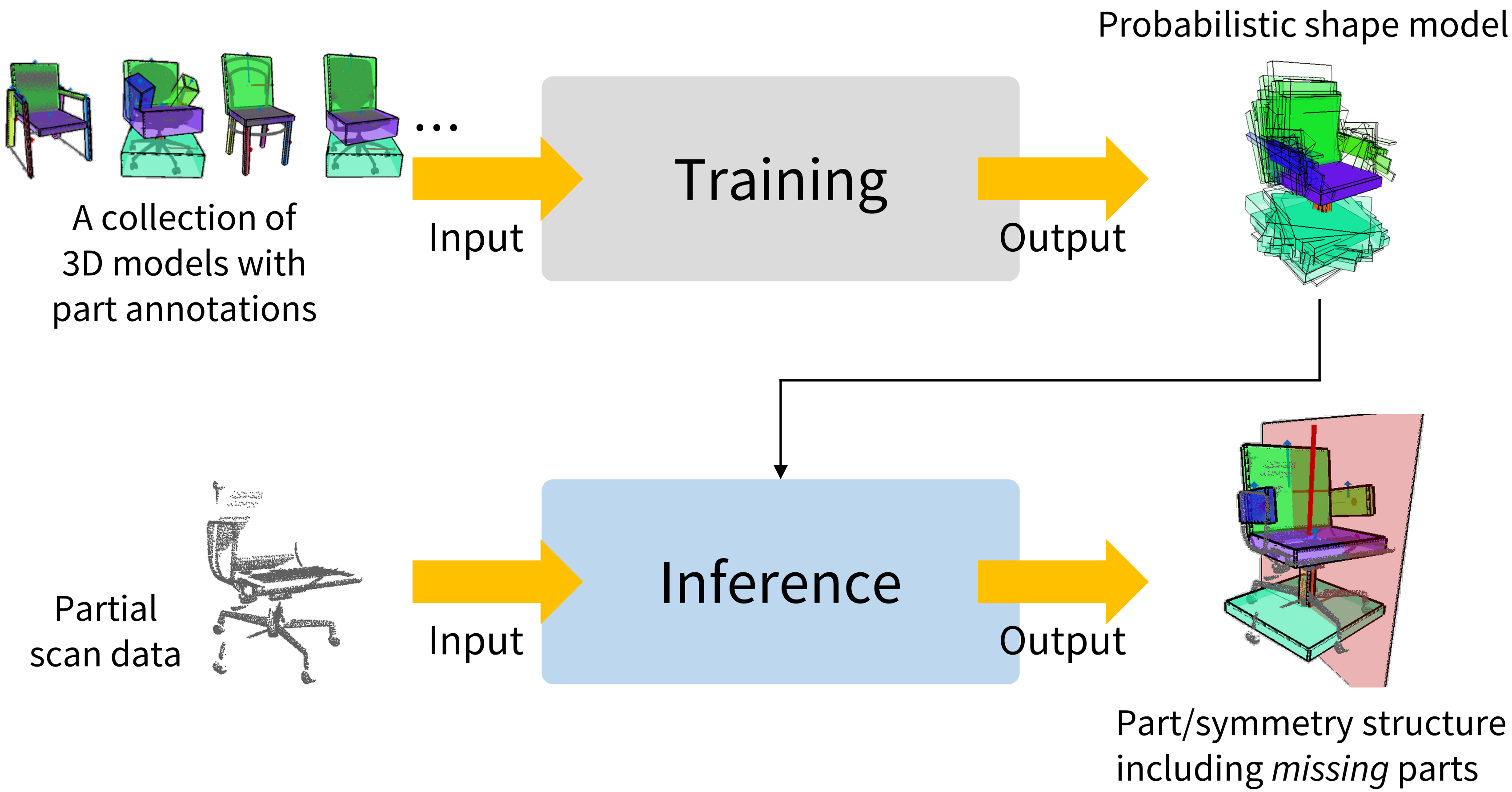
Probabilistic Part Relations

- Part parameters
  - Local coordinates + Scale
- Reflectional and rotational partial symmetries
  - On either a *single* part and/or *pairs* of parts.

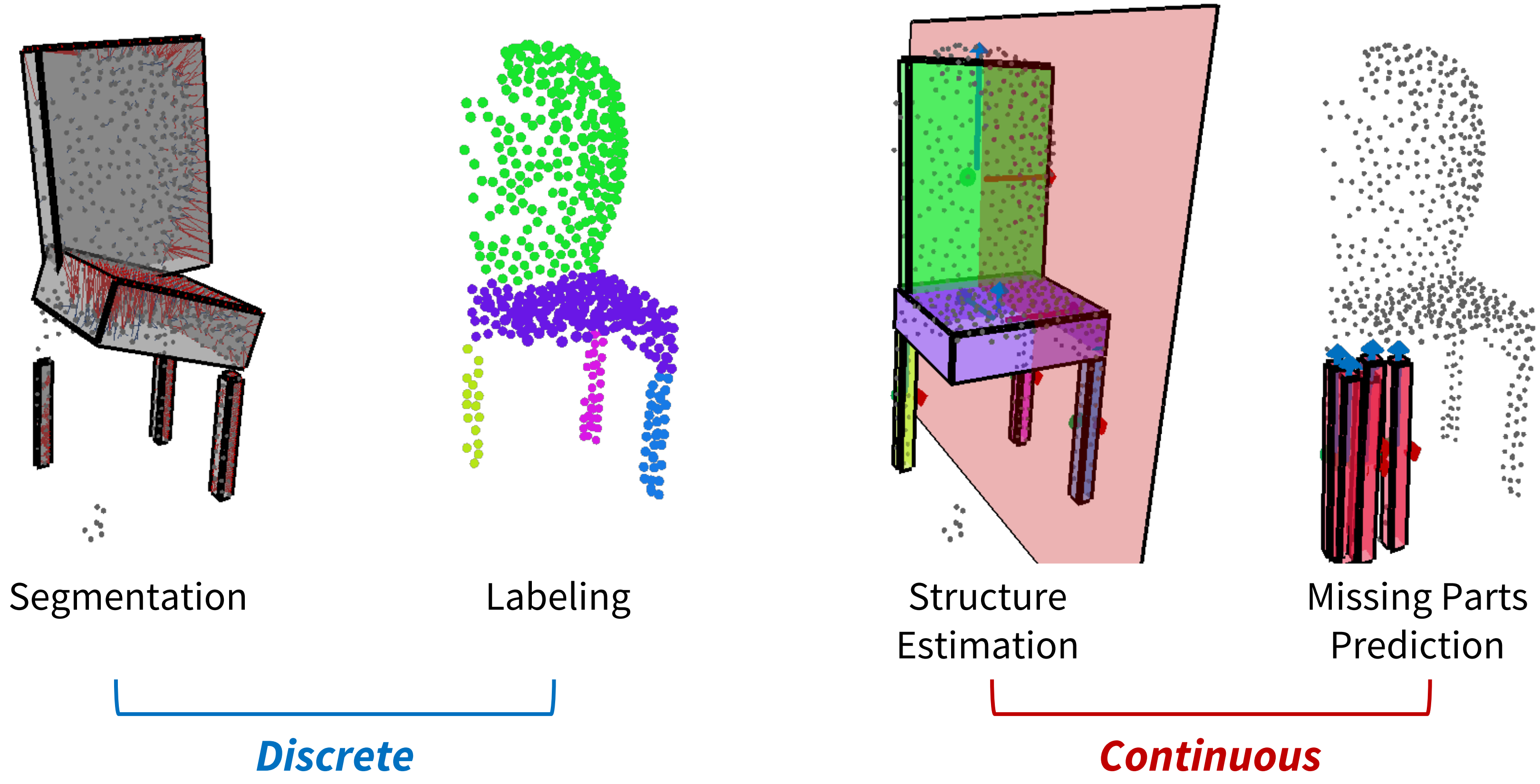
e.g.	Reflectional symmetry:	back, seat, armrests (pairs).
	Rotational symmetry:	column, legs.



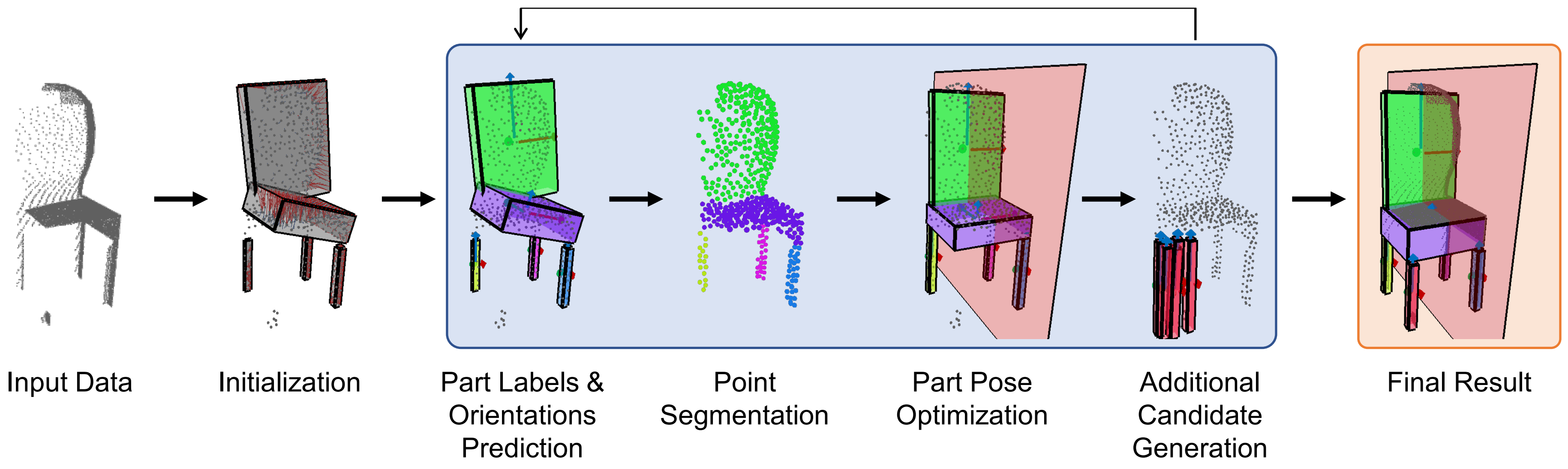
The Pipeline



Inference

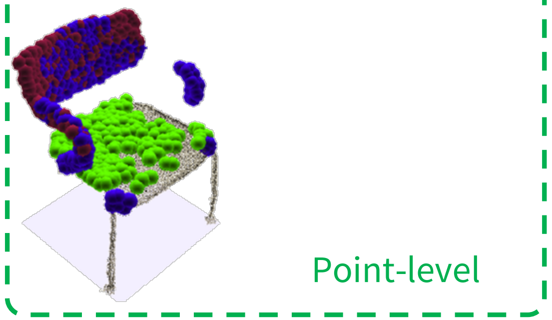

Segmentation
Labeling
Structure Estimation
Missing Parts Prediction
Discrete
Continuous

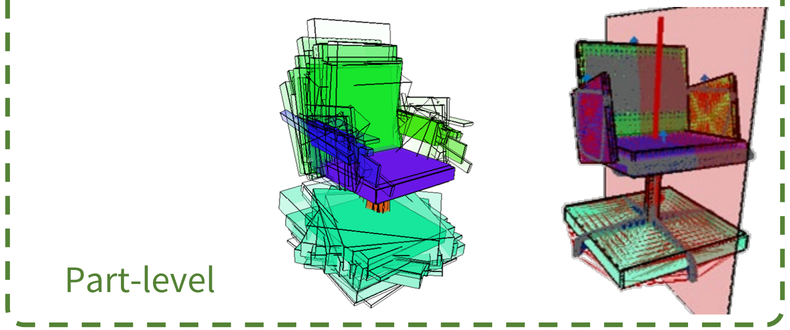
Inference Time



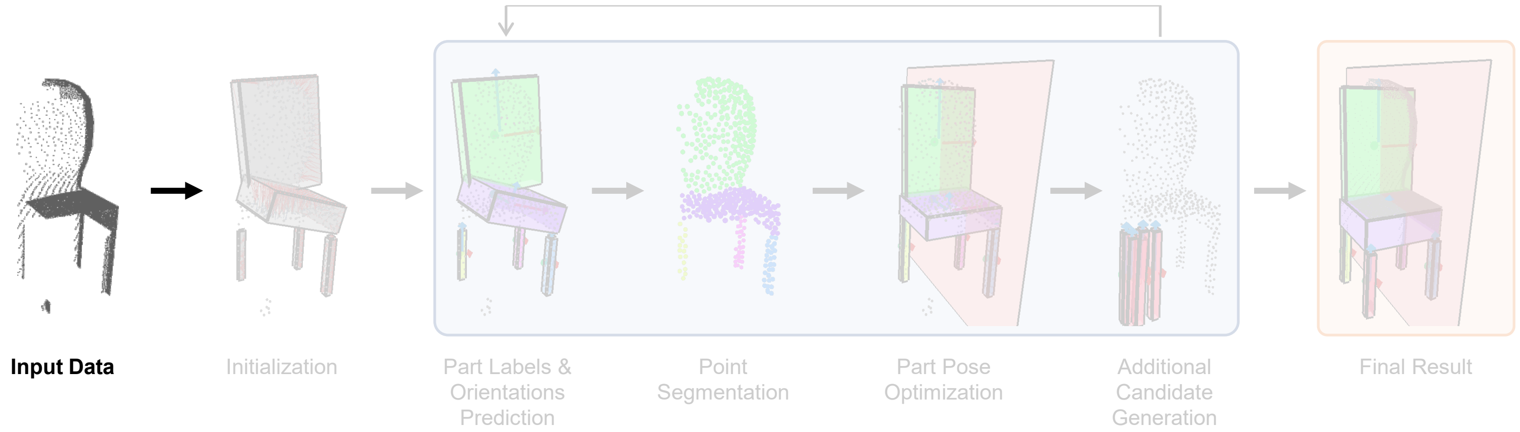
Energy function

$$\boldsymbol{E} = E_{pnt} + E_{smooth} + E_{SMD} + E_{pair} + E_{symm}$$

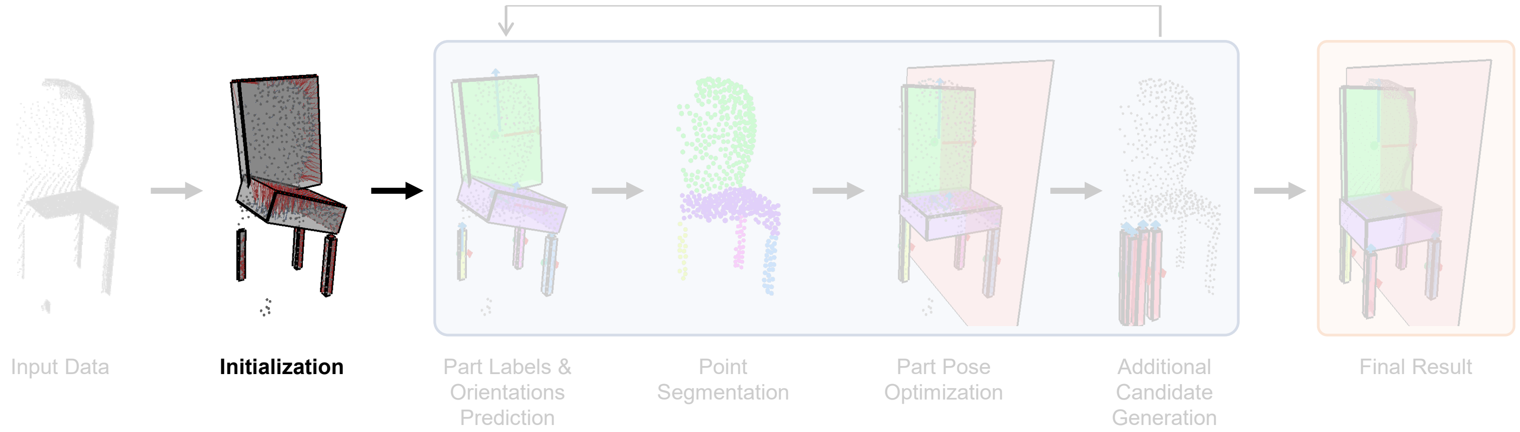




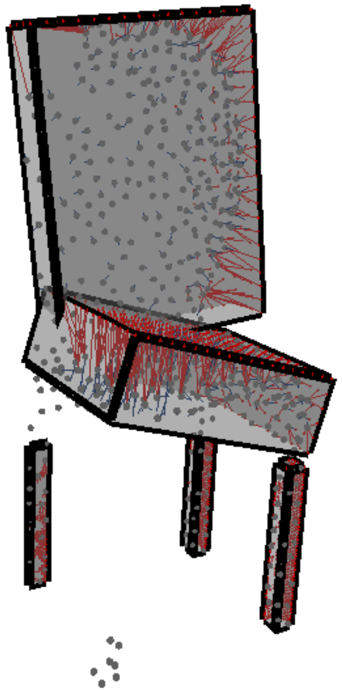
Inference Pipeline



Inference Pipeline

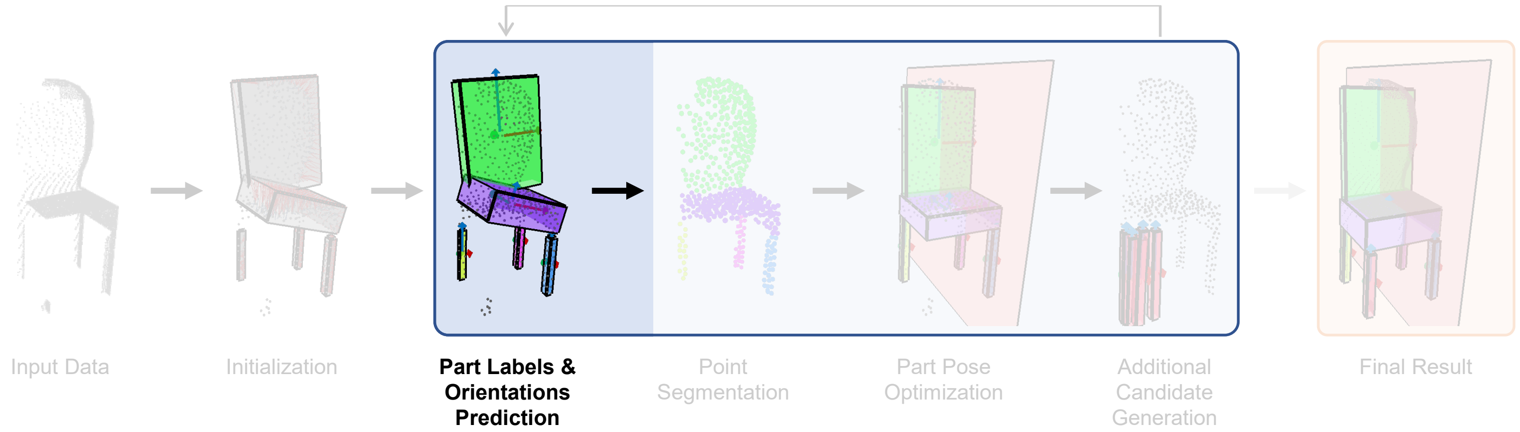


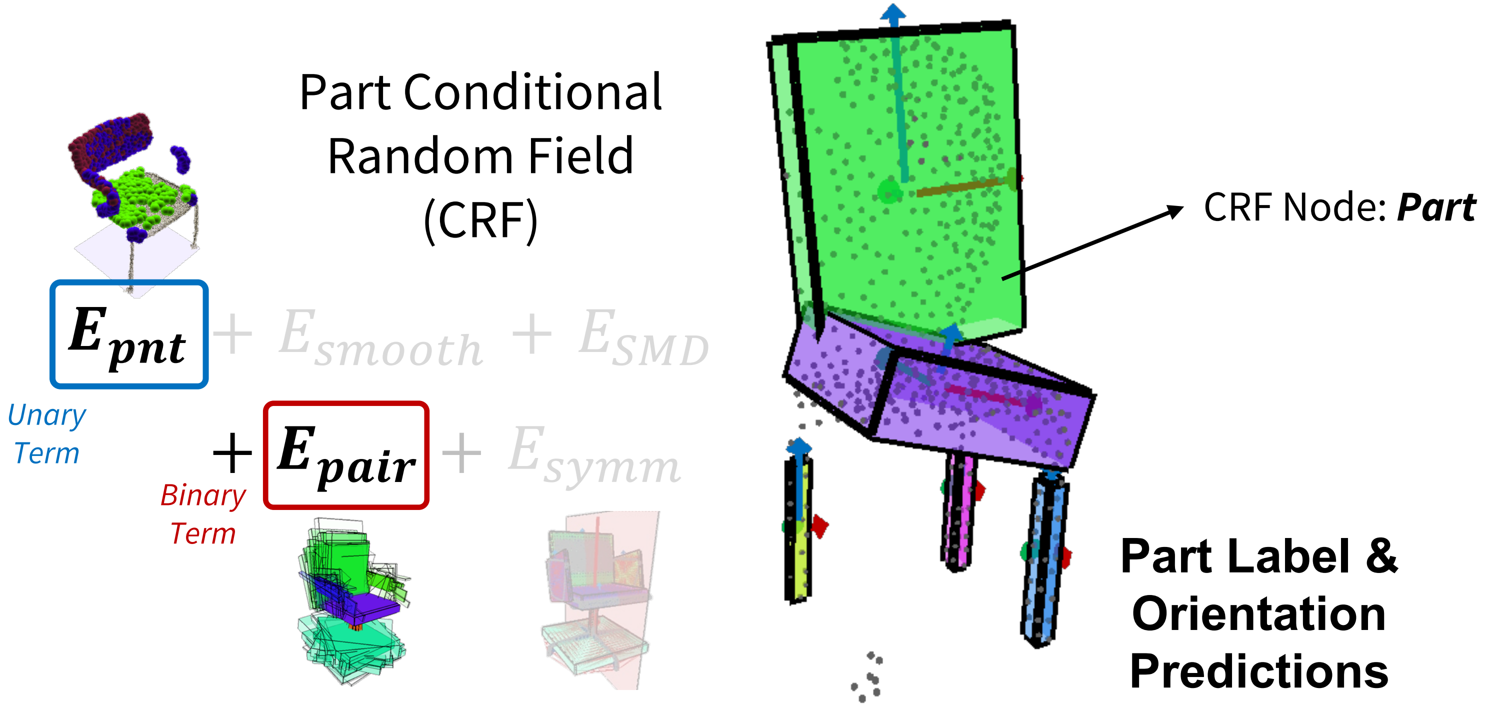
Clustering



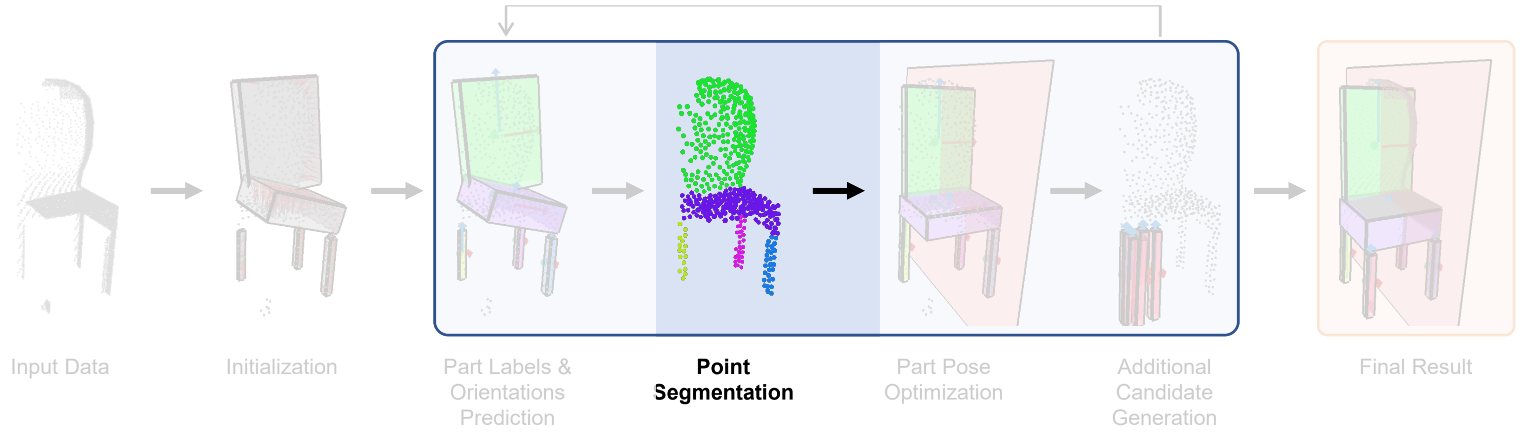
Initialization

Inference Pipeline

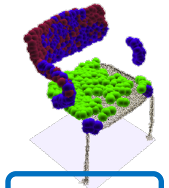




Inference Pipeline



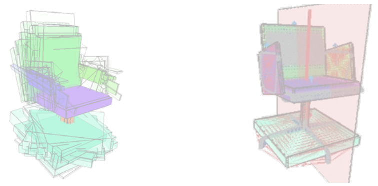
Point Conditional Random Field (CRF)

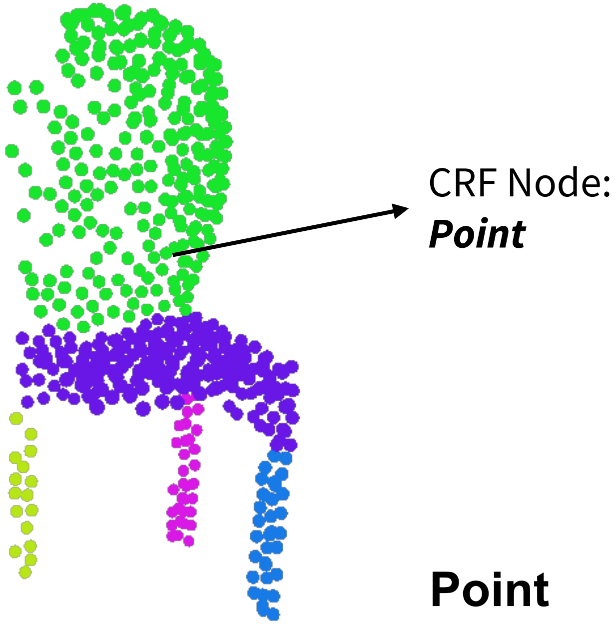


$$E_{pnt} + E_{smooth} + E_{SMD}$$

Unary Term | *Binary Term* | *Unary Term*

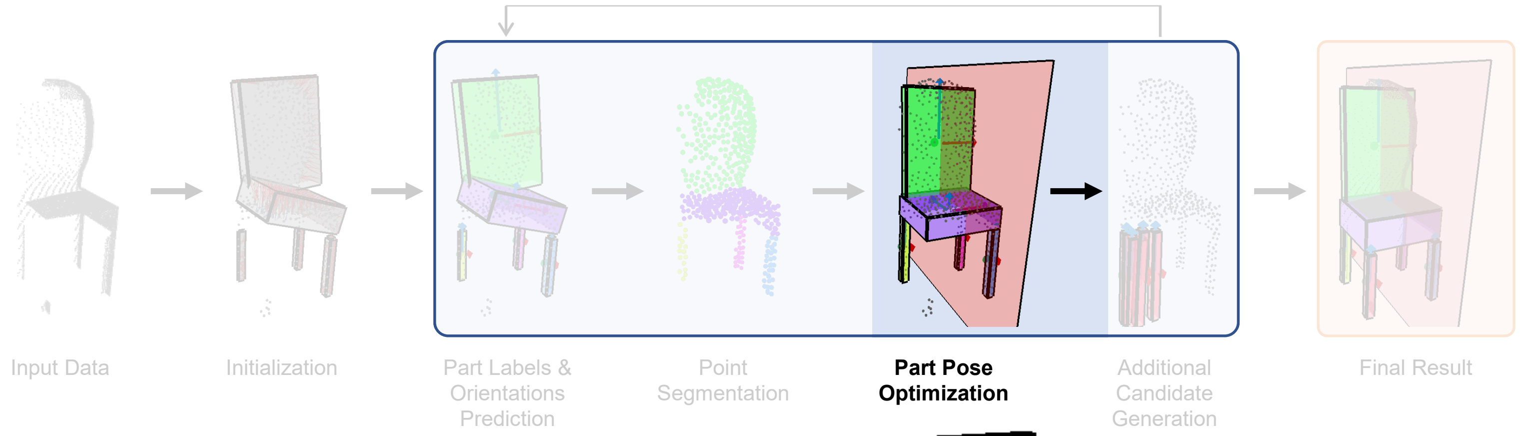
$$+ E_{pair} + E_{symm}$$





Point Segmentation

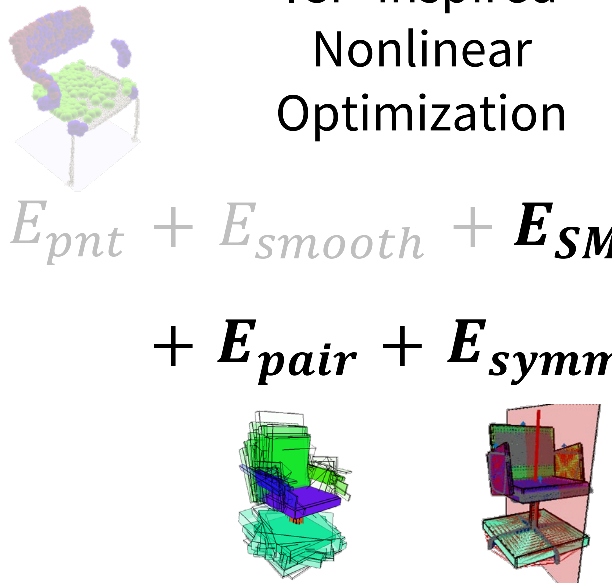
Inference Pipeline

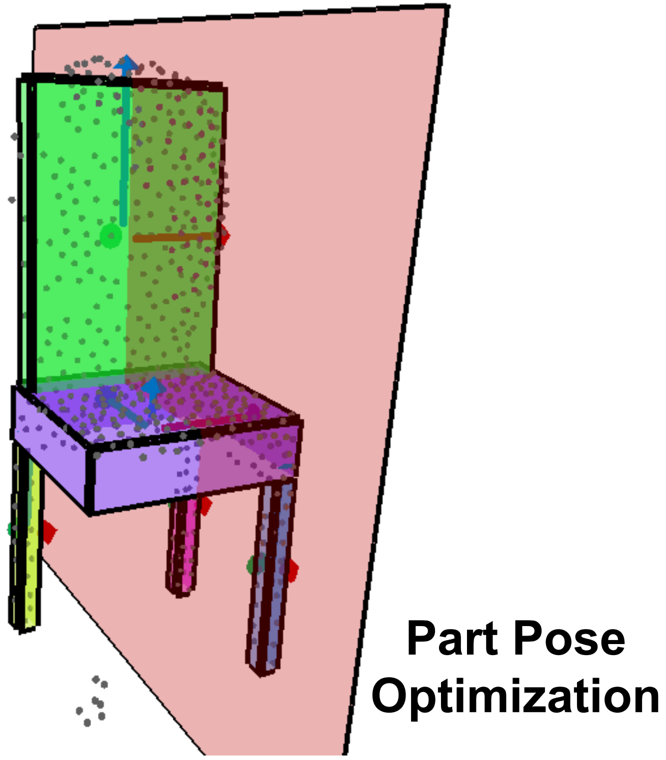


ICP-inspired Nonlinear Optimization

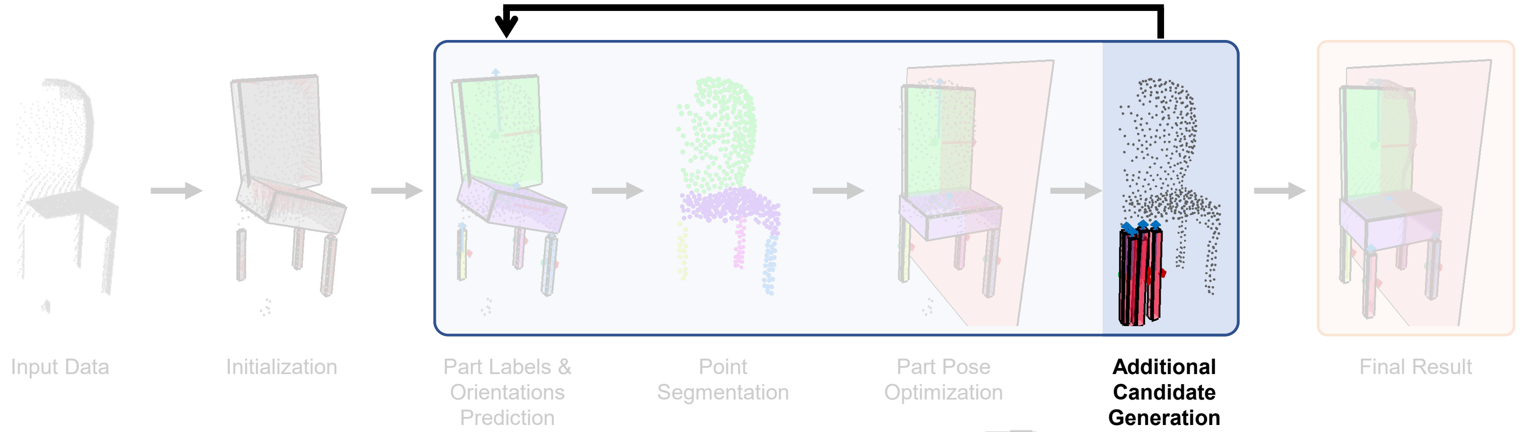
$$E_{pnt} + E_{smooth} + \boldsymbol{E_{SMD}}$$

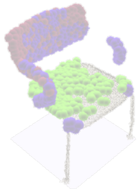
$$+ \boldsymbol{E_{pair}} + \boldsymbol{E_{symm}}$$



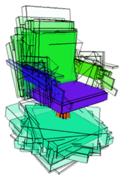


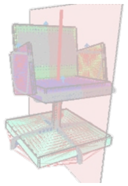
Inference Pipeline

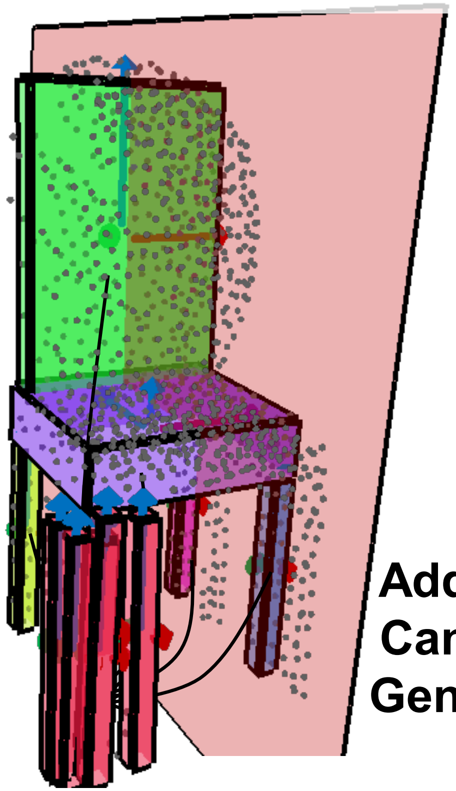




$$E_{pnt} + E_{smooth} + E_{SMD} + \boldsymbol{E_{pair}} + E_{symm}$$

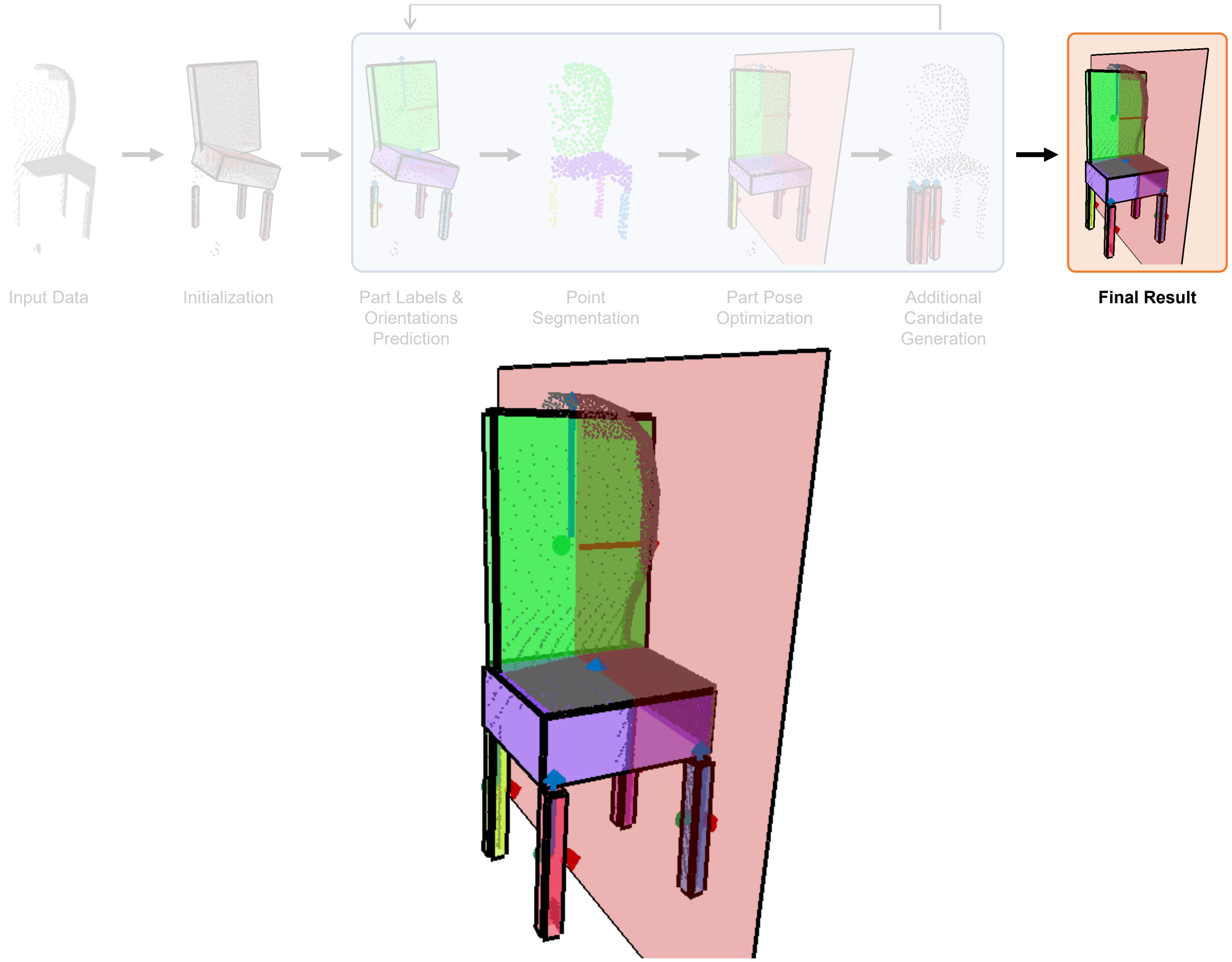






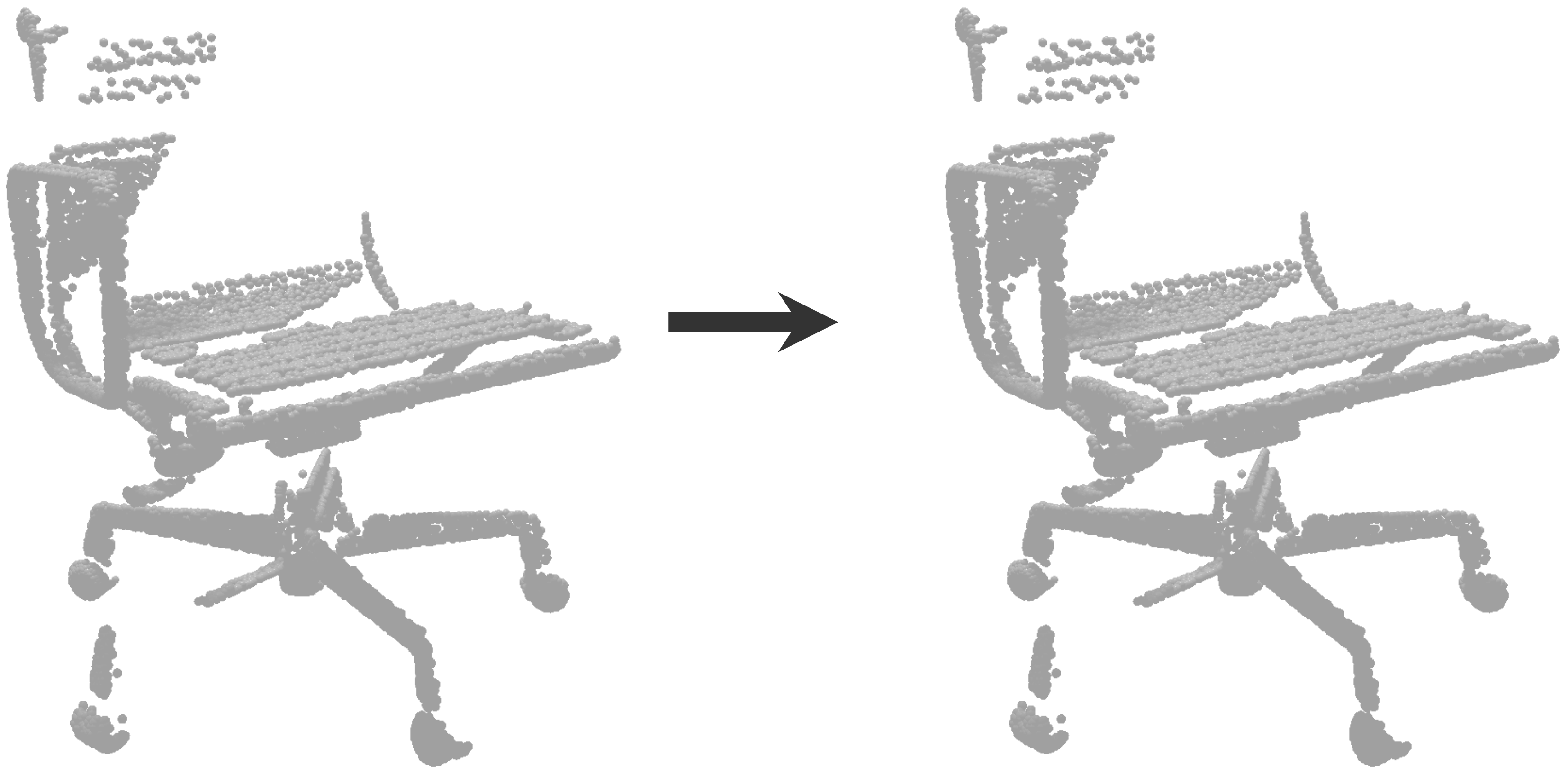
Additional Candidate Generation

Inference Pipeline



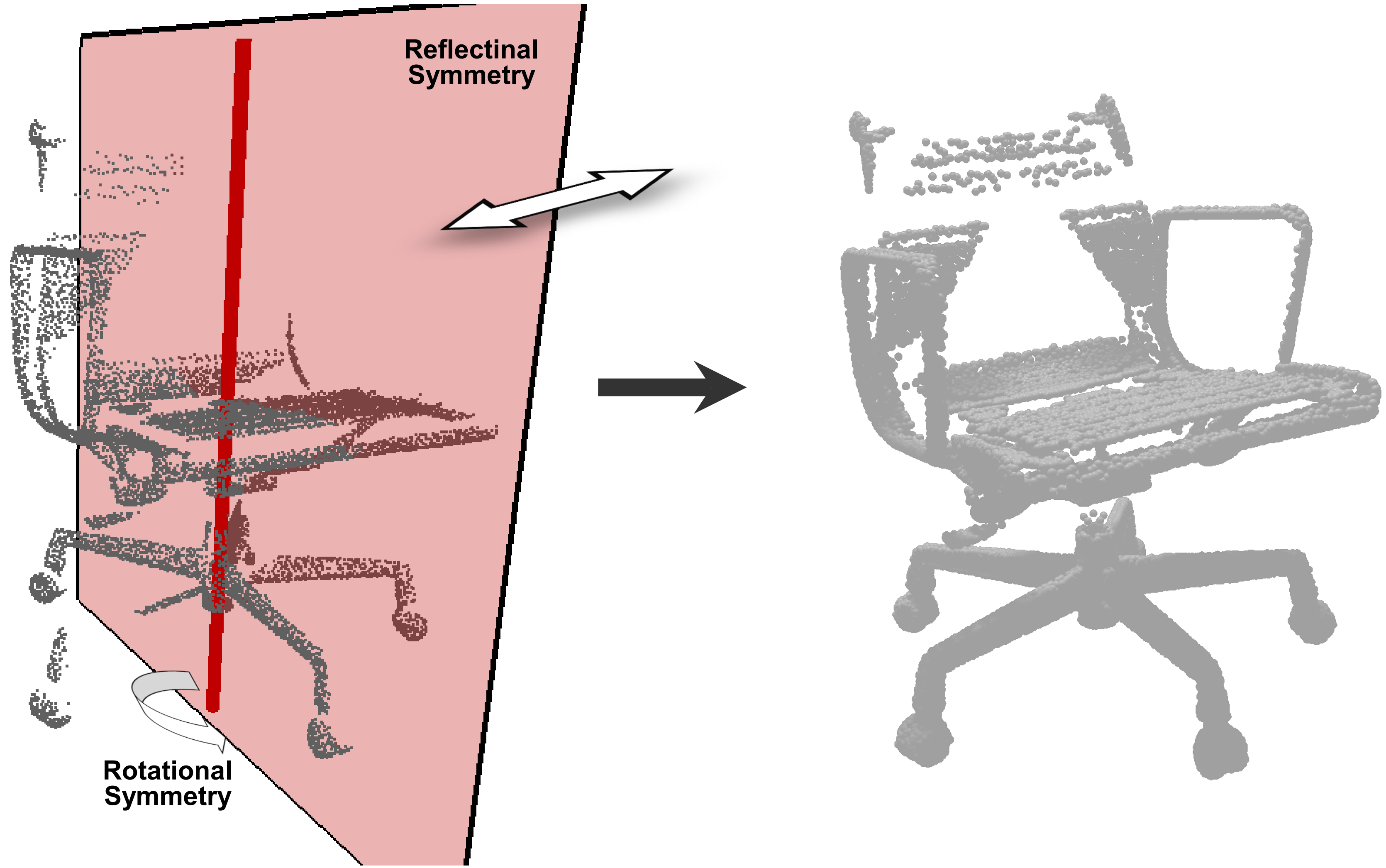
Completion Strategy

- Input



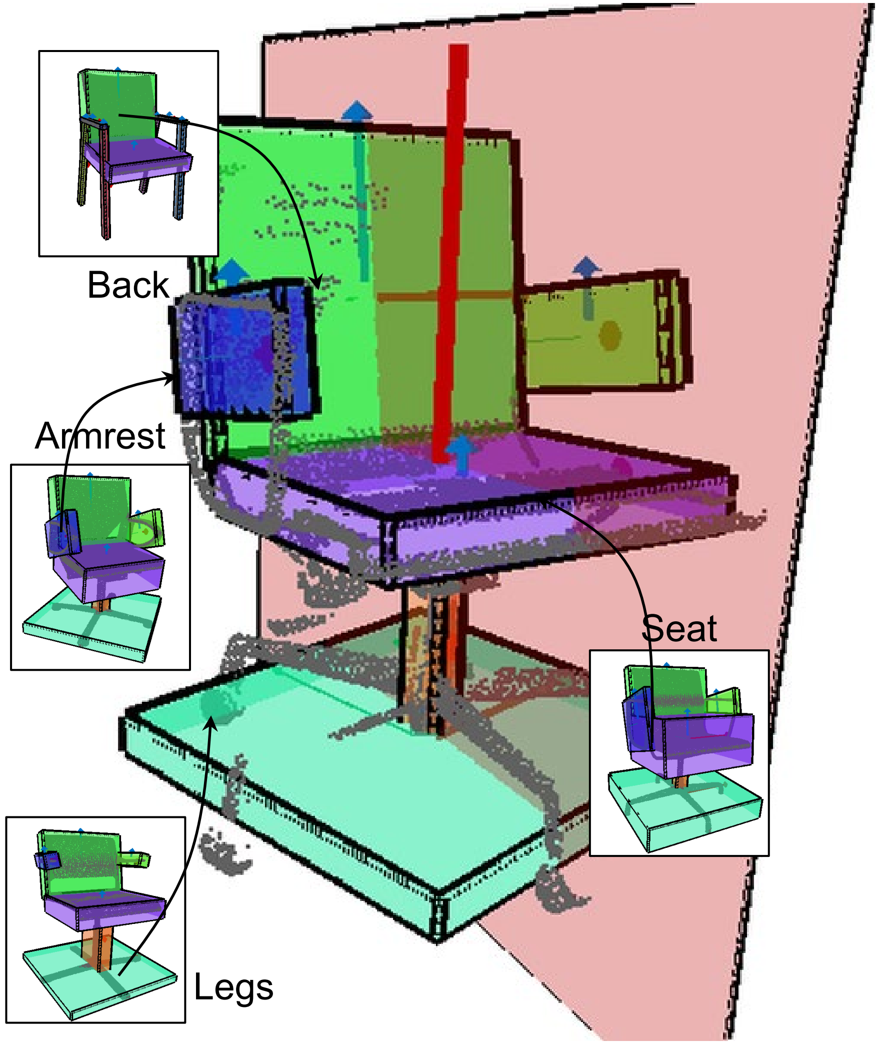
Completion Strategy

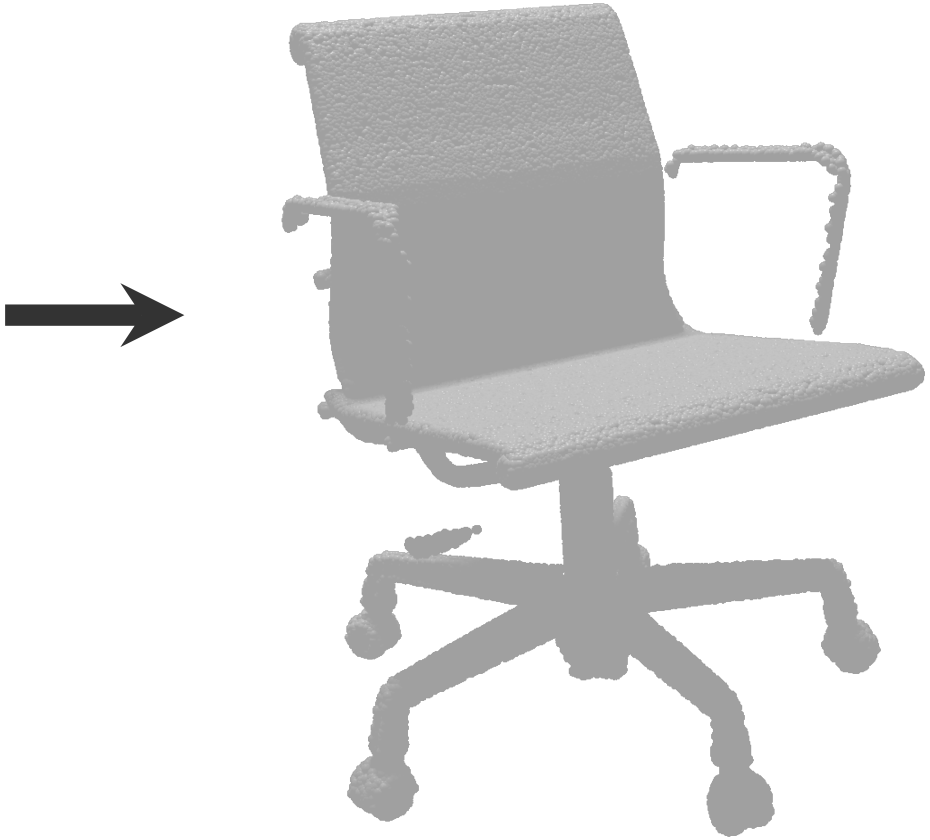
- Input → Symmetry



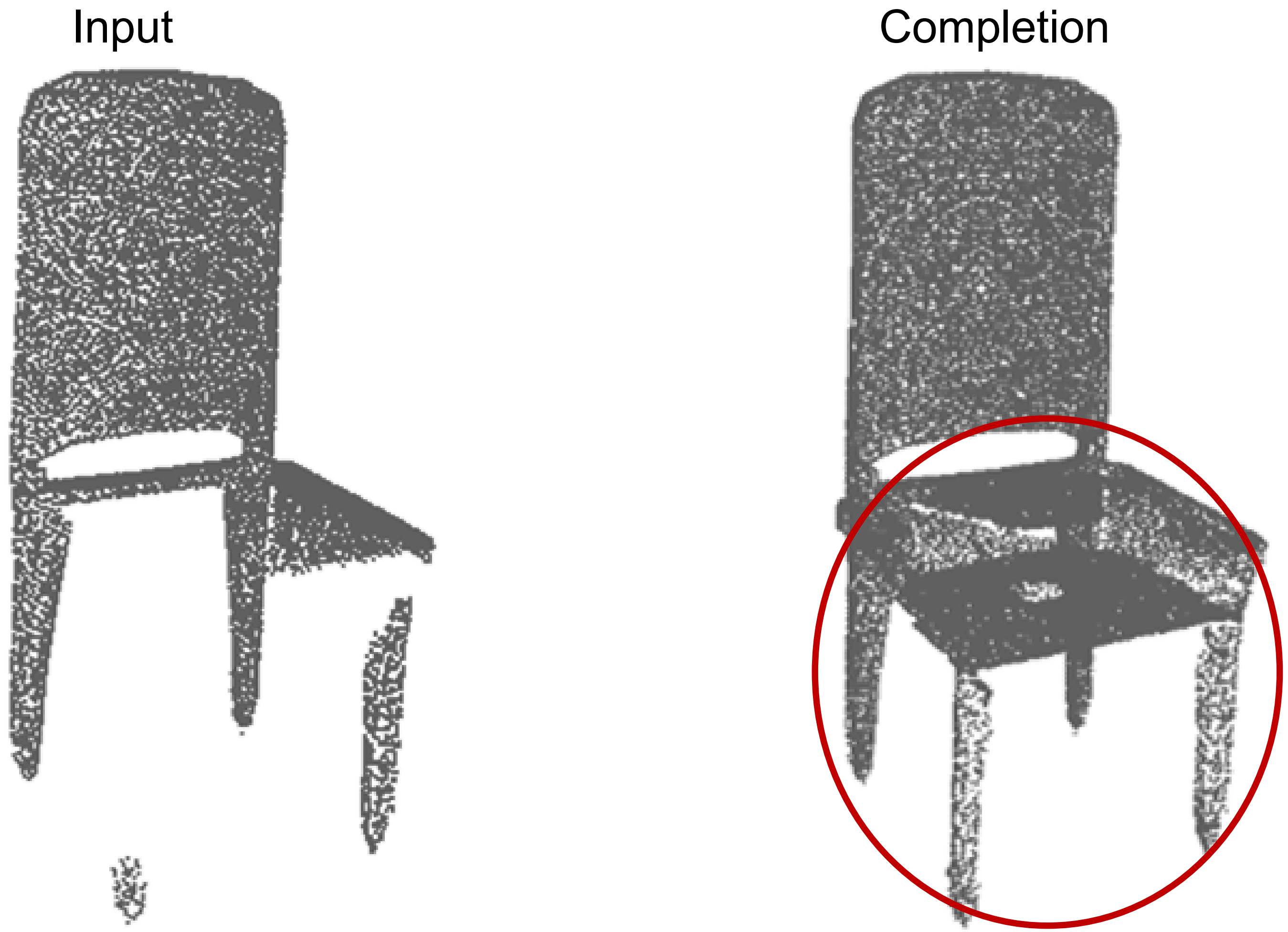
Completion Strategy

- Input → Symmetry → Database

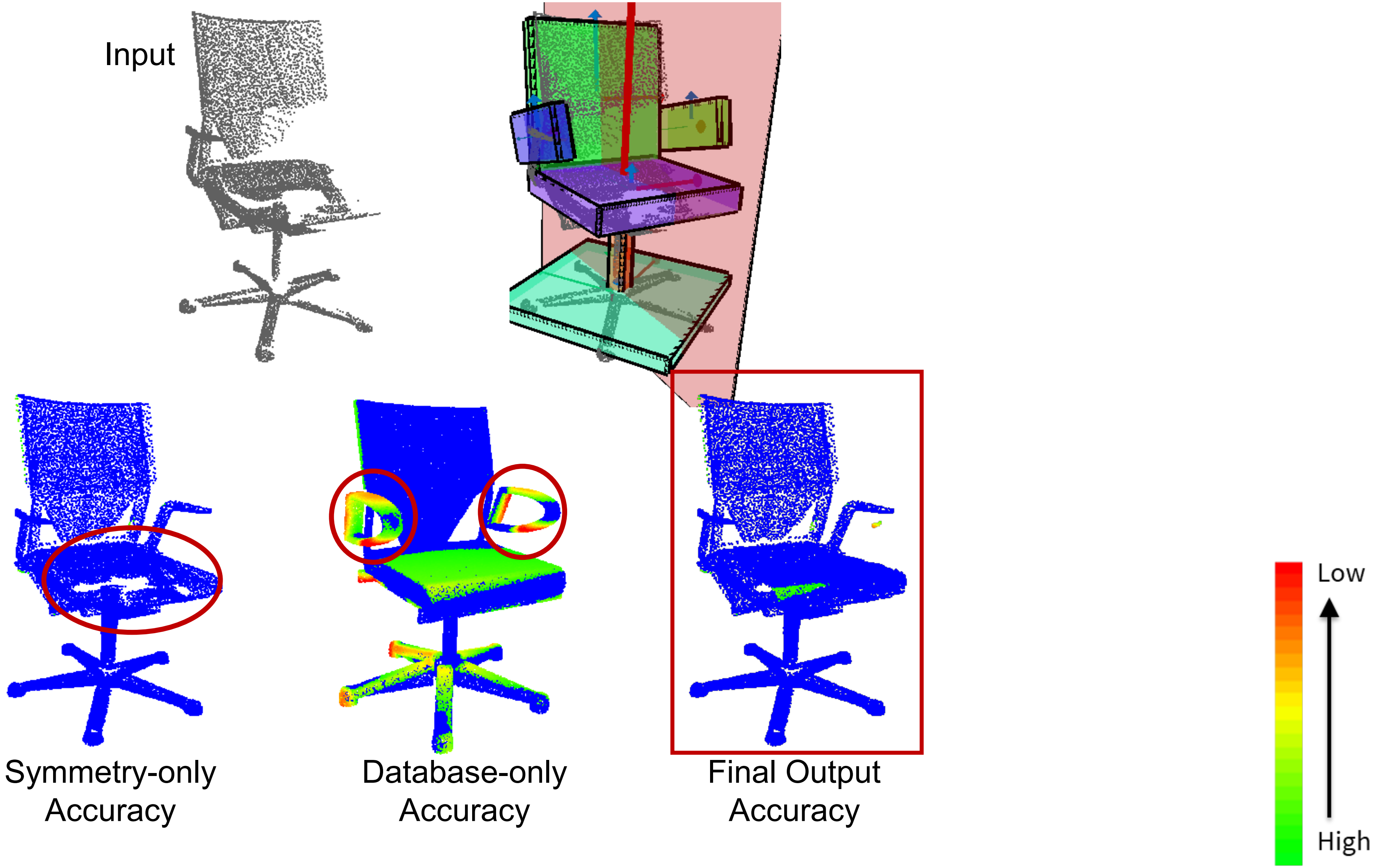




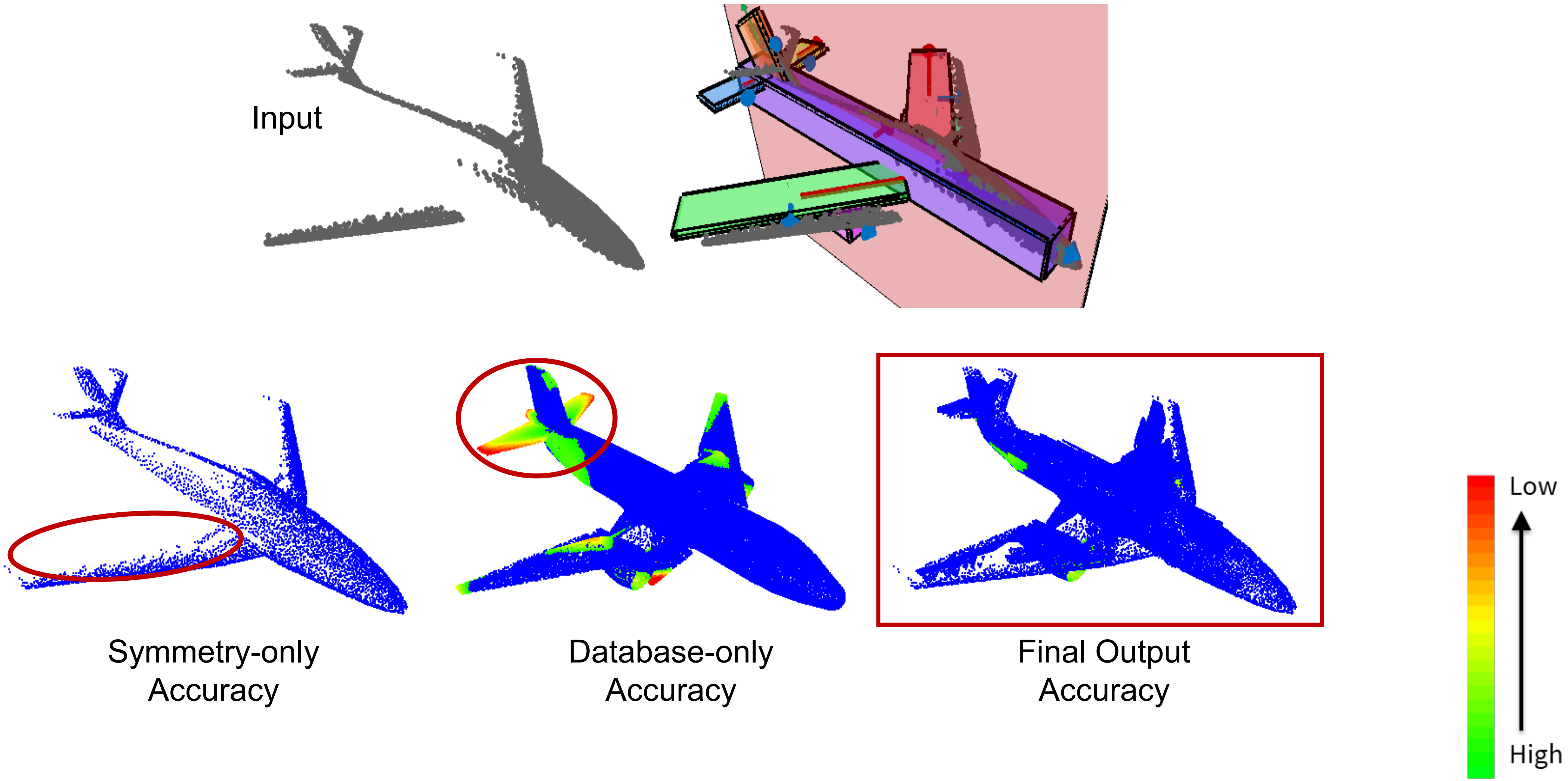
Qualitative Results



Comparison


Input
Symmetry-only
Accuracy
Database-only
Accuracy
Final Output
Accuracy
Low
High

Comparison


Input
Symmetry-only
Accuracy
Database-only
Accuracy
Final Output
Accuracy
Low
High

Real Scans

Input

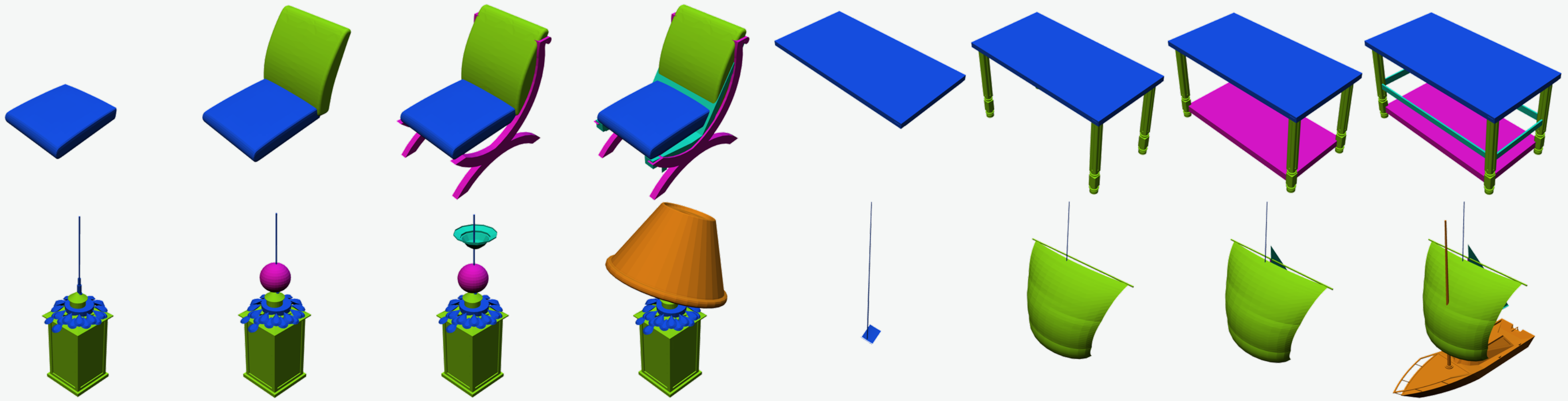
Output

Object Synthesis by Part Assembly

Minhyuk Sung, Hao Su, Vova Kim, Siddhartha Chaudhuri, Leonidas Guibas, Siggraph Asia ‘17
Minhyuk Sung, Anastasia Dubrovina, Vova Kim, , Leonidas Guibas, SGP ‘18

SIGGRAPH ASIA 2017 BANGKOK
CONFERENCE 27 – 30 November 2017
EXHIBITION 28 – 30 November 2017
BITEC, Bangkok, Thailand
SA2017.SIGGRAPH.ORG

ComplementMe: Weakly-Supervised Component Suggestions for 3D Modeling

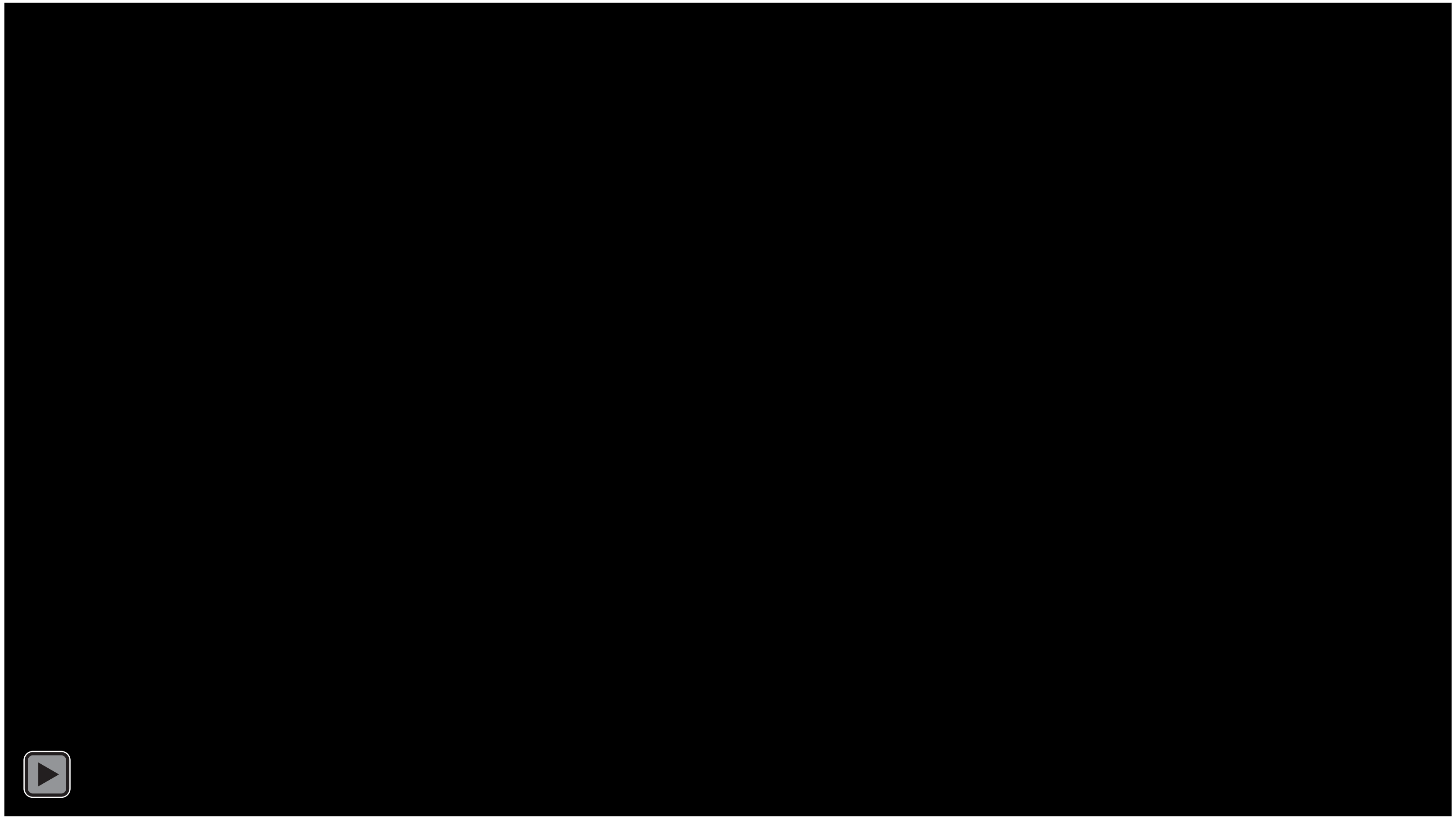


Minhyuk Sung[1], Hao Su[1,2], Vladimir G. Kim[3], Siddhartha Chaudhuri[4], Leonidas Guibas[1]

[1]Stanford University [2]University of California San Diego [3]Adobe Research [4]IIT Bombay

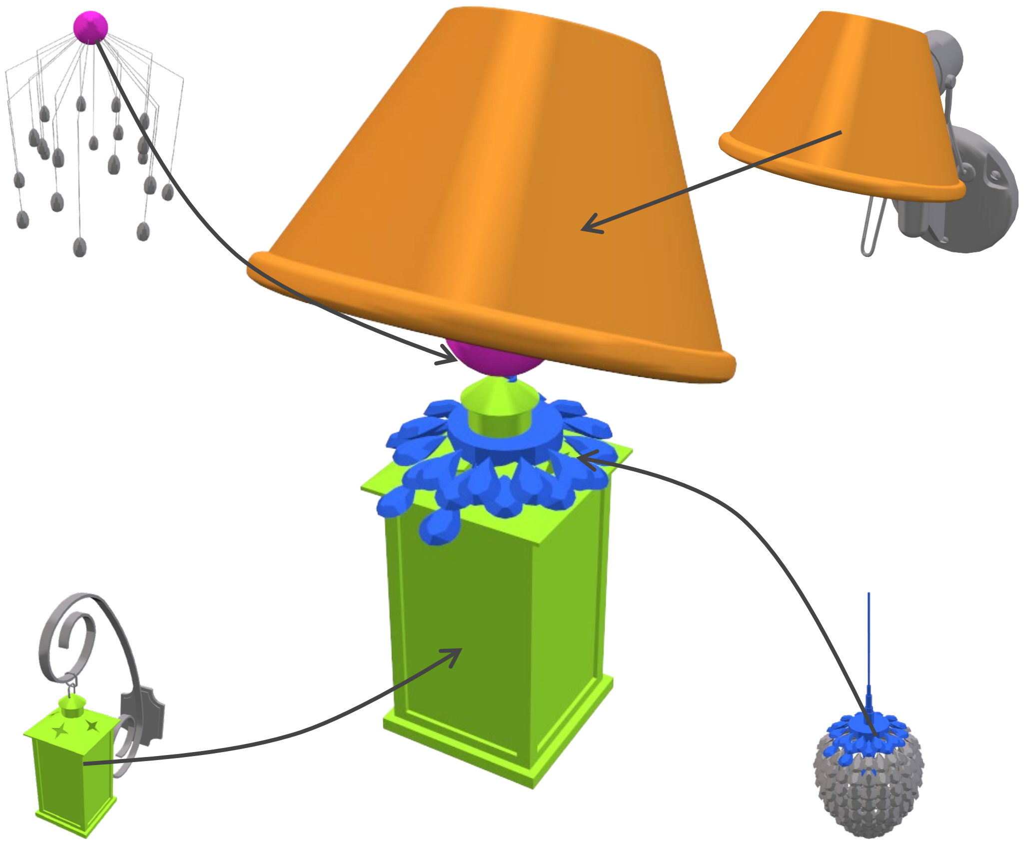
Motivation

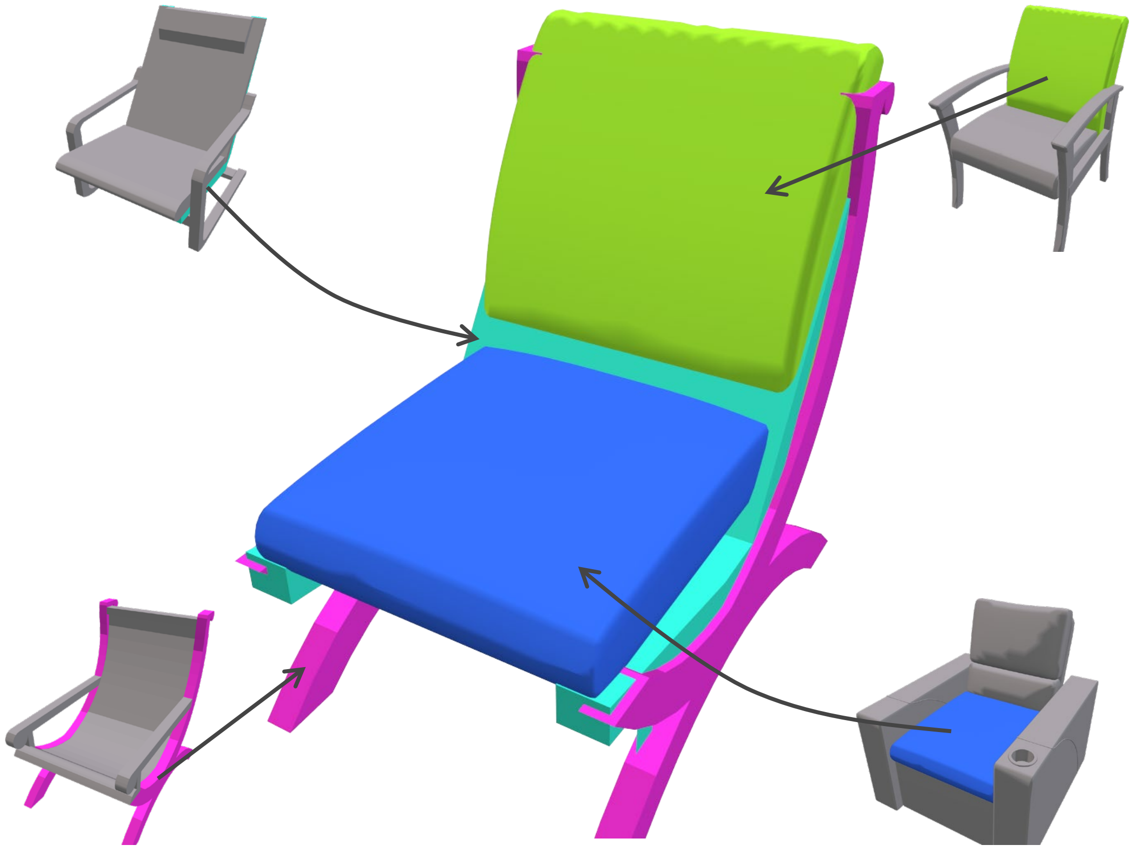
3D Modeling is time-consuming.



Composition-Based Modeling

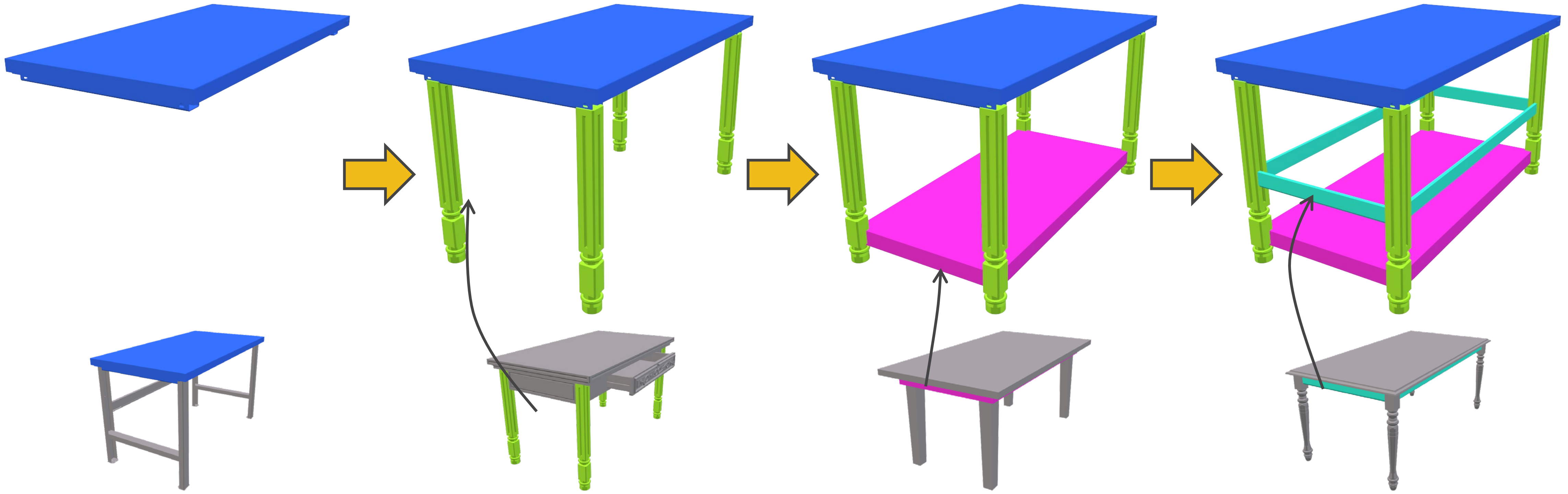
Create a shape by assembling components of 3D models in a large-scale repository.





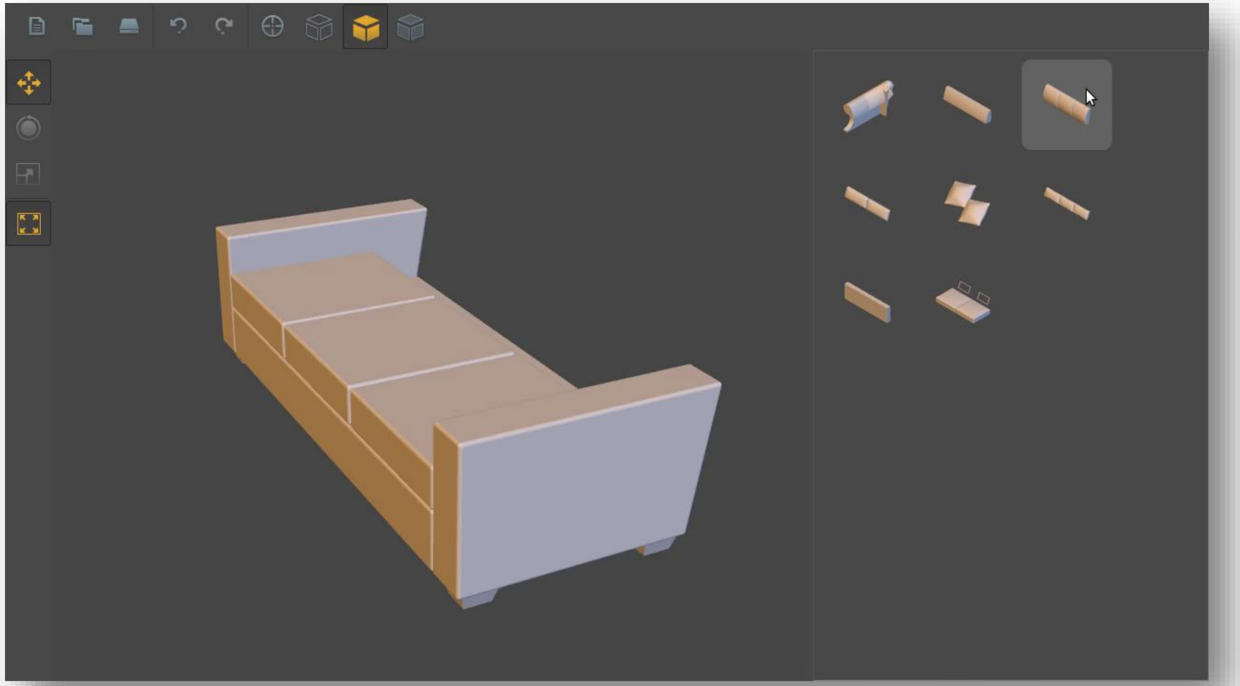
Composition-Based Modeling

- Propose an iterative *assembly* system.
- Suggest *complementary* parts and their locations at each time.

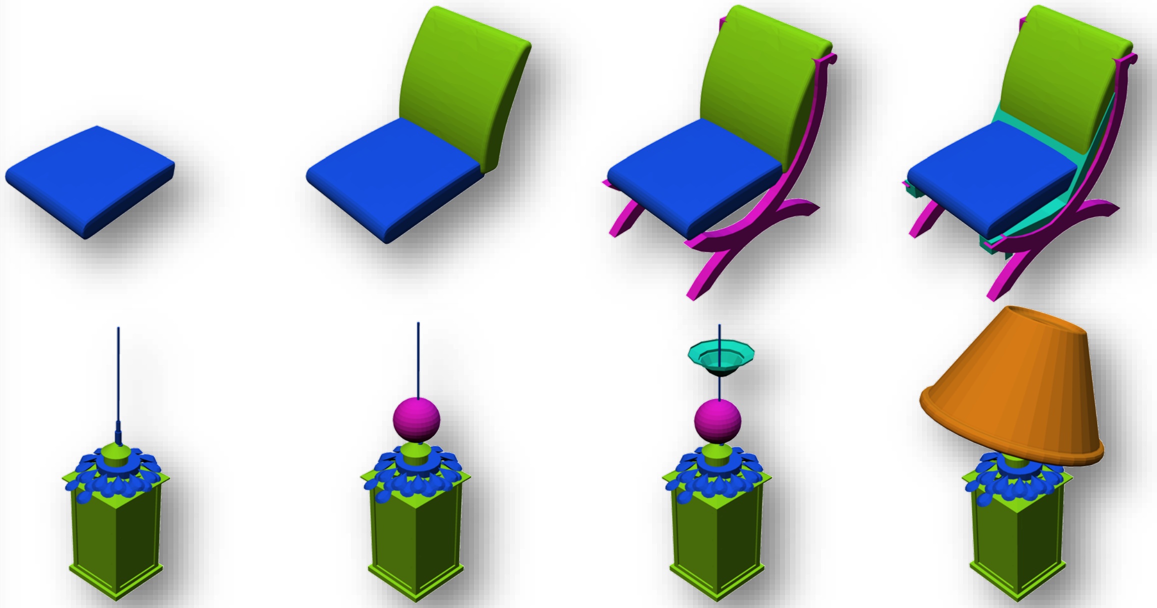


Composition-Based Modeling

Interactive design interface

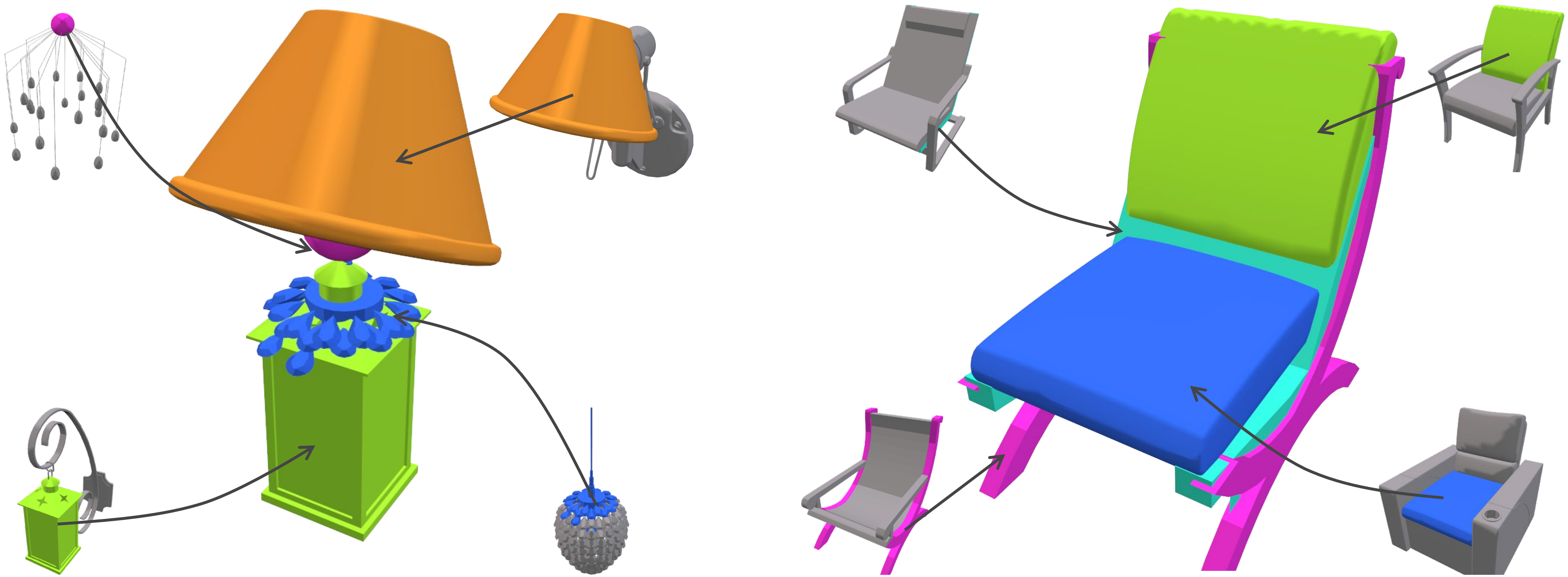


Automatic shape synthesis



Modeling by Assembly

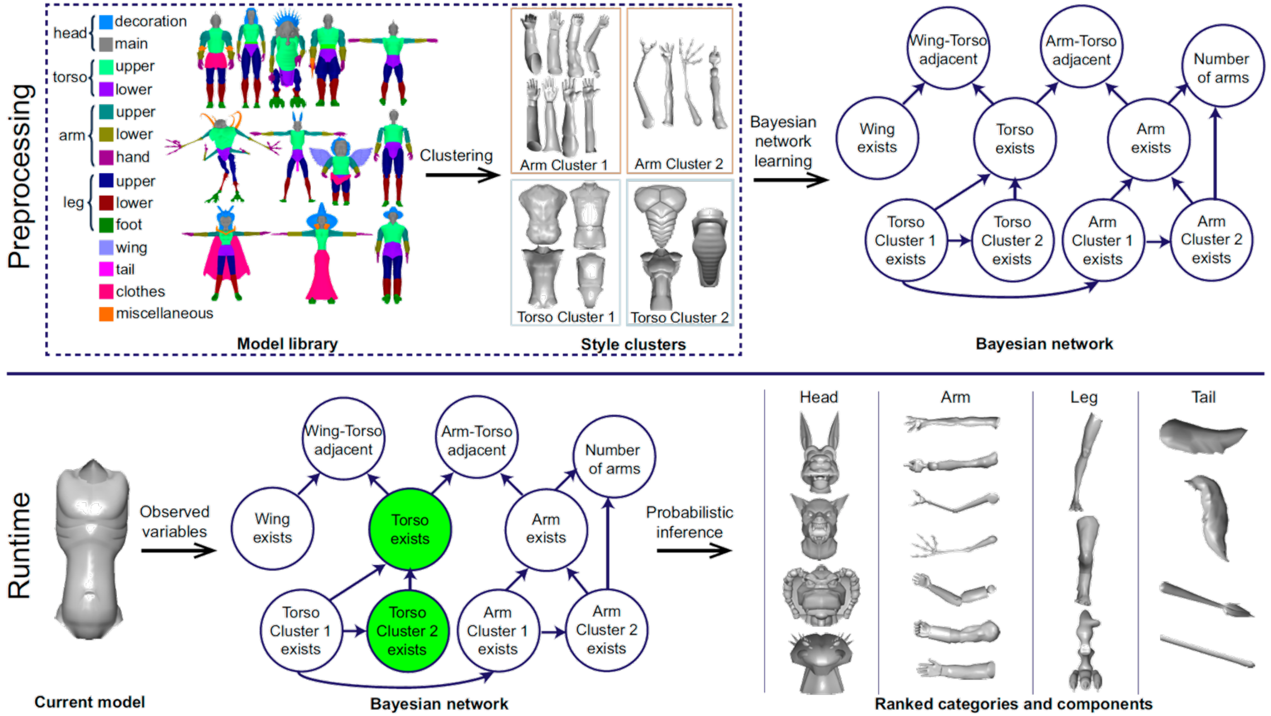
Create a high quality model by predicting *mid-level* information and *reusing* geometries of parts in the database.



Previous Work

Requires consistent part labels.

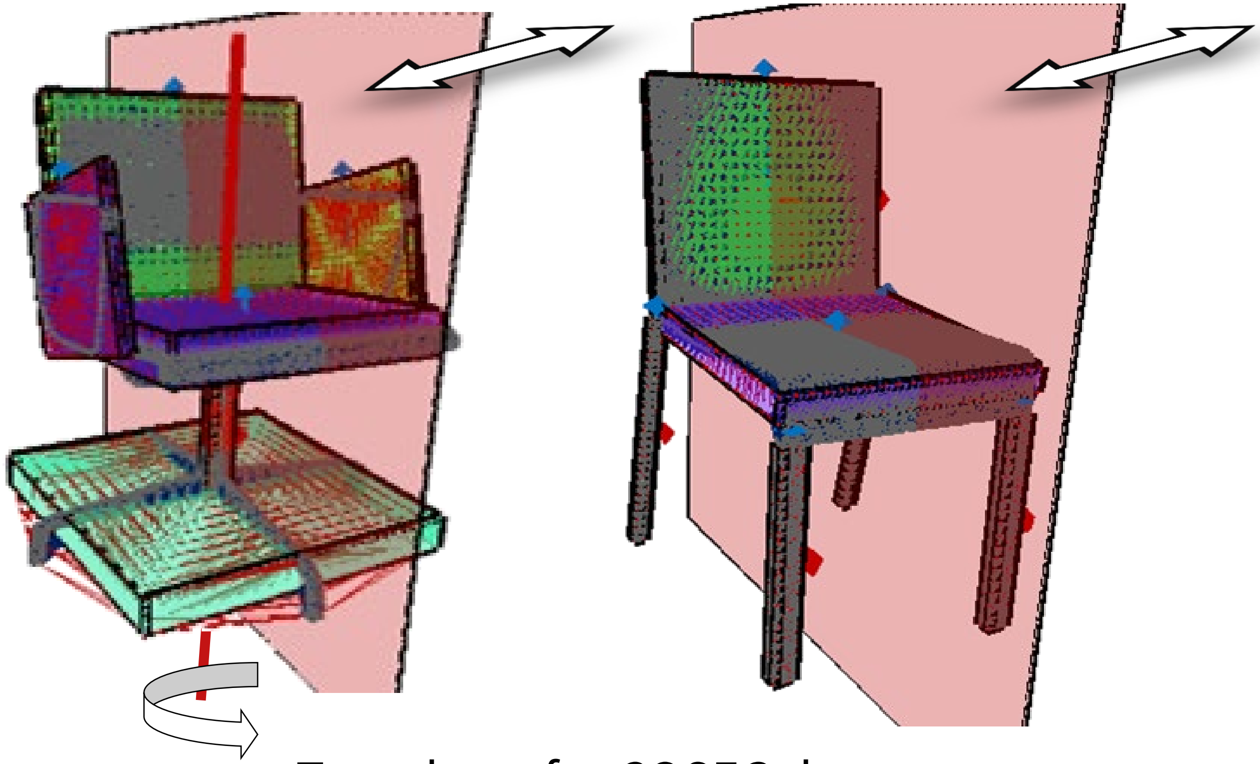
Chaudhuri et al., 2013



Chaudhuri et al., 2011

Limitations

Requires consistent part labels.

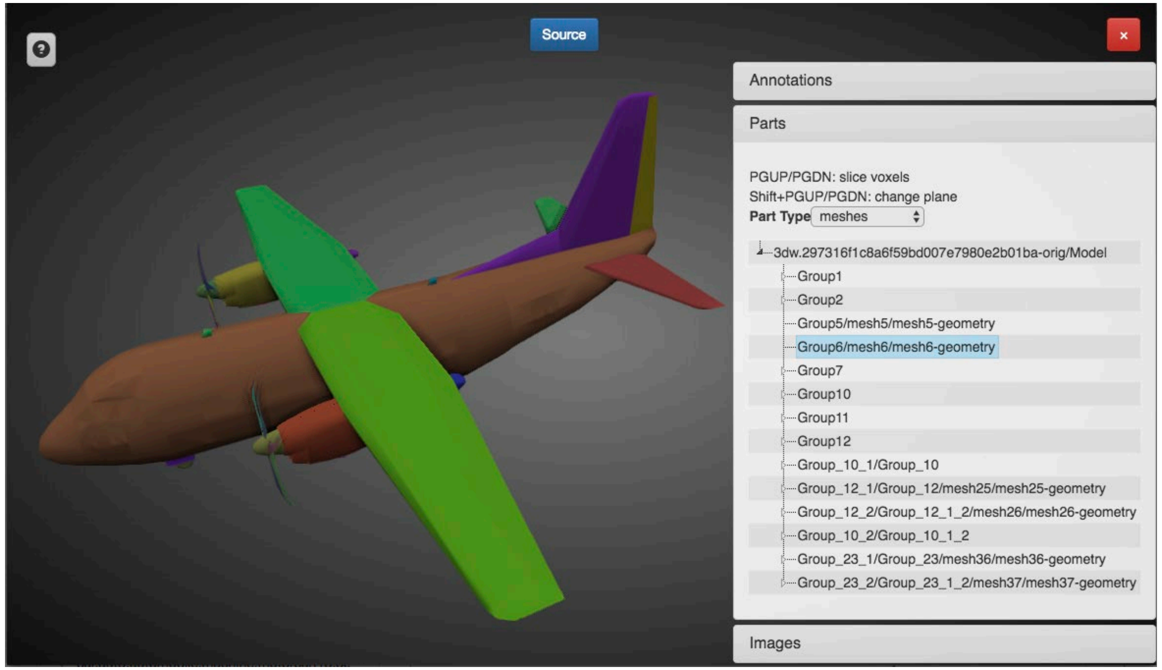


Templates for COSEG dataset
(~400 models)

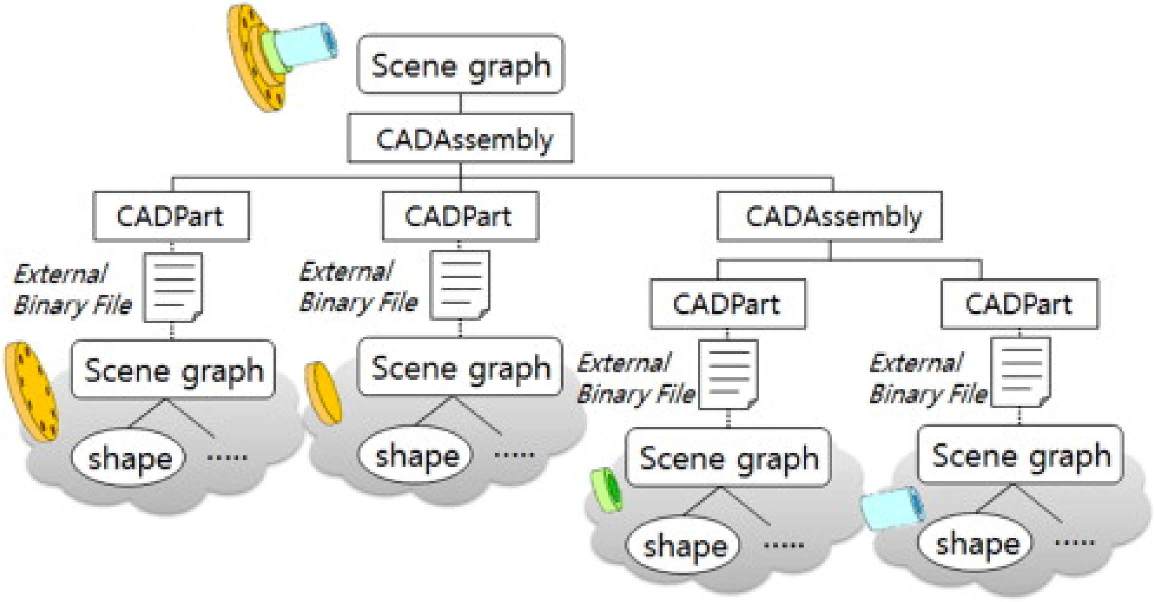
ShapeNet Dataset
(~3,000,000 models)

Observations

CAD data include *scene graphs*:
Part geometry + Hierarchical structure



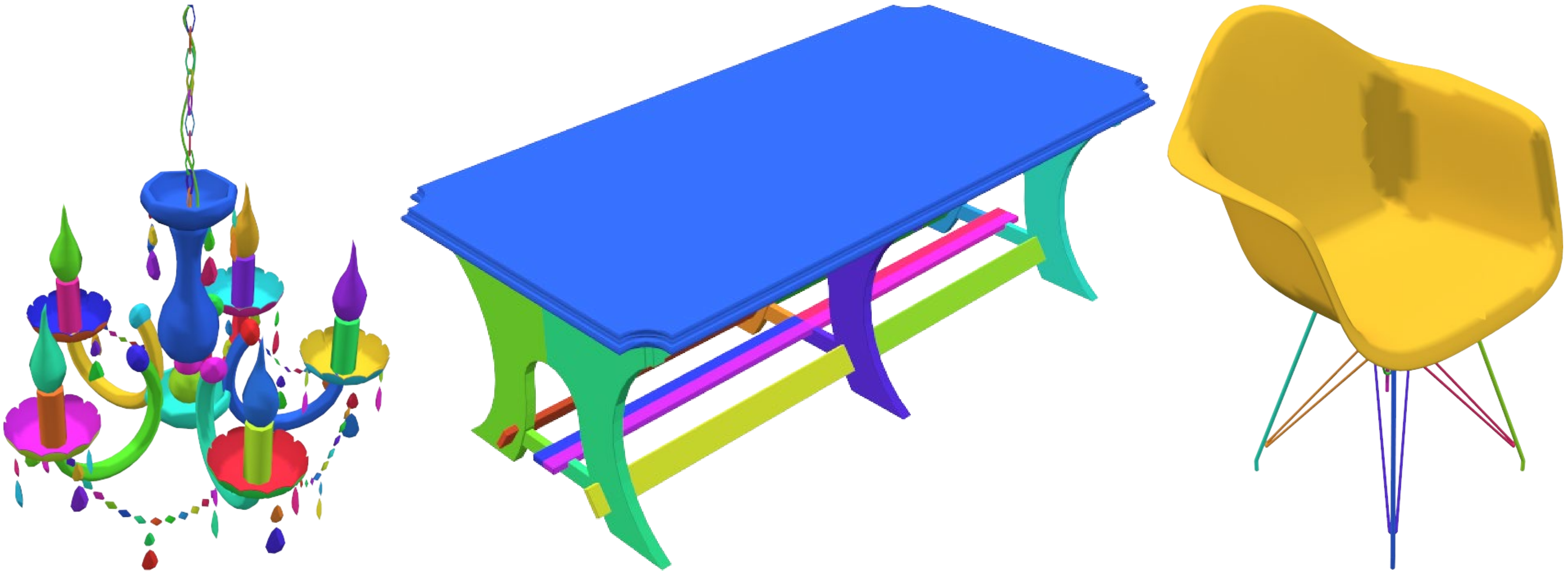
ShapeNet



Kim et al., 2015

Observations

(+) Provides natural part segmentations.



Observations

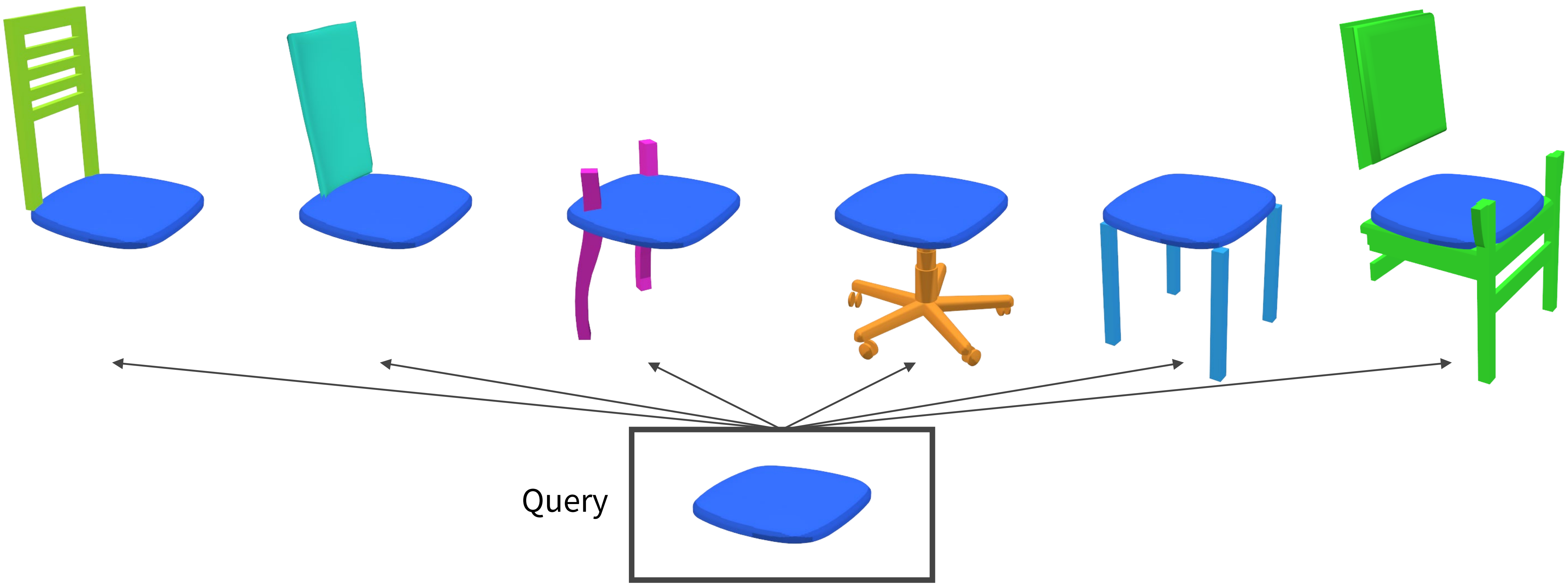
(+) Provides natural part segmentations.

(−) Inconsistent and unlabeled.

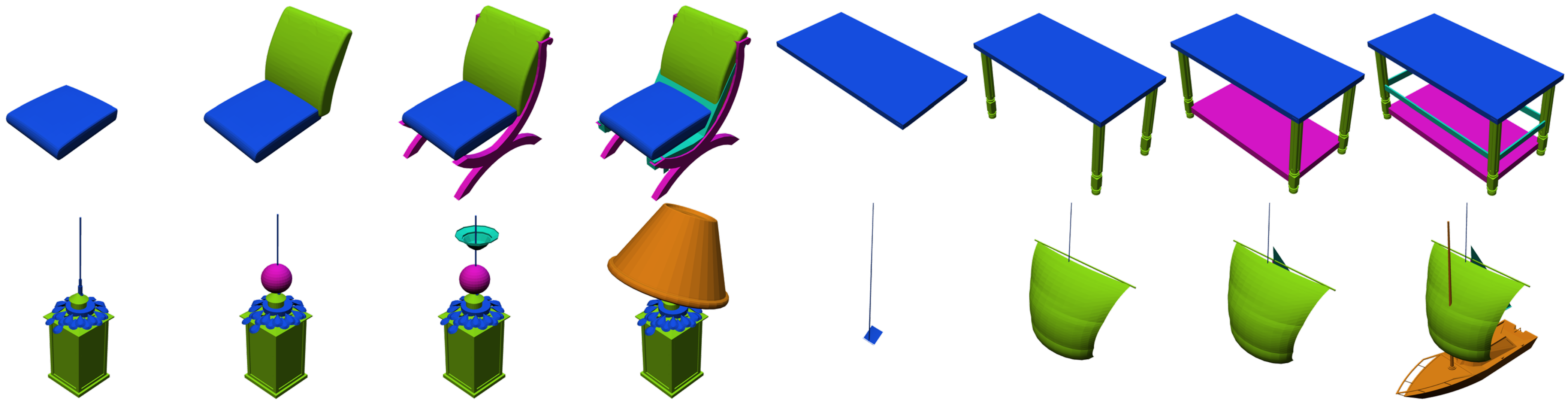
Goal

Predict complementary parts

using only geometric information

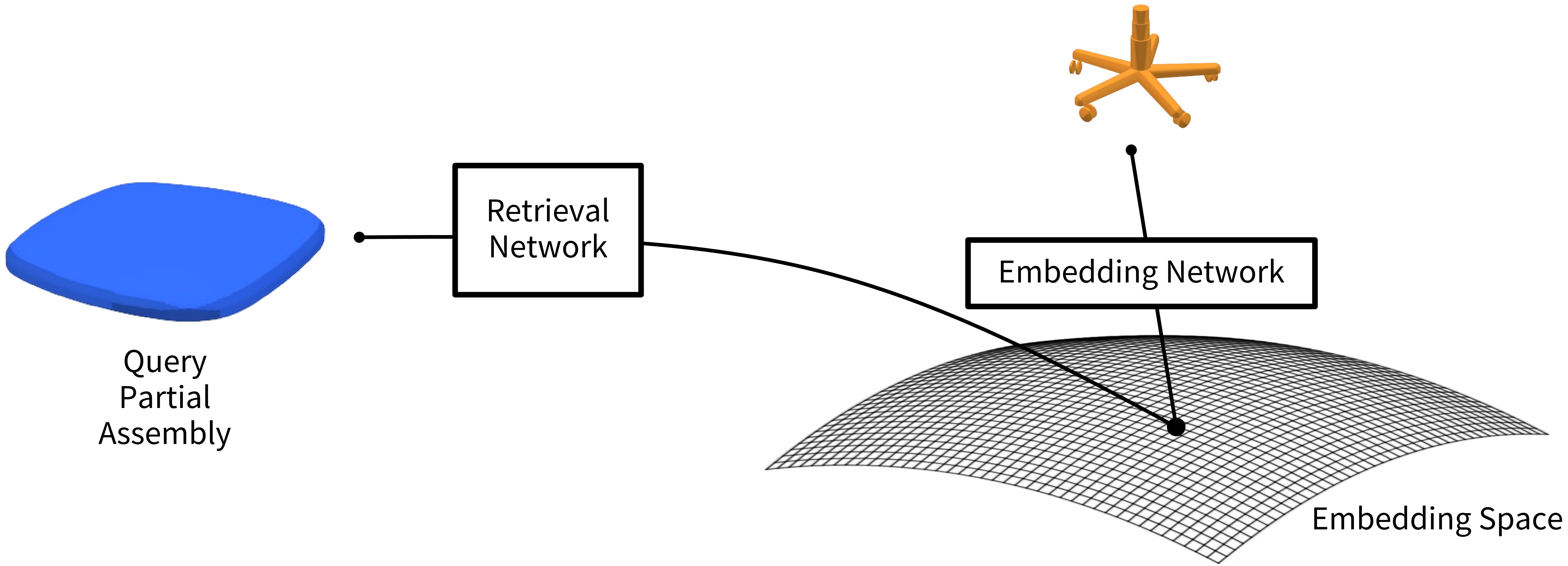


ComplementMe: Weakly-Supervised Component Suggestions



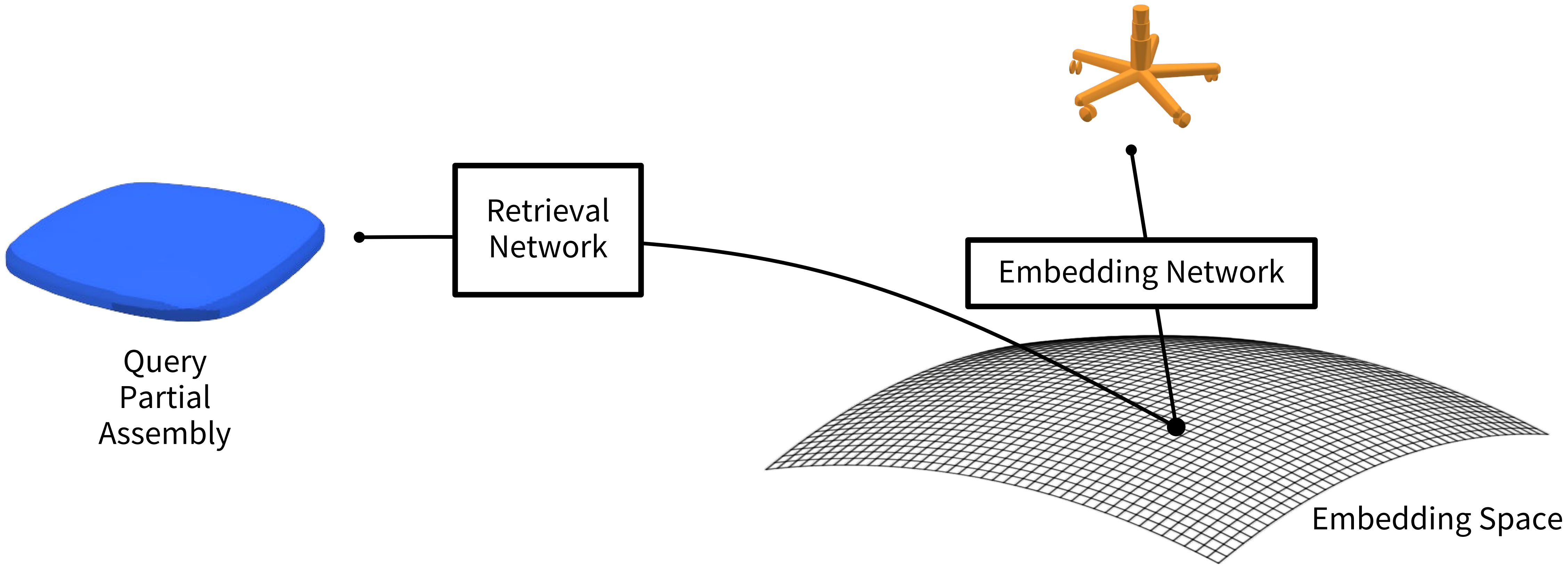
Retrieval and Embedding

Jointly map both the query shape and complements to an **embedding** space, and find the **nearest neighbors**.



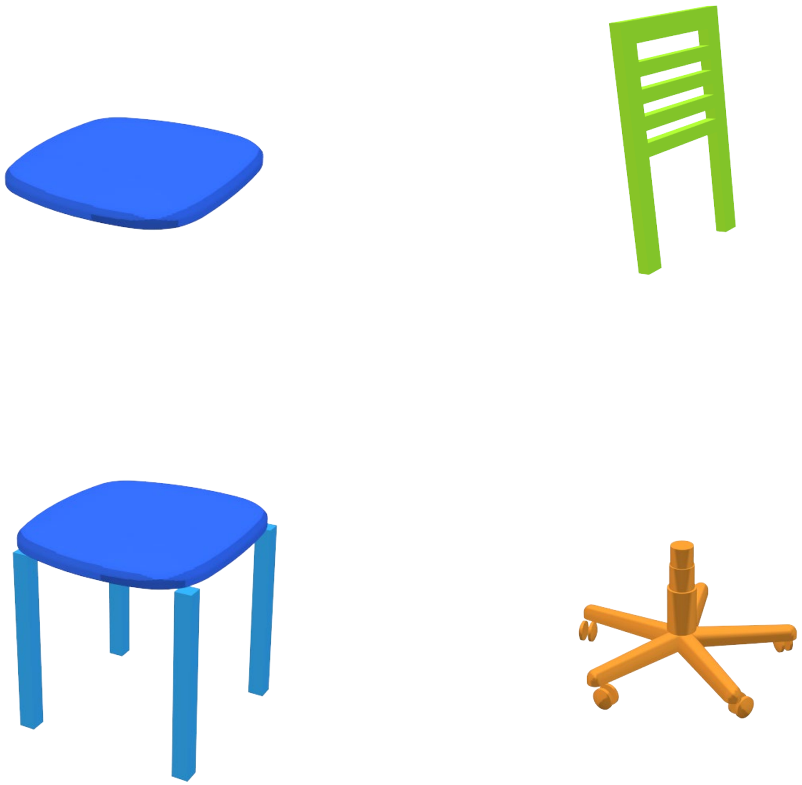
Retrieval and Embedding

- **Precompute** embedding coordinates of parts in the database.
- Compute only for the input shape in test time.



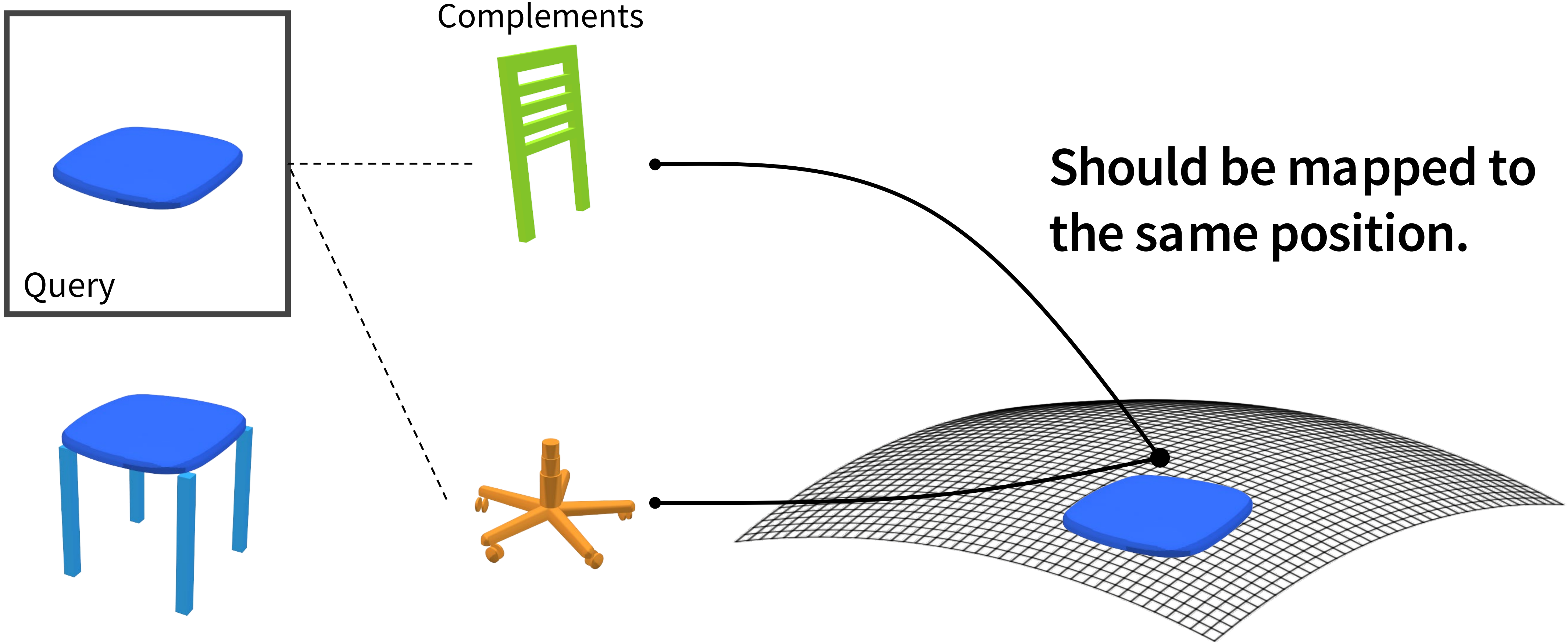
Retrieval and Embedding

Problem of the joint embedding when learning a **multi-valued** function:



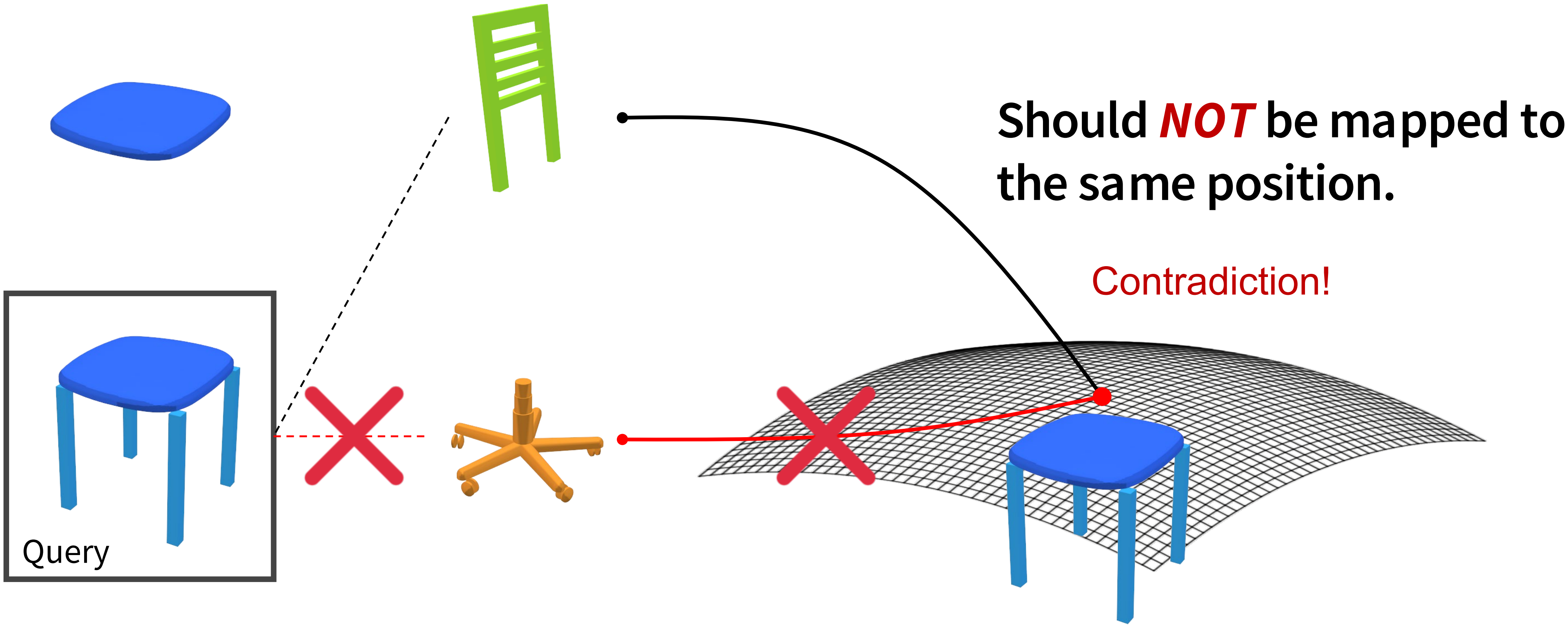
Retrieval and Embedding

Problem of the joint embedding when learning a **multi-valued** function:



Retrieval and Embedding

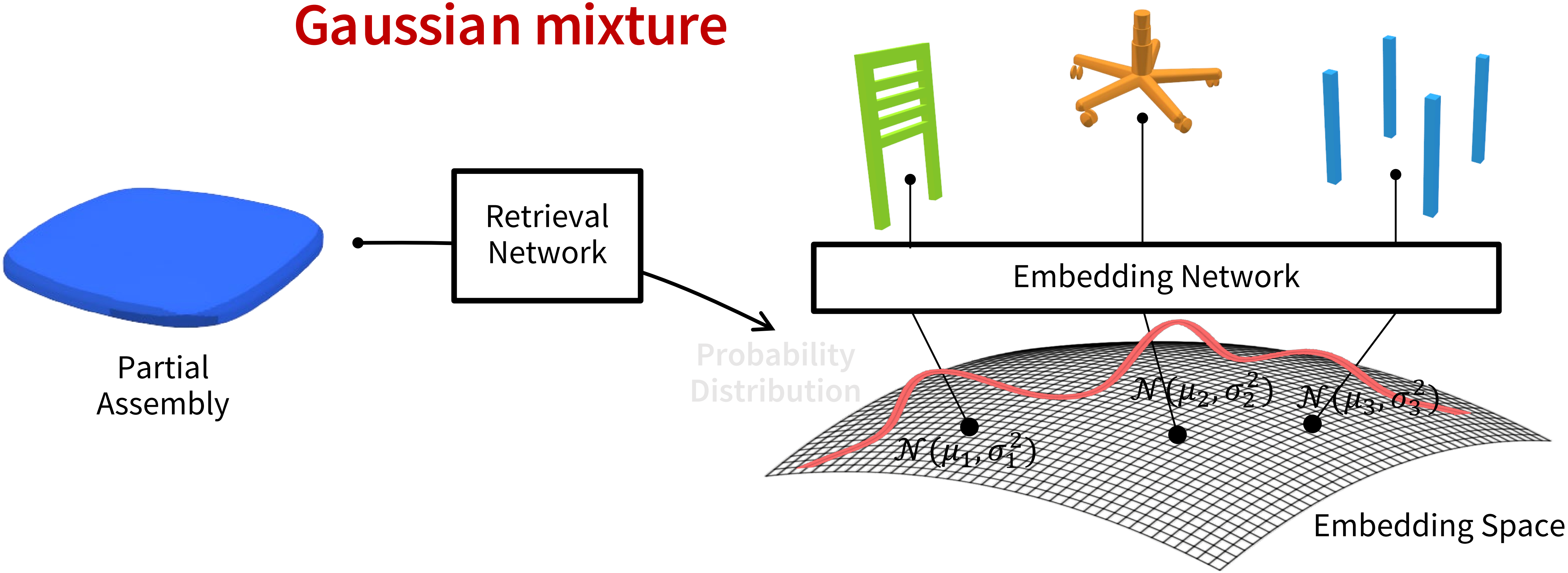
Problem of the joint embedding when learning a **multi-valued** function:



Retrieval and Embedding

Predict a **multimodal** probability distribution (Bishop 1994).

Gaussian mixture

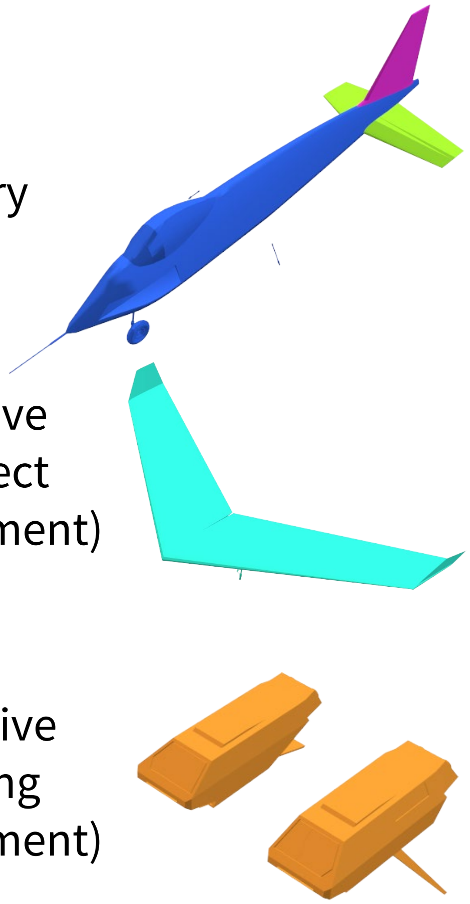


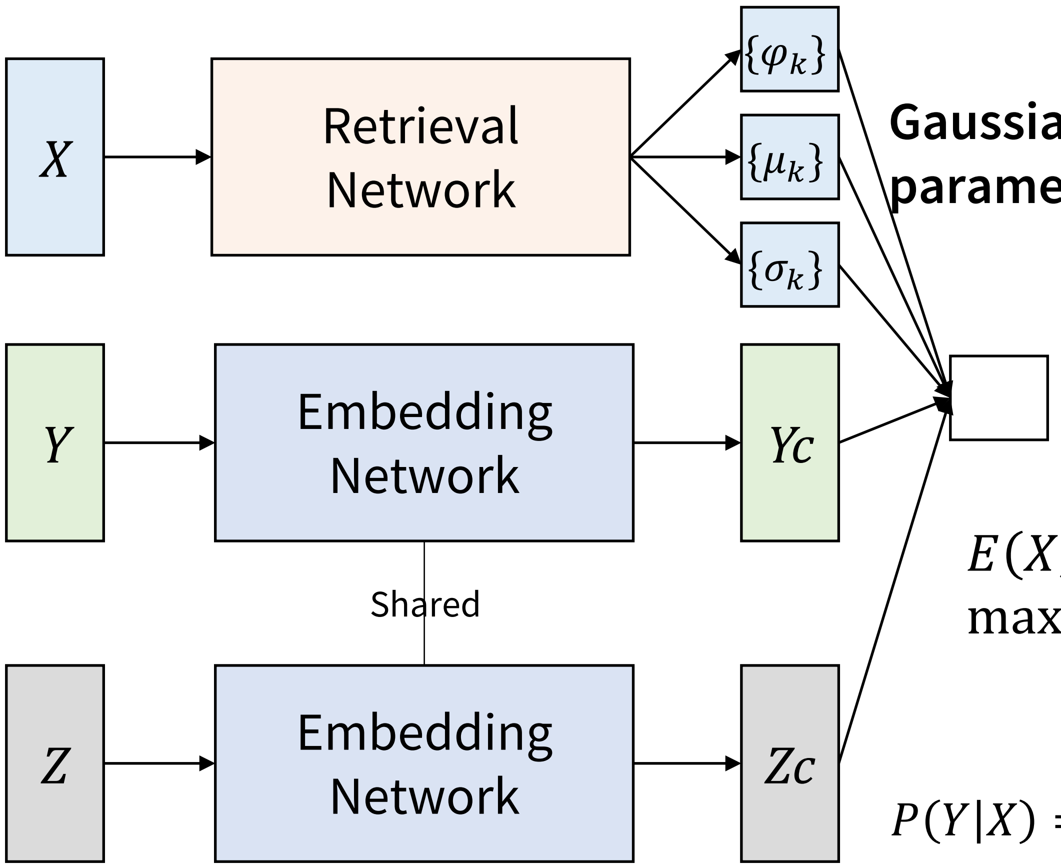
Neural Network Training

Query

Positive
(Correct
complement)

Negative
(Wrong
complement)





Gaussian mixture parameters

Relative margin loss
(Chechik *et al.* 2010)

$$E(X,Y,Z) = \max\{m + E(X,Y) - E(Y,Z), 0\}$$

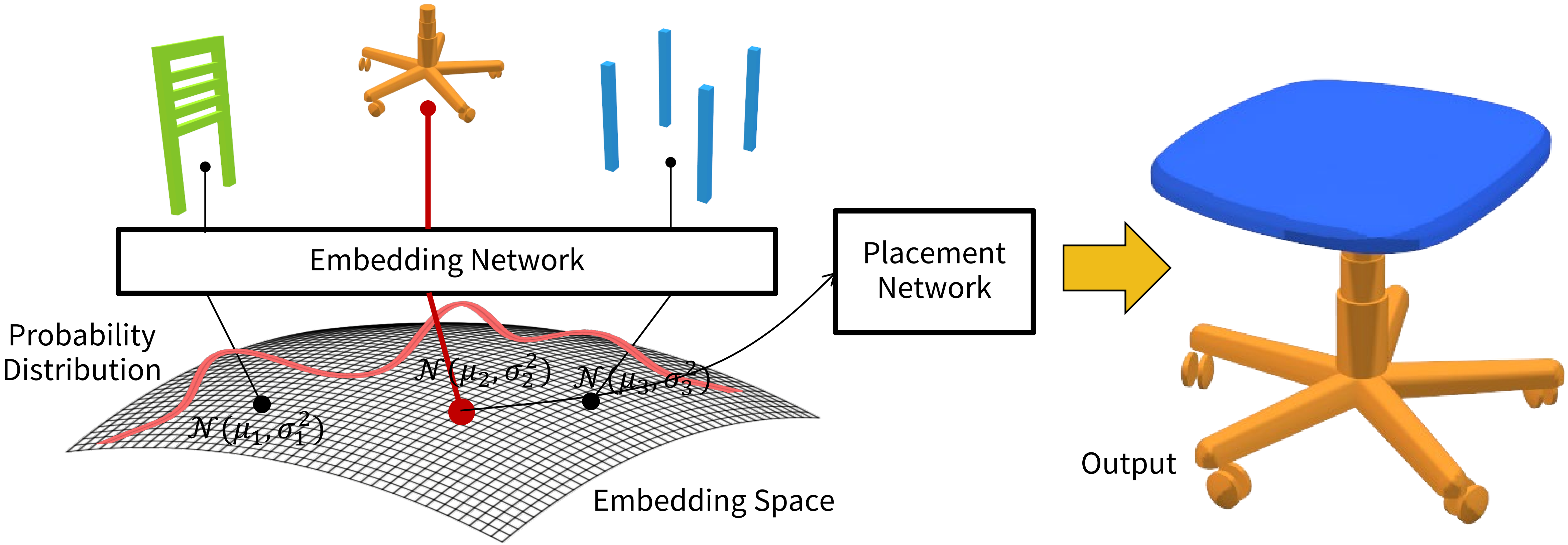
$$E(X,Y) = -\log P(Y|X)$$

$$P(Y|X) = \sum_k \varphi_k(X)\mathcal{N}(Y|\mu_k(X), \sigma_k(X)^2)$$

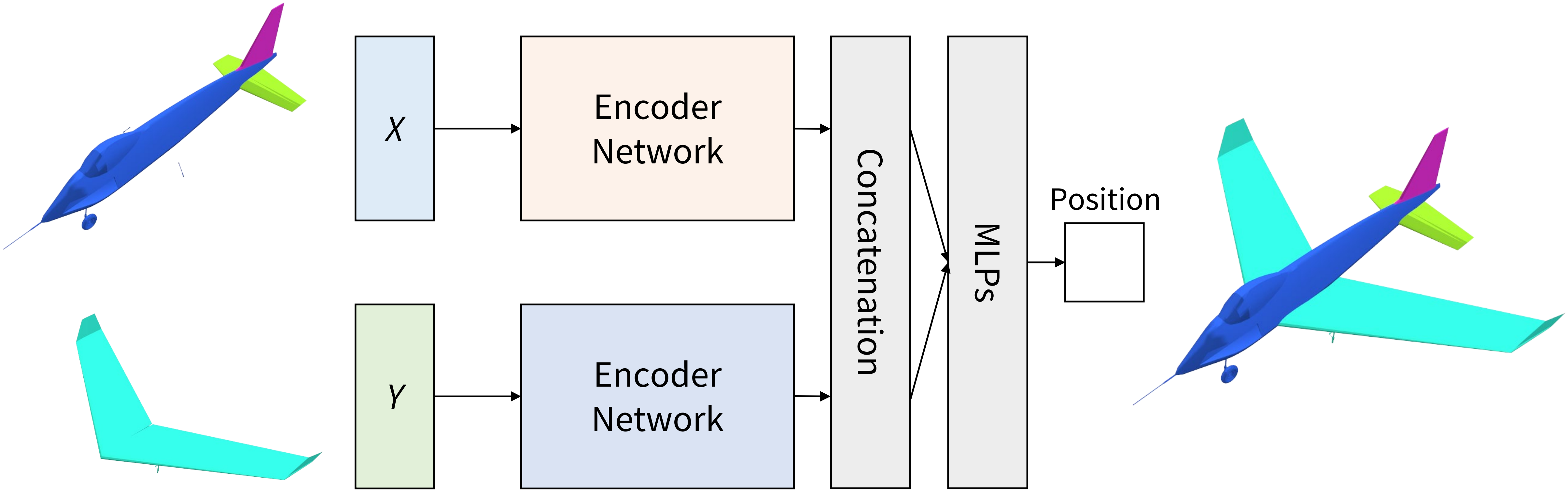
Embedding coordinates

Placement Network

- Sample a complement from the predicted distribution.
- Predict the location of the selected component.

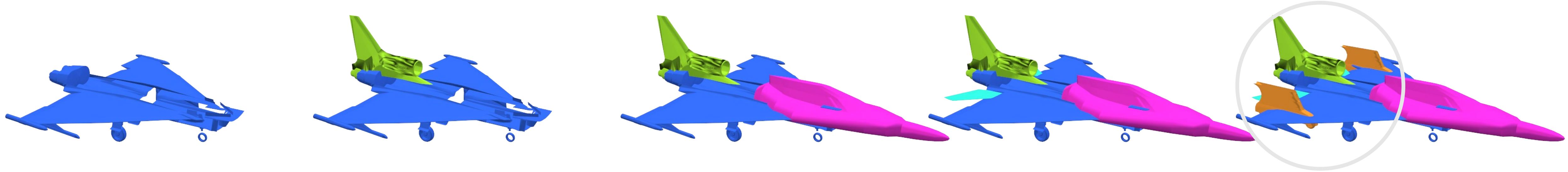


Placement Network



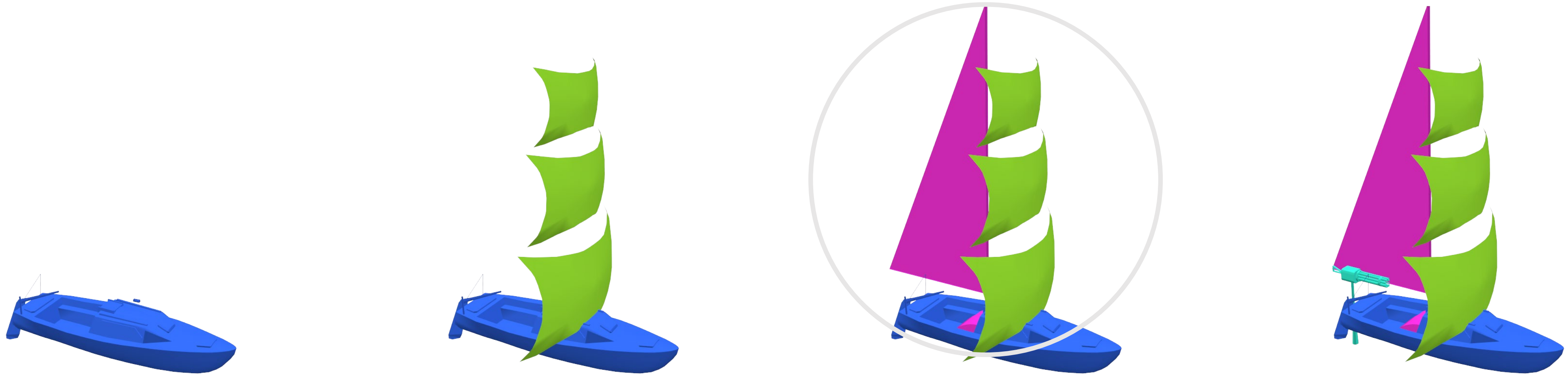
Automatic Shape Synthesis

Add the maximum probability part iteratively.



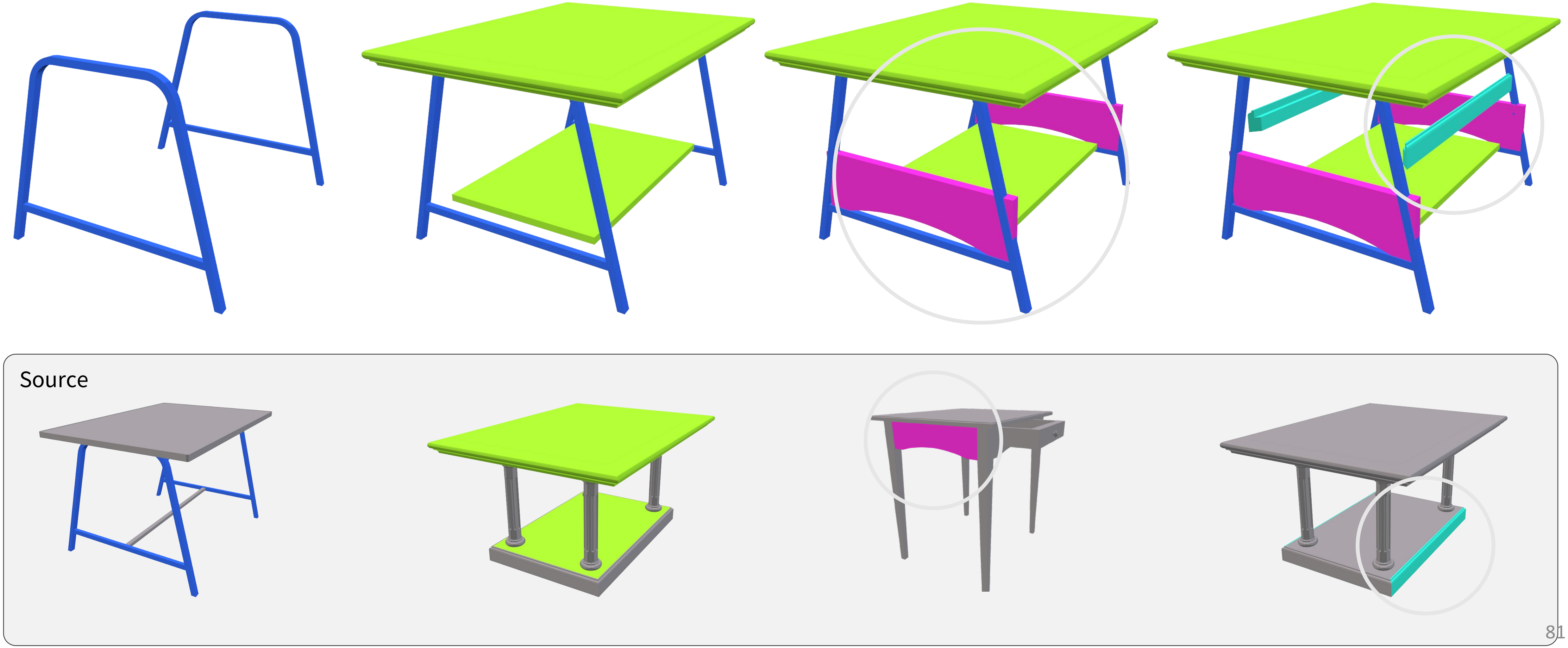
Source

Automatic Shape Synthesis

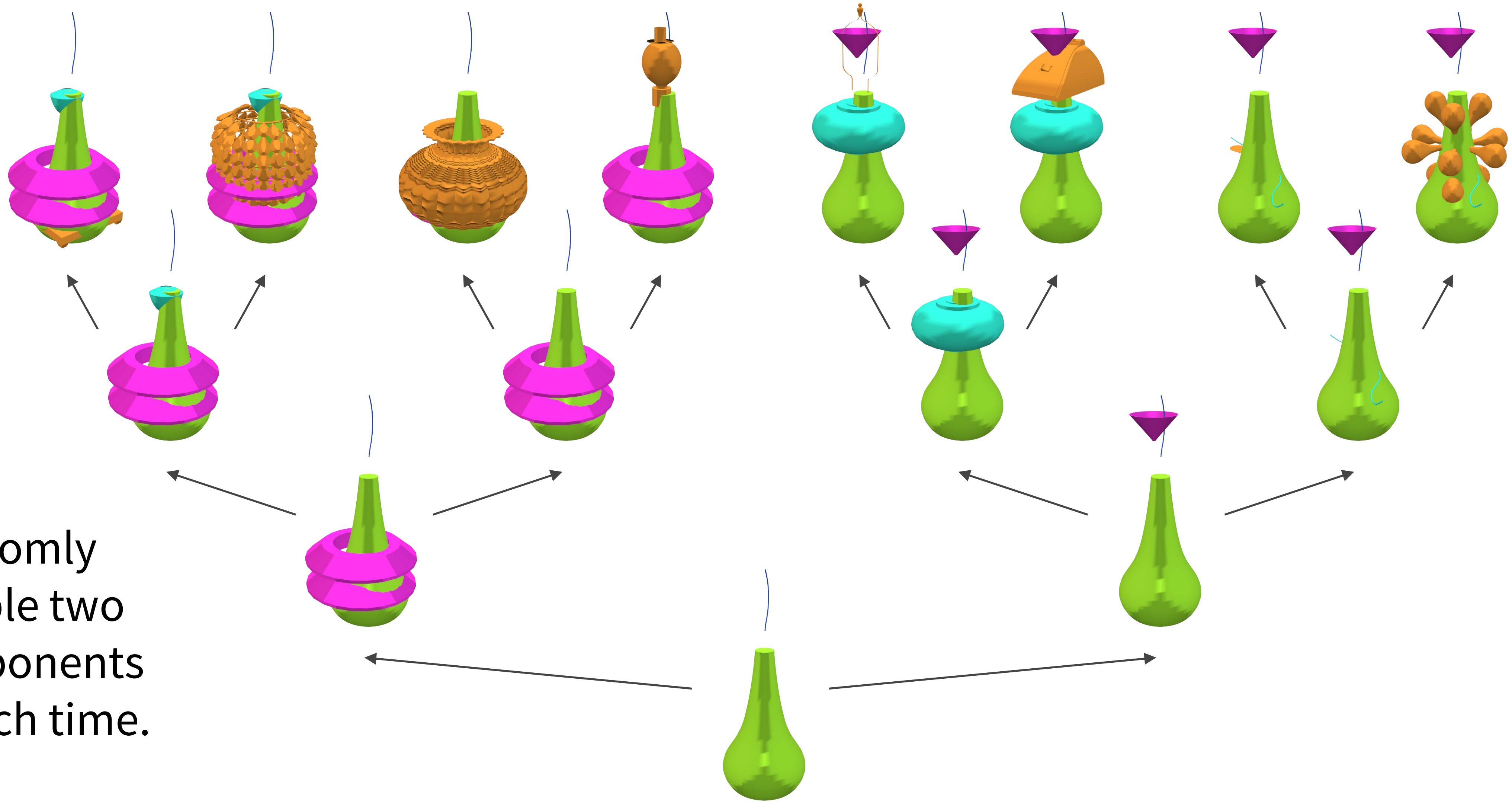


Source

Automatic Shape Synthesis



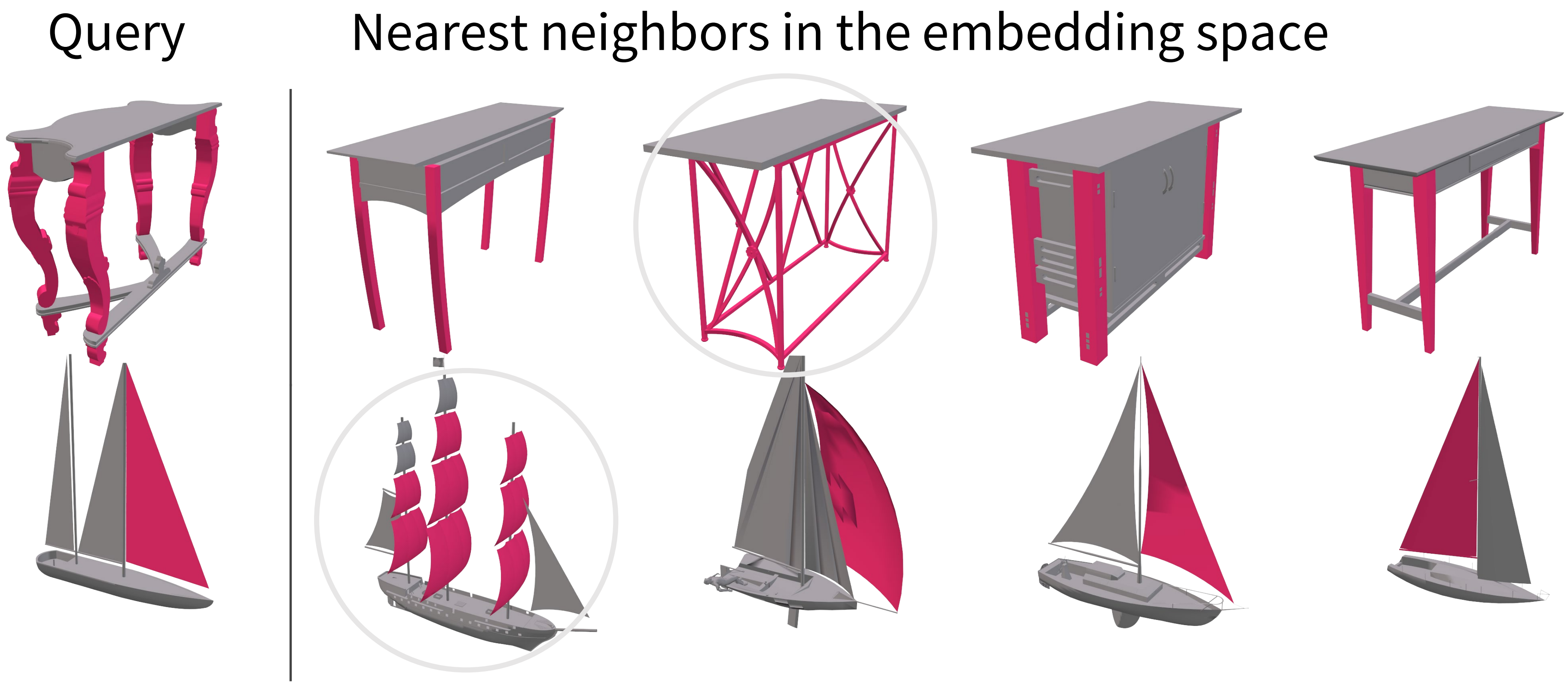
Automatic Shape Synthesis



Randomly sample two components at each time.

Observation

The retrieval network discovers *interchangeable* parts.

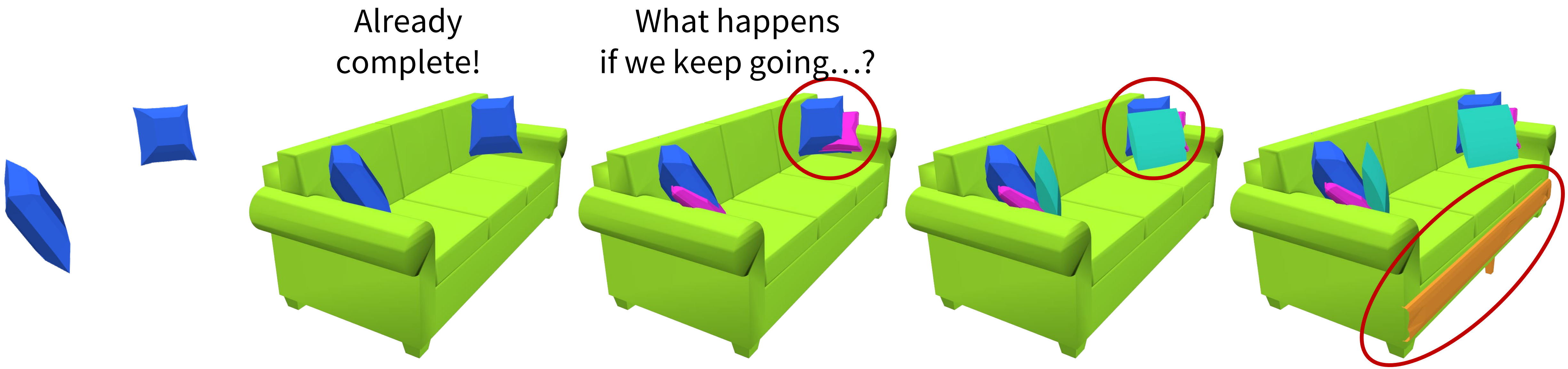


Can discover semantic relationships among parts!

Limitations

Limitations

- Accumulating noise in iterations.
- Missing notion of termination.



Learning Fuzzy Set Representations of Partial Shapes on Dual Embedding Spaces

Eurographics Symposium on Geometry Processing (SGP) 2018

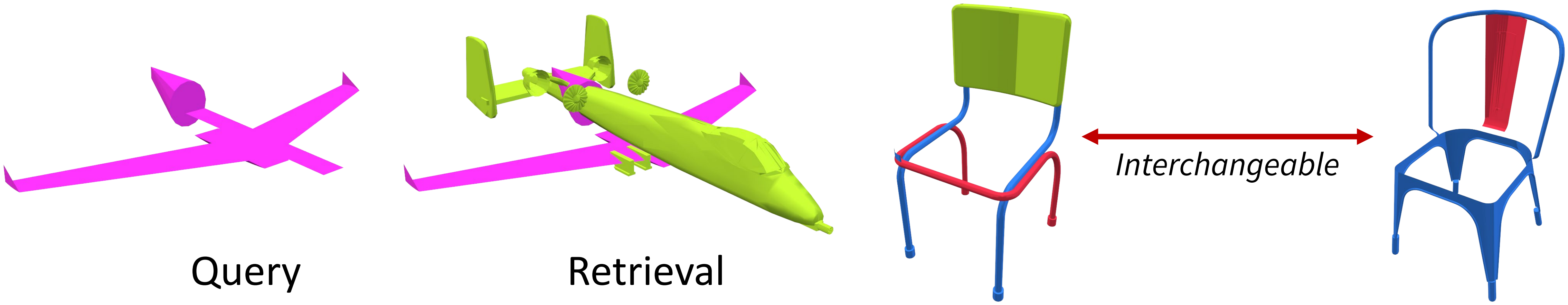
Minhyuk Sung[1], Anastasia Dubrovina[1], Vladimir G. Kim[2], and Leonidas Guibas[1]

[1]Stanford University [2]Adobe Research

Learn Relations Among Partial Shapes

Learn relations among *partial shapes*.

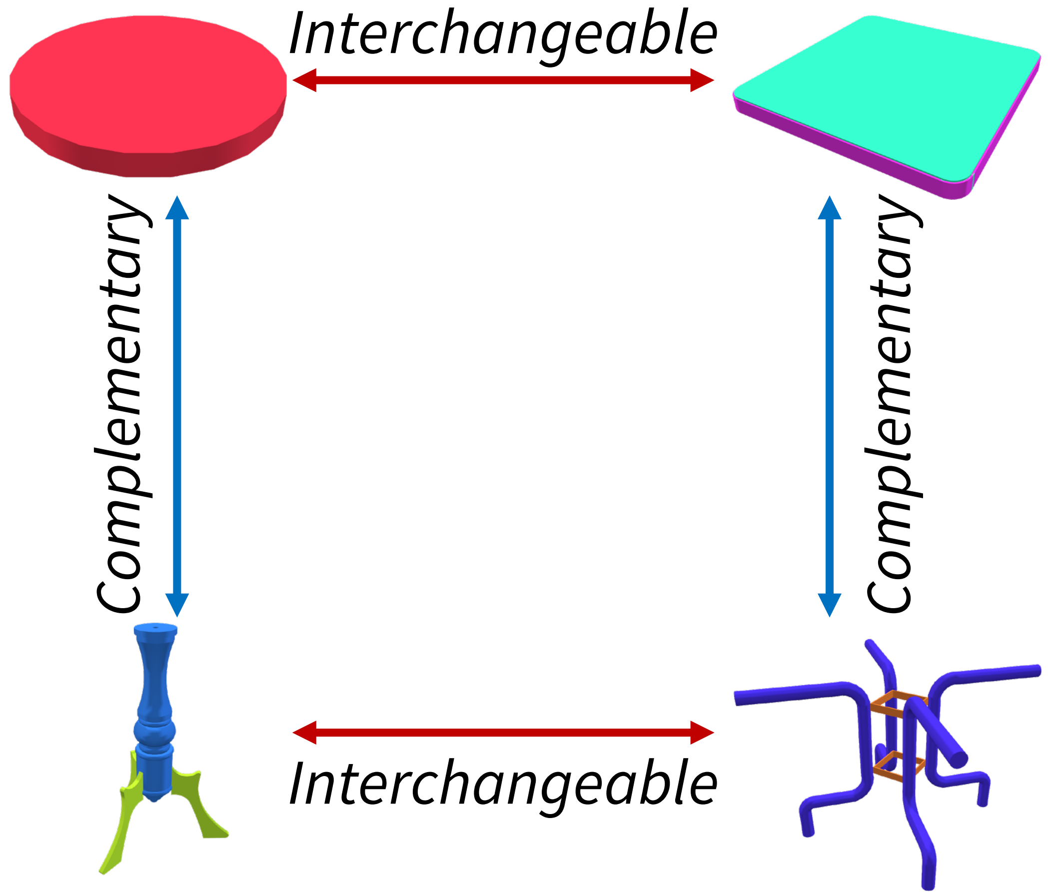
- Can complete an object with a single retrieval.
- Can discover group-to-group relations.



Relations Among Partial Shapes

Learn relations among *partial shapes*.

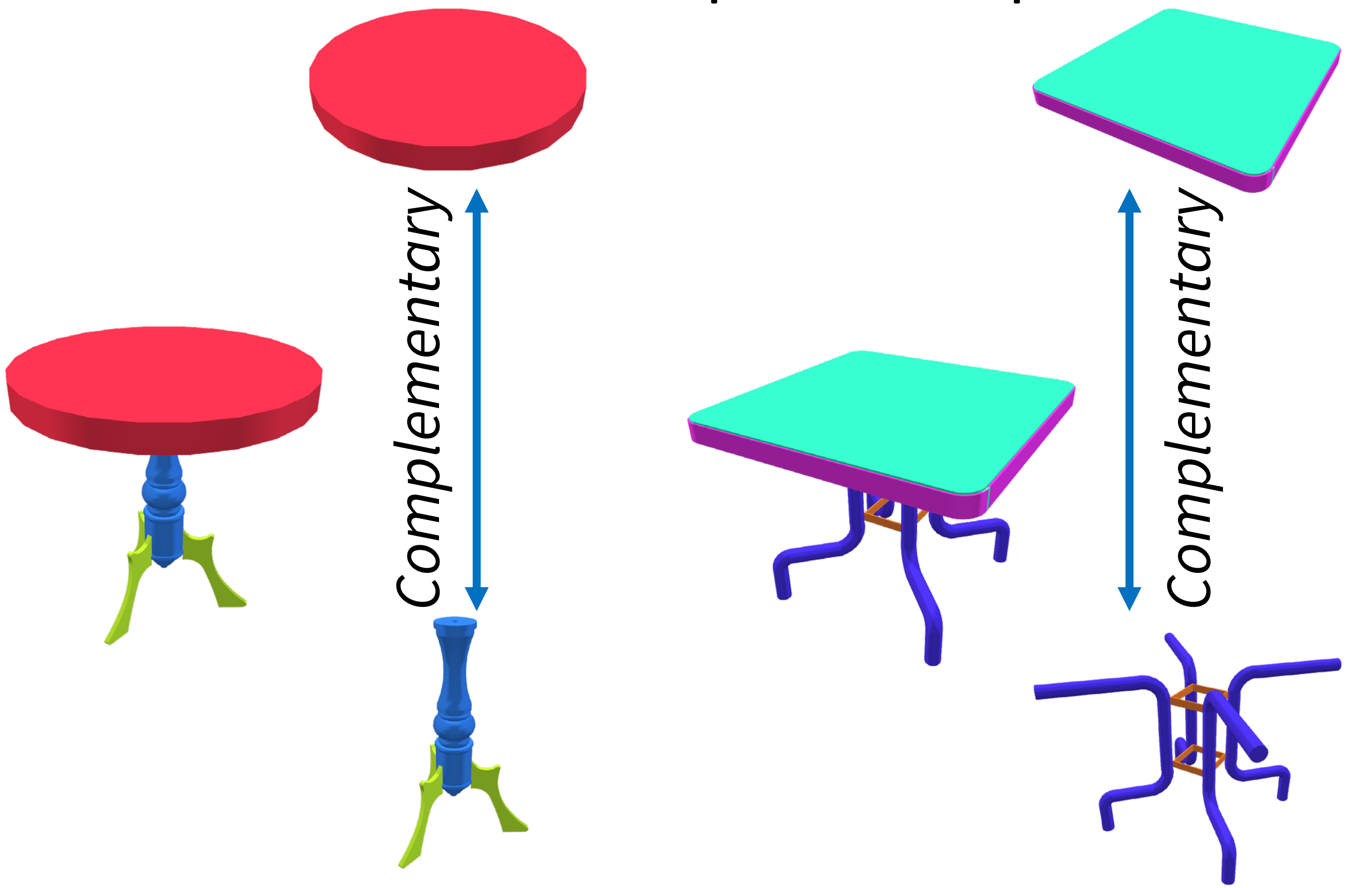
- Complementarity
- Interchangeability



Relations Among Partial Shapes

Complementarity

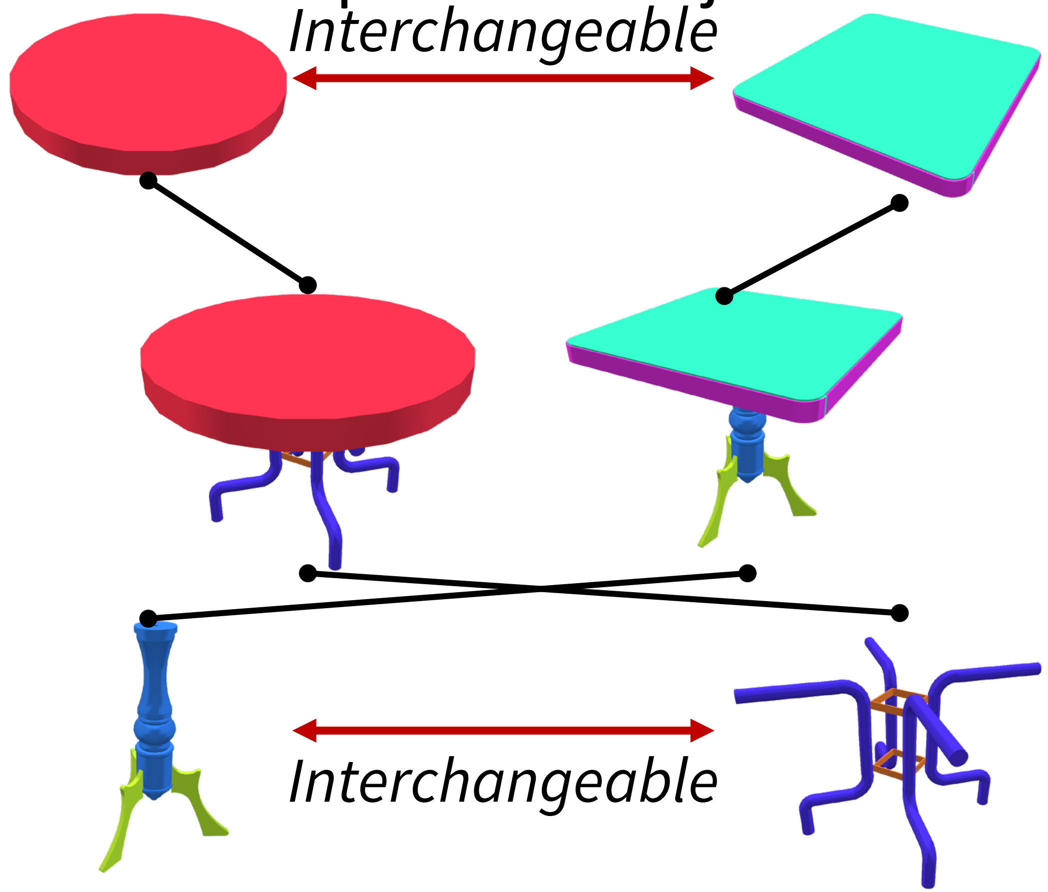
: Two partial shapes can be combined into a complete and plausible object.



Relations Among Partial Shapes

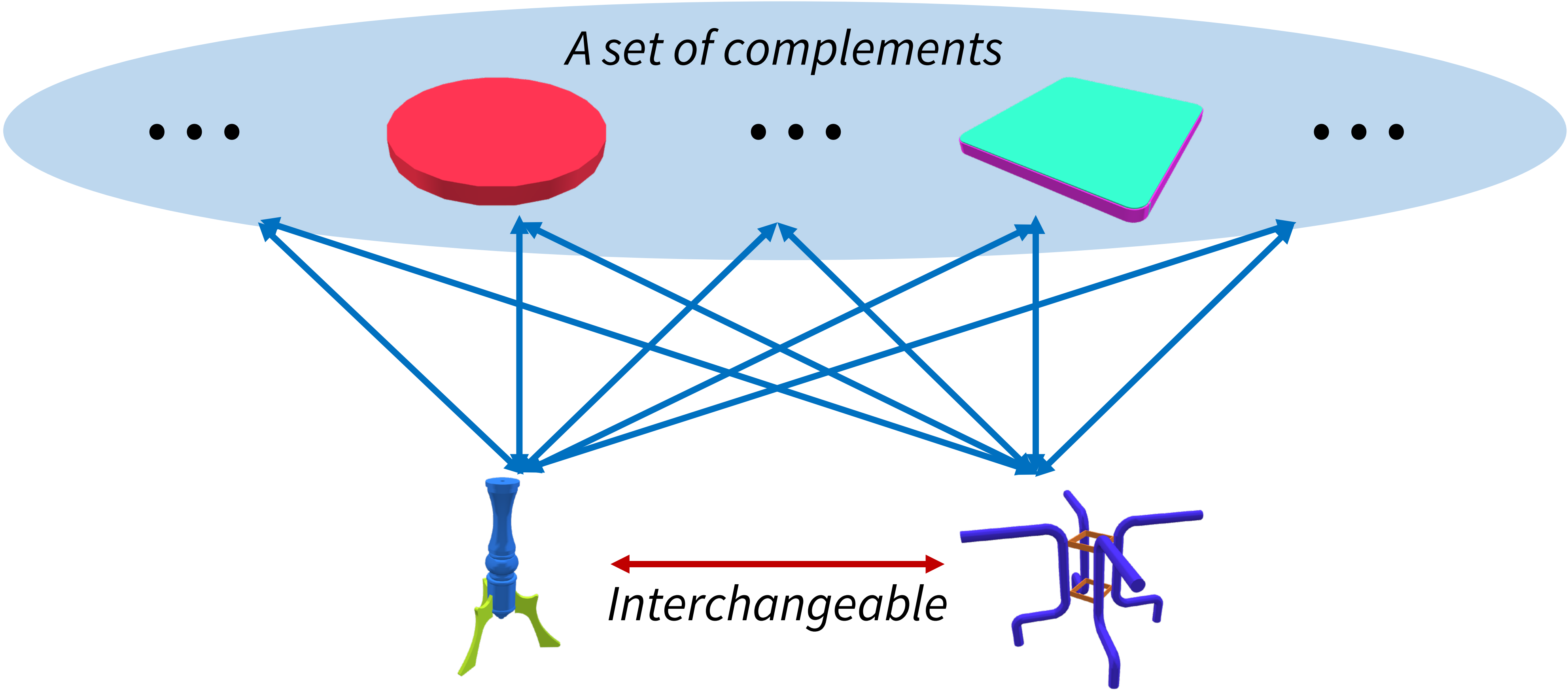
Interchangeability

: Replacing one with the other still produces a plausible object.



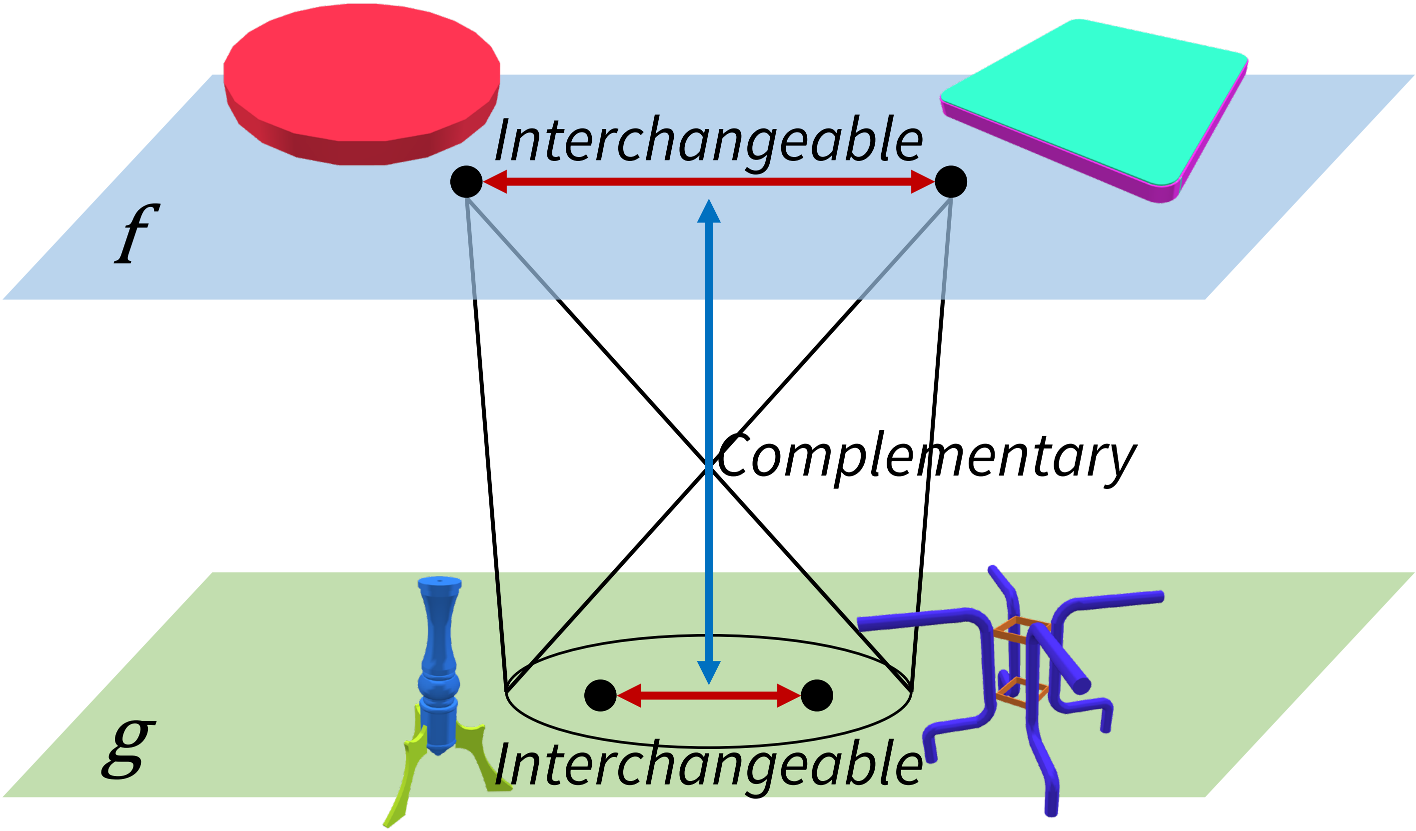
Relations Among Partial Shapes

Two partial shapes are *interchangeable* if they share the same set of *complements*.



Our Approach

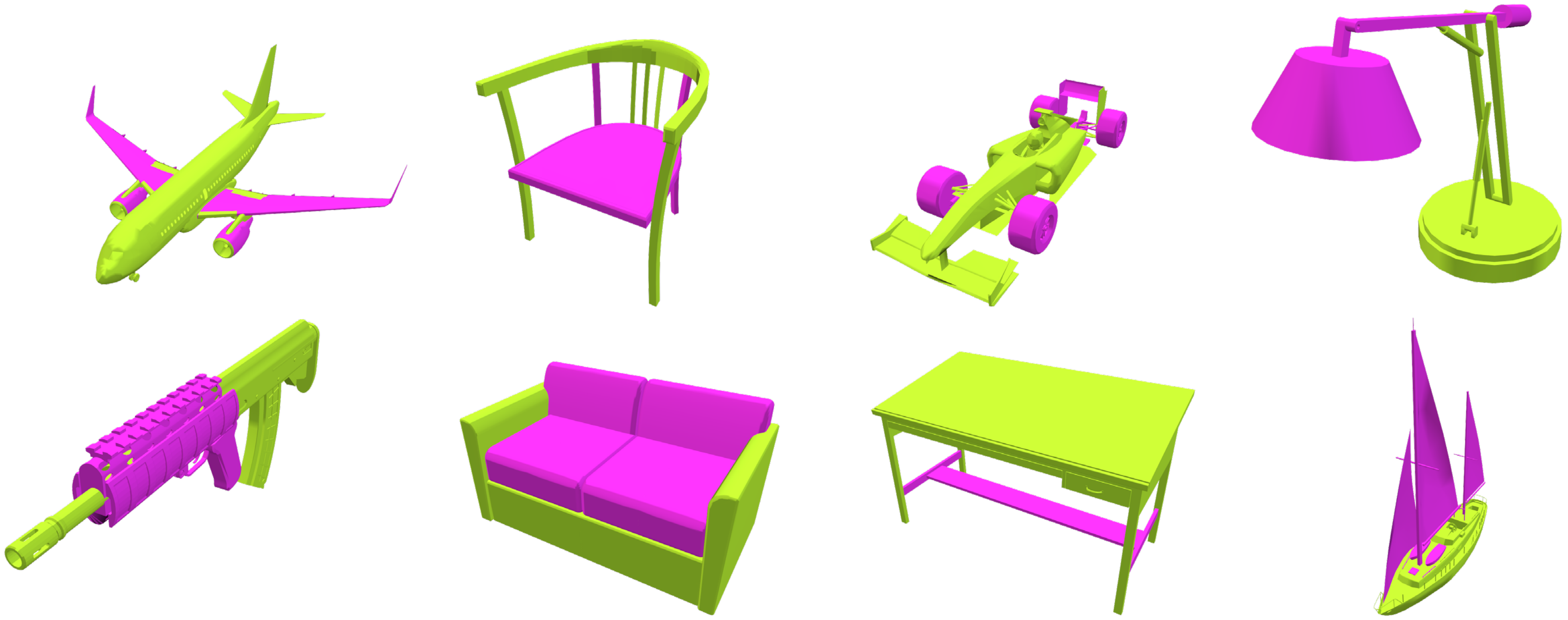
Jointly encode both *complementary* and *interchangeable* relations in a *dual* embedding space.



Our Approach

Learn *interchangeability* from *complementarity.*

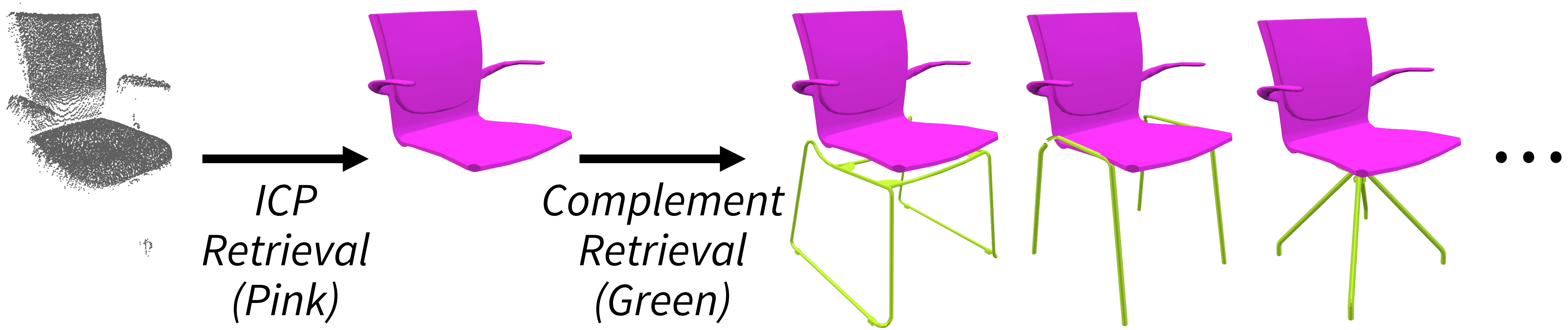
- *Complementary* pairs are created by splitting objects.
- No supervision for *interchangeability* is given.



Applications

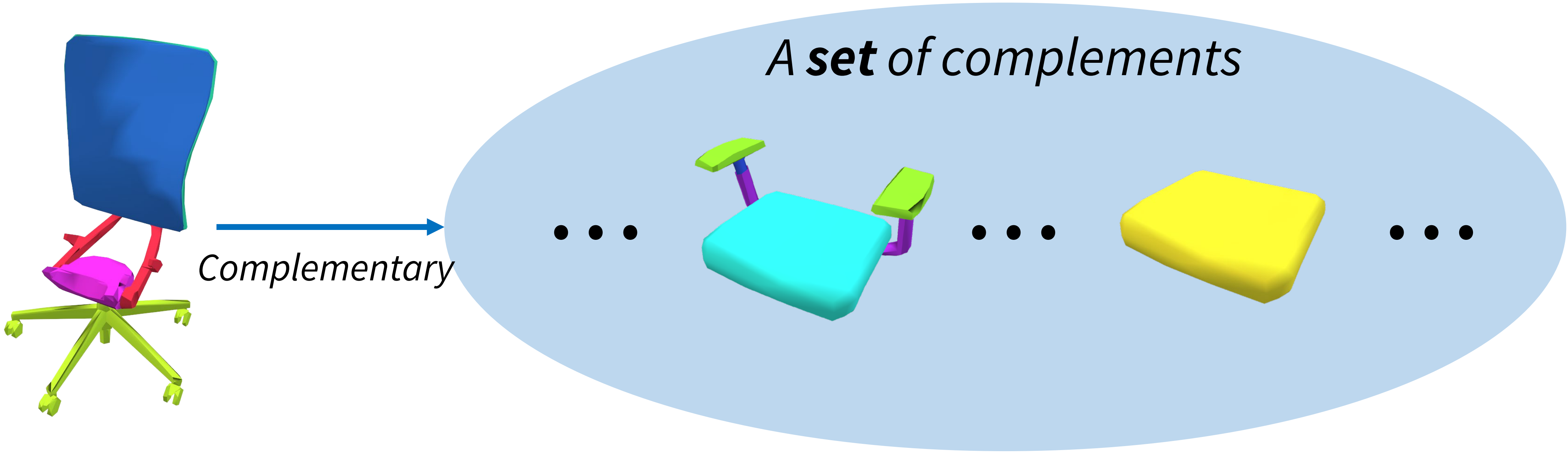
- Shape analysis

- Shape completion



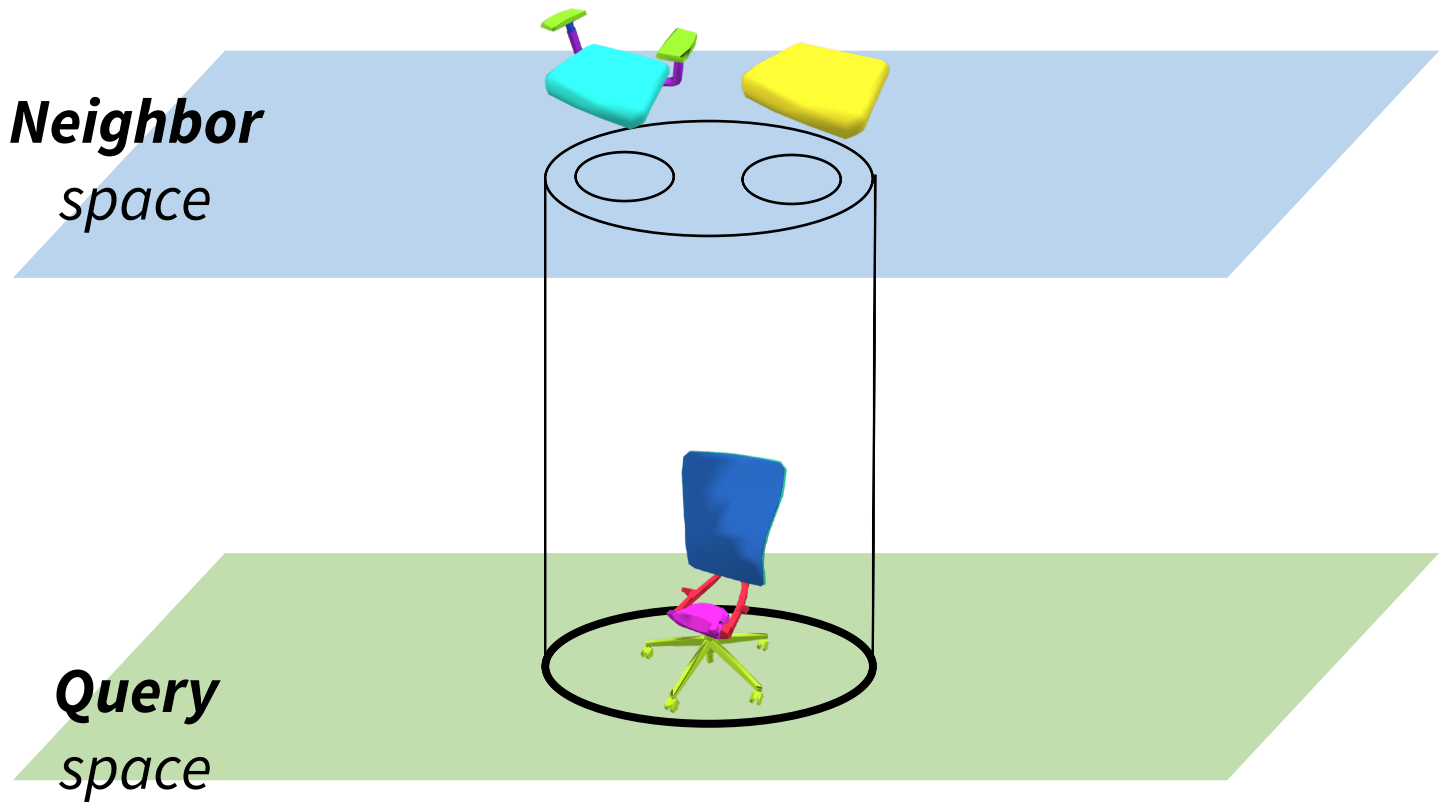
Embedding as Set Inclusion

Encode *1-to-N* mapping as *set inclusion*.

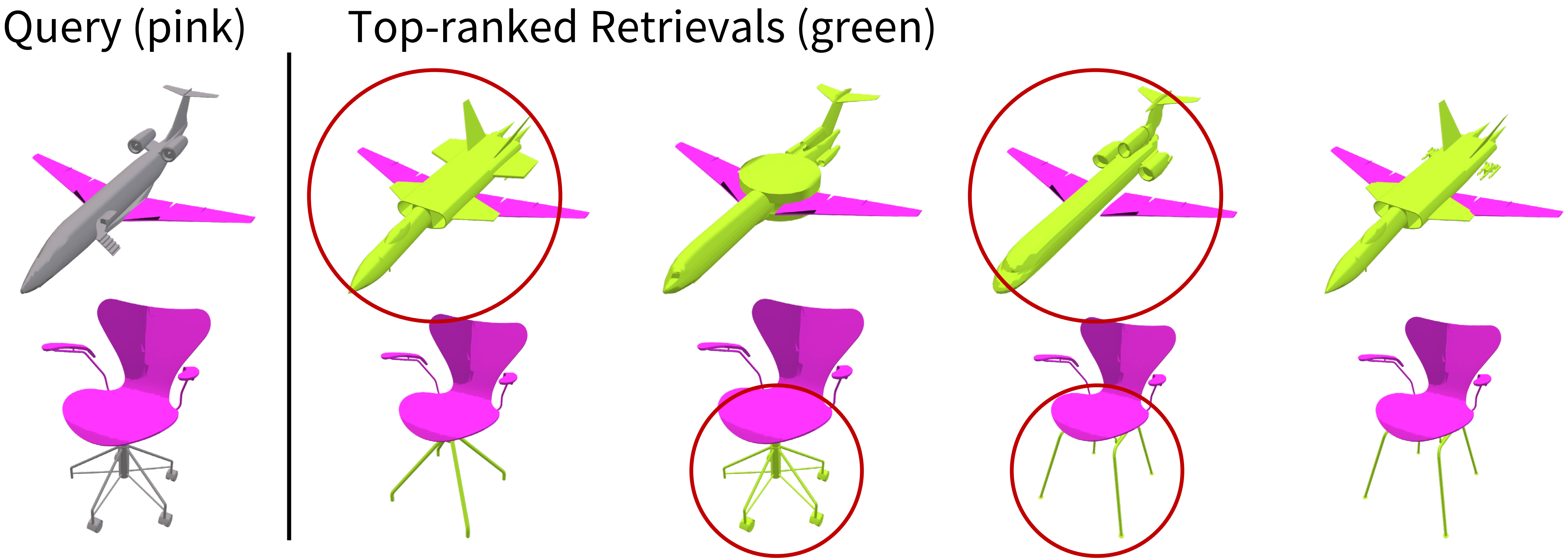


Embedding as Set Inclusion

Encode *1-to-N* mapping as *set inclusion*.

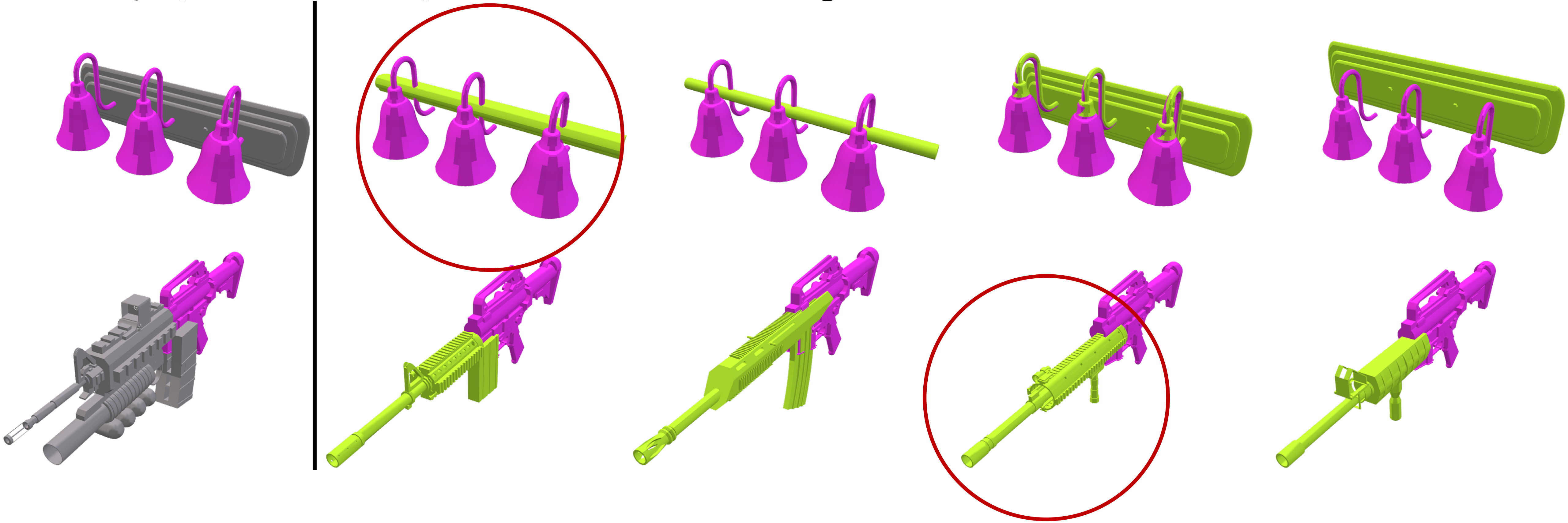


Complementary Shape Retrievals



Complementary Shape Retrievals

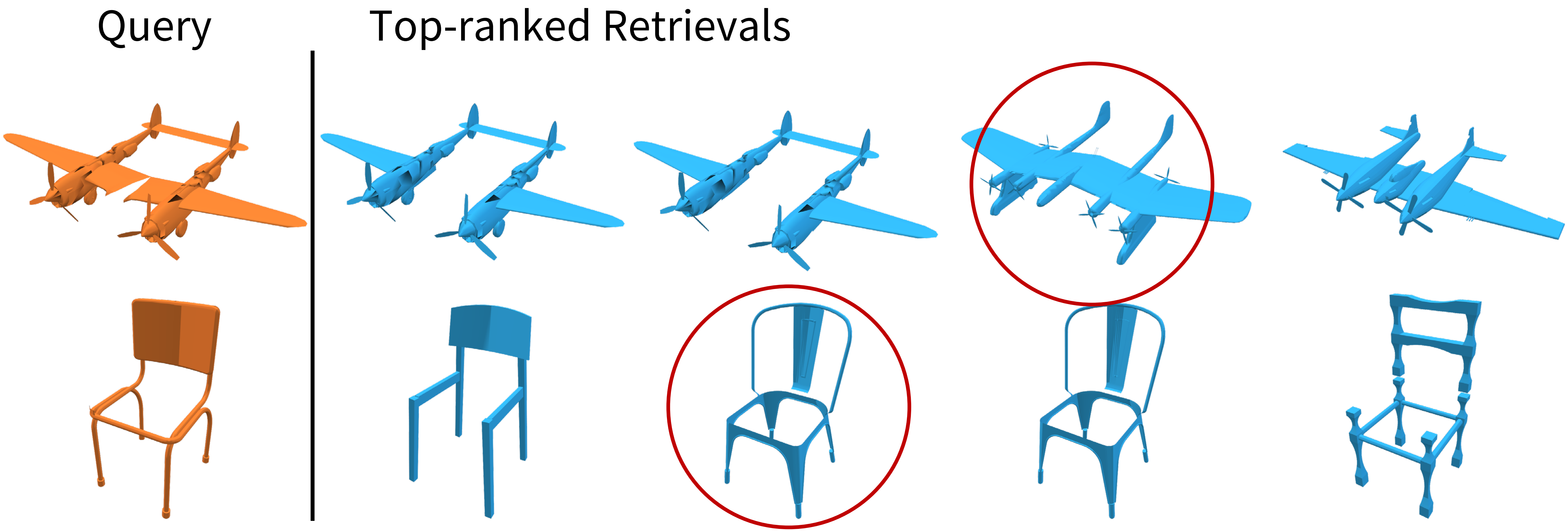
Query (pink) | Top-ranked Retrievals (green)



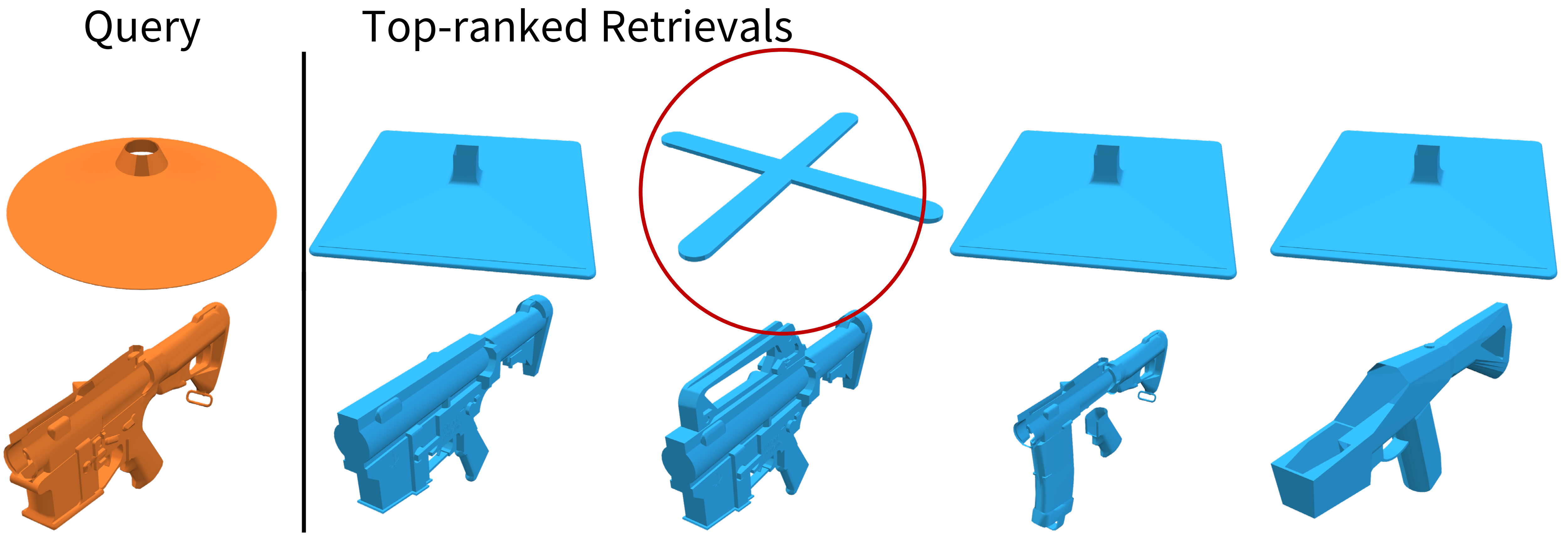
Complementary Shape Retrievals

Query (pink) | Top-ranked Retrievals (green)

Interchangeable Shape Retrievals

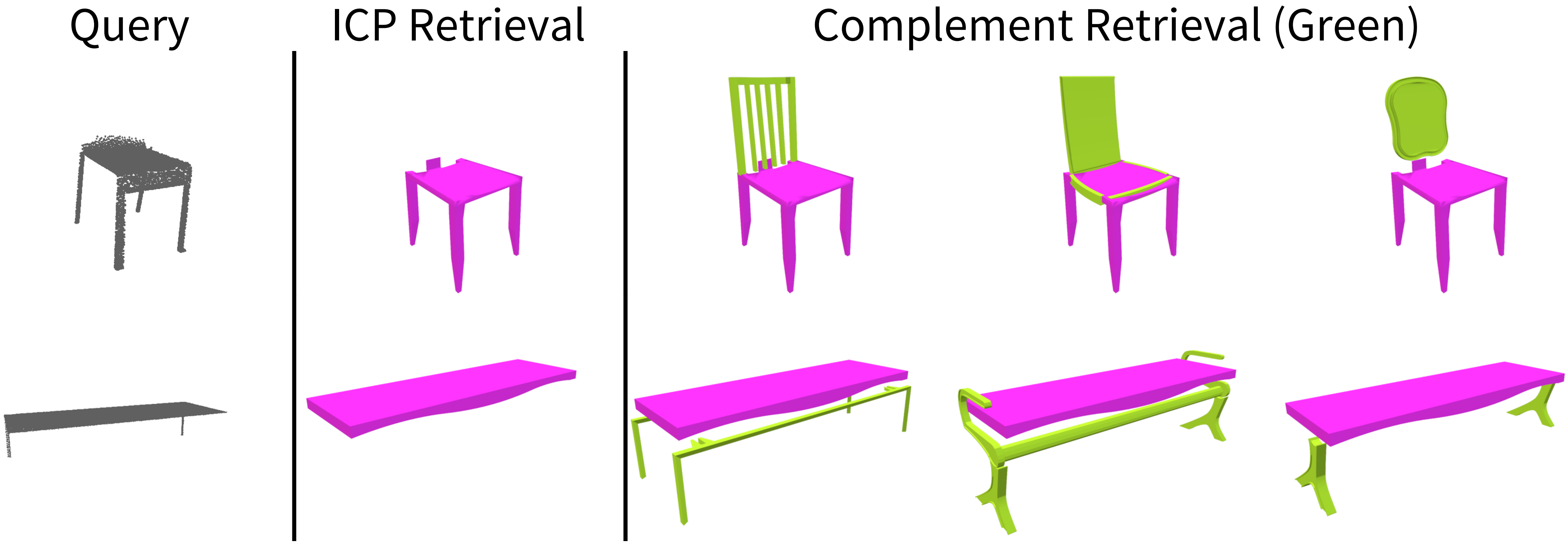


Interchangeable Shape Retrievals



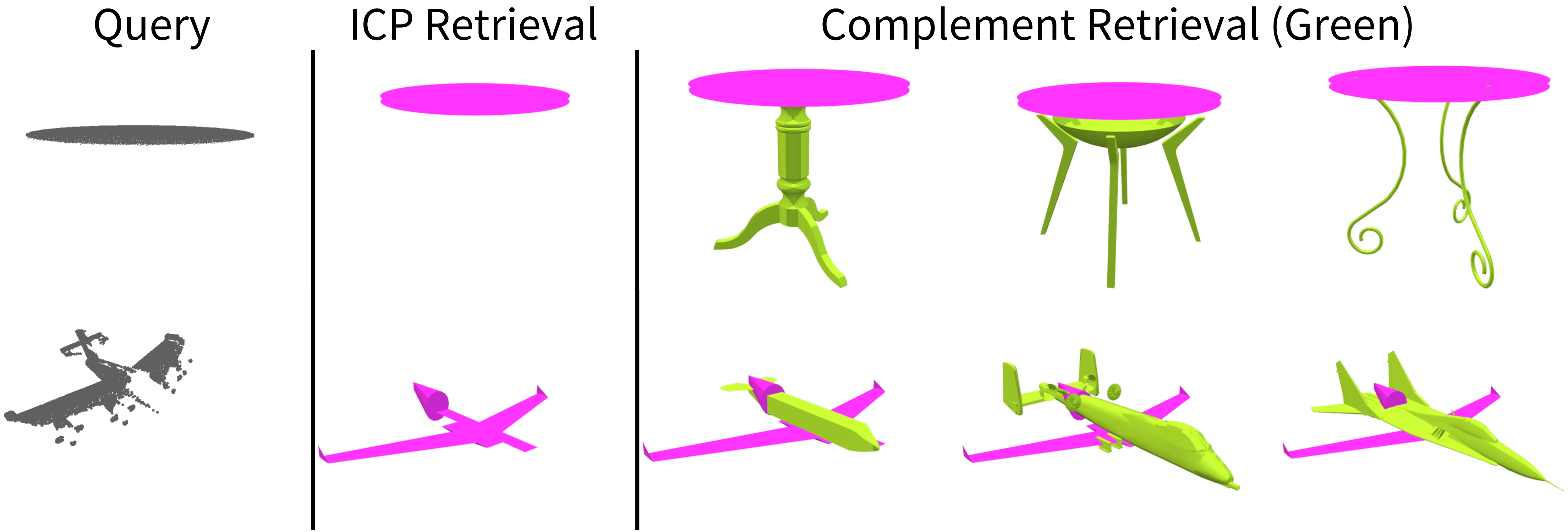
Partial Scan Completion

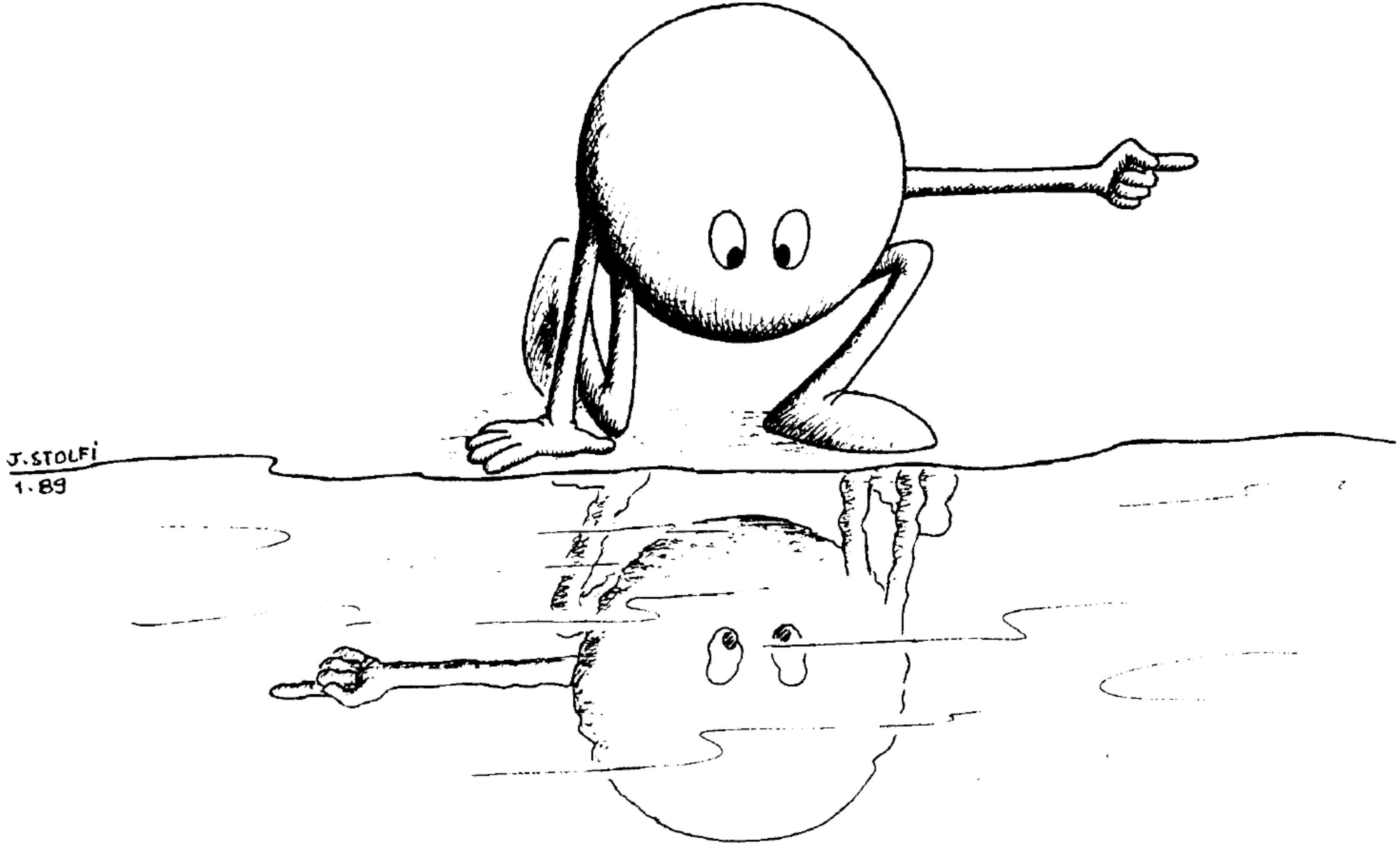
Completing synthetic scan data [Sung et al., 2015]



Partial Scan Completion

Completing synthetic scan data [Sung et al., 2015]




J.STOLFI
1.89