Neurosymbolic 3D Models:
Learning to Generate 3D Shape Programs

Daniel Ritchie

BROWN
Computer Science

This guy!

WHO AM I?

Brown University

- Located in Providence, Rhode Island
- #14 University in the US (US News)

Brown Computer Science Department

- 37 full-time faculty
- 2-year Masters program
- Fully-funded PhD program (5 years)
- #25 for CS Graduate Study (US News)

Brown Visual Computing

BVC

- Nine (9) faculty
- Active research in graphics, vision, HCI, visualization, ...
- Regularly publish in top visual computing venues (SIGGRAPH, CVPR, ICCV, ...)

Brown Visual Computing

BVC

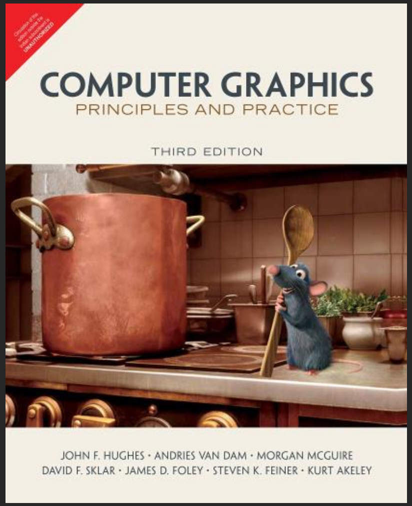
- **Andy van Dam:**
  co-founder of ACM SIGCRAPH (pre-cursor to SIGGRAPH)

Brown Visual Computing

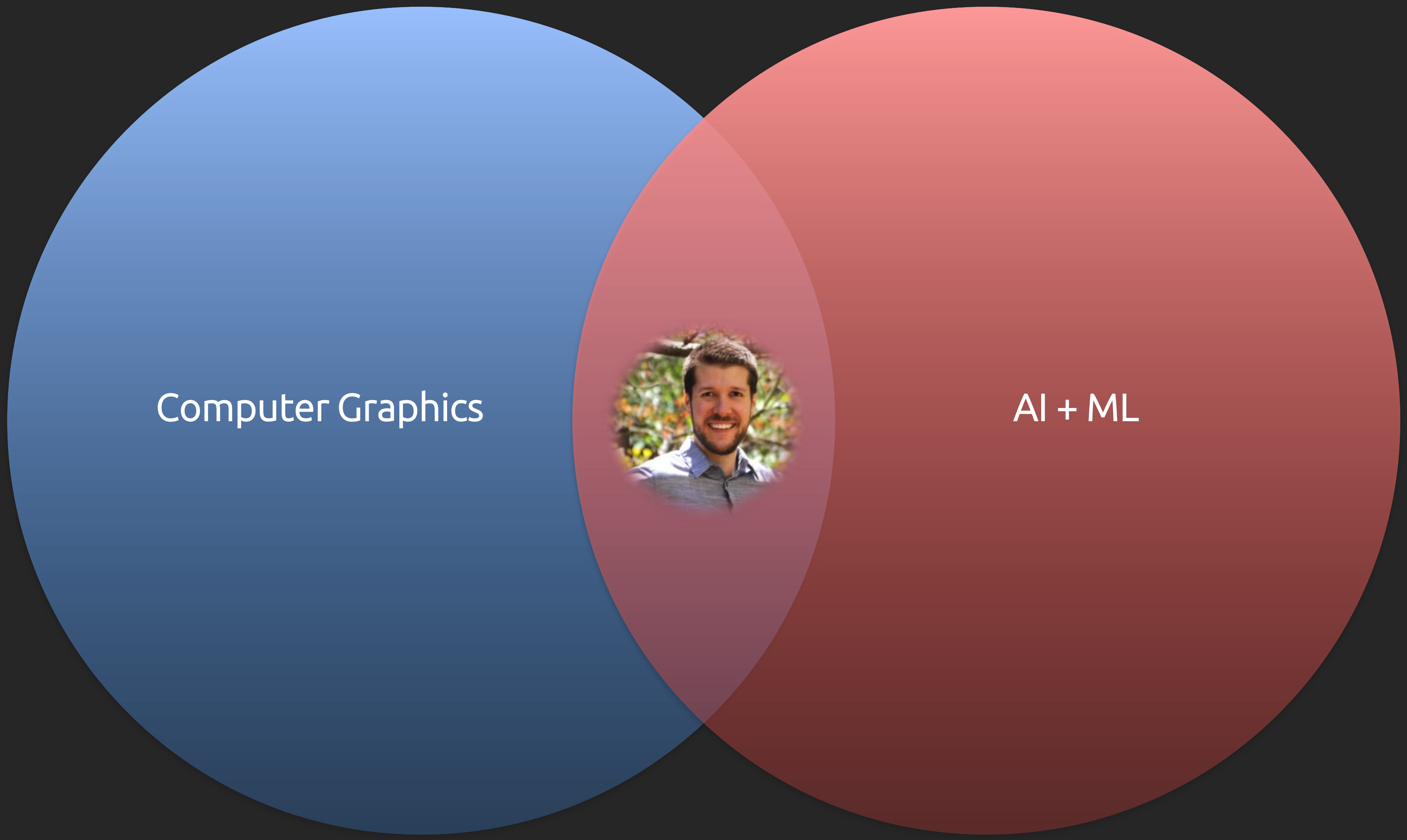
BVC

http://visual.cs.brown.edu/

- **Andy van Dam & Spike Hughes:** Authors of "Computer Graphics: Principles and Practice"



My Research (Broadly)



My Research (Specifically)

Generate

Generative Models

- Programs
- Deep Networks
- …

What are *neurosymbolic 3D models*, and how do they relate to all of this?

3D Structures

- Objects
- Scenes
- …

Infer

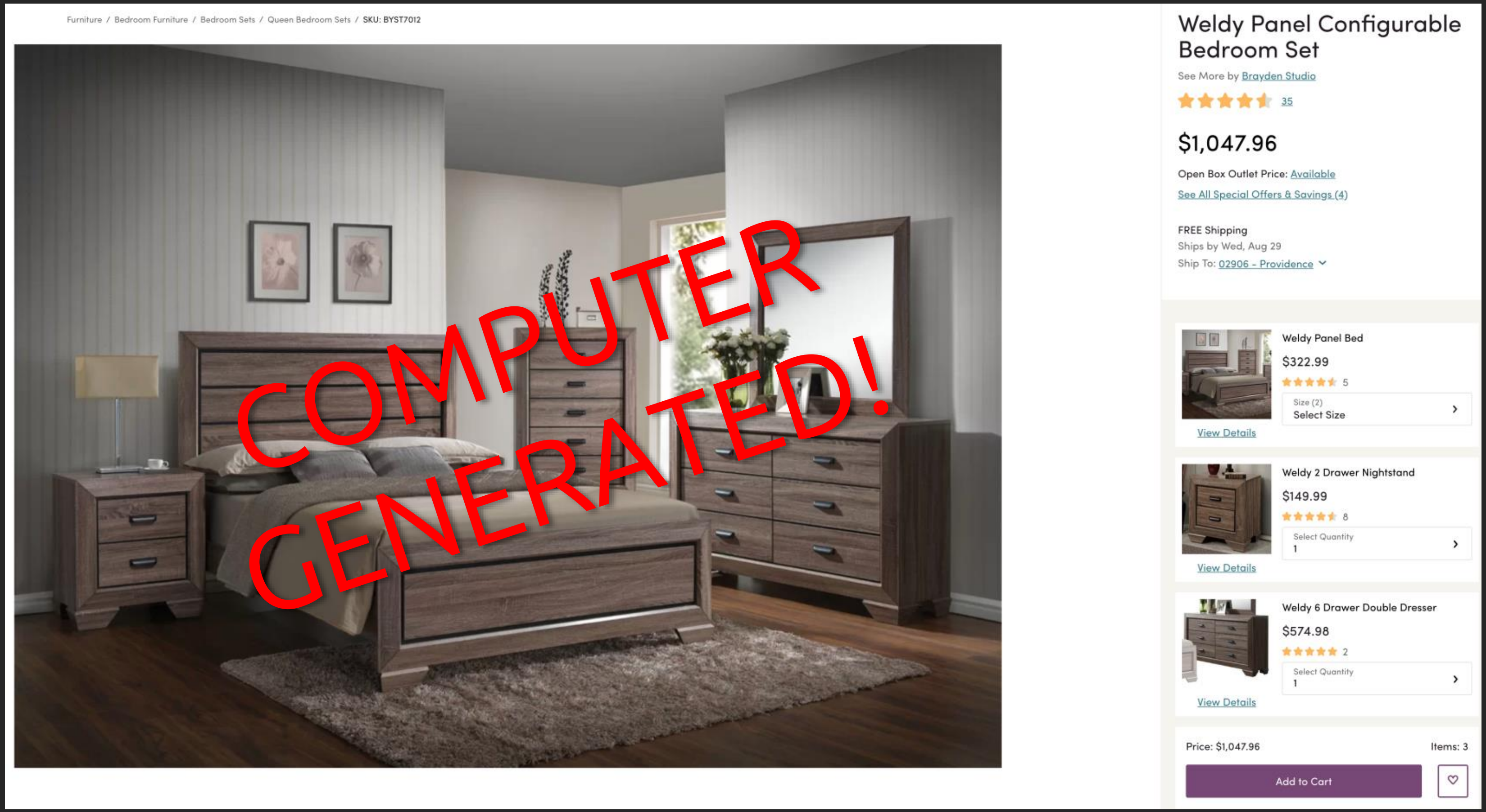
FIRST, A LITTLE BACKGROUND & MOTIVATION…

Increasing Demand for 3D Content

Traditional driver: Entertainment (Games, VR, ...)

Increasing Demand for 3D Content

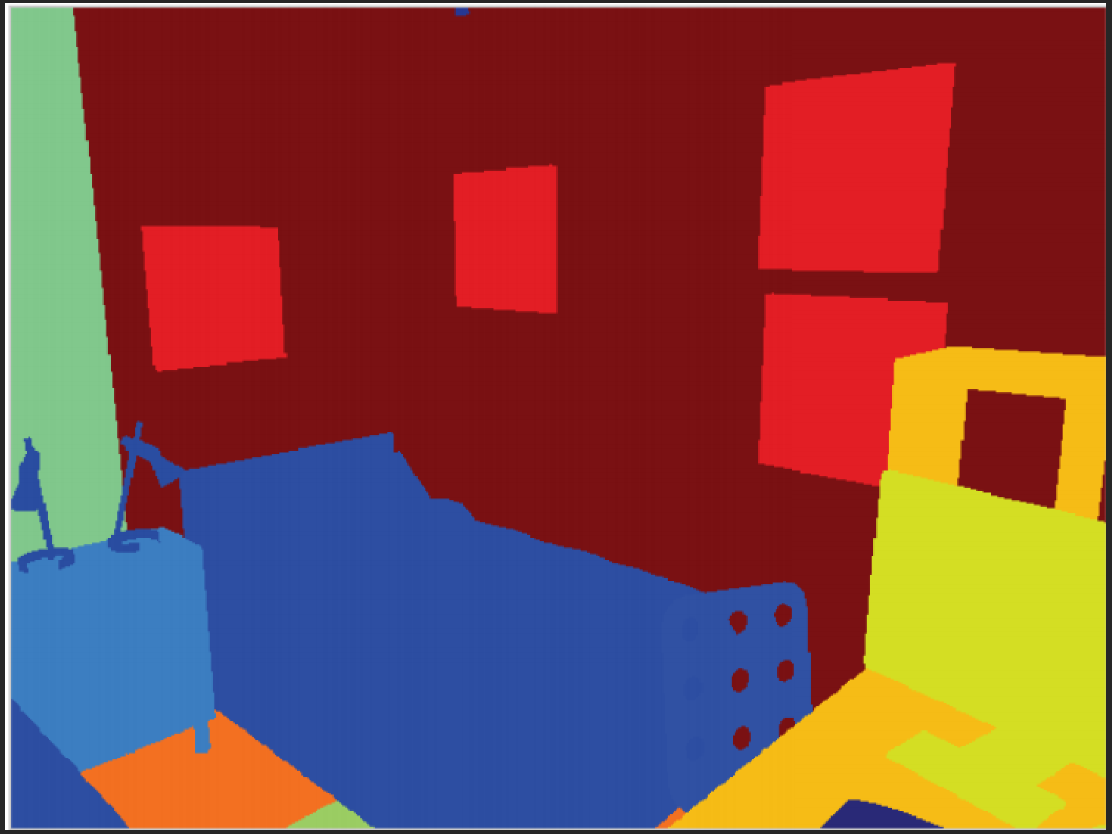
E-Commerce (esp. furniture / interior design)



Increasing Demand for 3D Content

New driver: Artificial Intelligence ("Graphics for AI")

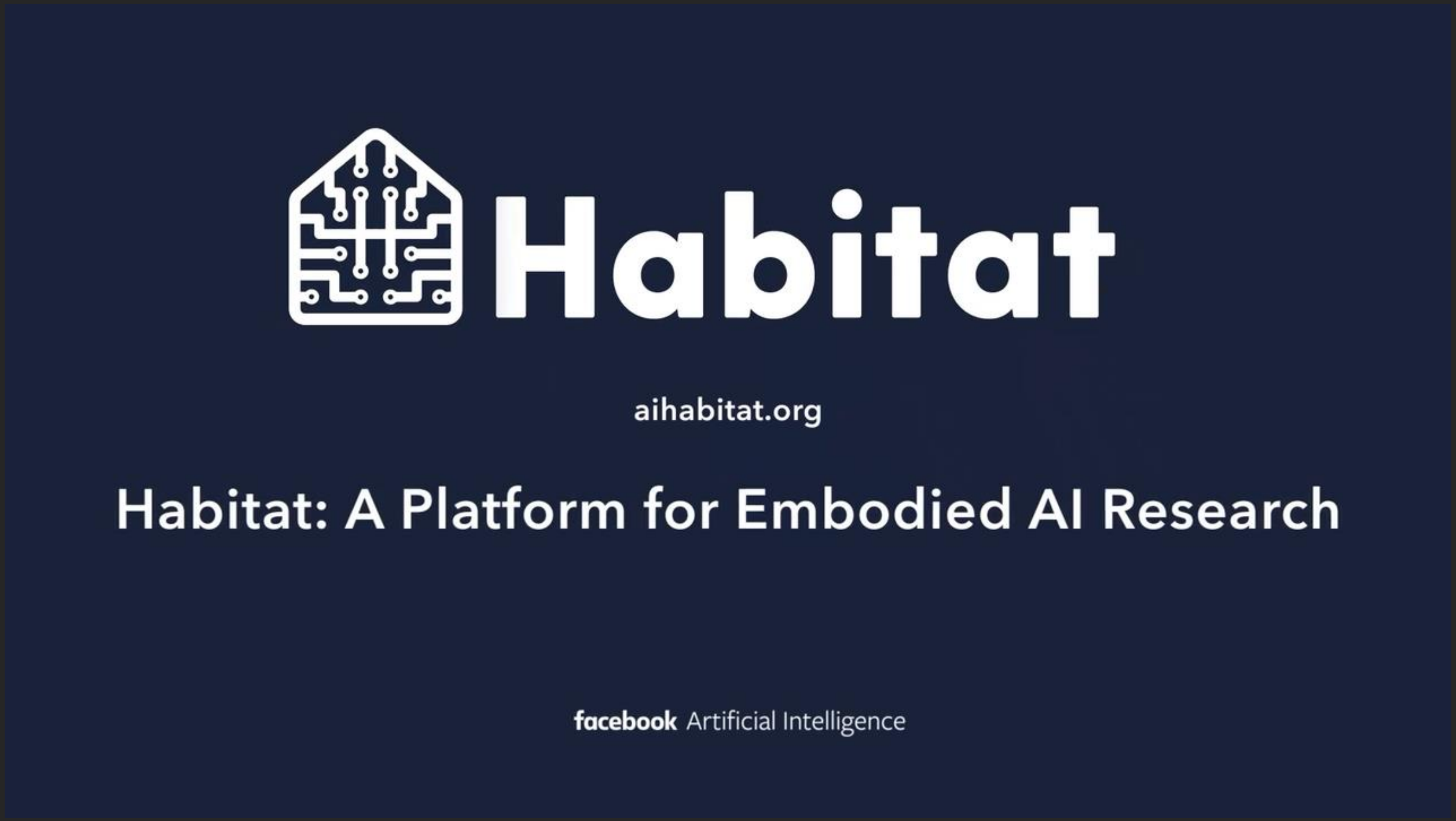
3D Scene



Semantic Segments

Increasing Demand for 3D Content

New driver: Artificial Intelligence ("Graphics for AI")



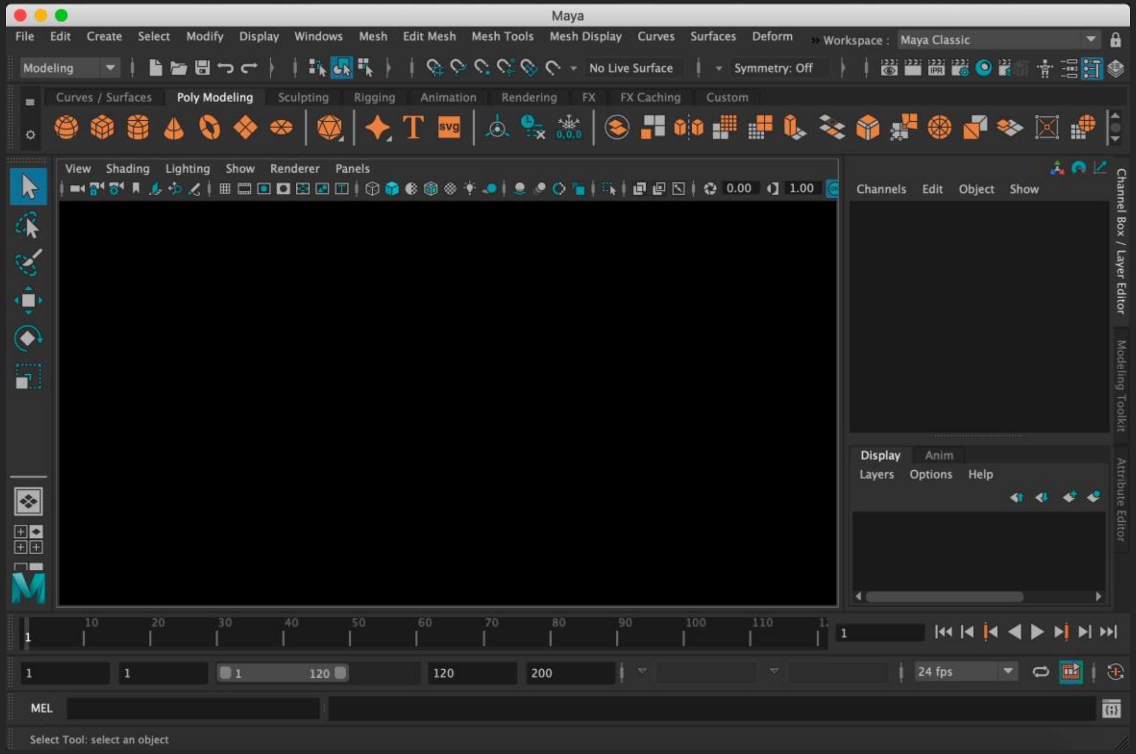
Increasing Demand for 3D Content

New driver: Artificial Intelligence ("Graphics for AI")

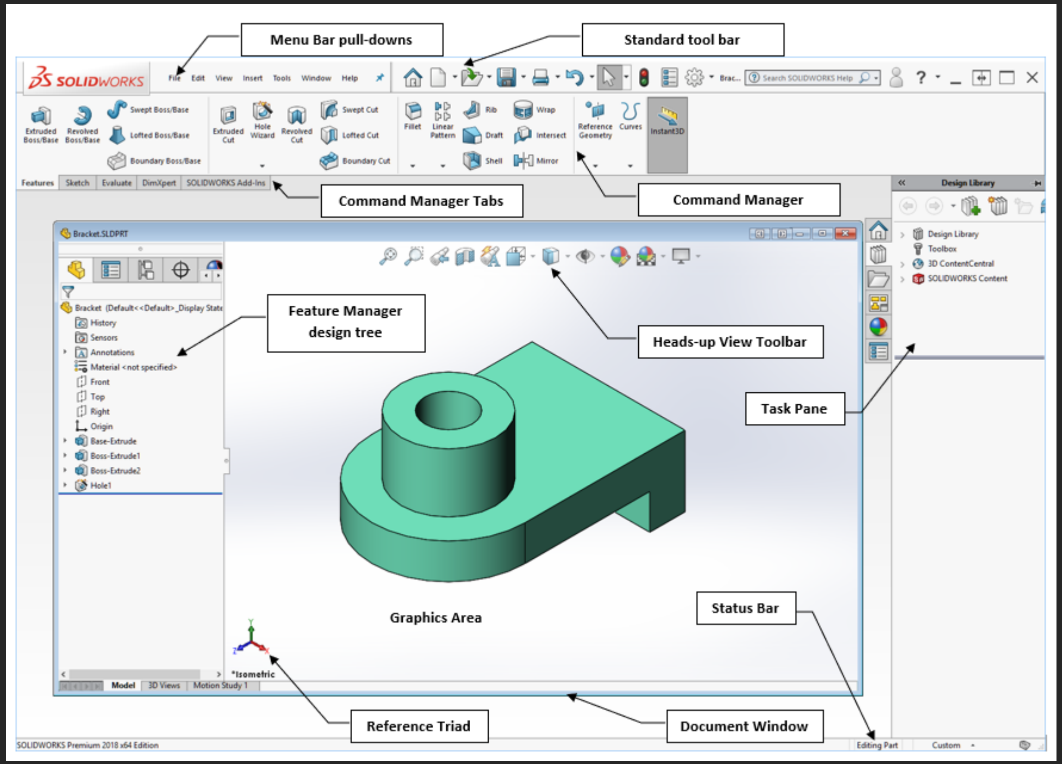
Learning to Generalize Kinematic Models to Novel Objects, Abbatematteo et al. 2019

Current Practice Can’t Meet Demand

Mannual 3D modeling: still slow, still hard to learn



Maya



Solidworks

“The difficulty of generating images has been overwhelmed by a five-thousand-fold improvement in price/performance of computing.

What remains hard is modeling…the grand challenges in three-dimensional graphics are to ***make simple modeling easy*** *and to* ***make complex modeling accessible to far more people****.”*

— Bob Sproull, 1990

Generative Models to the Rescue!?

For the purposes of this talk:

Generative model: a procedure which can be executed to generate novel instances of some 3D object class

Benefits of Generative Models

3D content generation at scale

SpeedTree, Unreal Engine

CityEngine

Benefits of Generative Models

Explore modeling possibilities

Learning Implicit Fields for Generative Shape Modeling , Chen & Zhang 2019

Benefits of Generative Models

Strong prior for vision systems

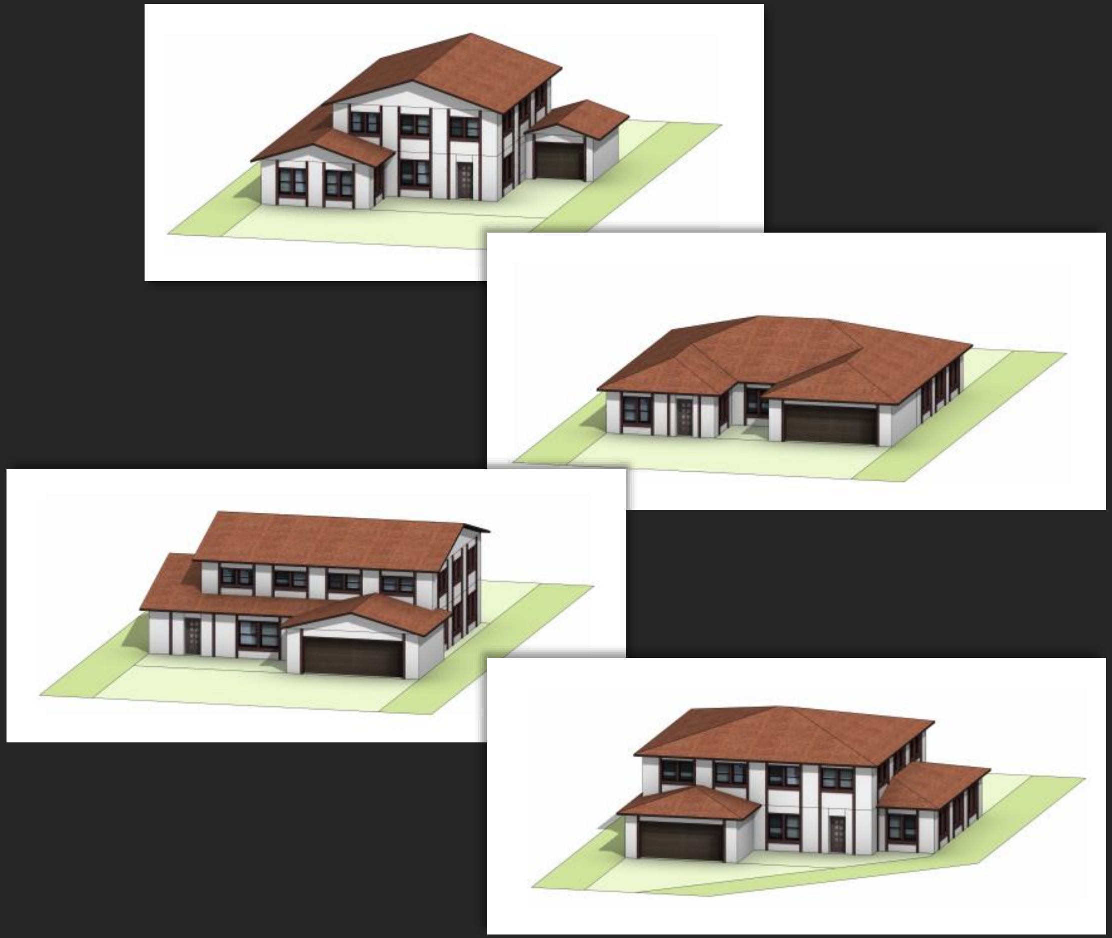
StructureNet: Hierarchical Graph Networks for 3D Shape Generation, Mo et al. 2019

Two Classes of Generative Model

Procedural Models

Pros:

- High quality output by construction



Advanced Procedural Modeling of Architecture, Schwartz & Muller 2015

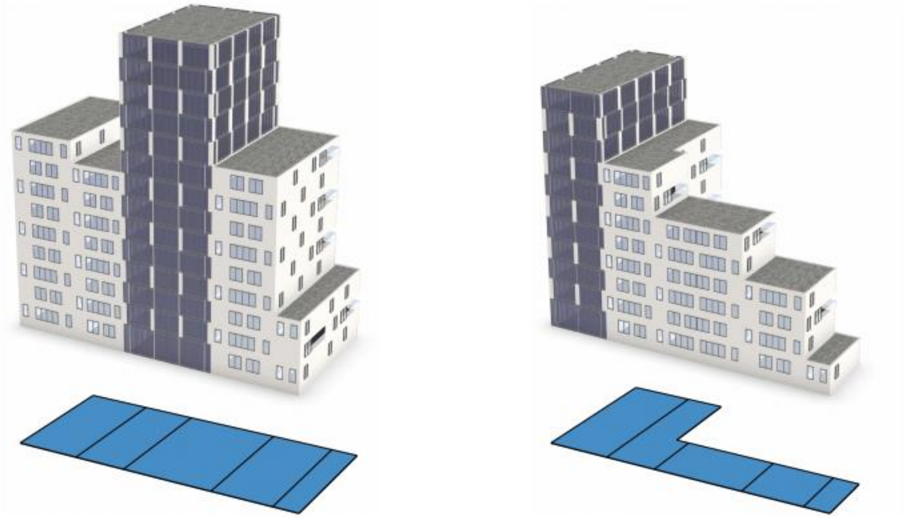
Two Classes of Generative Model

Procedural Models

Pros:

- High quality output by construction
- Interpretable & editable

```
Parcel --> split("x") { rand(8, 16): Footprint | ~1: Parcel }
Footprint --> event(IdentifyLargest) extrude(area()/6) Mass
Mass --> case { get("isLargest"): Offices | else: Apartments }
Offices --> ...
Apartments --> ...

event IdentifyLargest =
  with(A = map(n:$nodes, area(n)), largest = index(A, max(A)),
    foreach($nodes) { set("isLargest", $index == largest) } )
```



Advanced Procedural Modeling of Architecture, Schwartz & Muller 2015

Two Classes of Generative Model

Procedural Models

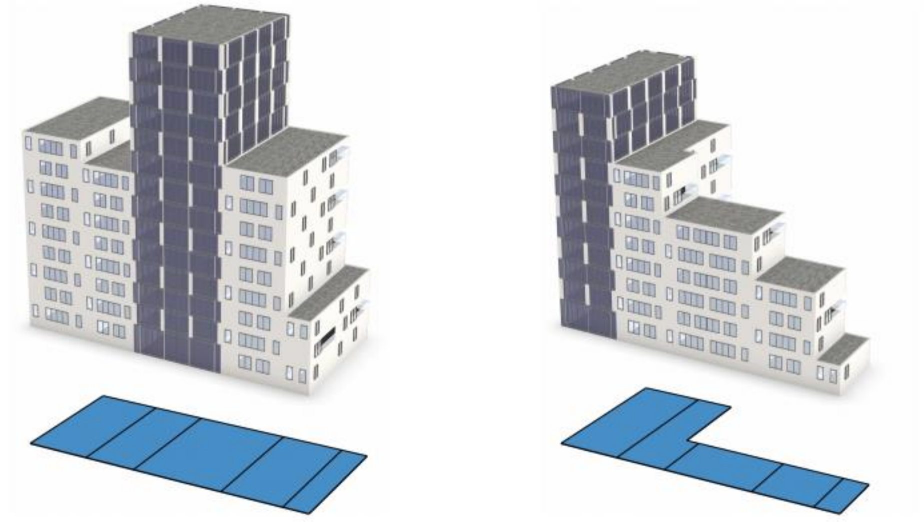
Pros:

- High quality output by construction
- Interpretable & editable

Cons:

- Difficult to author

```
Parcel --> split("x") { rand(8, 16): Footprint | ~1: Parcel }
Footprint --> event(IdentifyLargest) extrude(area()/6) Mass
Mass --> case { get("isLargest"): Offices | else: Apartments }
Offices --> ...
Apartments --> ...

event IdentifyLargest =
  with(A = map(n:$nodes, area(n)), largest = index(A, max(A)),
    foreach($nodes) { set("isLargest", $index == largest) } )
```



Advanced Procedural Modeling of Architecture, Schwartz & Muller 2015

Two Classes of Generative Model

Procedural Models

Pros:

- High quality output by construction
- Interpretable & editable

Cons:

- Difficult to author
- Limited output variety

Learning to Generalize Kinematic Models to Novel Objects, Abbatematteo et al. 2019

Two Classes of Generative Model

Learning Implicit Fields for Generative Shape Modeling , Chen & Zhang 2019

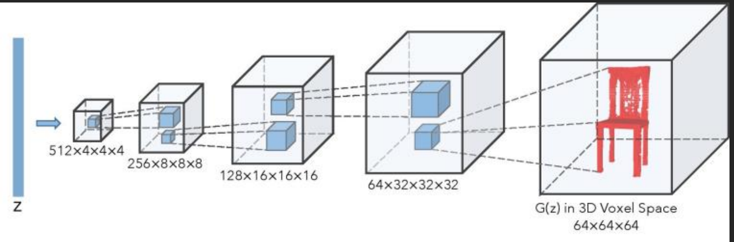
Deep Generative Models

Pros:

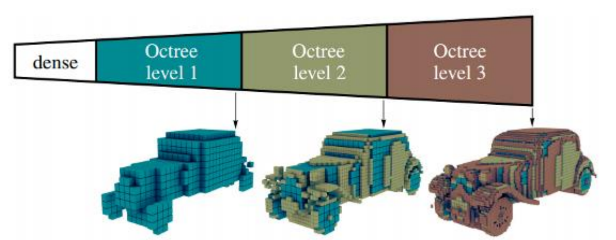
- Variety (any class of shape)
- Easy to author (“just add data”)

Recent High-Profile Successes

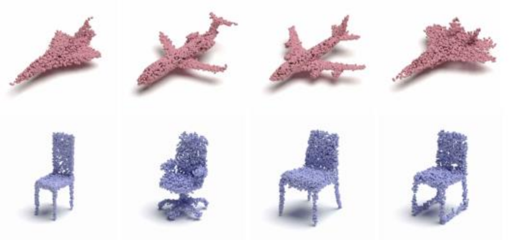
3D-GAN



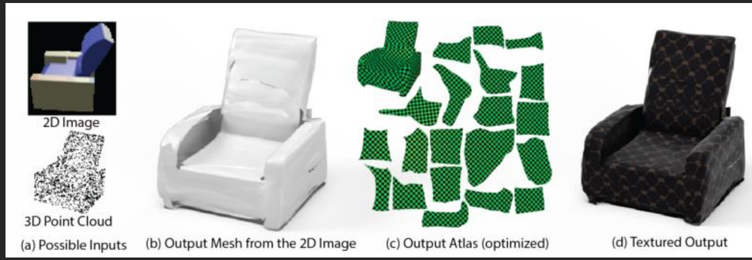
Octree Generating Nets



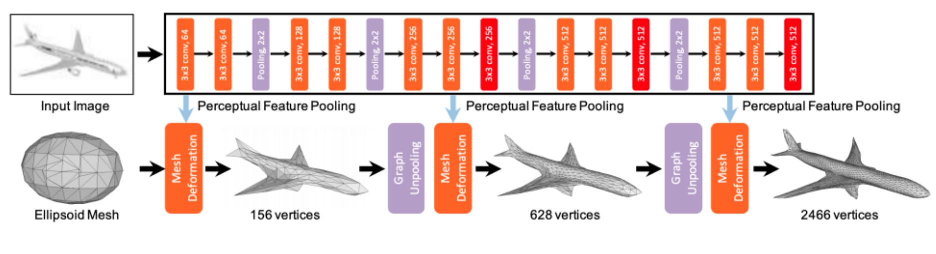
PointFlow



AtlasNet

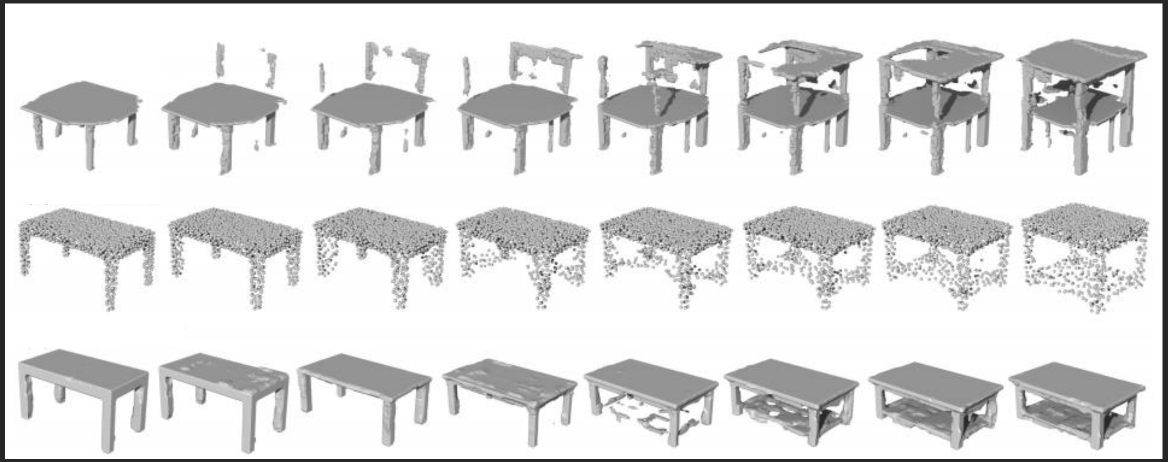


Pixel2Mesh



IM-Net

Two Classes of Generative Model



Deep Generative Models

Pros:

- Variety (any class of shape)
- Easy to author ("just add data")

Cons:

- Inconsistent output quality
- Inscrutable representation

Two Classes of Generative Model

Procedural Models

Pros:

- High quality output by construction
- Interpretable & editable

Cons:

- Difficult to author
- Limited output variety

Deep Generative Models

Pros:

- Variety (any class of shape)
- Easy to author ("just add data")

How can we get all of these…

Cons:

- Inconsistent output quality
- Inscrutable representation

Two Classes of Generative Model

Procedural Models

Pros:

- High quality output by construction
- Interpretable & editable

Cons:

- Difficult to author
- Limited output variety

Deep Generative Models

Pros:

- Variety (any class of shape)
- Easy to author ("just add data")

How can we get all of these…

Cons:

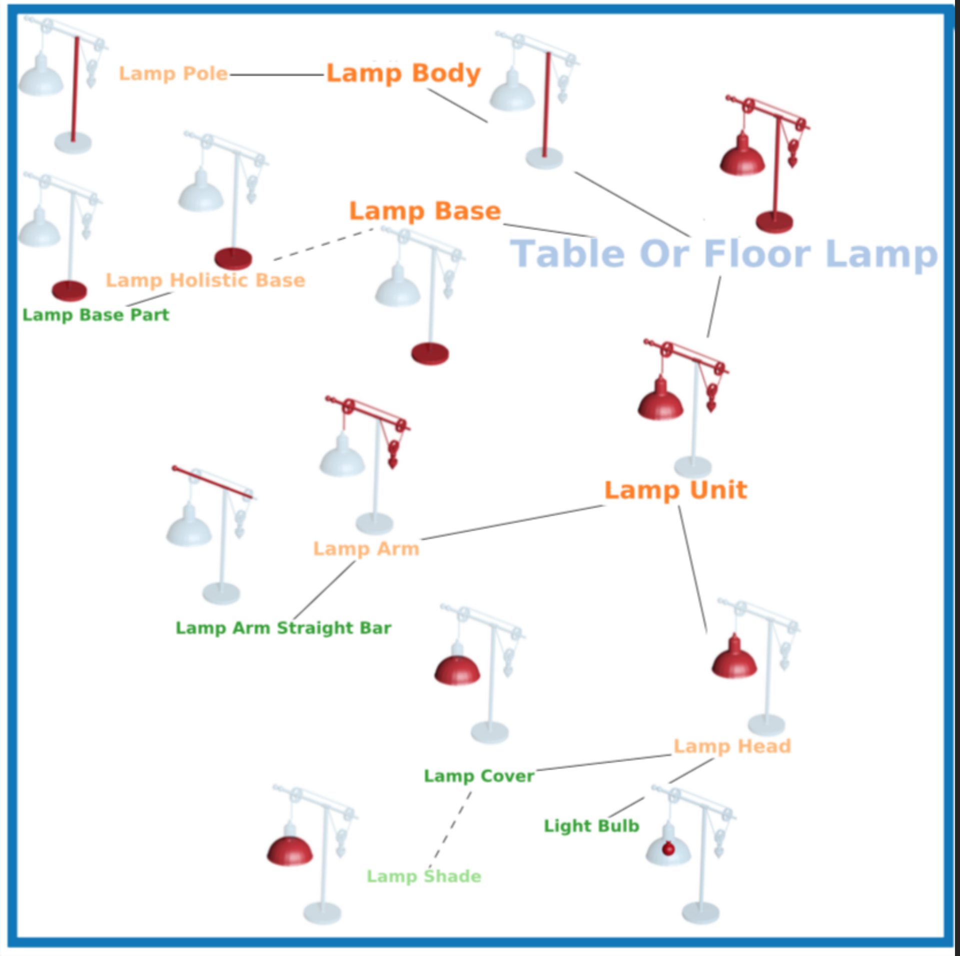
- Inconsistent output quality
- Inscrutable representation

…with none of these?

Generative Models Capture Variaton

Some modes can easily be expressed symbolically:

- Hierarchy

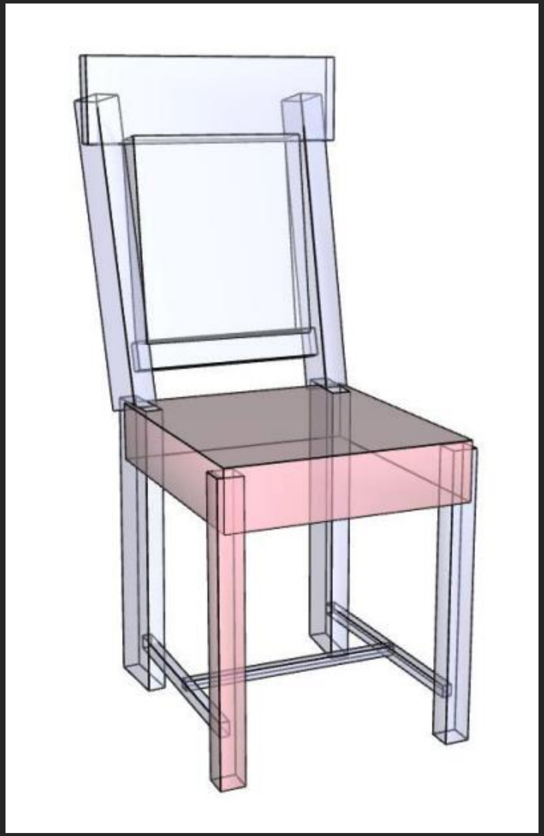


StructureNet: Hierarchical Graph Networks for 3D Shape Generation, Mo et al. 2019

Generative Models Capture Variaton

Some modes can easily be expressed symbolically:

- Hierarchy
- Connectivity

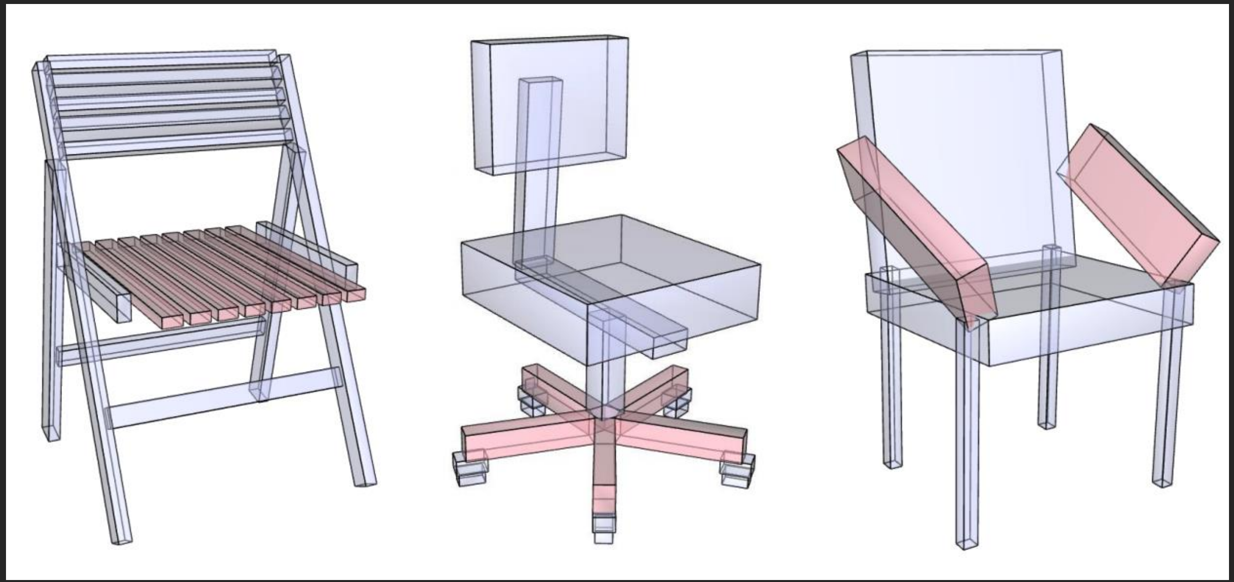


GRASS: Generative Recursive Autoencoders for Shape Structures, Li et al. 2018

Generative Models Capture Variaton

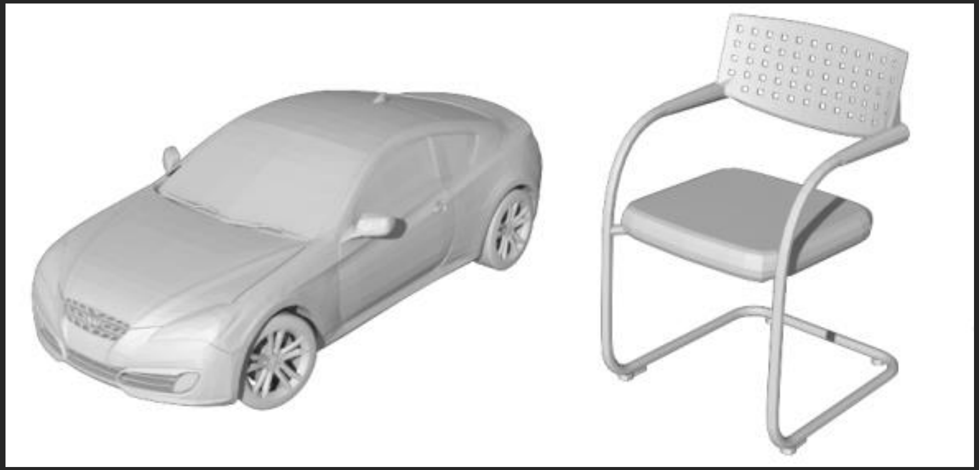
Some modes can easily be expressed symbolically:

- Hierarchy
- Connectivity
- Symmetry
- …



GRASS: Generative Recursive Autoencoders for Shape Structures, Li et al. 2018

Generative Models Capture Variaton



Learning Implicit Fields for Generative Shape Modeling , Chen & Zhang 2019

Some modes are hard to express symbolically:

- Fine-detailed geometry

Generative Models Capture Variaton

Learning Implicit Fields for Generative Shape Modeling , Chen & Zhang 2019

Some modes are hard to express symbolically:

- Fine-detailed geometry
- Complex inter-part correlations
- ...

Generative Models Capture Variaton

Some modes can easily be expressed symbolically:

- Hierarchy
- Connectivity
- Symmetry
- ...

Some modes are hard to express symbolically:

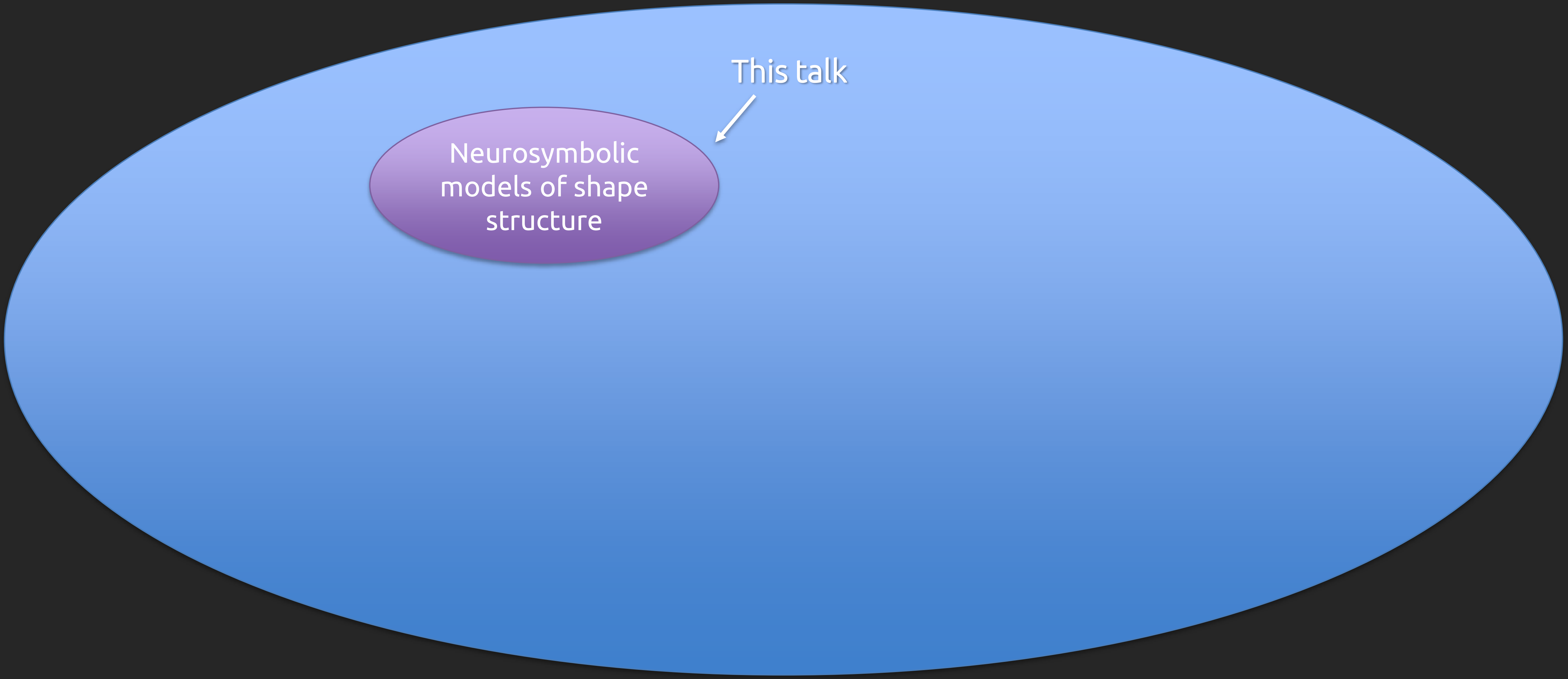
- ...etry
- ... correlations
- ...

Design Philosophy:
Use symbols where possible
Use neural nets for everything else

Neurosymbolic 3D Model:

A generative model of a class of 3D objects which models some modes of variability via explicit symbols and others via a neural latent space

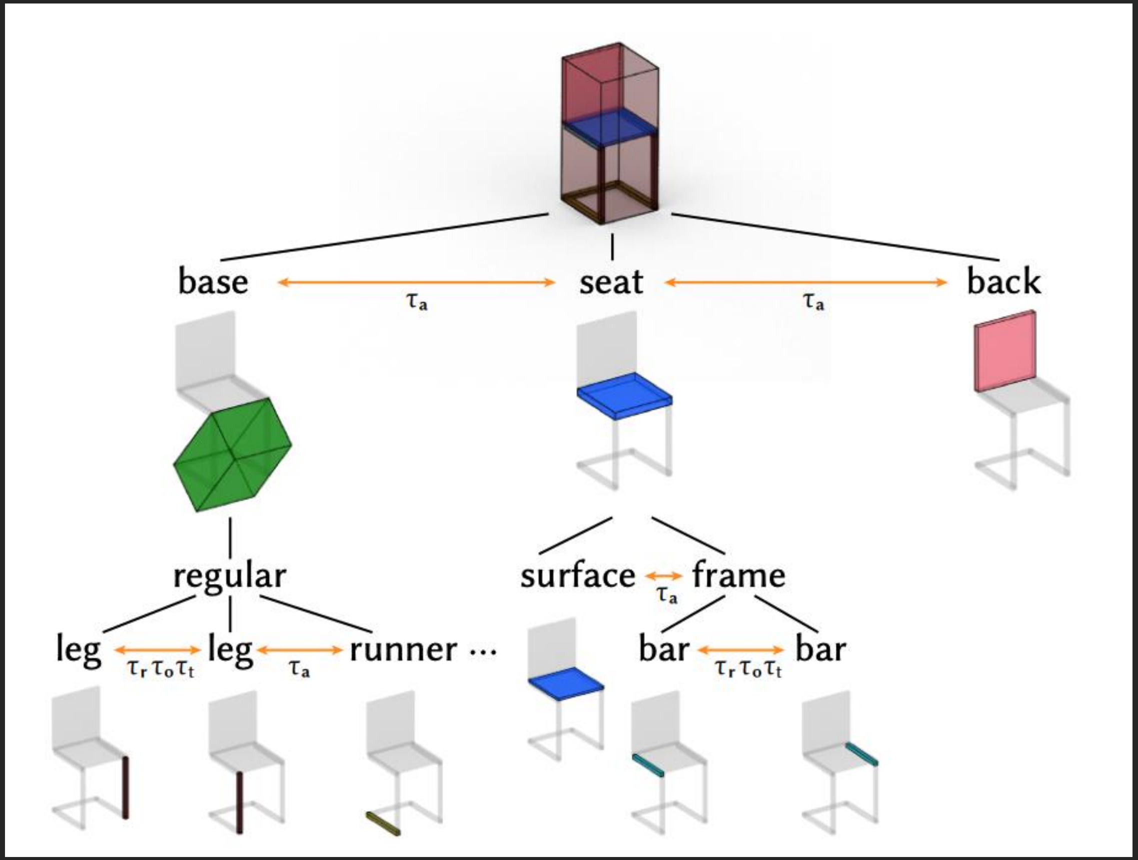
Neurosymbolic 3D Model Design Space



NEUROSYMBOLIC MODELS OF SHAPE STRUCTURE

What Do I Mean by *Shape Structure*?

- Parts (as oriented bounding boxes)
- Relations
  - Hierarchy, connectivity, symmetry, …
- Useful despite low geometric detail
  - Ex: robot motion planning → infer all parts + relations given point cloud observation
- Focus on *manufactured* objects
  - E.g. chairs, tables, airplanes…



StructureNet: Hierarchical Graph Networks for 3D Shape Generation, Mo et al. 2019

What Do I Mean by *Shape Structure*?

- Parts (as oriented bounding boxes)
- Relations
  - Hierarchy, connectivity, symmetry, …
- Useful despite low geometric detail
  - Ex: robot motion planning → infer all parts + relations given point cloud observation
- Focus on *manufactured* objects
  - E.g. chairs, tables, airplanes…
  - Can extend to organic objects via e.g. generalized cylinder decomposition

Generalized Cylinder Decomposition, Zhou et al. 2015

The “Holy Grail” of Structure Modeling

A single, interpretable procedural model that generates the structures of every object in a given shape class (e.g. chairs, airplanes)

But…

Two Classes of Generative Model

Procedural Models

Pros:

- High quality output by construction
- Interpretable & editable

Cons:

- Difficult to author
- Limited output variety

Can a strategic use of neural nets eliminate these?

Eliminating Procedural Cons

Problem: Hard to author

Solution: Train a neural net to write them for us

Problem: Limited output variety

Solution: Latent space of neural net will capture the variability that the symbolic program does not

ShapeAssembly: Learning to Generate Programs for 3D Shape Structure Synthesis [SIGGRAPH Asia 2020]

R. KENNY JONES, Brown University
THERESA BARTON, Brown University
XIANGHAO XU, Brown University
KAI WANG, Brown University
ELLEN JIANG, Brown University
PAUL GUERRERO, Adobe Research
NILOY MITRA, University College London
DANIEL RITCHIE, Brown University

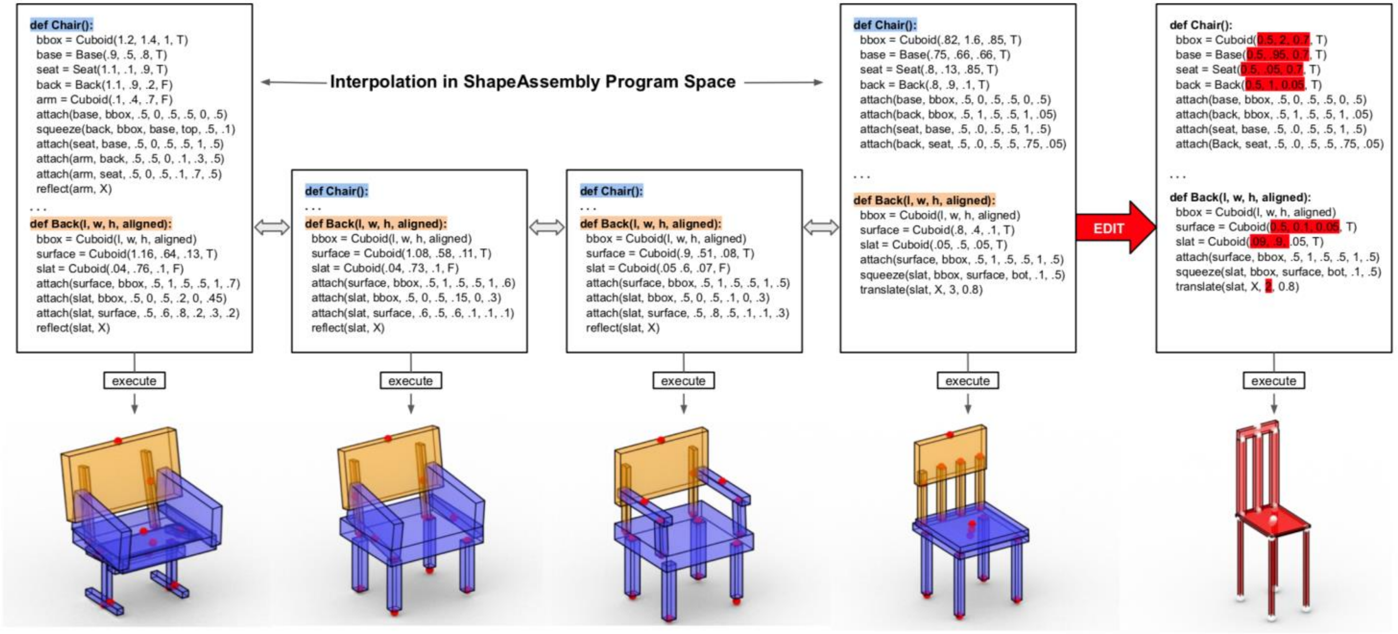
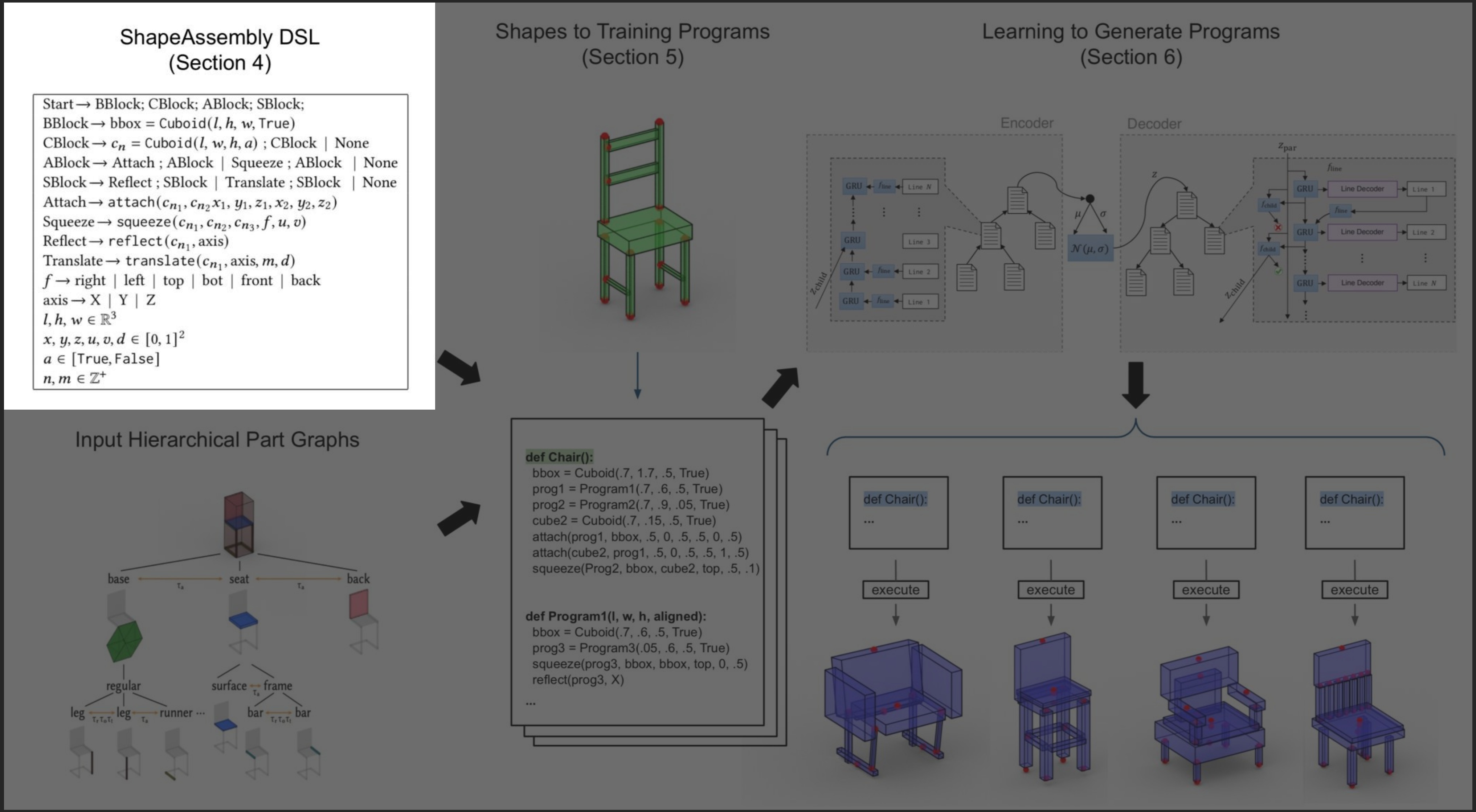


Fig. 1. We present a deep generative model which learns to write novel programs in ShapeAssembly, a domain-specific language for modeling 3D shape structures. Executing a ShapeAssembly program produces a shape composed of a hierarchical connected assembly of part proxies cuboids. Our method develops a well-formed latent space that supports interpolations between programs. Above, we show one such interpolation, and also visualize the geometry these programs produce when executed. In the last column, we manually edit the continuous parameters of a generated program, in order to produce a variant geometric structure with new topology.

A Neurosymbolic 3D Modeling Pipeline



ShapeAssembly

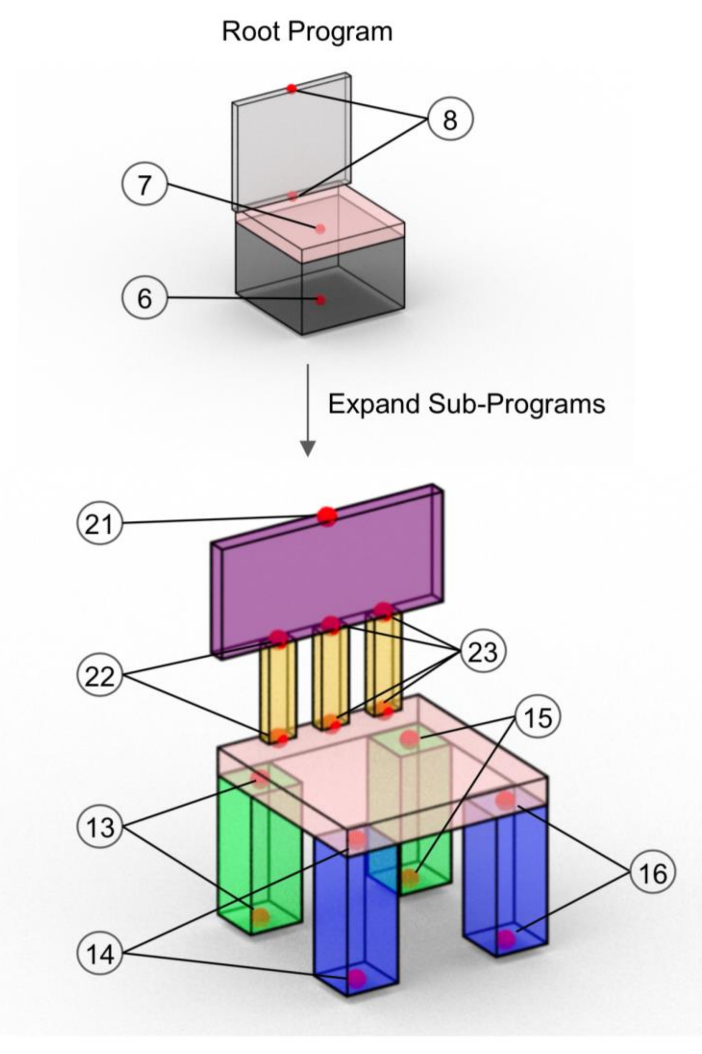
An “assembly language” for part-based shapes

Low-level instructions

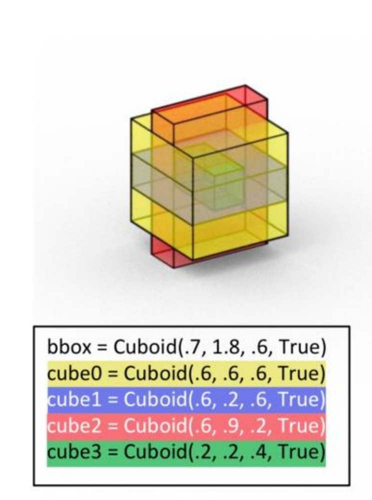
Operates by assembling parts

Anatomy of a ShapeAssembly Program

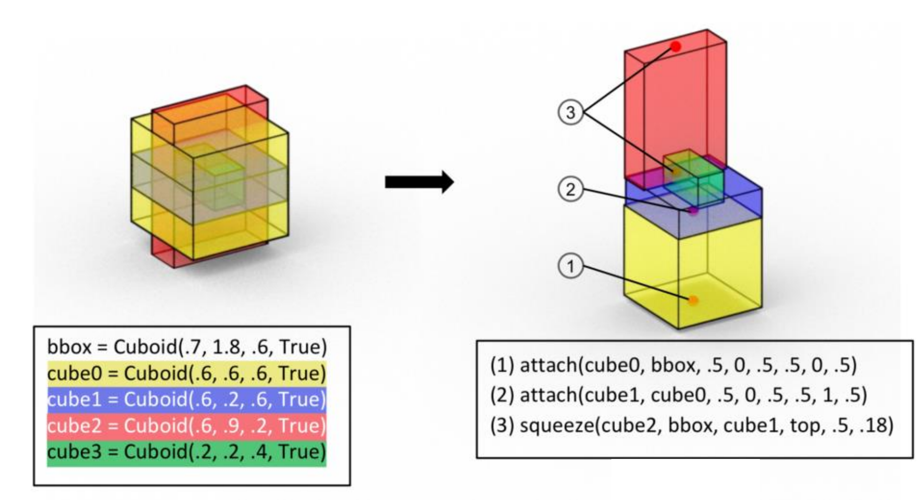
```
def Chair():
  bbox = Cuboid(1, 1.5, .8, True)
  base = Base(.8, .5, .8, True)
  cube1 = Cuboid(.8, .1, .8, True)
  back = Back(.9, .8, .07, True)
  attach(base, bbox, .5, 0, .5, .5, 0, .5)
  attach(cube1, base, .5, 0, .5, .5, 1, .5)
  squeeze(back, bbox, cube1, top, .5, .05)

def Base(l, w, h, aligned):
  bbox = Cuboid(l, w, h, aligned)
  cube0 = Cuboid(.2, .5, .2, True)
  cube1 = Cuboid(.2, .5, .2, True)
  squeeze(cube0, bbox, bbox, top, .1, .1)
  squeeze(cube1, bbox, bbox, top, .1, .8)
  reflect(cube0, X)
  reflect(cube1, X)

def Back(l, w, h, aligned):
  bbox = Cuboid(l, w, h, aligned)
  cube0 = Cuboid(.9, .4, .07, True)
  cube1 = Cuboid(.1, .4, .05, True)
  attach(cube0, bbox, .5, 1, .5, .5, 1, .5)
  squeeze(cube1, bbox, cube0, bot, .3, .5)
  translate(cube1, X, 2, .5)
```



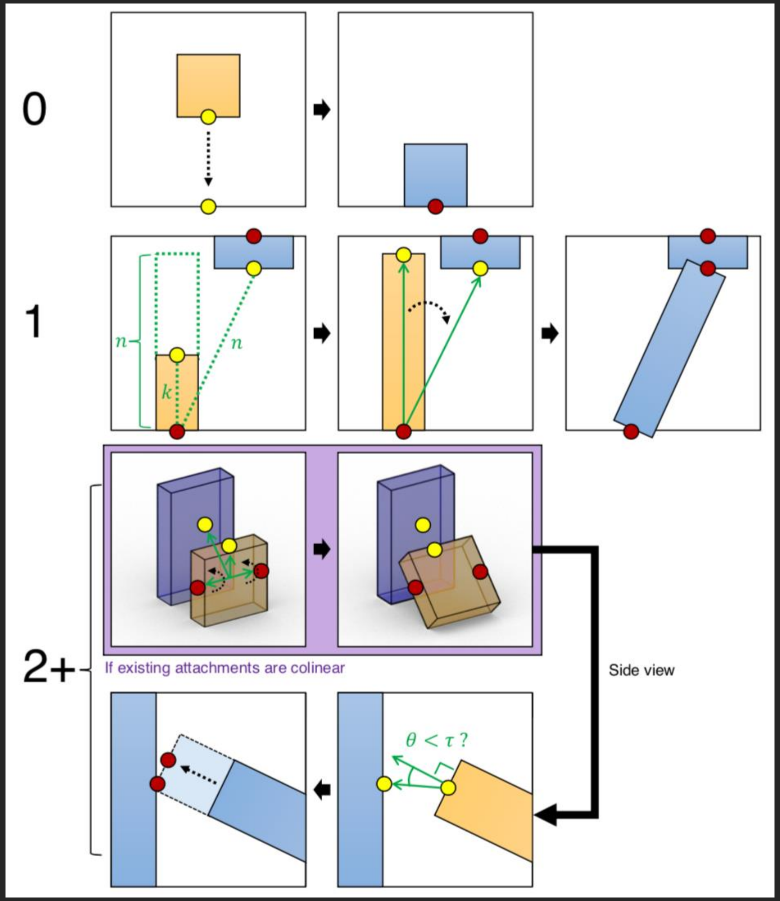
Execution Semantics



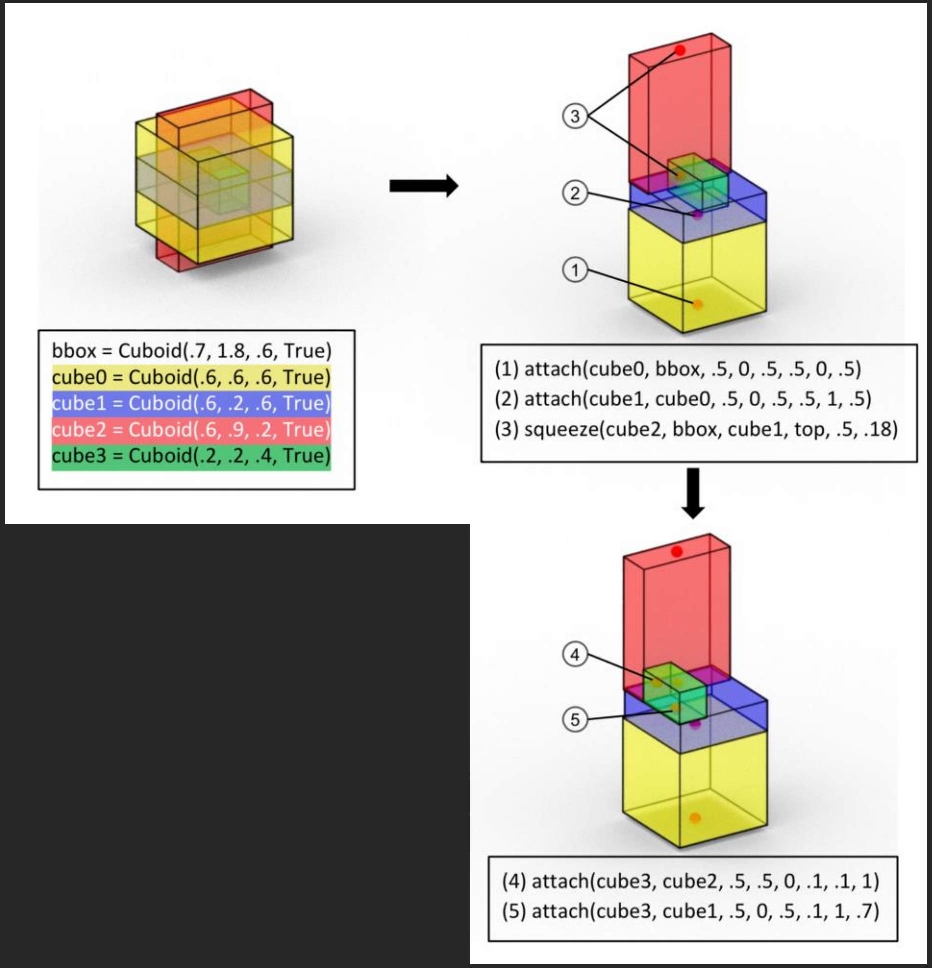
Execution Semantics



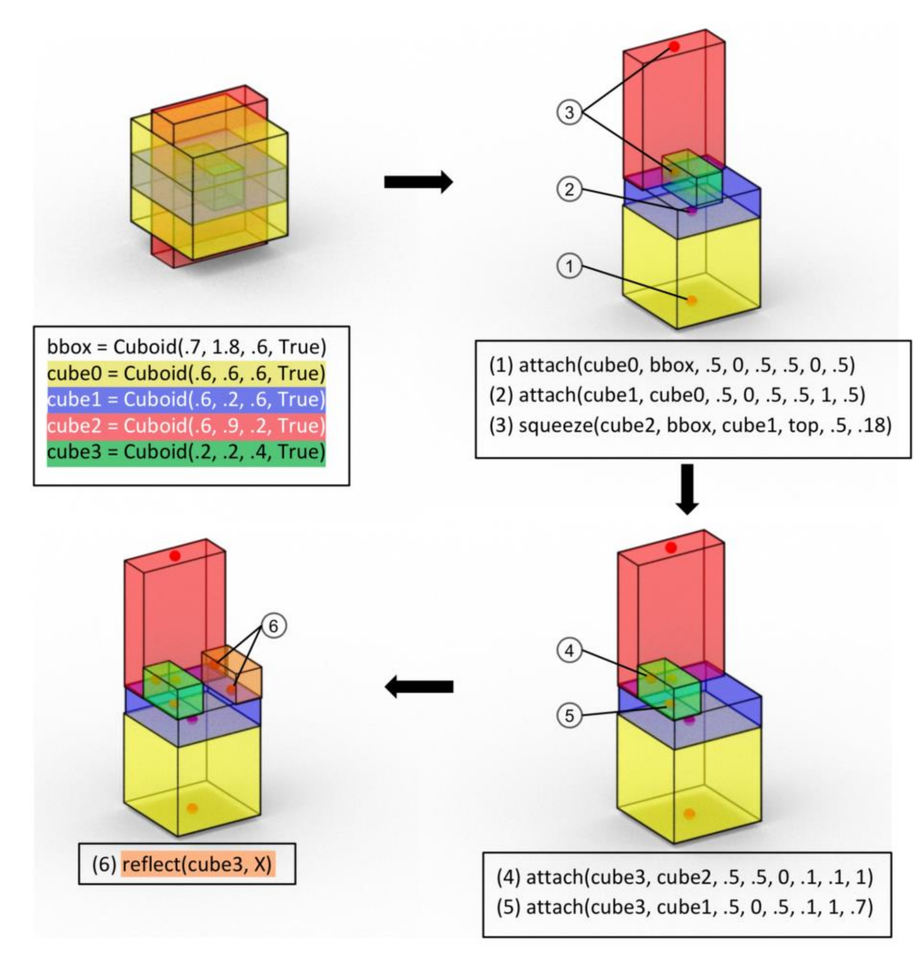
Semantics of **`attach`**



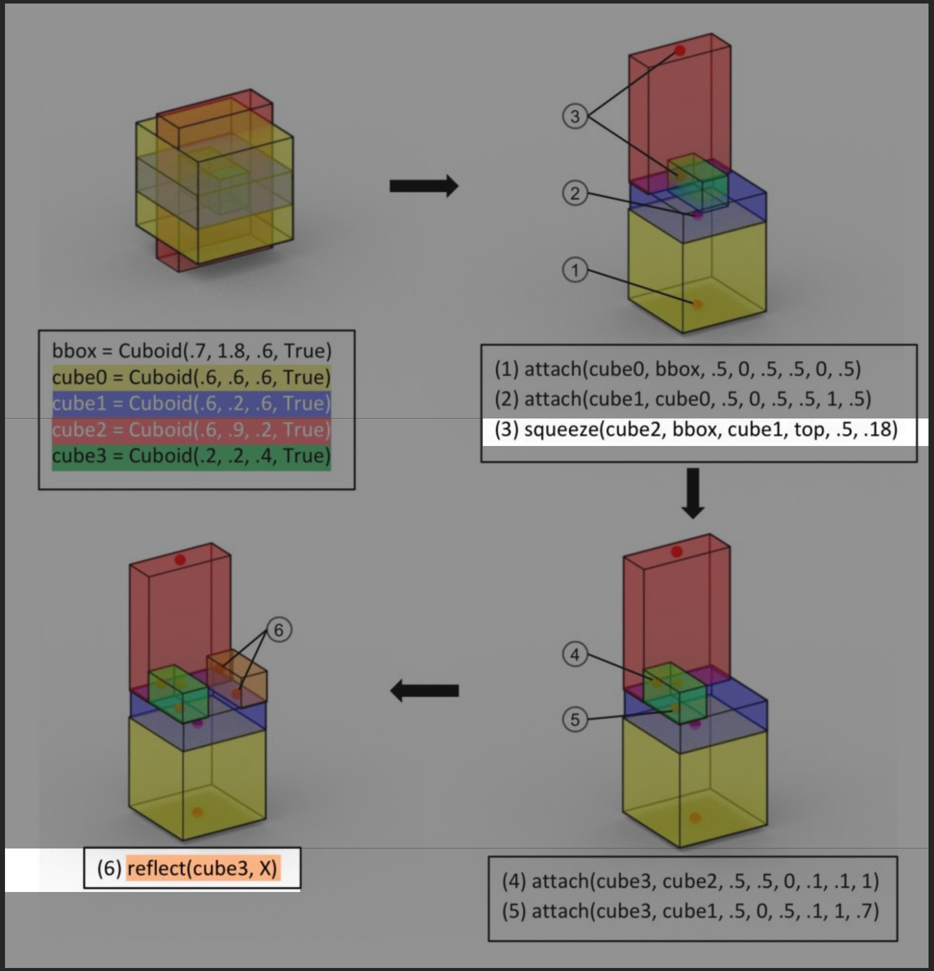
Execution Semantics



Execution Semantics



Execution Semantics

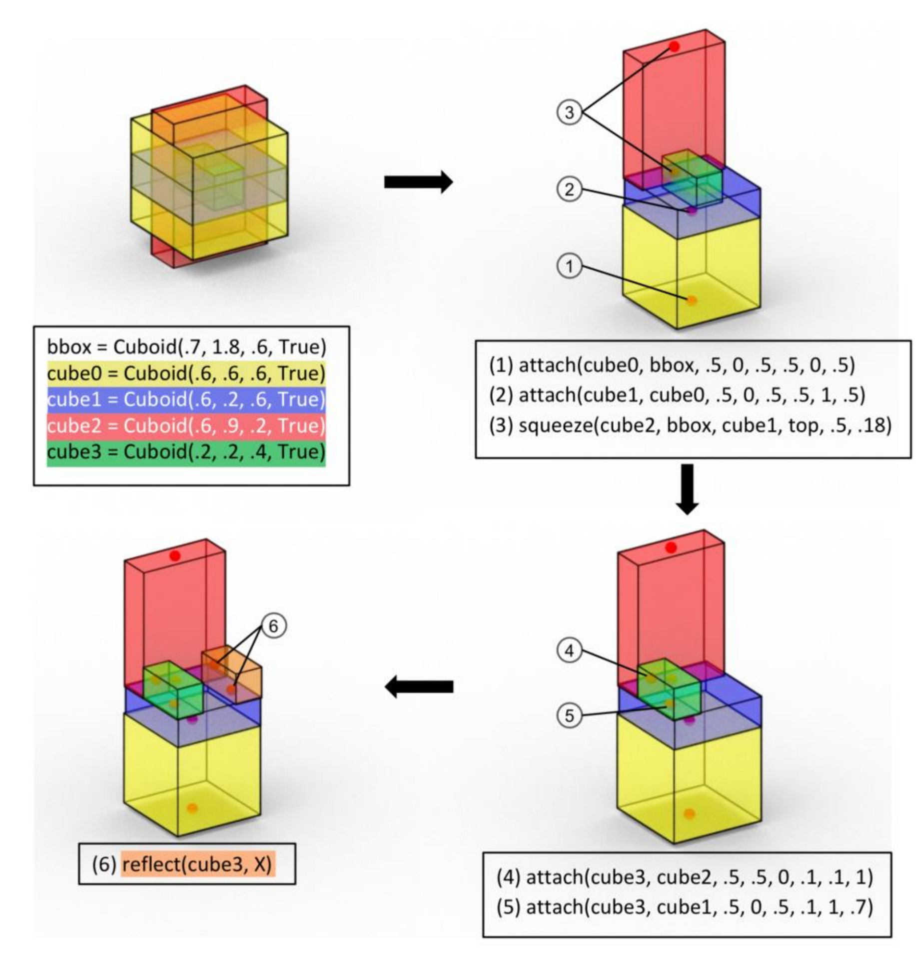


Macros:

`squeeze`, `reflect`, `translate`

expand into multiple `Cuboid` + `attach` statements

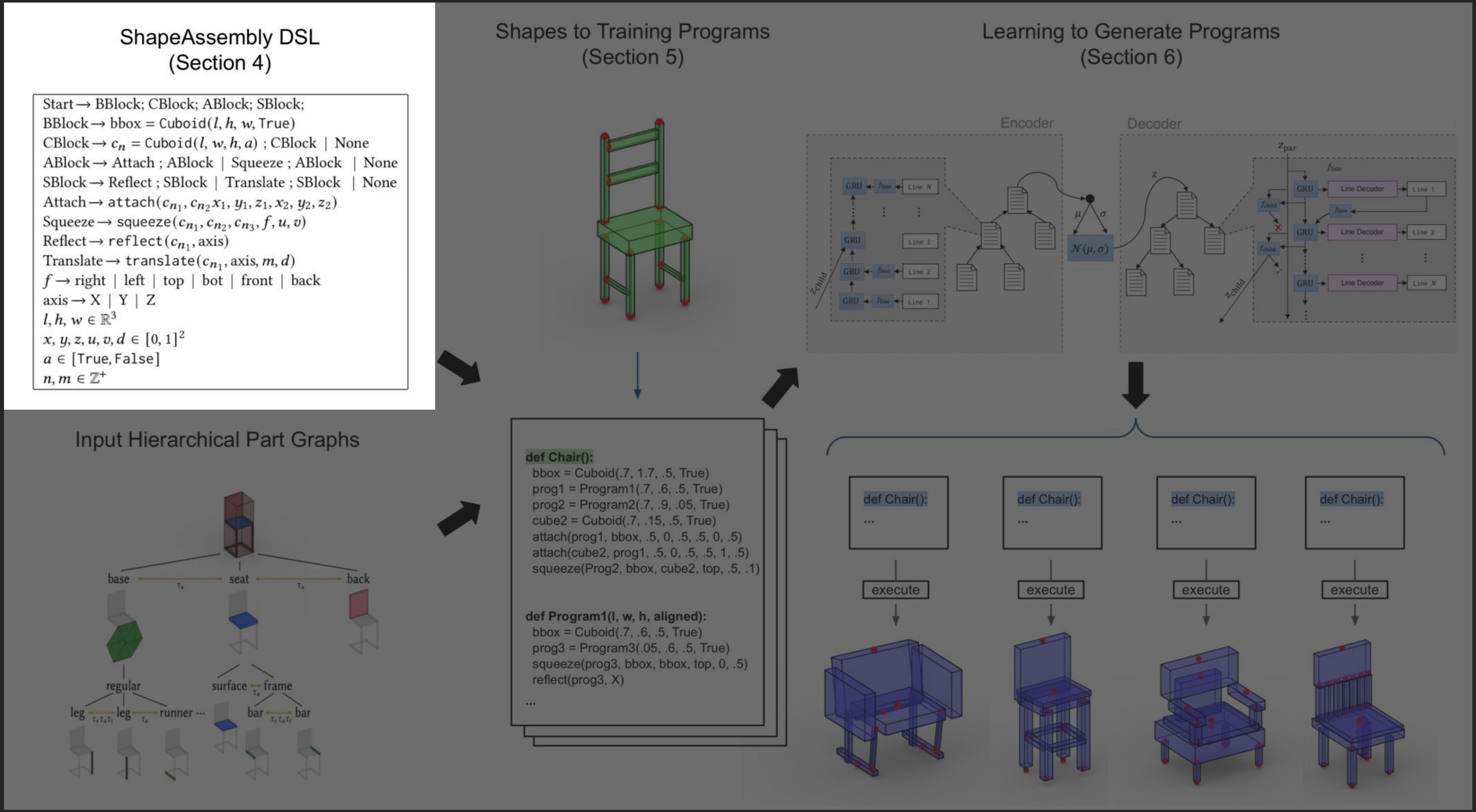
Execution Semantics



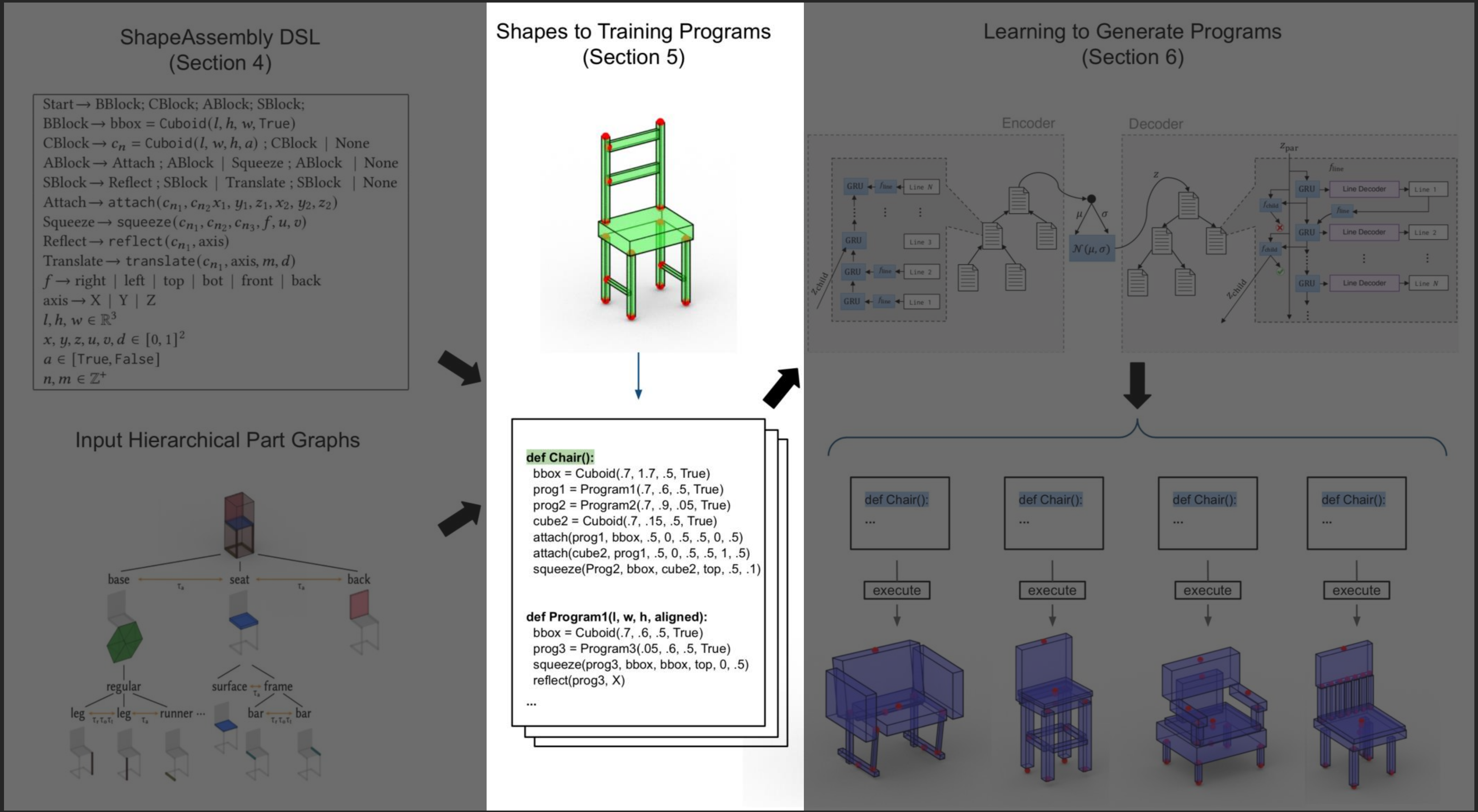
Differentiable execution:

Output geometry is differentiable with respect to continuous parameters of input program

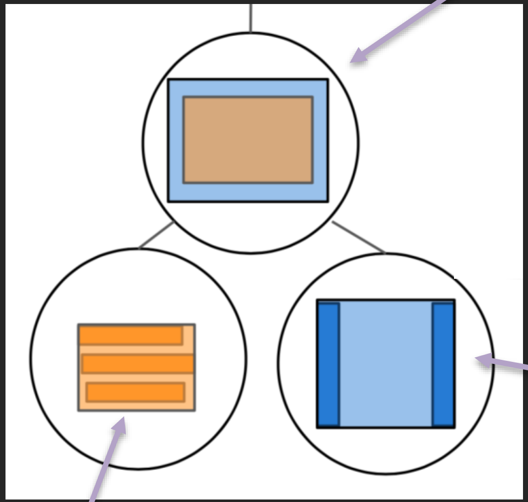
A Neurosymbolic 3D Modeling Pipeline



A Neurosymbolic 3D Modeling Pipeline

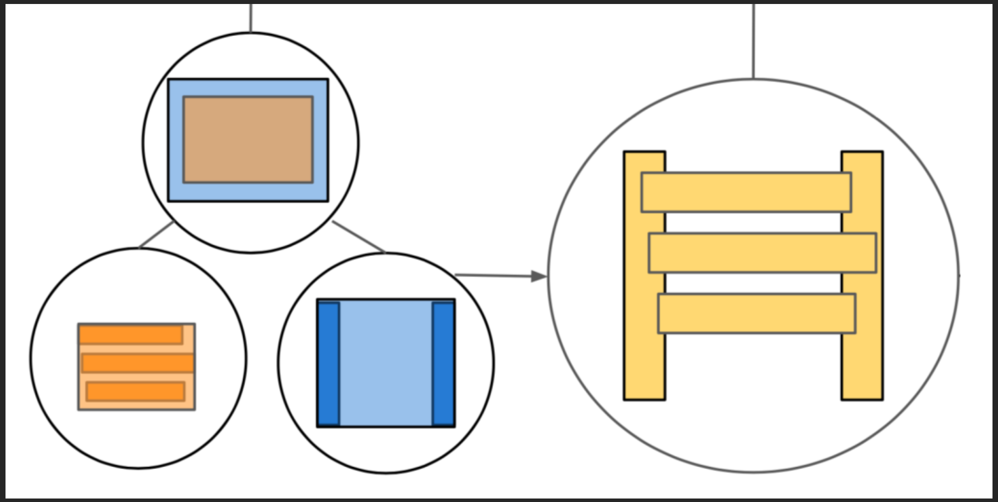


Extracting Programs from Shapes



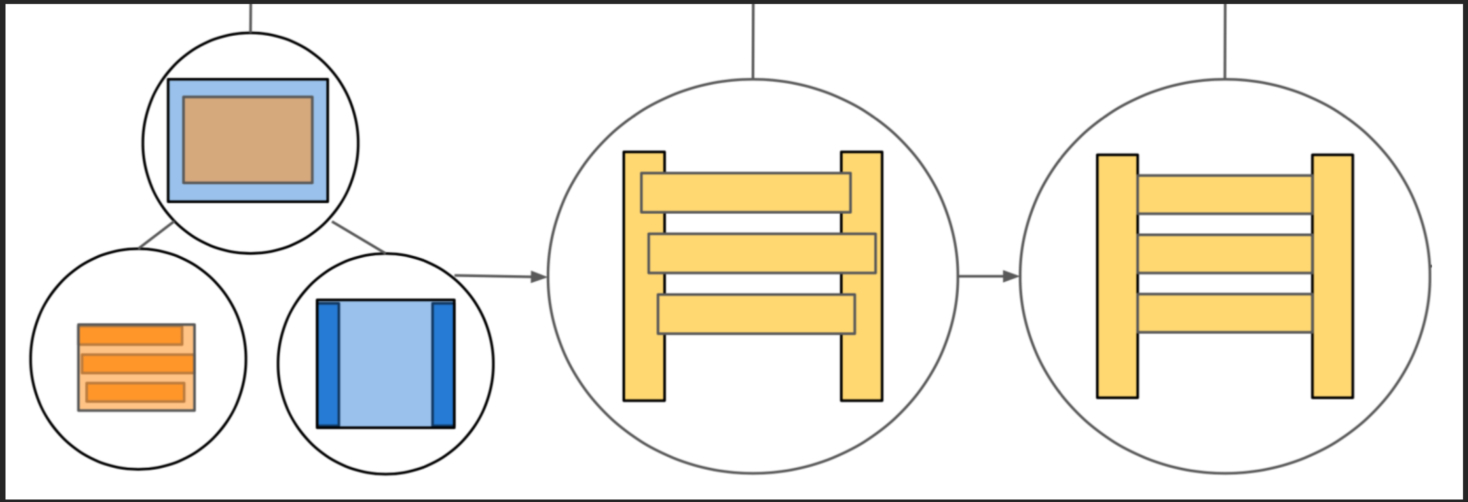
Local region of an
input hierarchical
part graph

Extracting Programs from Shapes



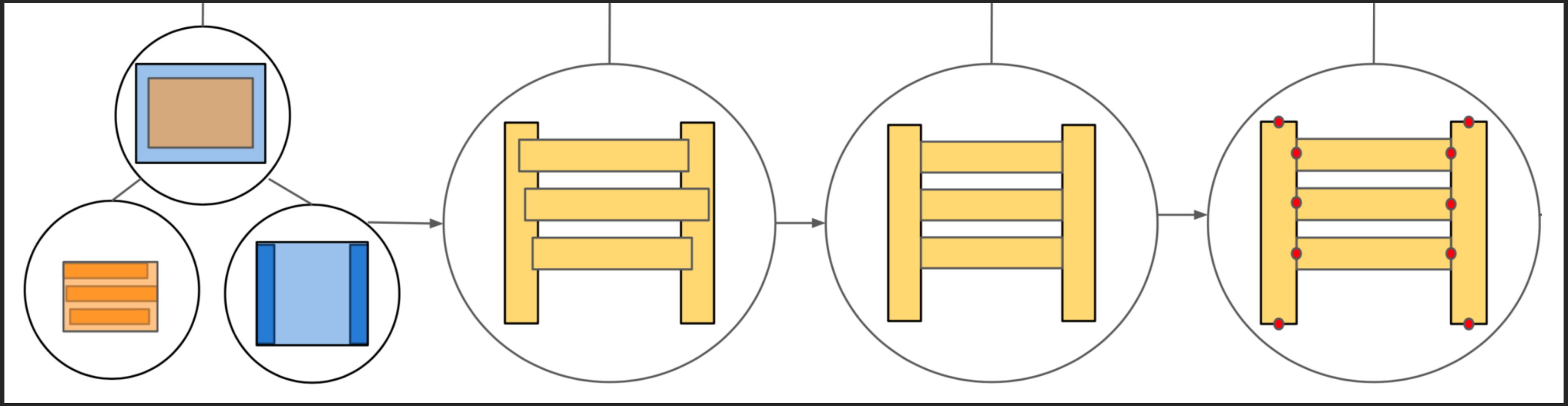
Locally flattening the hierarchy to make interacting leaf parts siblings

Extracting Programs from Shapes



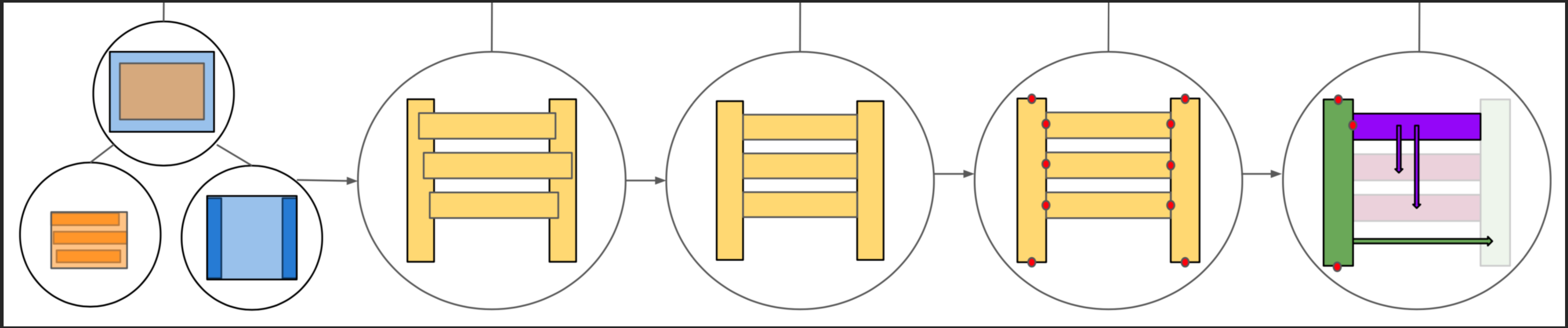
Shortening leaf parts that intersect other leaf parts

Extracting Programs from Shapes



Locating
attachment points
between parts

Extracting Programs from Shapes

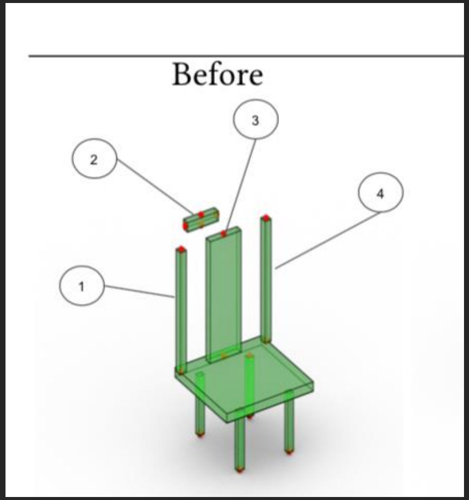


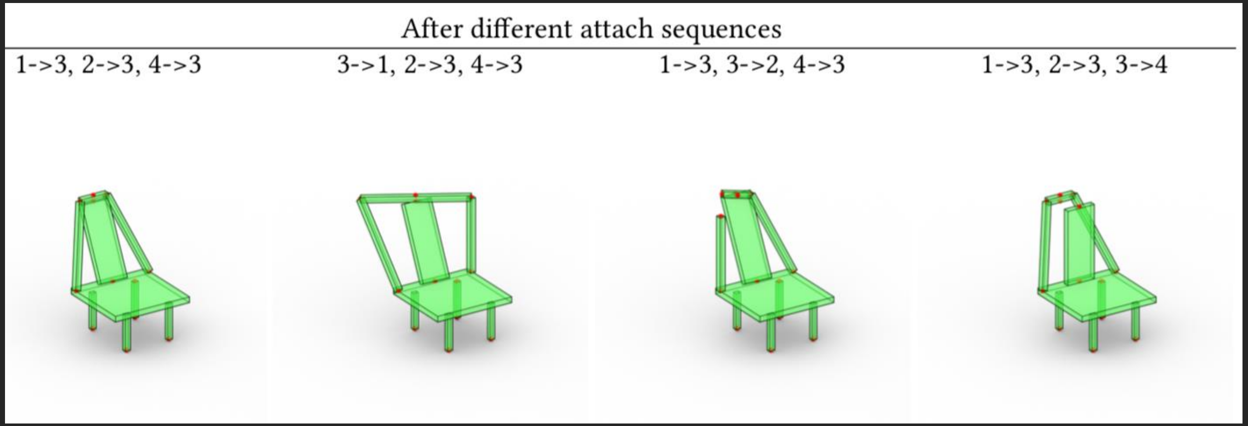
Forming leaf parts
into symmetry
groups

Extracting Programs from Shapes

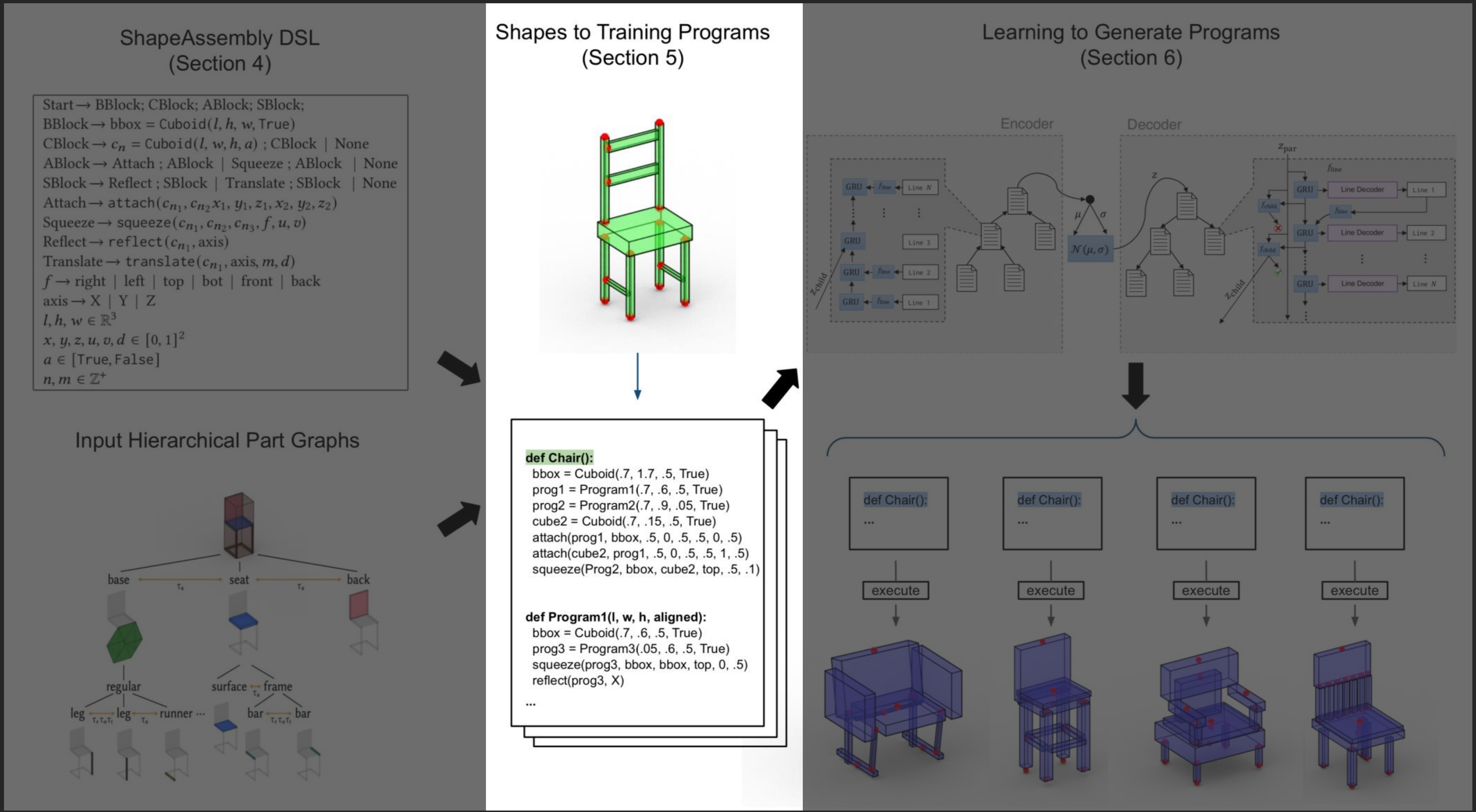
Ordering Attachments

- Due to imperative semantics, attach order matters
- Heuristics to prune possible orders, then check which one produces output that best fits the shape

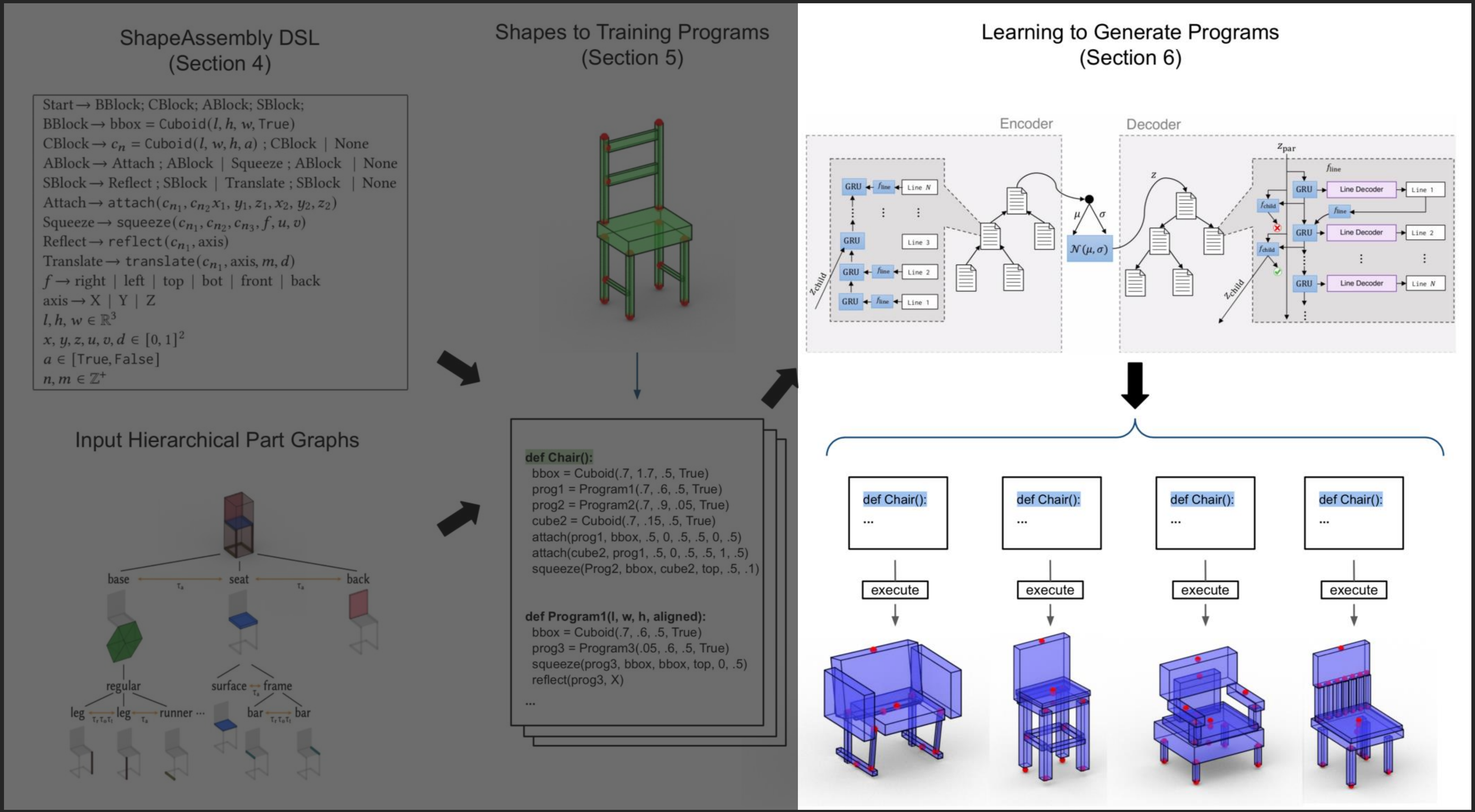




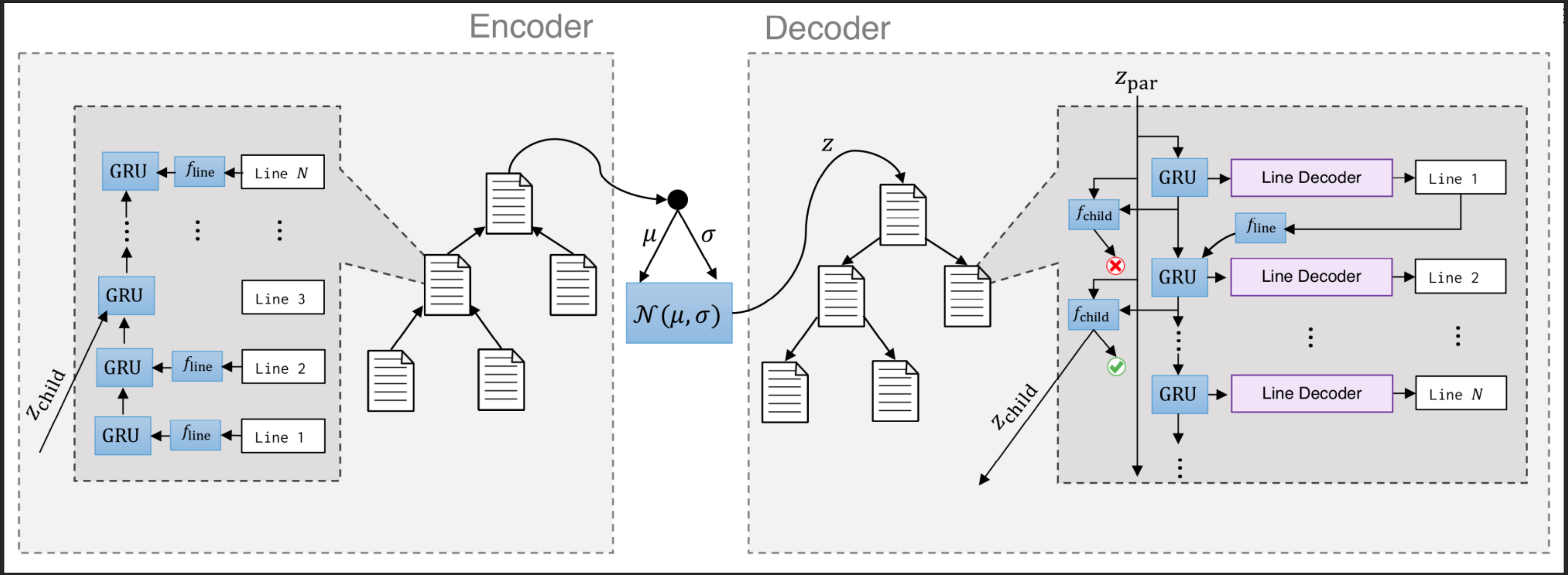
A Neurosymbolic 3D Modeling Pipeline



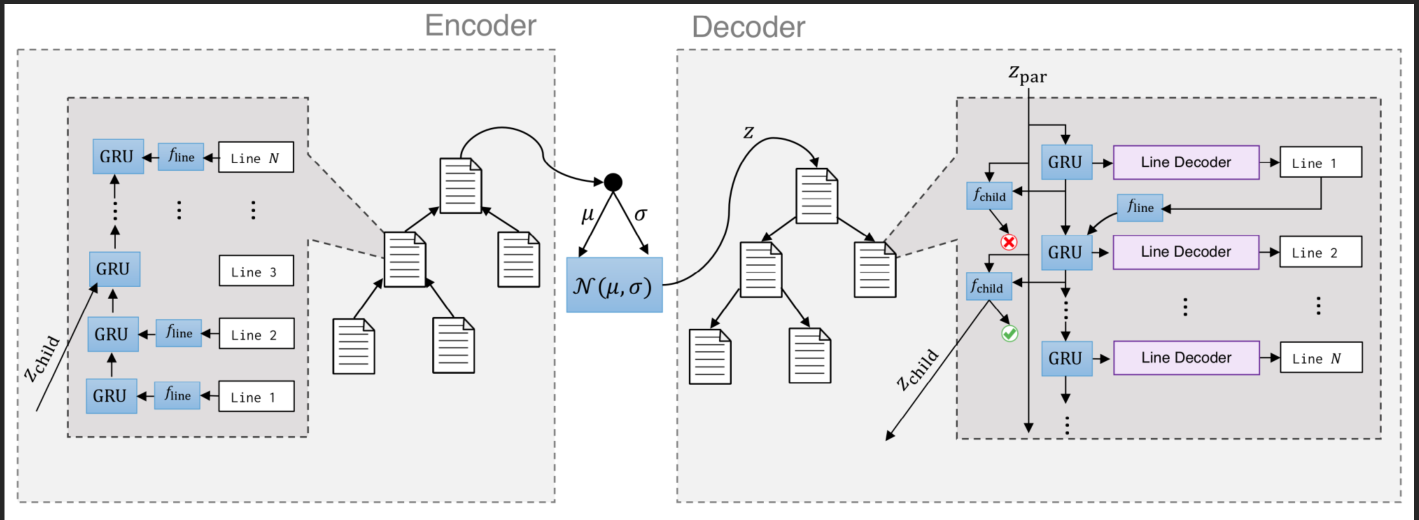
A Neurosymbolic 3D Modeling Pipeline

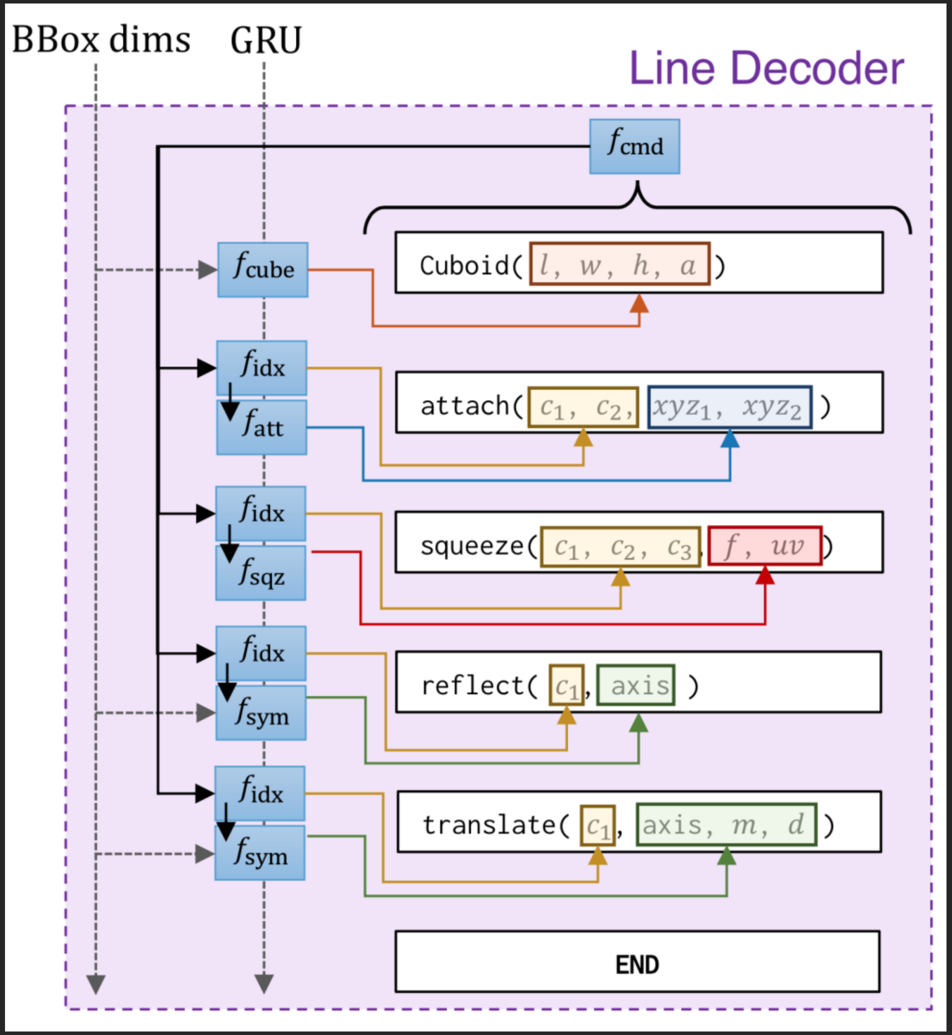


Learning to Write ShapeAssembly Programs



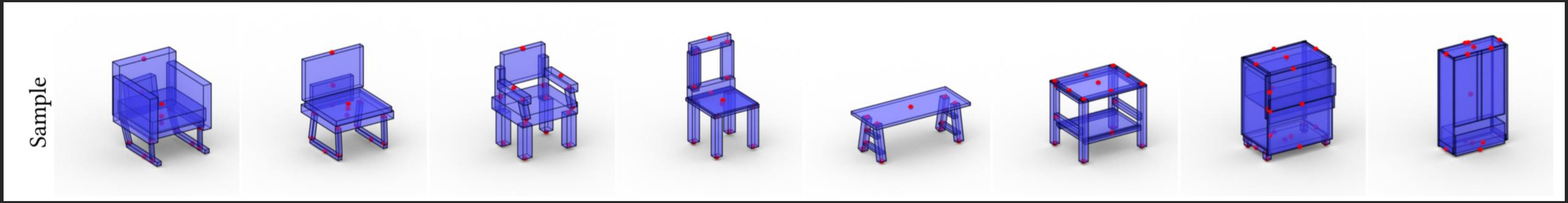
Learning to Write ShapeAssembly Programs





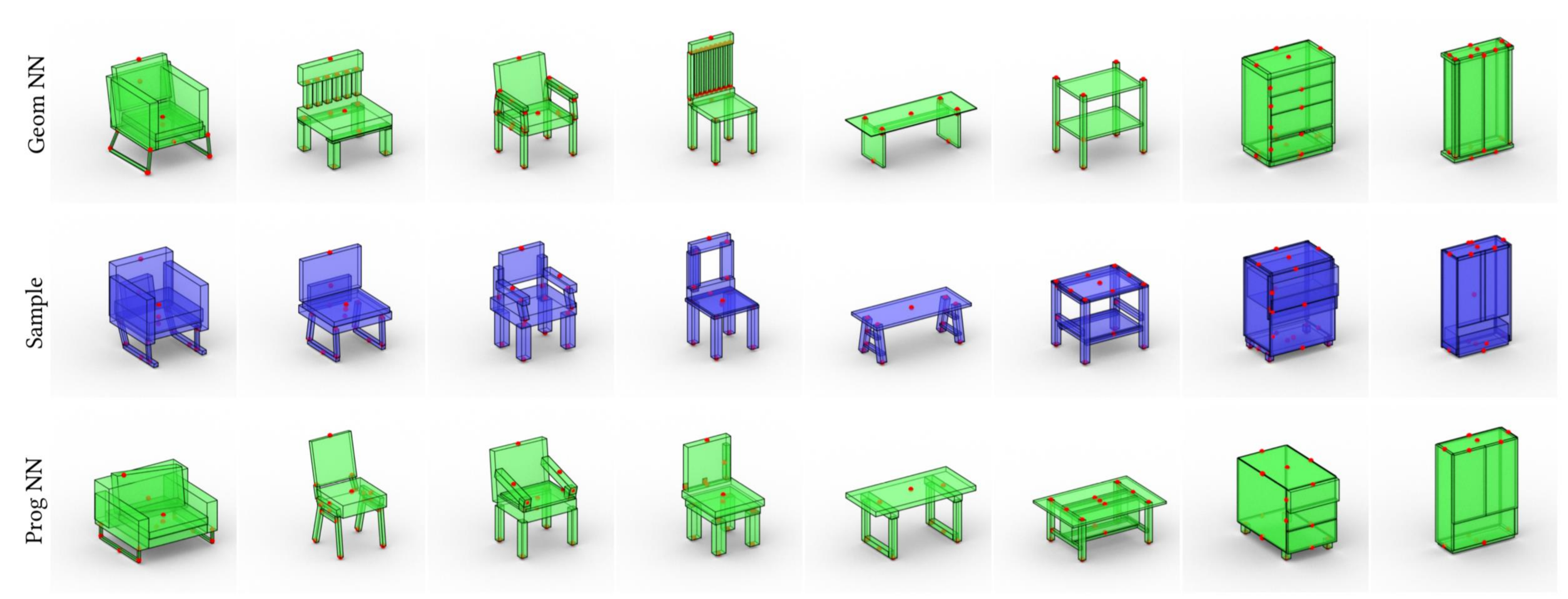
WHAT CAN YOU DO WITH IT?

Novel Shape Generation

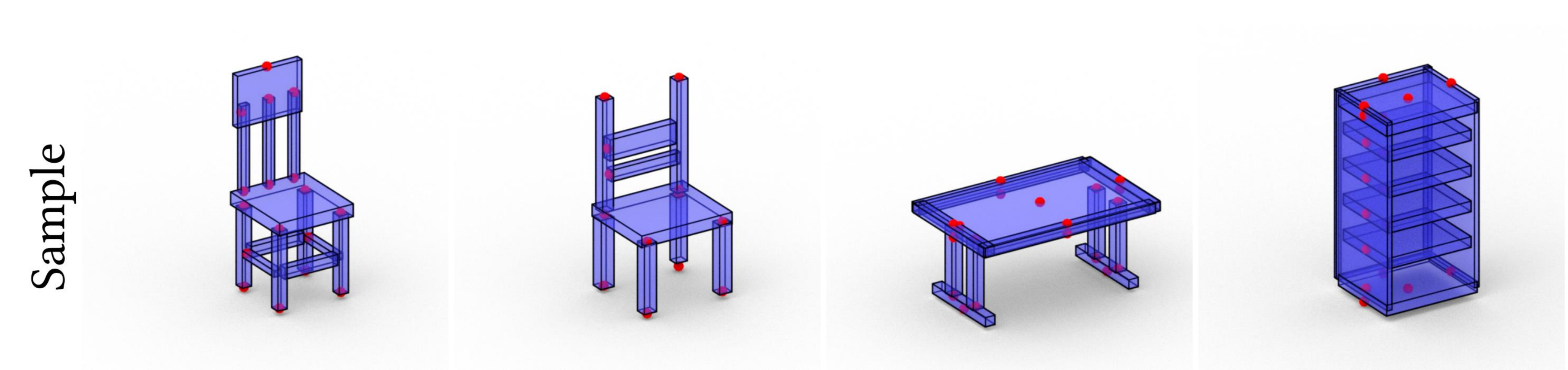


Novel Shape Generation

Novel Shape Generation



Editing Generated Programs



Editing Generated Programs

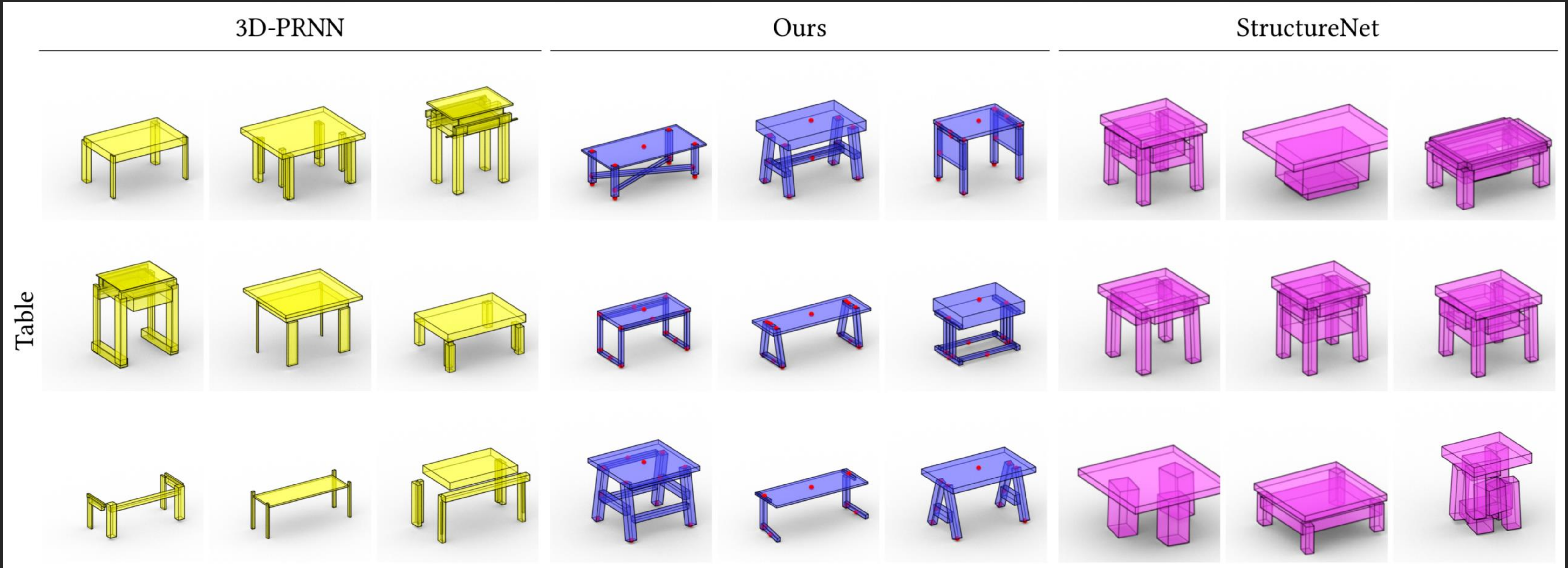
Comparison Conditions

- 3D PRNN:
  Sequence of boxes, but no hierarchy or relations

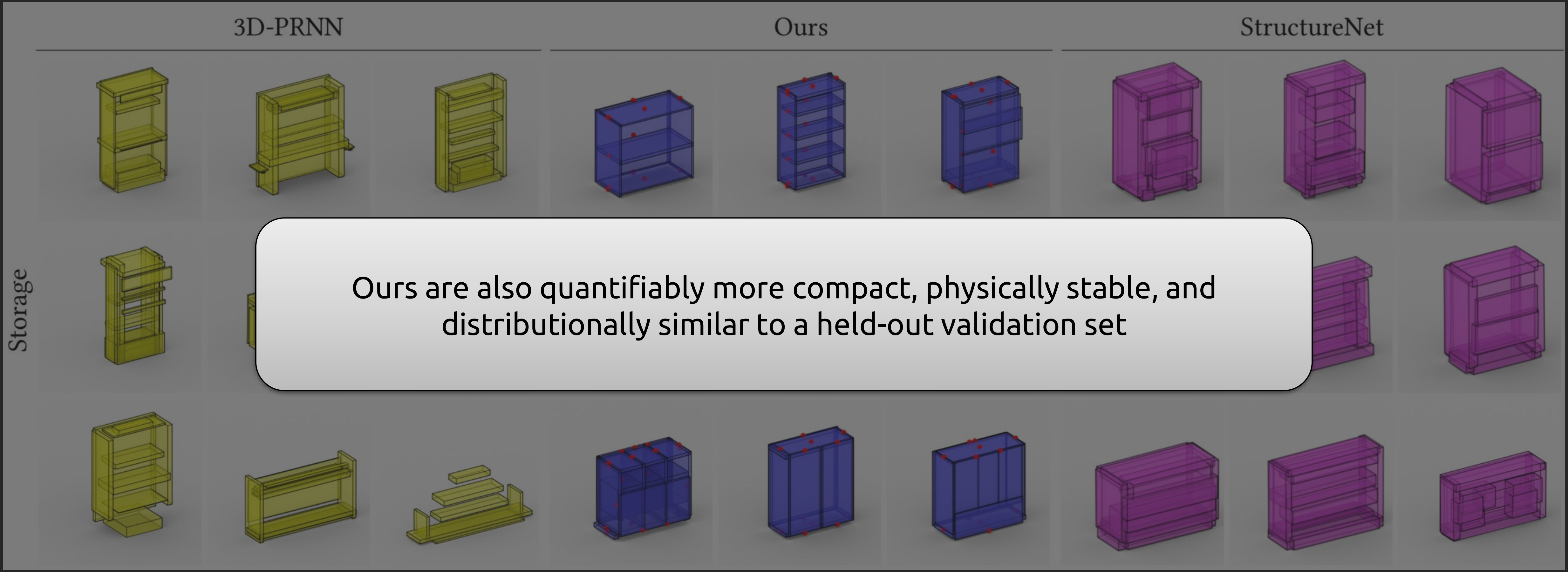
- StructureNet:
  Hierarchy of boxes w/ symmetry relations, but no explicit parametric attachments

Ours Generates Better Novel Shapes

Ours Generates Better Novel Shapes

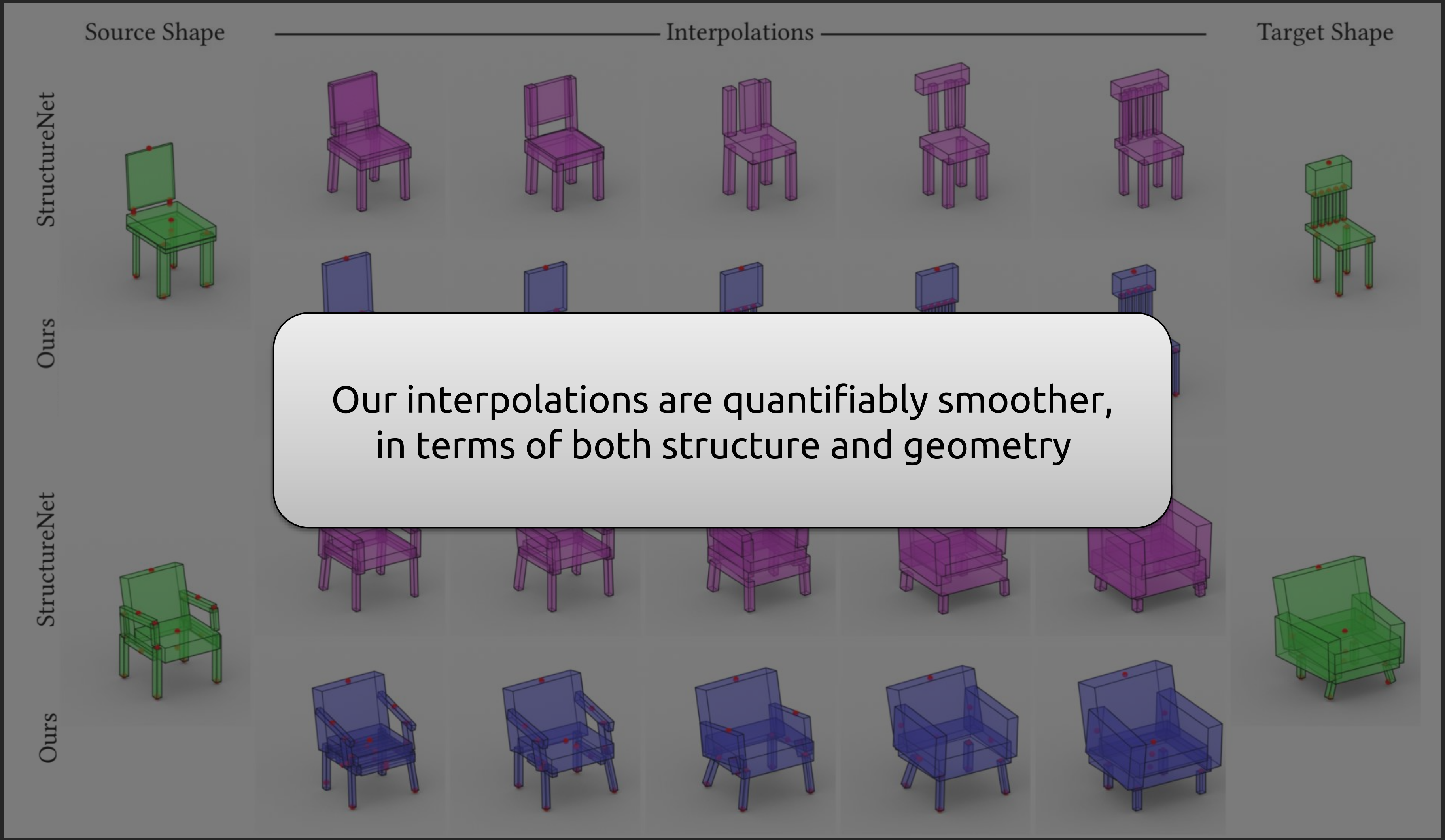


Ours Generates Better Novel Shapes



Ours are also quantifiably more compact, physically stable, and distributionally similar to a held-out validation set

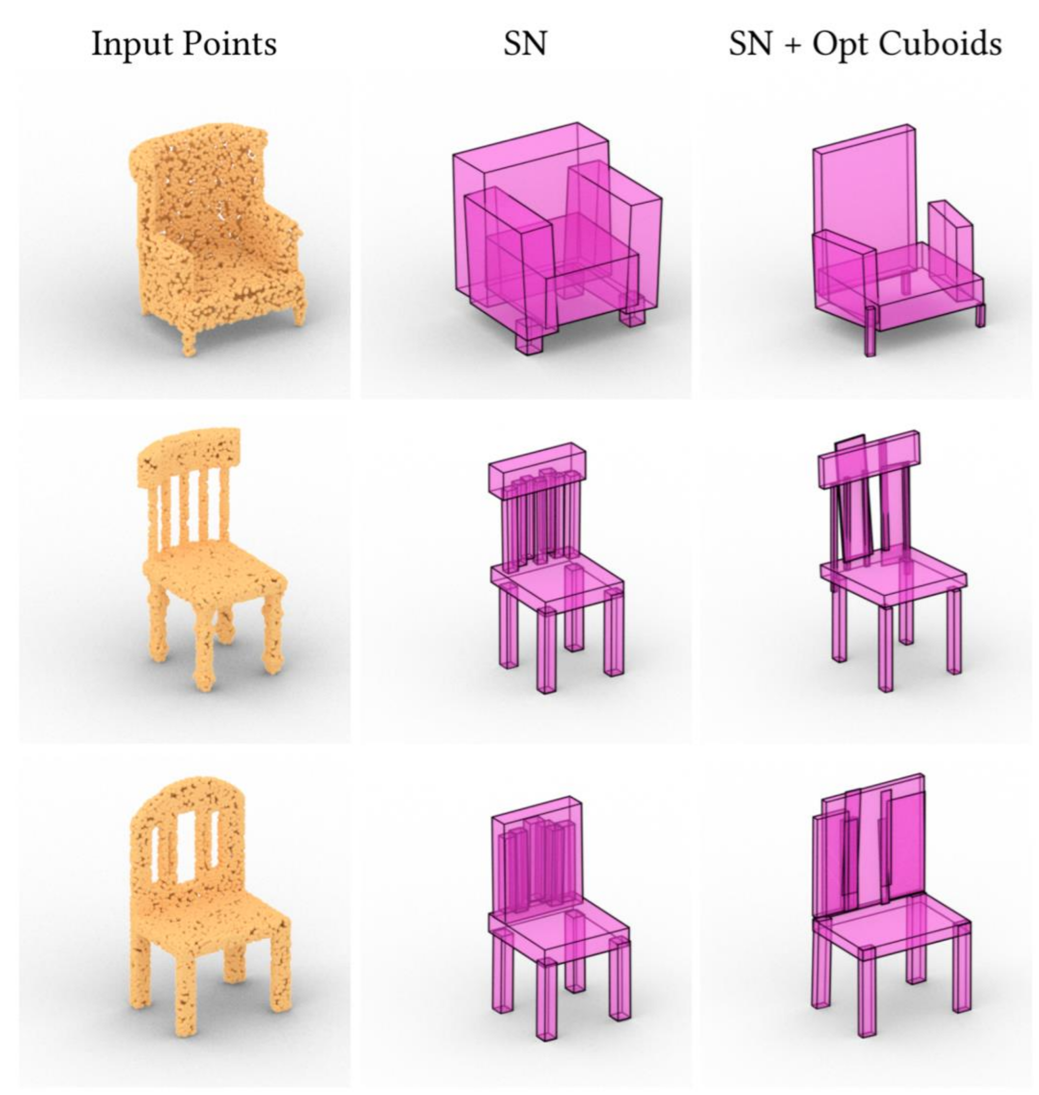
Ours Produces Better Interpolation



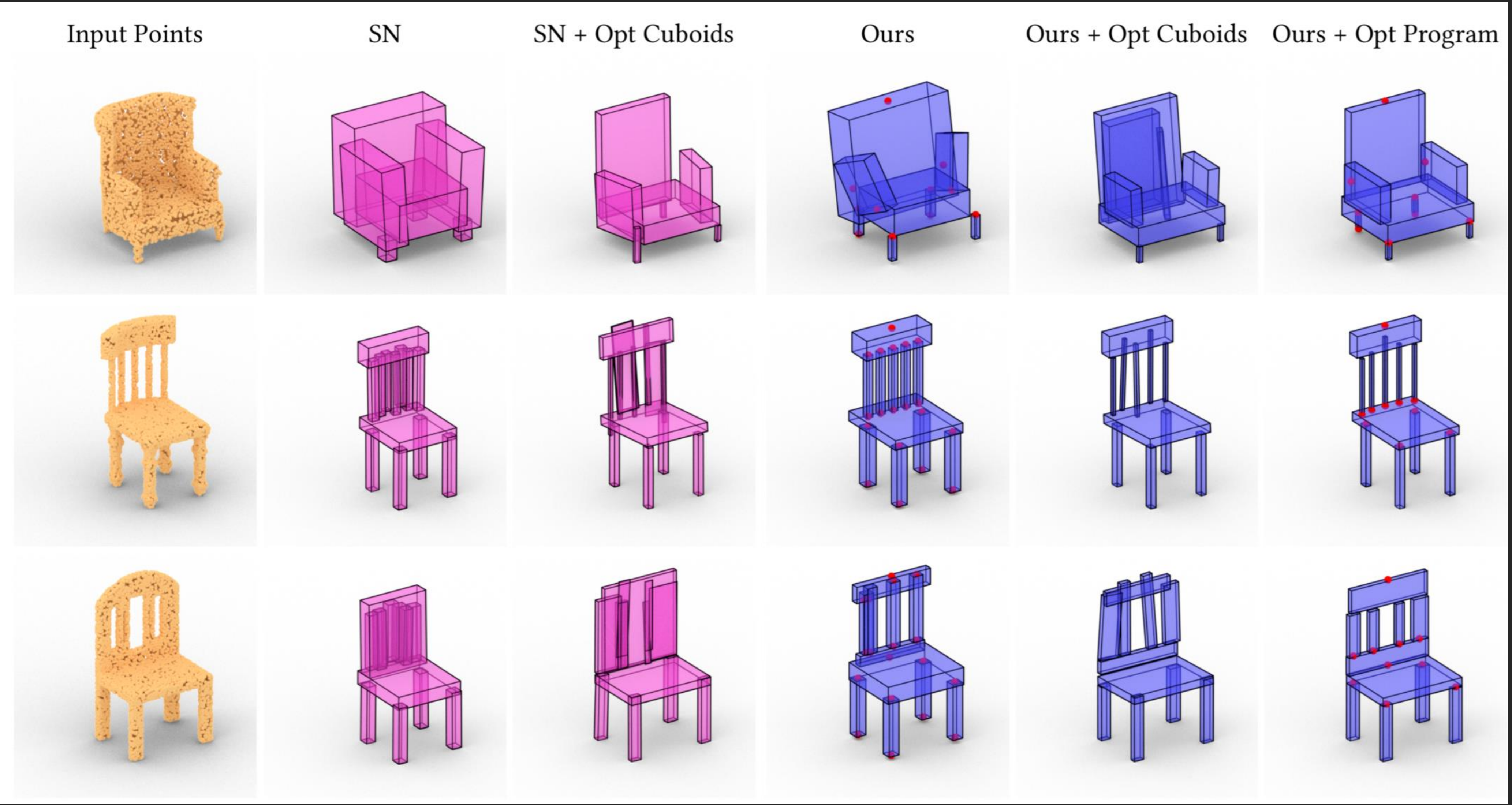
Our interpolations are quantifiably smoother, in terms of both structure and geometry

Point Cloud “Parsing”

Point Cloud “Parsing”

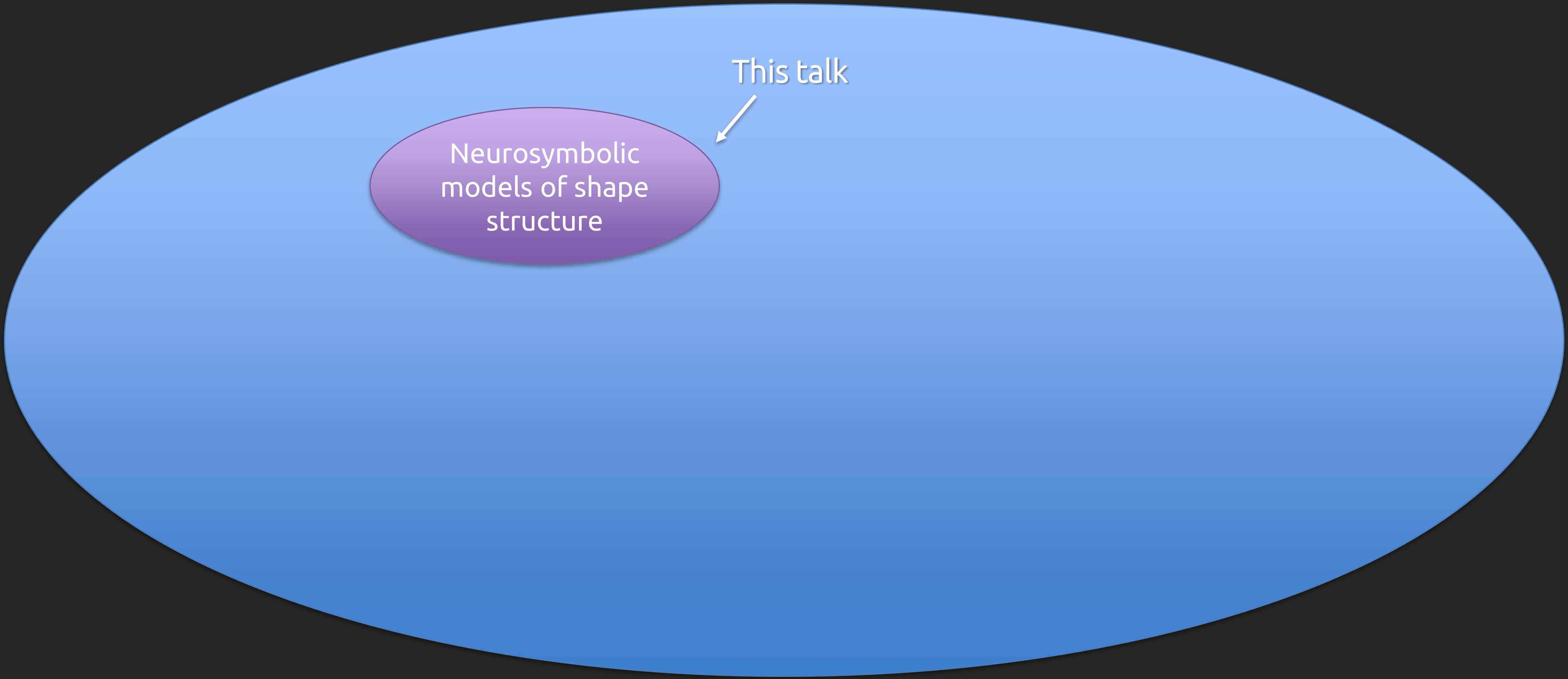


Point Cloud “Parsing”

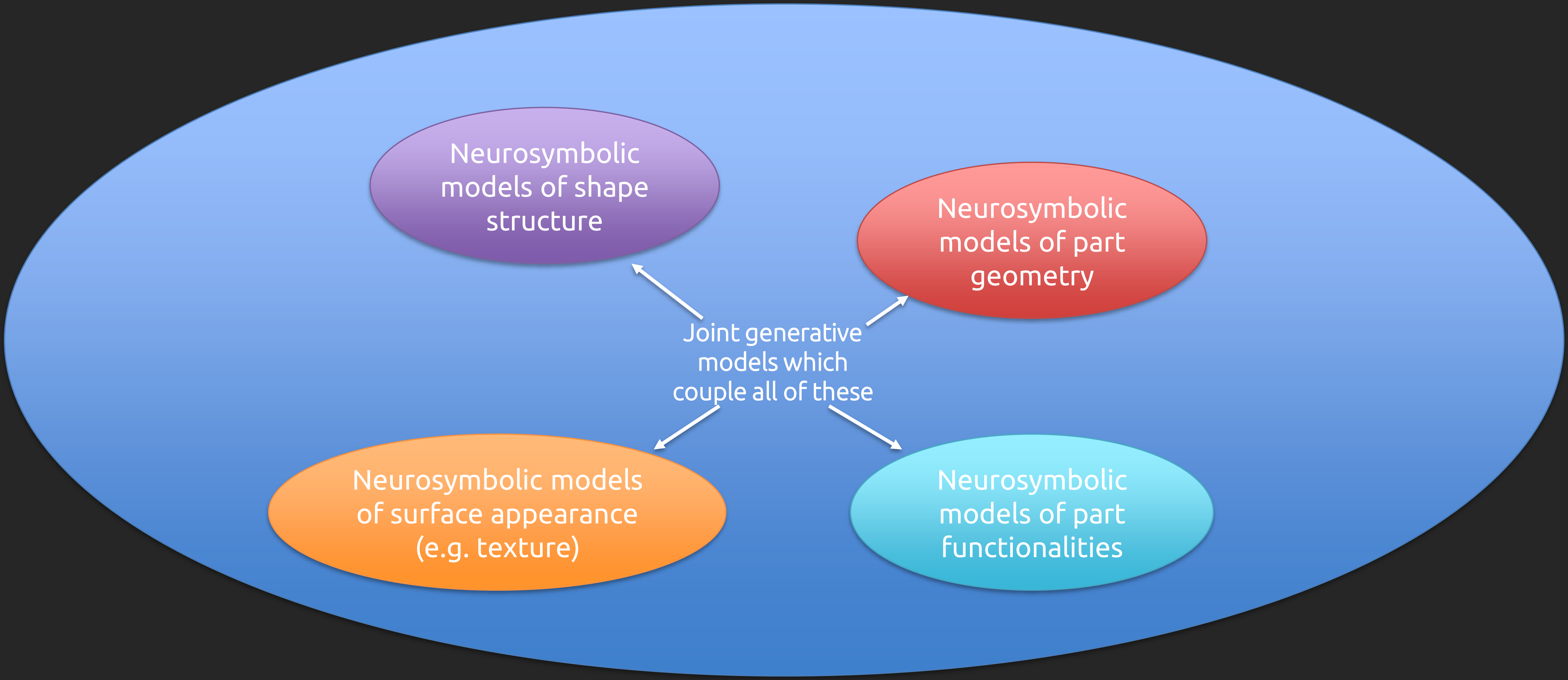


WHAT’S NEXT?

Neurosymbolic 3D Model Design Space



Neurosymbolic 3D Model Design Space



THANKS!