The Role of Generative AI in Shaping the Next Generation of the Metaverse

Mubbasir Kapadia, Roblox
Honglu Zhou, NEC Labs
Derek Liu, Roblox Research
Daniel Ritchie, Brown University
Kartik Ayyar, Roblox

Thursday July 27, 5:45 pm – 7:30 pm HST
Ballroom B

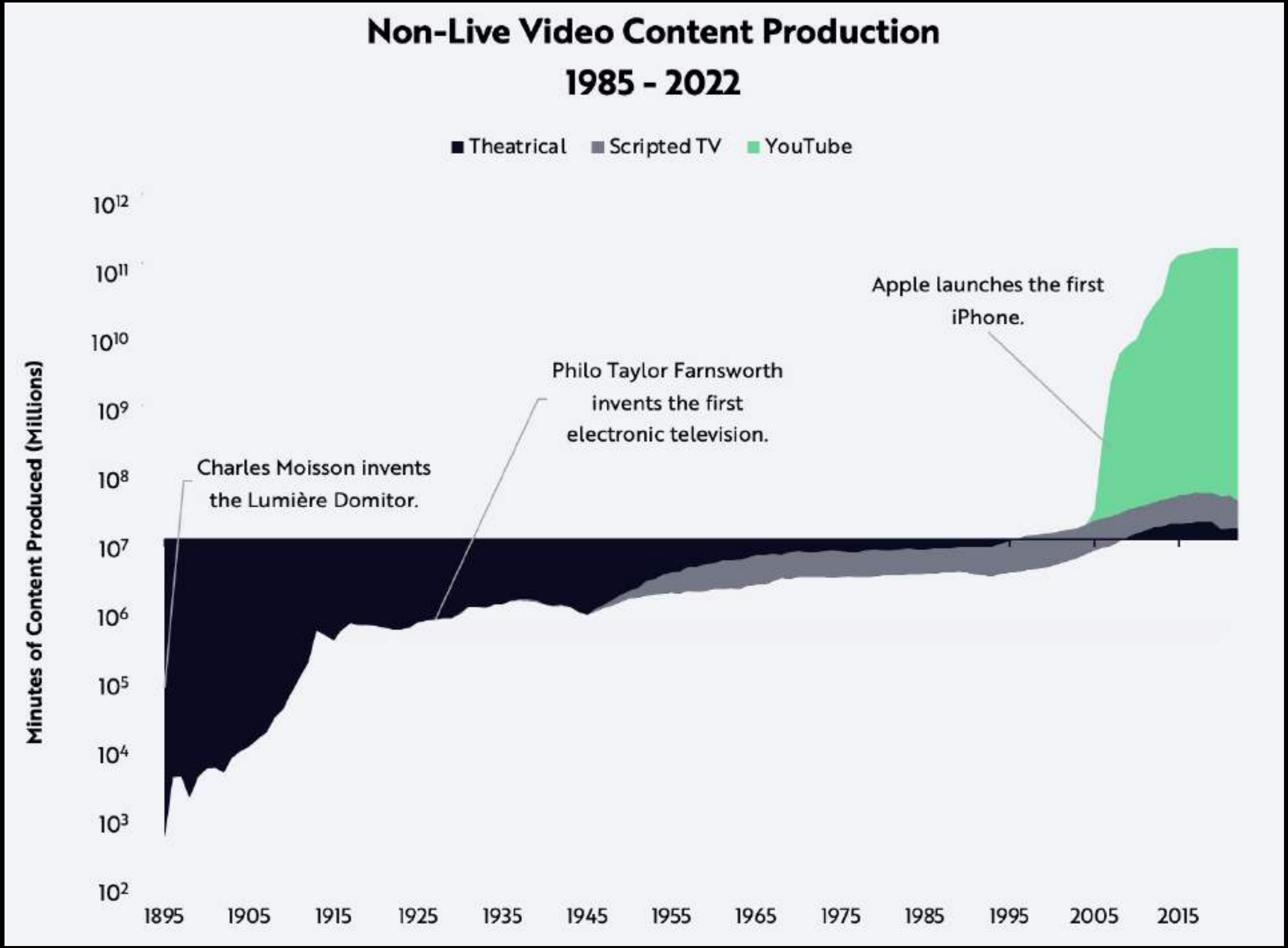
Link to social page

Let's talk about video games

- Largest entertainment market in the world (3.2B players, 180M USD annual spend)
- Video game creation restricted to game studios with 10’s/100’s of employees with expertise in programming, 3D content creation


What if all video gamers could become developers ?

Evolution of Content Creation and Consumption

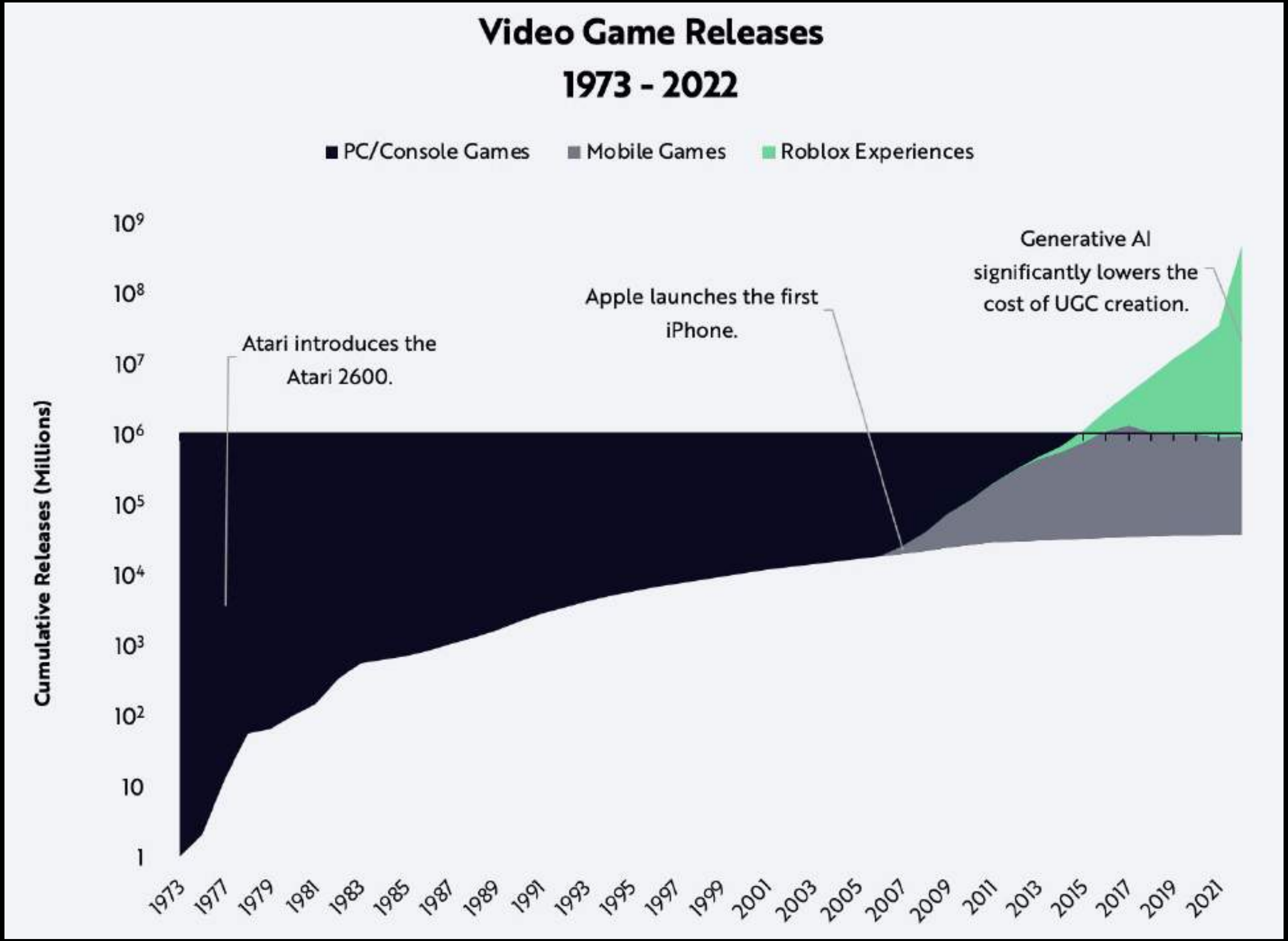


Television: Scripted TV surpassed theatrical releases (annual minutes)

iPhone: Youtube scales to 1B minutes of content by 2011

2022: Youtube content approaches ~15B minutes. 4000 times scripted TV + theatrical content

Meaningful cost declines in video content production democratize the creative process



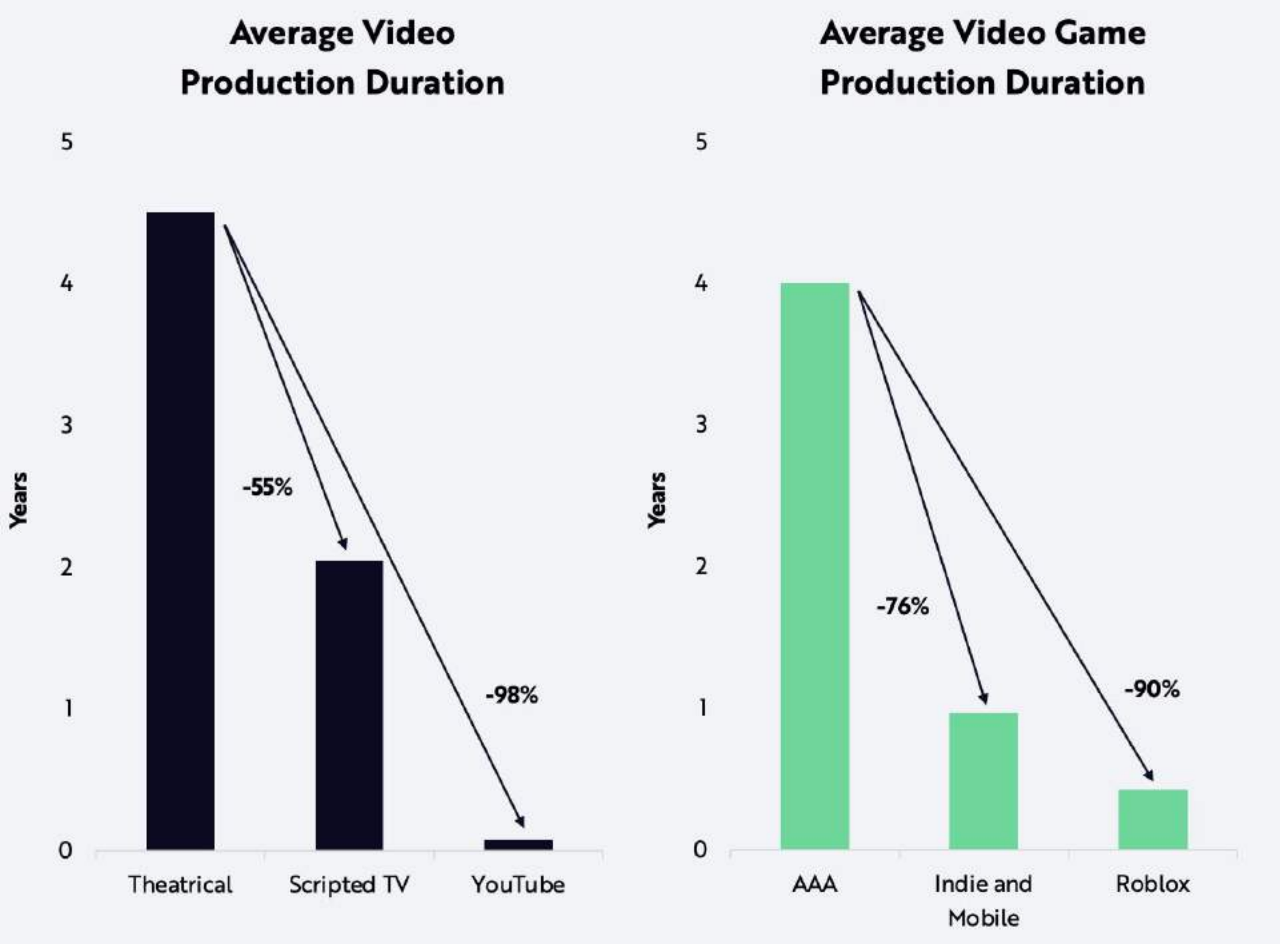
Similar trend in gaming

2009: Mobile games overtakes PC + console gaming

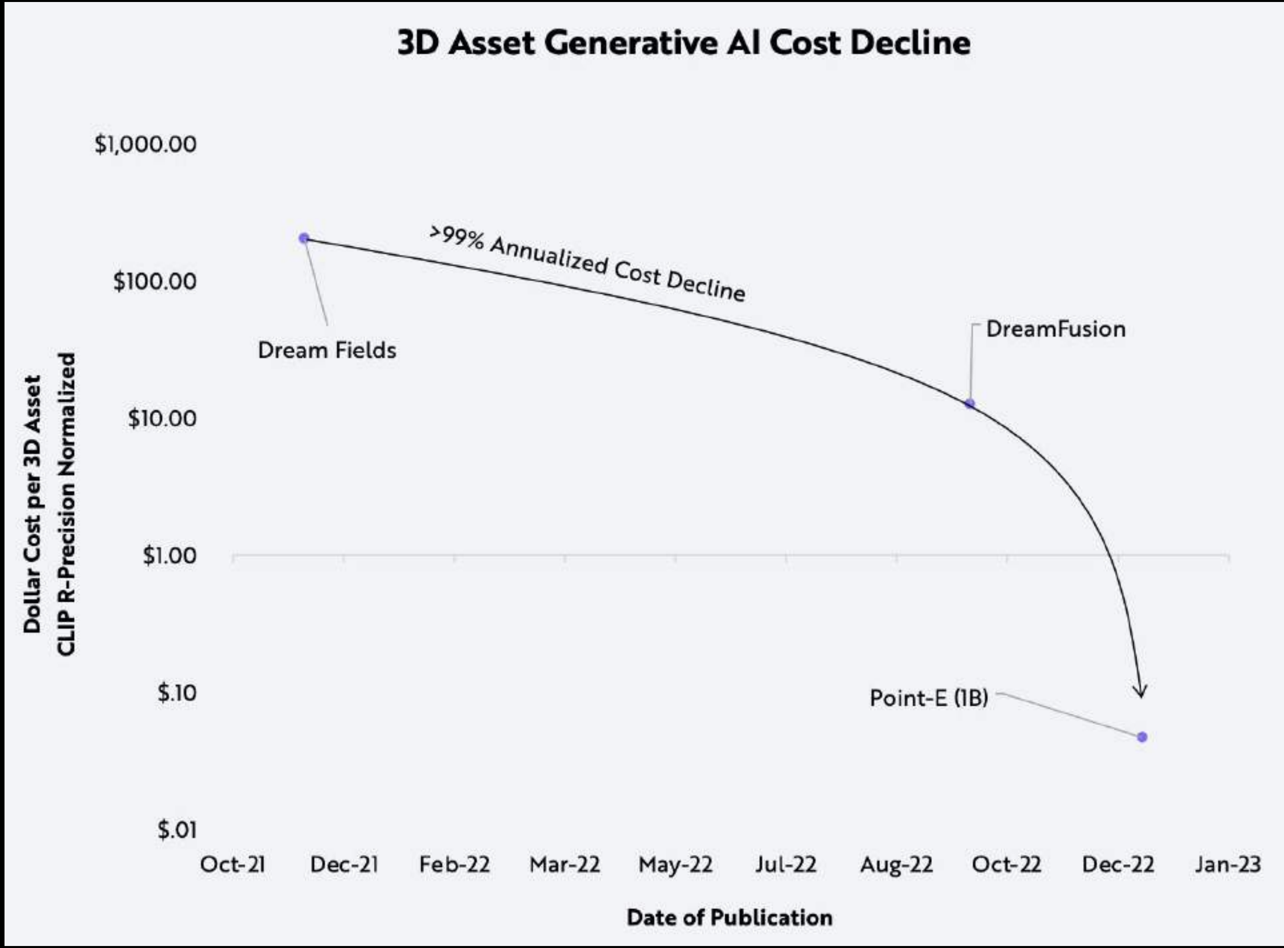
2017: Roblox studio overtakes PC, console, and mobile titles with 2.5M experiences

Today: Roblox offers ~470M experiences – 530 times more than PC, console, and mobile games

Advances in AI lowers the cost of UGC creation



Production cost collapse of video game creation commensurate with video creation



Dream Fields (2021): Reconstruct 3D models from NL by using NeRF to infer multi-view images in 3D space.

DreamFusion (2022): 3D asset generation without needing 3D training data

Cost to generate 3D asset 94% in 9 months

Point-E (2022): 3D asset generation in 1.5 mins (compared to 200 hrs for Dream Field, 12 hours for DreamFusion)

99% cost reduction (0.05$)

An inflection point for gaming

“Generative AI could be an important catalyst for video games … and generate 3D content much faster and cost effectively than existing approaches.”

- **Dan Sturman, CTO Roblox**


What if all video gamers could become developers ?


What if all video gamers could become developers ?
Every player can be a creator

Overview

- Recent trends in multimodal content generation, encompassing
- Application of neuro-symbolic representations for 3D Generative AI
- Geometric Learning on Discrete surfaces in 3D content creation
- Practical implementations of Generative AI within Roblox

Agenda

Duration	Presenter	Talk Title
15 mins	Honglu Zhou, NEC Labs	Illuminating the Metaverse: Unveiling NEC Labs' Journey in Revolutionizing AIGC with Compositionality
20 mins	Derek Liu, Roblox Research	Geometric Learning on Discrete Surface Meshes
30 mins	Daniel Ritchie, Brown University	Neuro-symbolic Methods for 3D Generative AI
30 mins	Kartik Ayyar	Generative AI in Action at Roblox

\Orchestrating a brighter world NEC

Illuminating the Metaverse: Unveiling NEC Labs' Journey in Revolutionizing AIGC with Compositionality

Presenter: Honglu Zhou
Affiliation: NEC Laboratories America, Inc. (NEC Labs)
July 26, 2023

Current AIGC: flexible, accessible, and stunning

They lack crucial capabilities!

Compositionality

StyleT2I, LCG (NEC Labs)

glasses
smile
55 y/o

Video Generation

LFDM (NEC Labs)

jogging

3D Content Generation

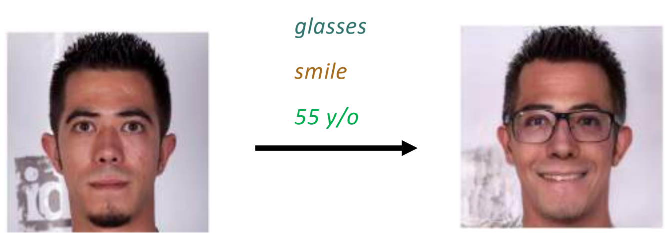
Relightify (Papantoniou, Foivos Paraperas, et al. 2023)

3D avatar

StyleT2I: Toward Compositional and High-Fidelity Text-to-Image Synthesis

Zhiheng Li[1,2] Martin Renqiang Min[1] Kai Li[1] Chenliang Xu[2]
[1]NEC Laboratories America [2]University of Rochester

CVPR 2022



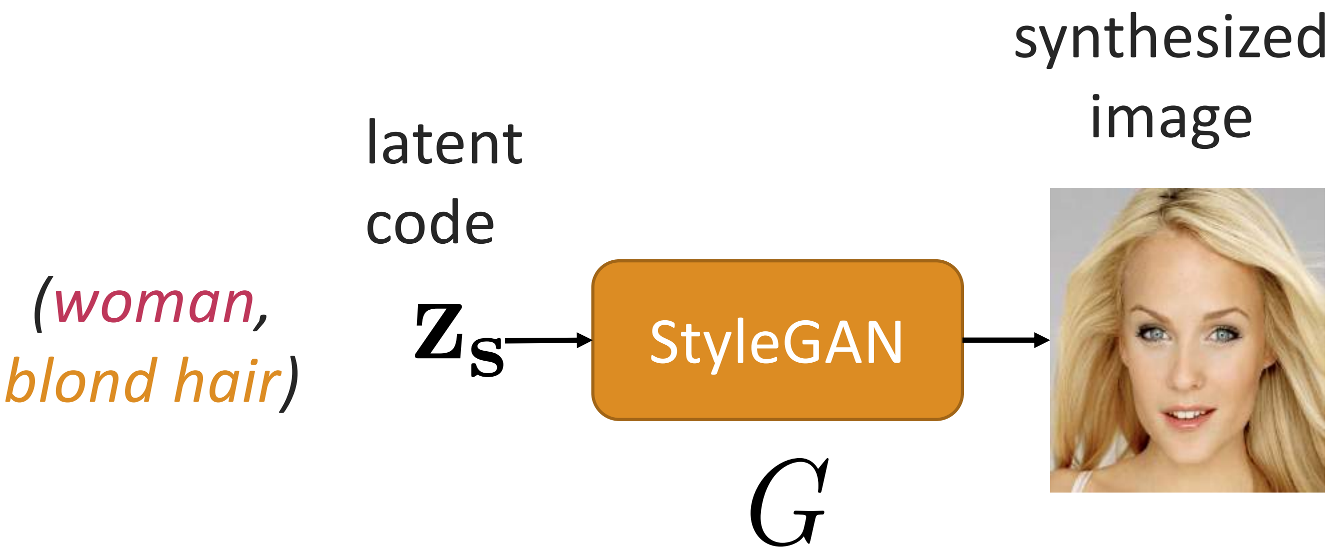
Lacking compositionality could have severe implications

		ControlGAN	DAE-GAN	TediGAN	**StyleT2I (Ours)**
Text Input: "*He* is *wearing lipstick*."					
Attribute Composition	*he*	✓	✓	✗	✓
	wearing lipstick	✗	✗	✓	✓
High Fidelity		✗	✗	✓	✓

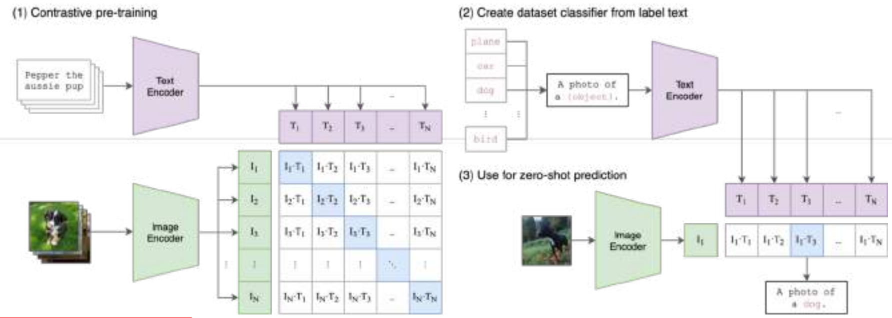
\Orchestrating a brighter world NEC

Hypothesis

There exists a latent direction that corresponds to the composition of multiple attributes in StyleGAN's latent space.



Overview of StyleT2I



$$\mathcal{L}_{\text{norm}} = \max(||\mathbf{s}||_2 - \theta, 0)$$

Norm Penalty

$\mathcal{L}_{\text{norm}}$

random latent code $\mathbf{z}$

The woman has blond hair.

text

$\mathbf{t}$

Text-to-Direction

$\mathbf{s}$ sentence direction

if inference?

no

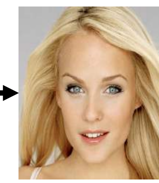
$\mathbf{z} + \mathbf{s}$

text-conditioned code $\mathbf{z_s}$

StyleGAN

G

synthesized image



$\mathcal{L}_{\text{contras}}$

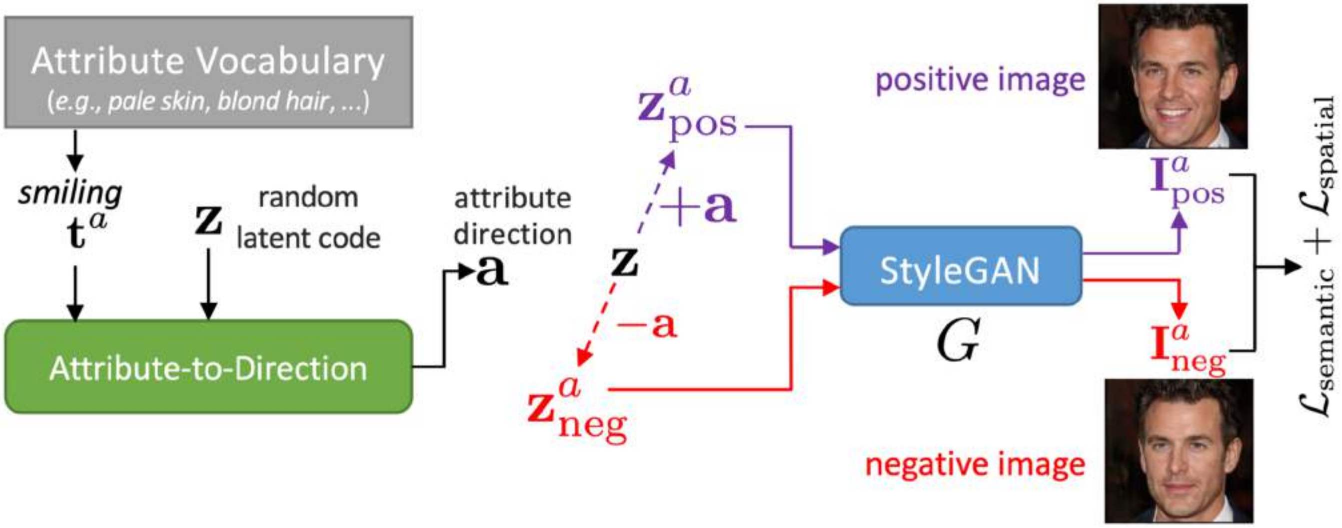
CLIP-guided Contrastive Loss

Semantically Aligned

$$\mathcal{L}_{\text{contras}}(\mathbf{I}_i) = -\log \frac{\exp(\cos(E^{\text{img}}_{\text{CLIP}}(\hat{\mathbf{I}}_i), E^{\text{text}}_{\text{CLIP}}(\mathbf{t}_i)))}{\sum_{j \neq i}^{B} \exp(\cos(E^{\text{img}}_{\text{CLIP}}(\hat{\mathbf{I}}_i), E^{\text{text}}_{\text{CLIP}}(\mathbf{t}_j)))}$$

Disentangled attribute representations

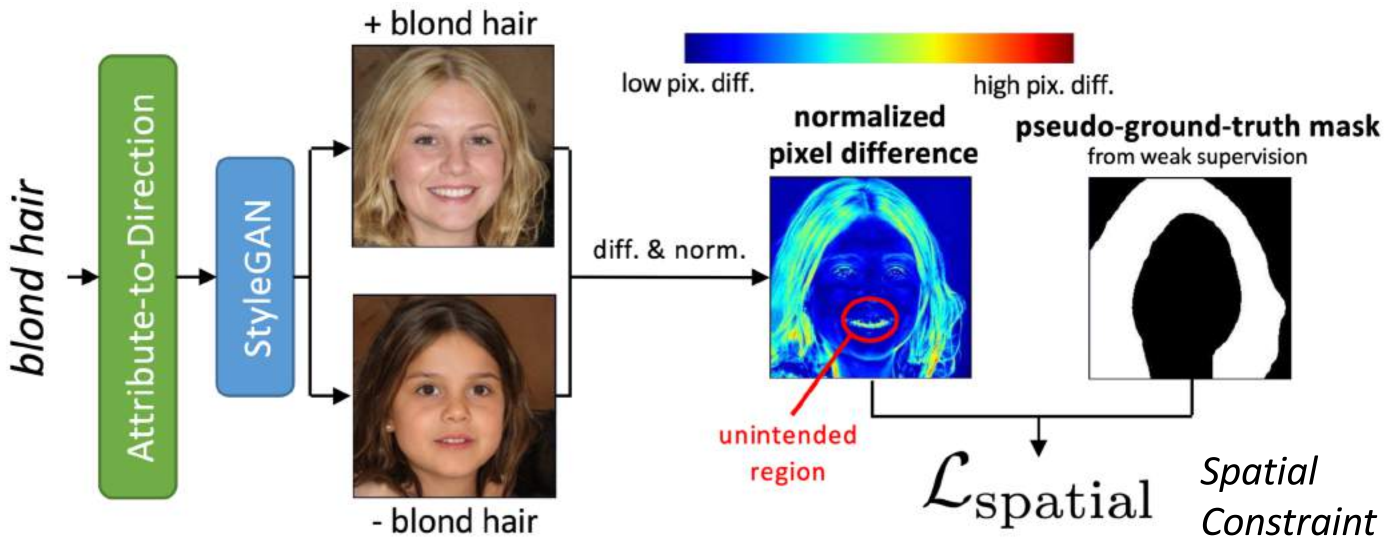
The compositional text-to-image model needs to be sensitive to each independent attribute described in the text.



$$\mathcal{L}_{\text{semantic}} = \max(\cos(E^{\text{img}}_{\text{CLIP}}(\mathbf{I}^a_{\text{neg}}), E^{\text{text}}_{\text{CLIP}}(\mathbf{t}^a)) - \cos(E^{\text{img}}_{\text{CLIP}}(\mathbf{I}^a_{\text{pos}}), E^{\text{text}}_{\text{CLIP}}(\mathbf{t}^a)) + \alpha, 0)$$

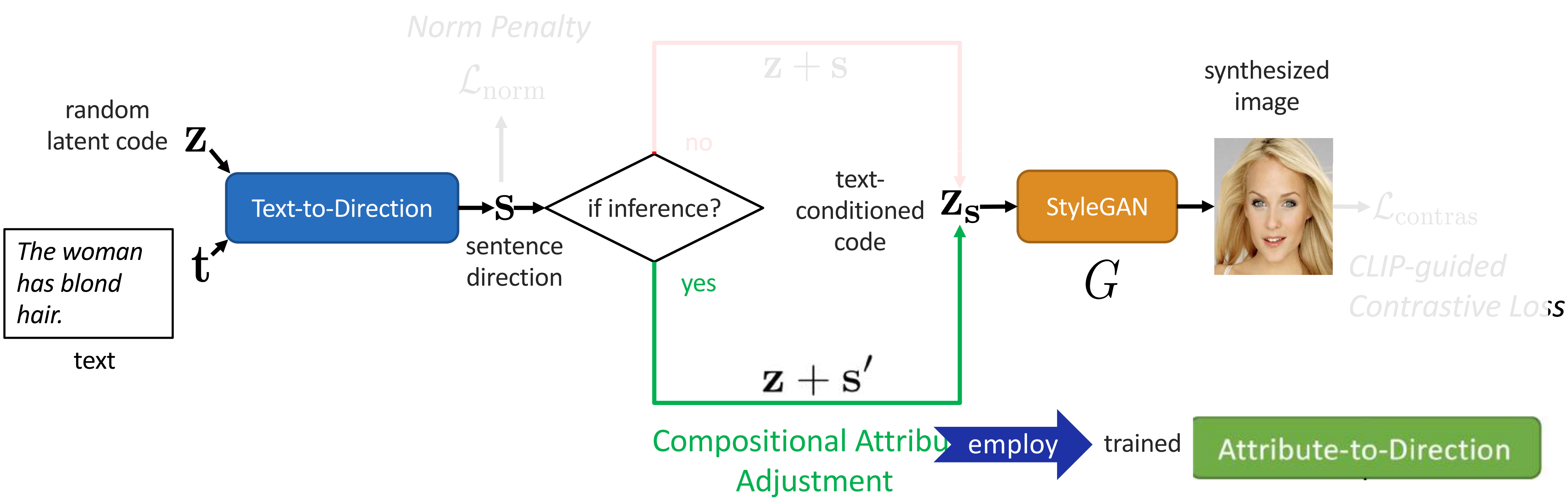
Disentangled attribute representations

The compositional text-to-image model needs to be sensitive to each independent attribute described in the text.



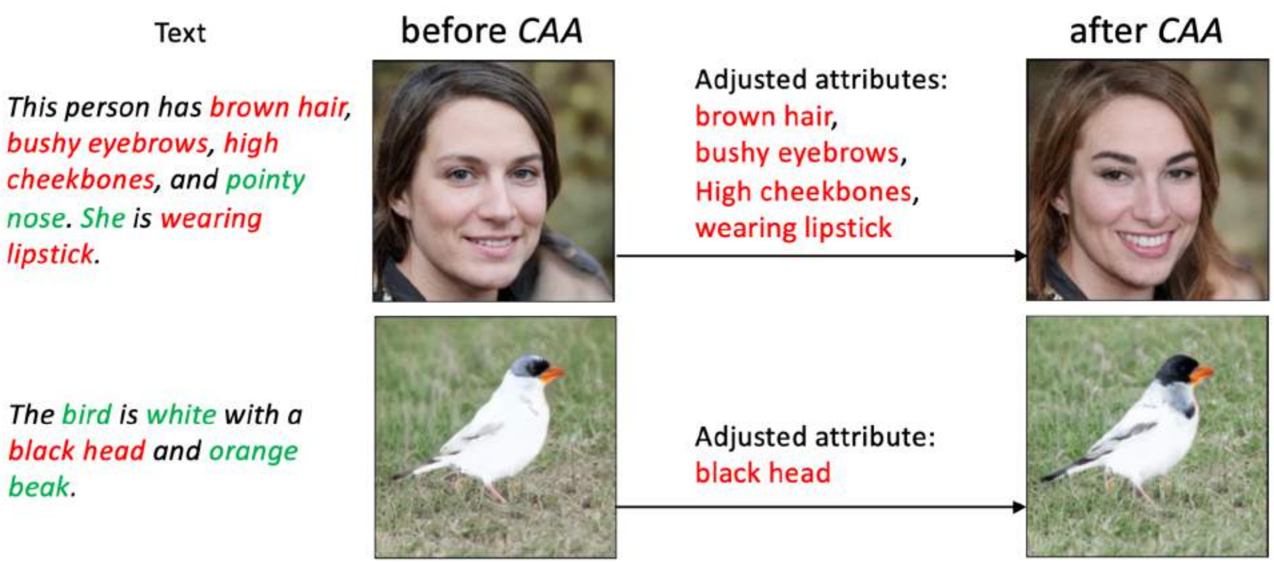
Spatial Constraint Loss

Overview of StyleT2I



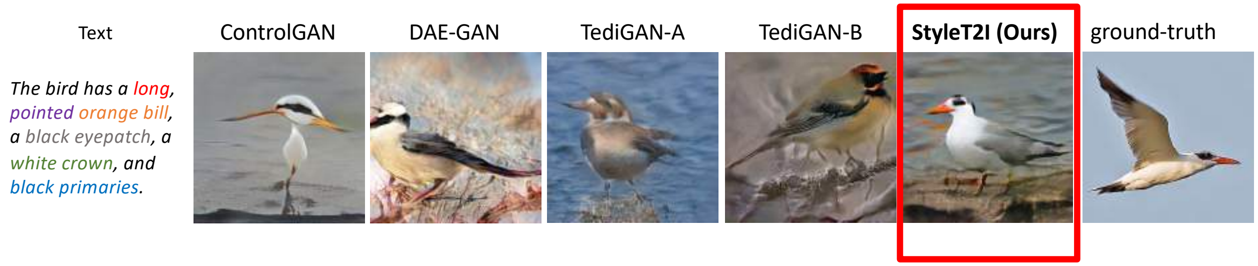
Adjust wrongly predicted attributes at inference time

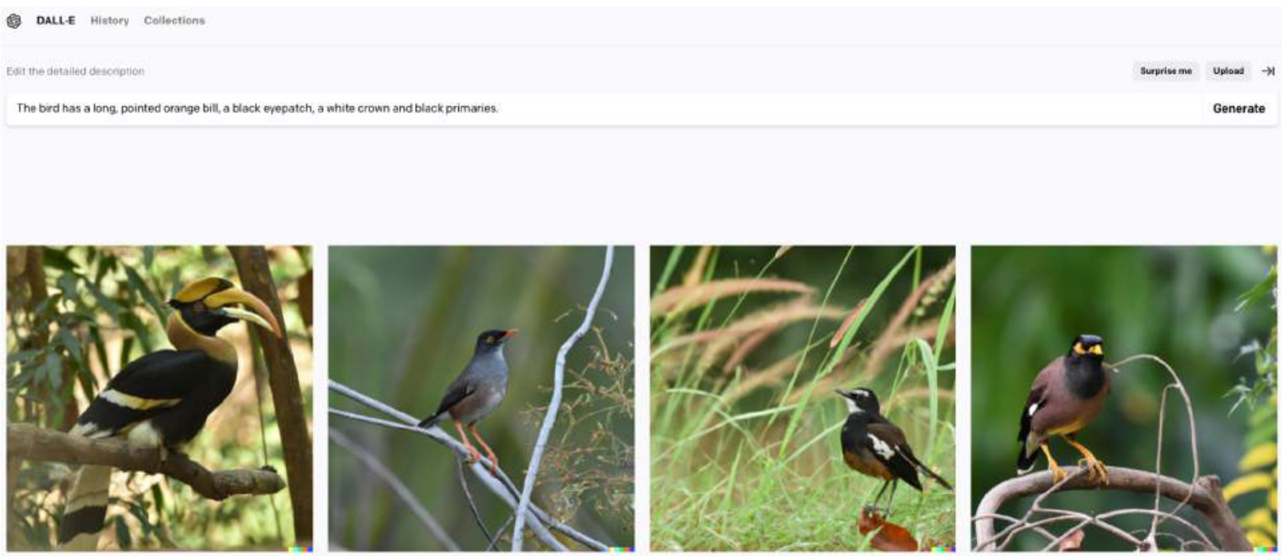
Compositional Attribute Adjustment (CAA): The attribute directions (from Attribute-to-Direction) can be used to adjust the sentence direction (from Text-to-Direction) .



$$\mathbf{A} = \{\mathbf{a}_i \mid \cos(\mathbf{a}_i, \mathbf{s}) \leq 0\}, \quad \mathbf{s}' = \mathbf{s} + \sum_{\mathbf{a}_i \in \mathbf{A}} \frac{\mathbf{a}_i}{||\mathbf{a}_i||_2}$$

Qualitative Results



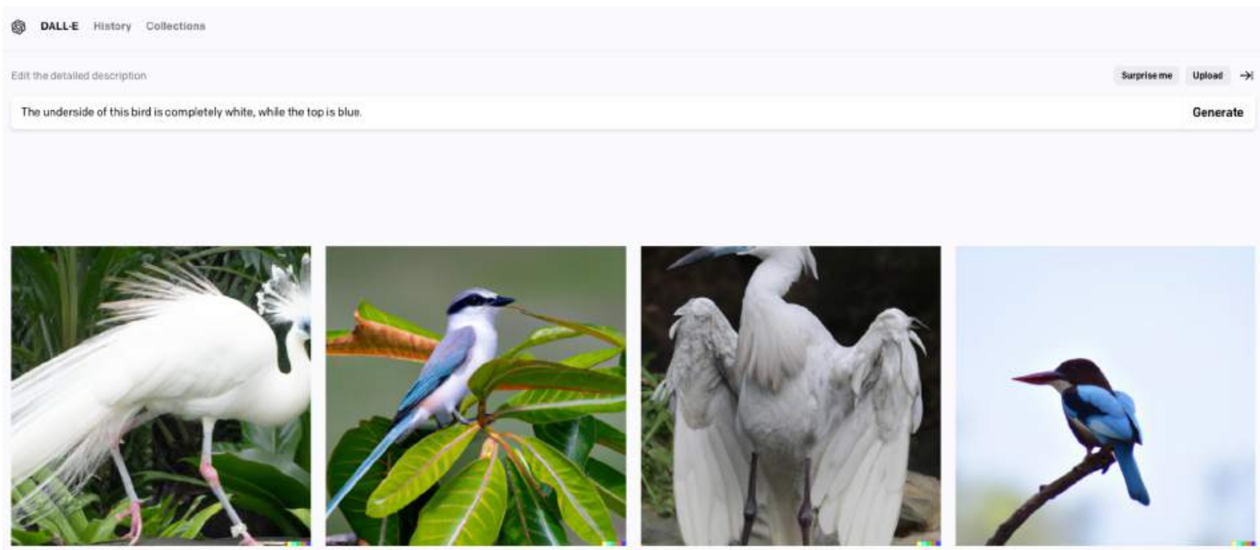


Stable Diffusion

Qualitative Results

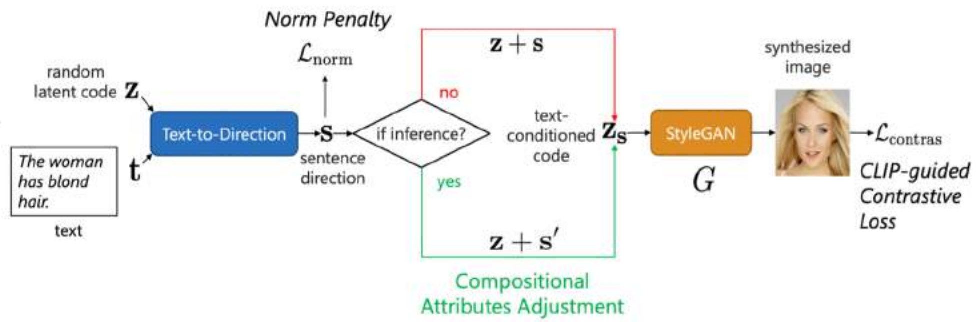
Text

The underside of this bird is completely white, while the top is blue.



Stable Diffusion

Afterthoughts of StyleT2I



Limitations:
- Closed attribute vocabulary
- Fine-tune CLIP might be necessary
- The Spatial Constraint is not helpful to disentangle a few attributes that share the same spatial region, e.g., “bushy eyebrow” and “arched eyebrow”

Lessons learned:
- Training a module to better navigate a pre-trained generator’s latent space
- Pre-trained vision-language foundation models such as CLIP can be helpful for AIGC to align user’s intent with generated content
- Aligning the global (sentence) representation with fine-grained local (attribute) representation can improve quality and compositionality
- Test-time adaptation methods such as Compositional Attribute Adjustment can be super useful

Future work:
Complex scene images synthesis for disentangling different objects and backgrounds

Exploring Compositional Visual Generation with Latent Classifier Guidance

Changhao Shi[1] Haomiao Ni[2] Kai Li[4] Shaobo Han[4] Mingfu Liang[3] Martin Renqiang Min[4]

[1]University of California, San Diego

[2]The Pennsylvania State University

[3]Northwestern University

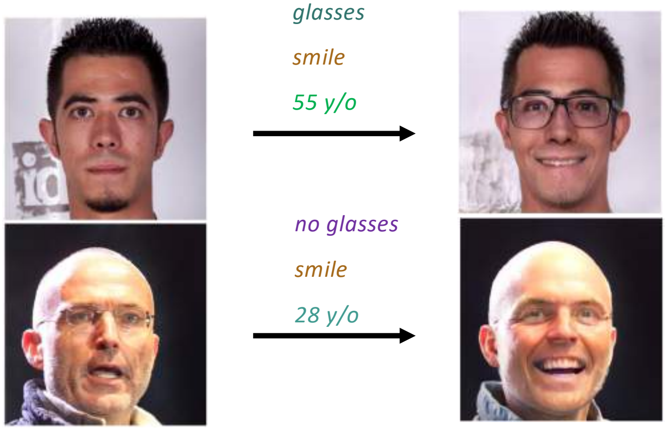
[4]NEC Laboratories America

CVPR 2023 Workshop

Key idea & Results of LCG (Latent Classifier Guidance)

For compositional image manipulation, the conditional ELBO of DDPM (De-noising Diffusion Probabilistic Models) is given by:

$$\mathbb{E}_{q(z_{1:T}|x_0)}\left[\sum_{t=1}^{T}\left[\sum_{i=1}^{n}\log p(y^i|z_{t-1}) + \log p(\hat{z}|z_{t-1})\right]\right] + \mathcal{L}_{uncond} + C$$



3 attributes: smiling, young, wavy hair.
The middle figure is from unconditional generation.
The + direction(i.e., apply attributes positively) is towards the right.

Afterthoughts of LCG (Latent Classifier Guidance)

$$\mathbb{E}_{q(z_{1:T}|x_0)}\left[\sum_{t=1}^{T}\left[\sum_{i=1}^{n}\log p(y^i|z_{t-1}) + \log p(\hat{z}|z_{t-1})\right]\right] + \mathcal{L}_{uncond} + C$$

Limitations:

- Closed attribute vocabulary
- The diffusion model always pulls the sample toward high density region. As a result, keeping images realistic is at the cost of losing identity preservation

Lessons learned:

- Training a latent diffusion model with auxiliary latent classifier guidance can facilitate non-linear manipulations of the latent space of a pre-trained generator for finer control of compositional content generation

Future work:

- Performance of latent classifier guidance in out-of-distribution settings
- Generating unseen classes and unseen sub-concept of an existing class

Conditional Image-to-Video Generation with Latent Flow Diffusion Models

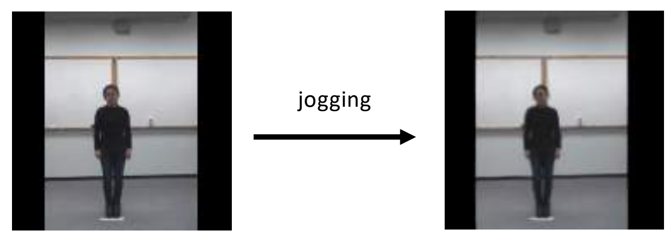
Haomiao Ni[1] Changhao Shi[2] Kai Li[3] Sharon X. Huang[1] Martin Renqiang Min[3]

[1]The Pennsylvania State University

[2]University of California, San Diego

[3]NEC Laboratories America

CVPR 2023



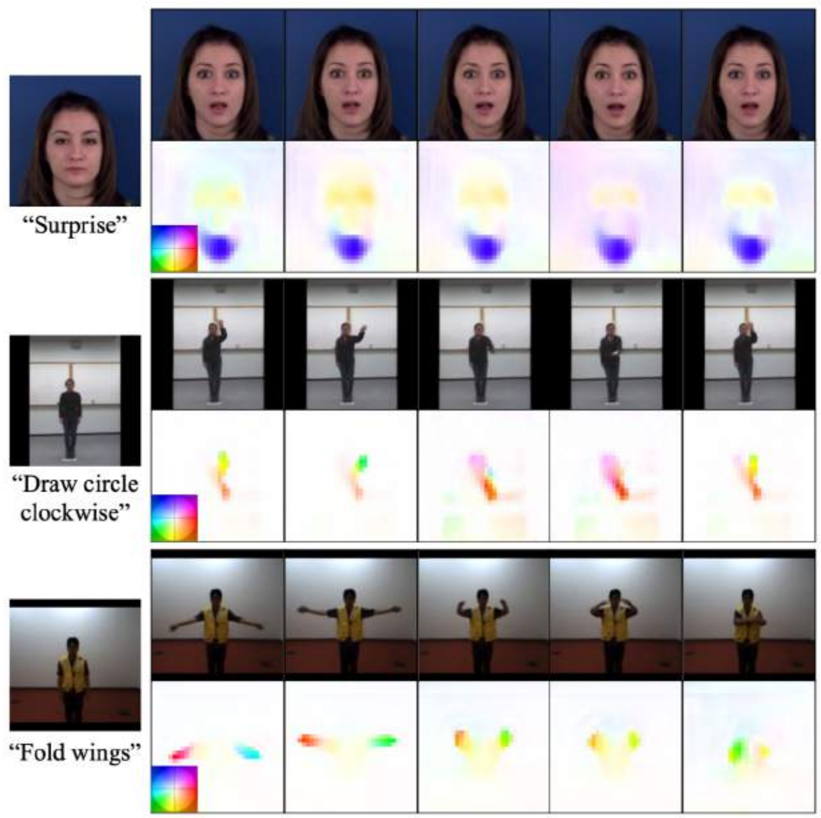
Text-conditioned image-to-video generation

jogging

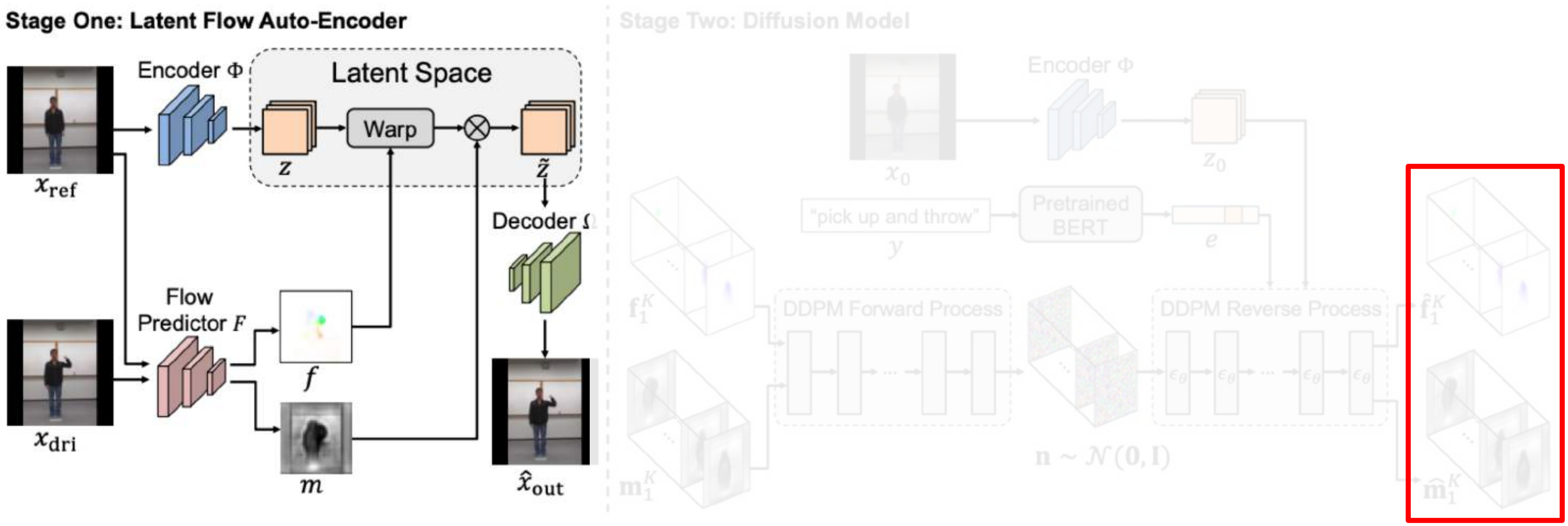
Synthesizing an optical flow sequence !

Time

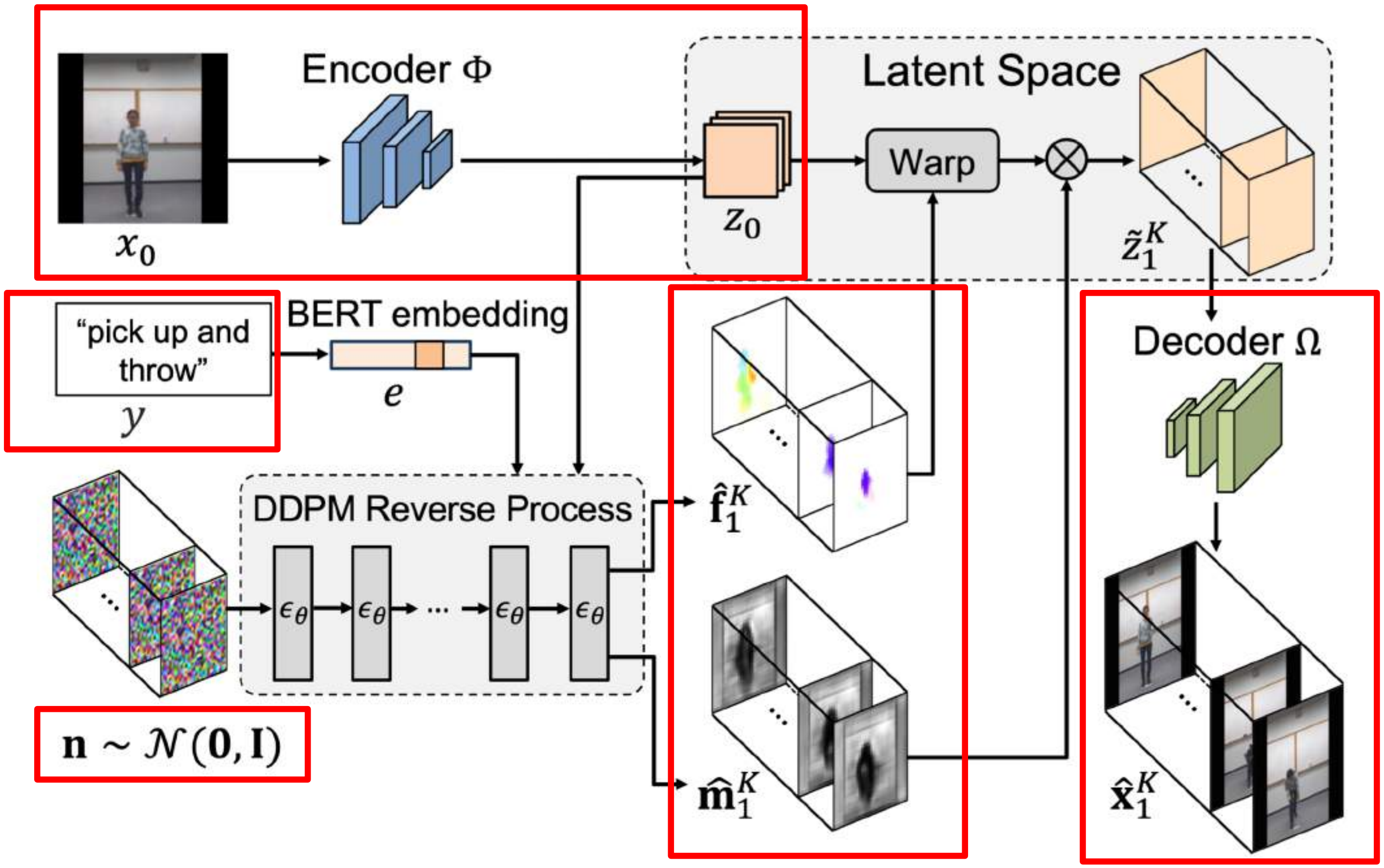
Source Image
&
Text Condition



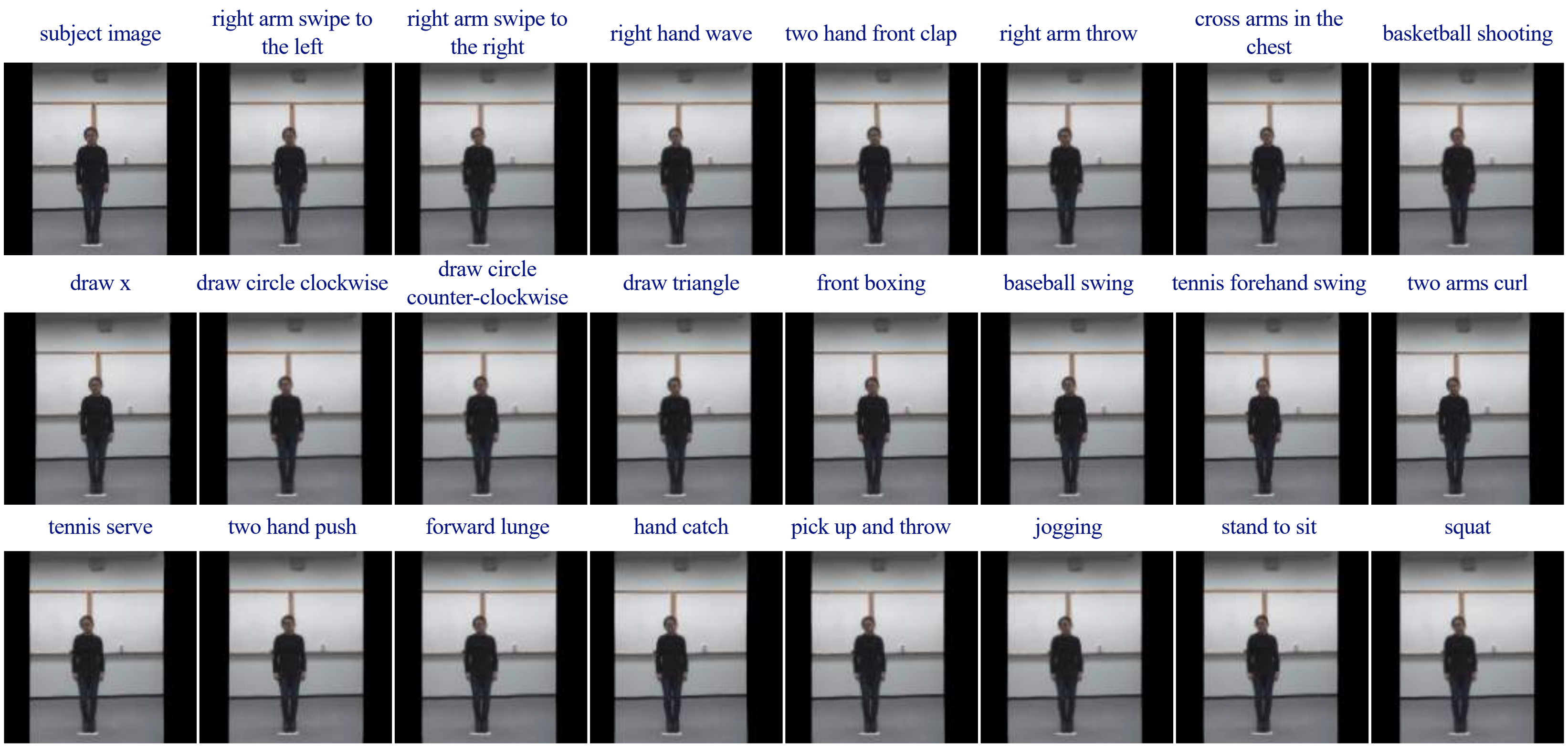
Training overview of LFDM



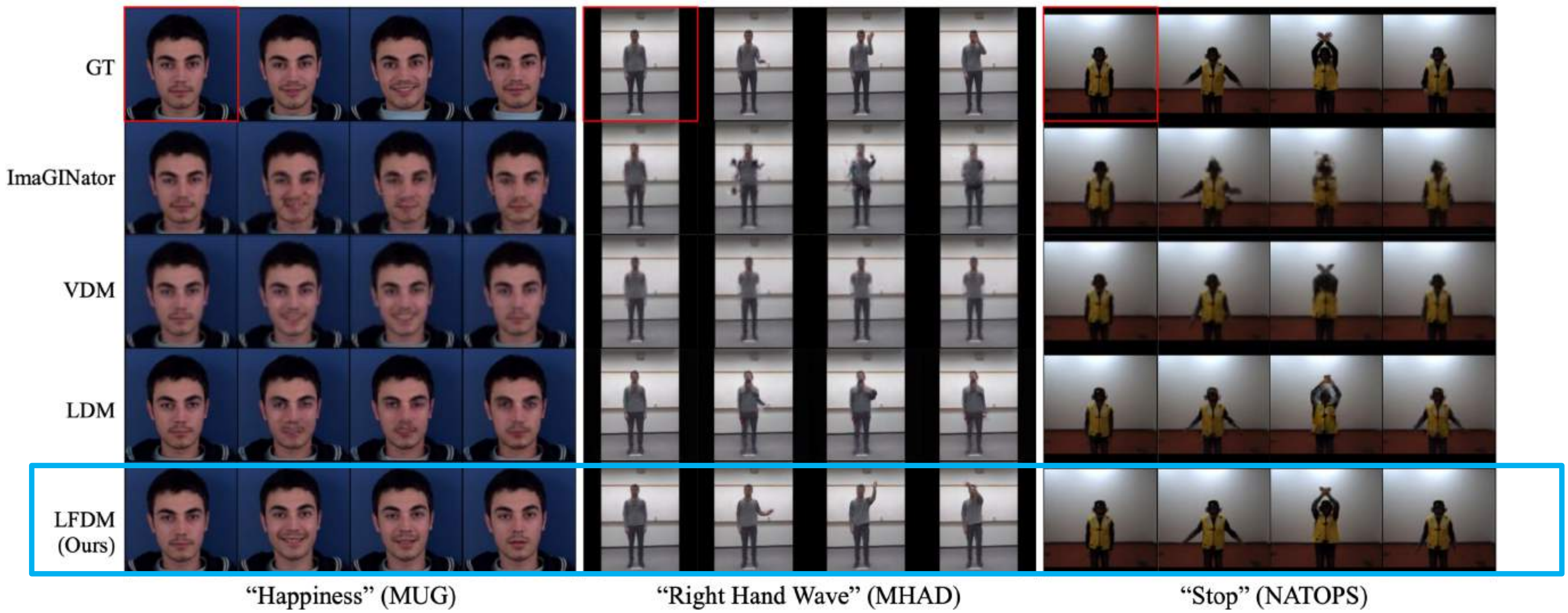
Inference overview of LFDM



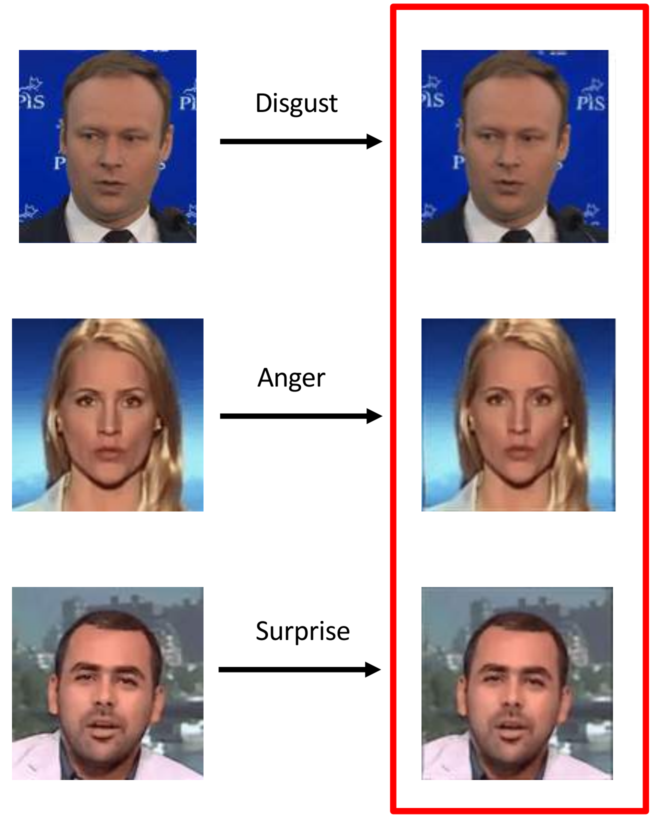
Qualitative Results



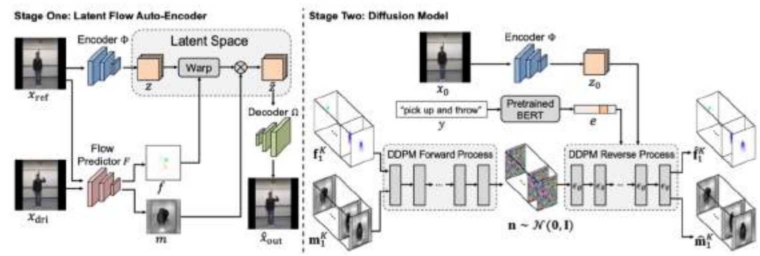
Qualitative Results



Qualitative Results



Afterthoughts of LFDM (Latent Flow Diffusion Model)



Limitations:
- Conditioned on the class labels instead of natural text descriptions
- Generation of a multi-subject flow sequence
- Generation of long videos
- 1000-step DDPM at inference is slow compared to GAN models, and thus frame resolution is hard to scale up

Lessons learned:
- Warp-based design can be more robust for generation of action/motion sequence
- Two-stage disentangled framework allows flexibility; potentially one can fine-tune the latent-to-pixel decoder on new target datasets for better spatial content generation quality without the need to retrain the whole framework including the latent flow diffusion model
- Diffusion models operating on the latent flow space, which is much more concise (simple and low-dimensional) than the RGB pixel space, are efficient and easier to model and train

Future work:
Generation of 3D content, e.g., 3D talking face generation, 3D human motion generation.

Generation of 3D content for Metaverse

Text prompt: (*left*) the person is walking forward (*middle*) the person’s left hand is holding a cup (*right*) the person’s right hand is waving

Tevet, Guy, et al. "Human motion diffusion model." *ICLR 2023.*

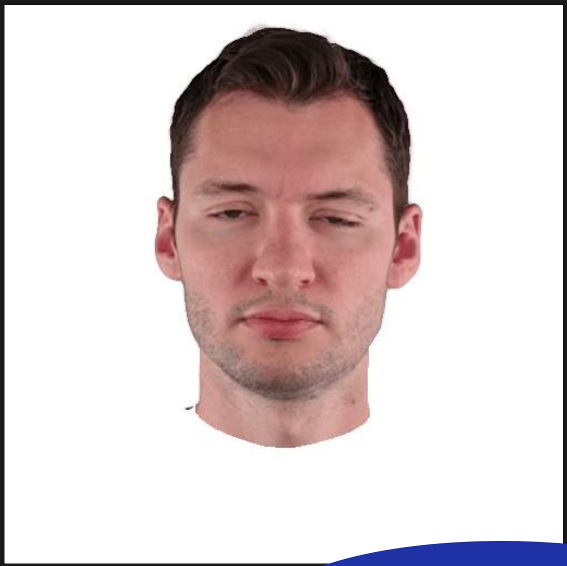
Text prompt: the person is walking forward and the left hand is holding a cup and the right hand is waving

sample 1

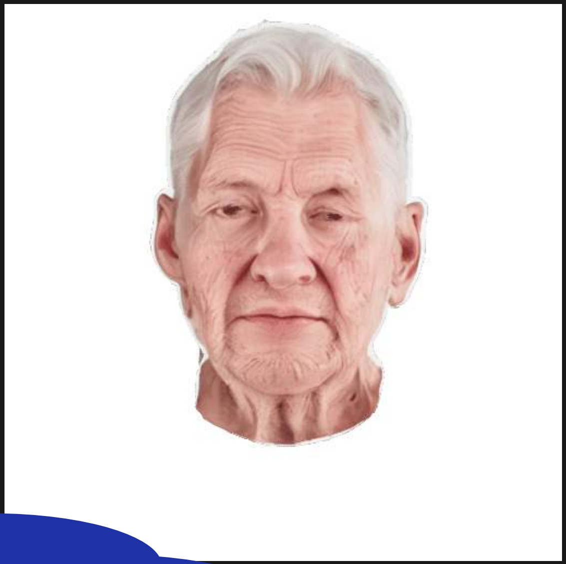
sample 2

sample 3

Tevet, Guy, et al. "Human motion diffusion model." *ICLR 2023.*



“He should look 100 years old”



How about “he should look 100 years old with reading glasses and he is smiling”?

Li, Shaoxu. "Instruct-Video2Avatar: Video-to-Avatar Generation with Instructions." arXiv preprint arXiv:2306.02903 (2023).

Failure cases of ”Instruct-Video2Avatar”:

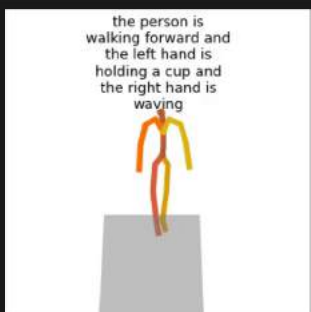
(1) fails to maintain the expression

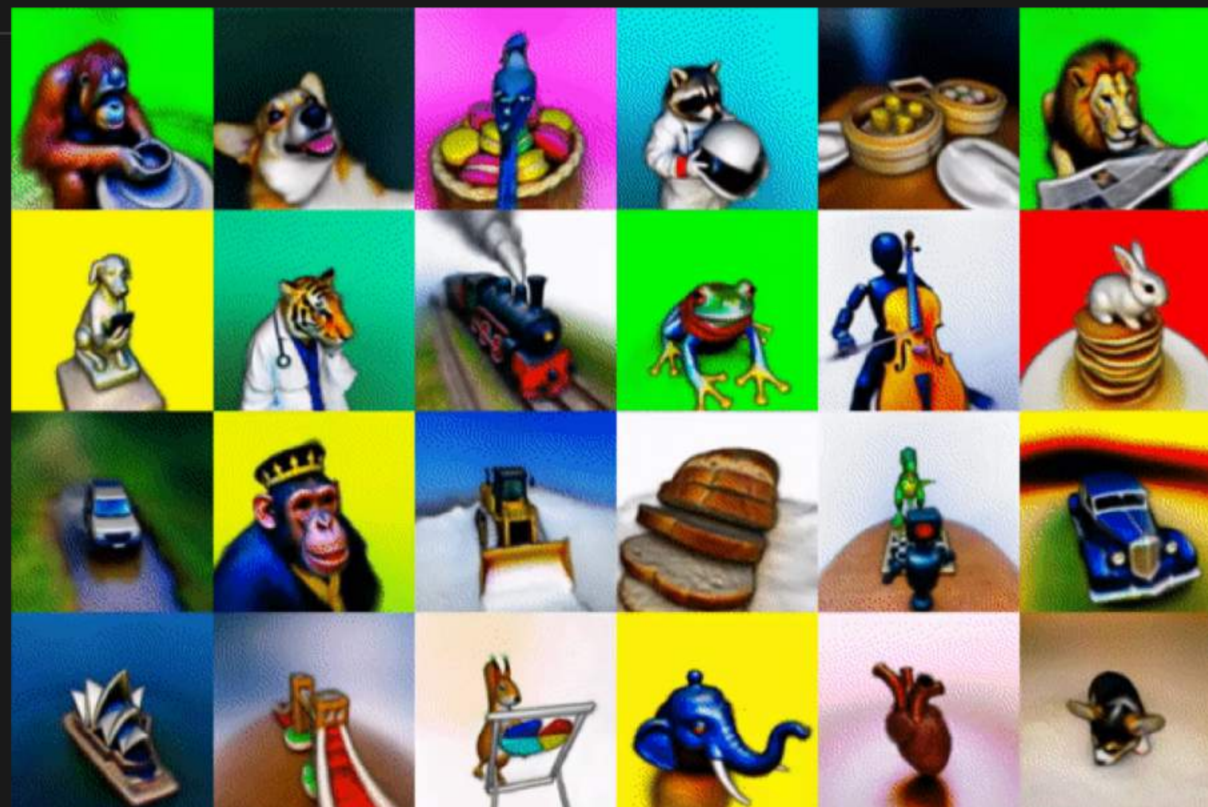
(2) the glasses are not independent of the deformable face

Li, Shaoxu. "Instruct-Video2Avatar: Video-to-Avatar Generation with Instructions." arXiv preprint arXiv:2306.02903 (2023).

Chen, Dave Zhenyu, et al. "Text2tex: Text-driven texture synthesis via diffusion models." arXiv preprint arXiv:2303.11396 (2023).

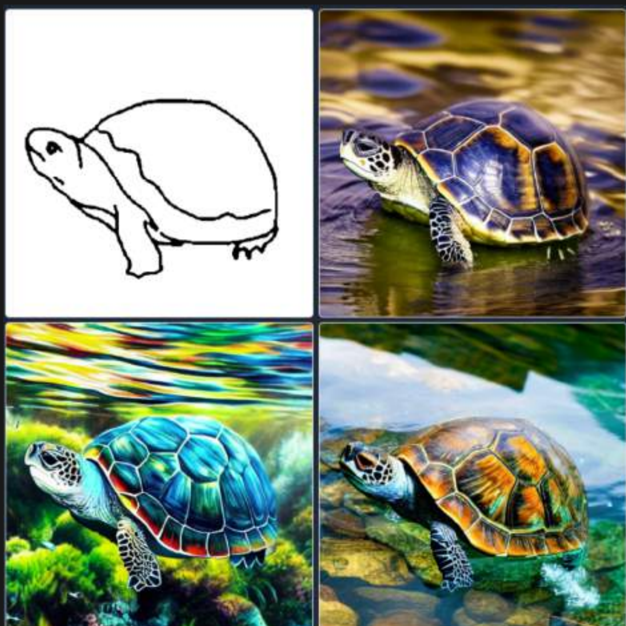
Lessons learned from NEC Labs' research – Part I



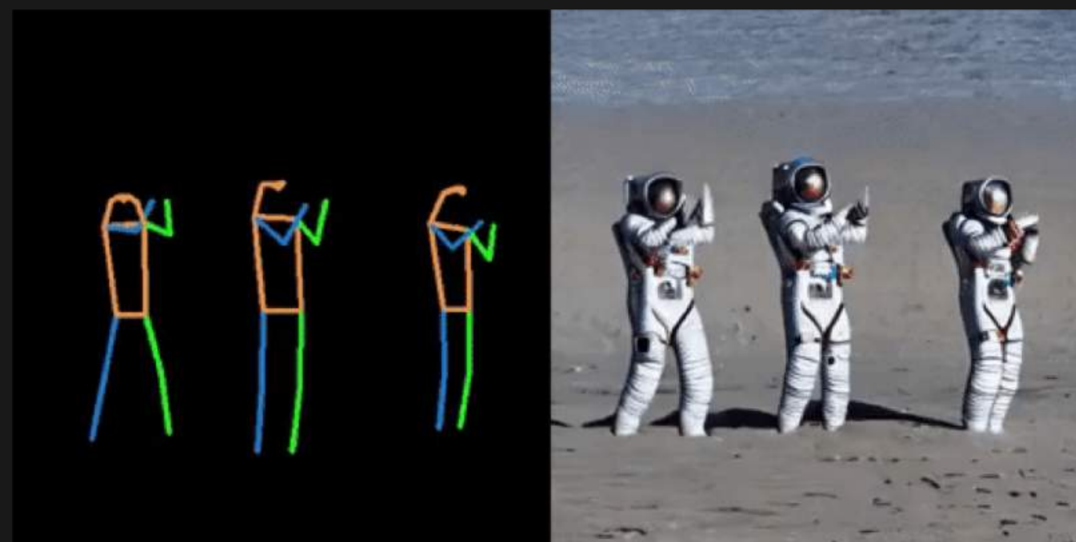


Poole, Ben, et al. "DreamFusion: Text-to-3d using 2d diffusion." arXiv preprint arXiv:2209.14988 (2022).

Adapting to user intent requires compositionality.

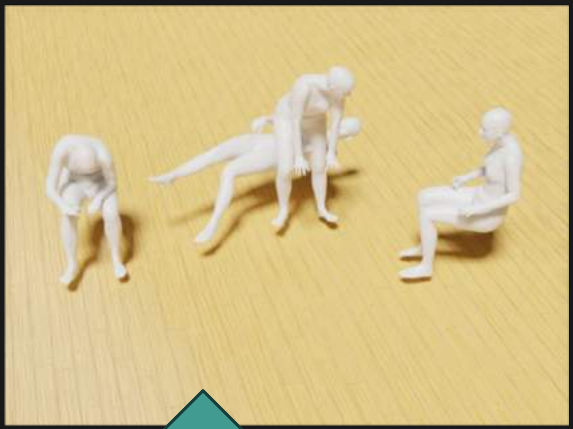


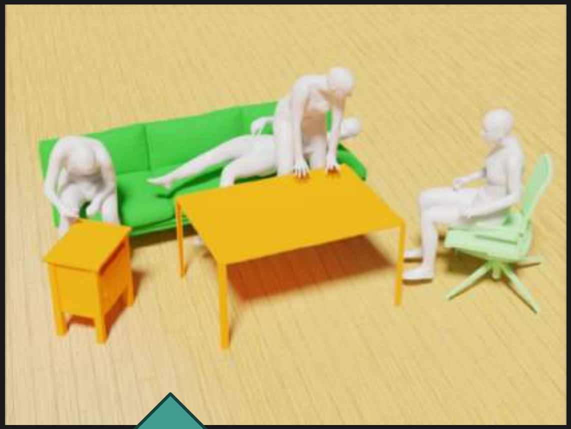
Zhang, Lvmin, and Maneesh Agrawala. "Adding conditional control to text-to-image diffusion models." arXiv preprint arXiv:2302.05543 (2023).



Ma, Yue, et al. "Follow Your Pose: Pose-Guided Text-to-Video Generation using Pose-Free Videos." arXiv preprint arXiv:2304.01186 (2023).

Lessons learned from NEC Labs' research – Part II





...

text

human meshes

3D scenes

scenes w/ textures

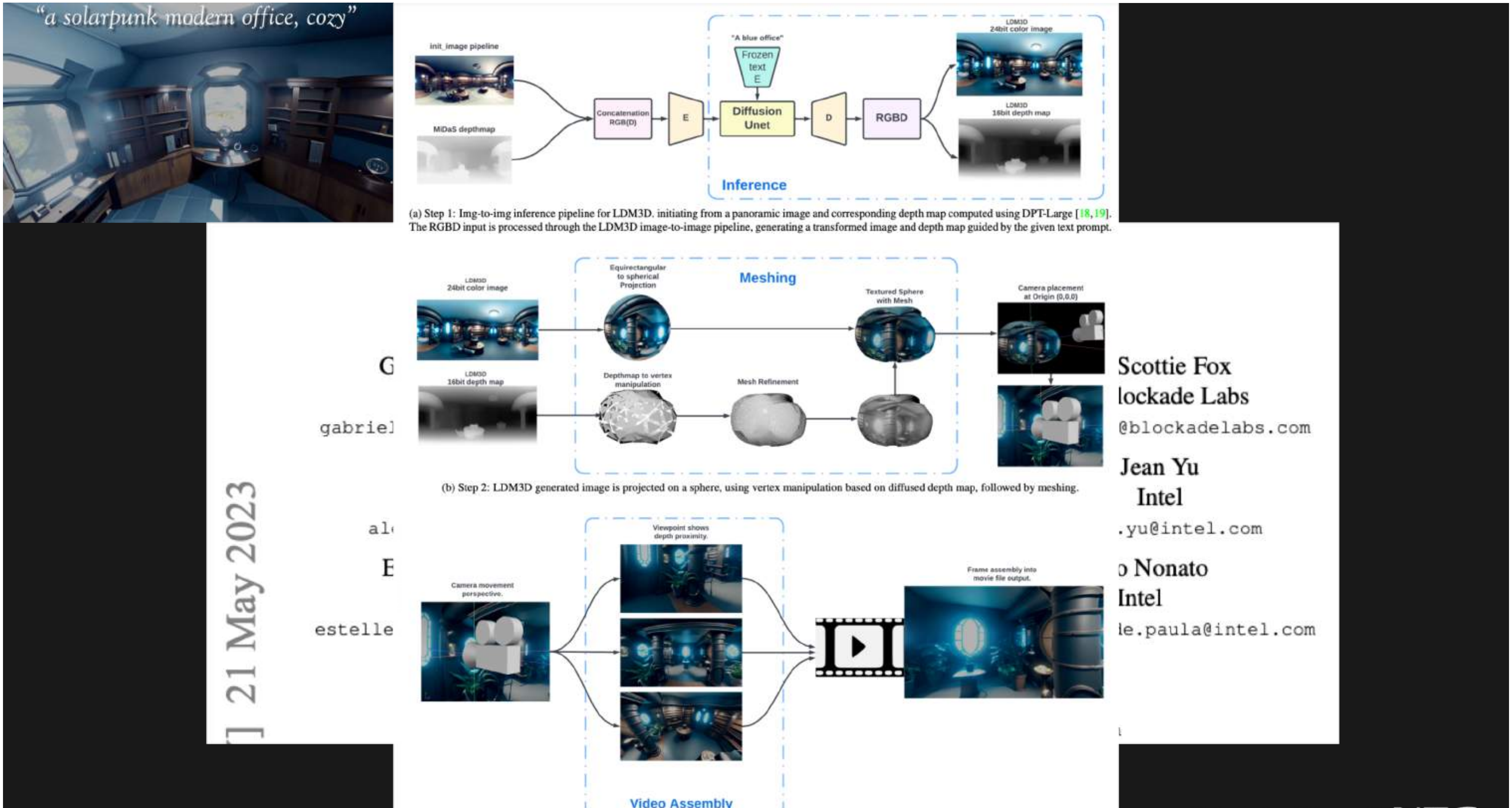
Ye, Sifan, et al. "Scene synthesis from human motion." SIGGRAPH Asia 2022 Conference Papers. 2022.

Generating highly controlled content is a challenge.

Lessons learned from NEC Labs' research – Part III

Flow/velocity-based latent space manipulation might be possible.


“a solarpunk modern office, cozy”



(a) Step 1: Img-to-img inference pipeline for LDM3D. initiating from a panoramic image and corresponding depth map computed using DPT-Large [18,19]. The RGBD input is processed through the LDM3D image-to-image pipeline, generating a transformed image and depth map guided by the given text prompt.

(b) Step 2: LDM3D generated image is projected on a sphere, using vertex manipulation based on diffused depth map, followed by meshing.

(c) Step 3: Image generation from different viewpoints, and video assembly.

Takeaways

1. Adapting to user intent requires compositionality.

2. Generating highly controlled content is a challenge.

3. Flow/velocity-based latent space manipulation might worth consideration.

Honglu Zhou
hozhou@nec-labs.com

Geometric Learning on Discrete Surface Meshes

Hsueh-Ti Derek Liu

ROBLOX

Neurosymbolic Models for 3D Generative AI

Daniel Ritchie

Brown University

BROWN
Computer Science


Me
You

Generative AI in Action at Roblox

Brent Vincent
Kartik Ayyar (@ayyar)
Roblox Creator engineering

ROBLOX

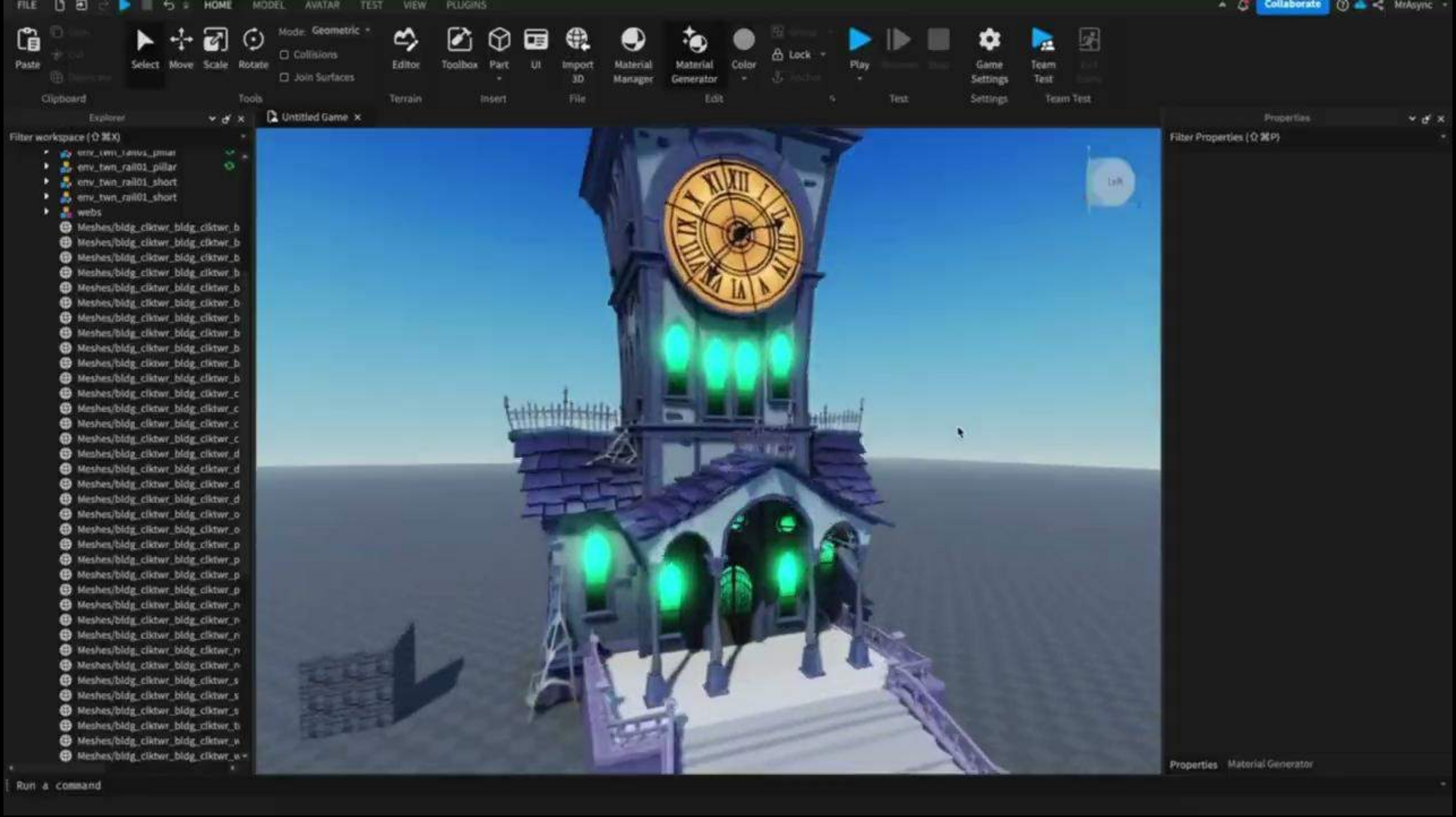
What is Roblox and why does generative AI matter to Roblox?

Roblox is platform for immersive experiences

66.1 million DAU as last reported publicly

Vision: *“Enabling Creation of Anything, Anywhere, by Anyone”*

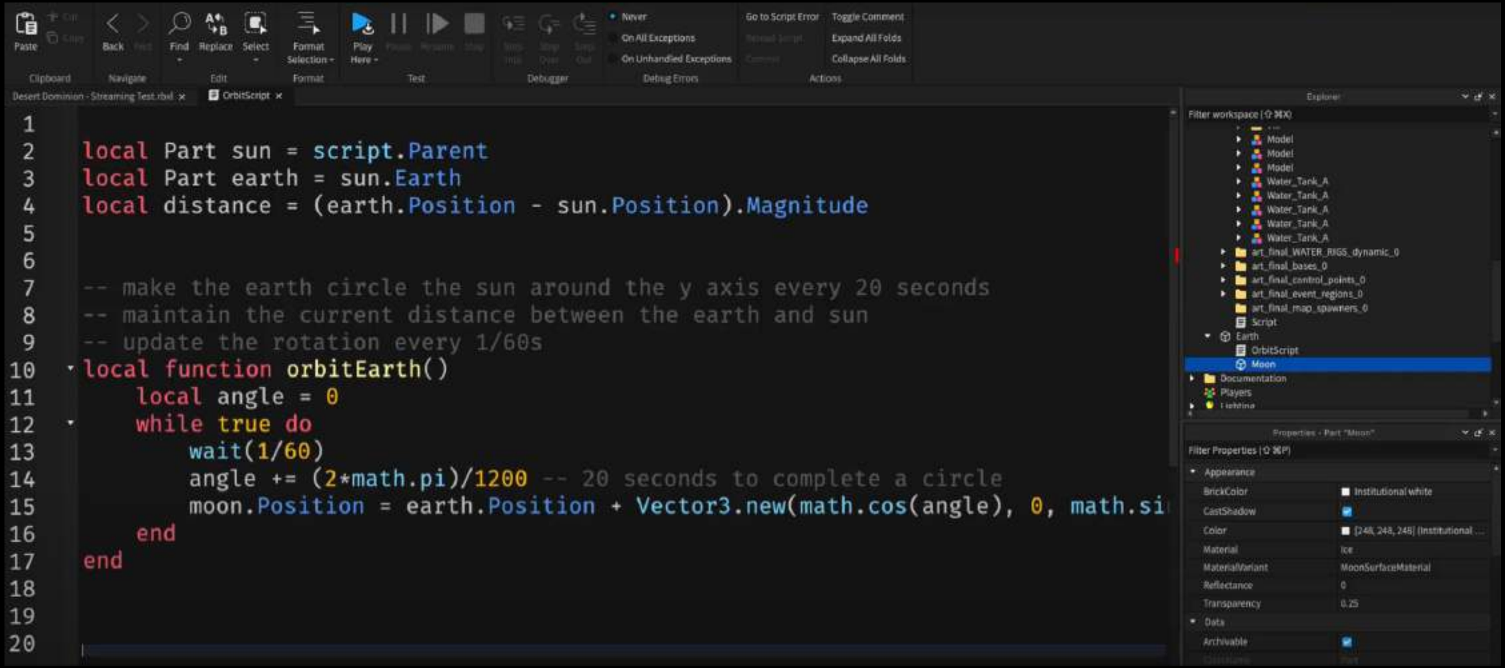
Generative AI can make Roblox creators of every skill level more productive


FILE
HOME
MODEL
AVATAR
TEST
VIEW
PLUGINS
Collaborate
MrAsync
Paste
Select
Move
Scale
Rotate
Mode: Geometric
Collisions
Join Surfaces
Editor
Toolbox
Part
UI
Import 3D
Material Manager
Material Generator
Color
Lock
Play
Game Settings
Team Test
Clipboard
Tools
Terrain
Insert
File
Edit
Test
Settings
Team Test
Explorer
Filter workspace (⇧⌘X)
Untitled Game
Properties
Filter Properties (⇧⌘P)
env_twn_rail01_pillar
env_twn_rail01_short
env_twn_rail01_short
webs
Meshes/bldg_clktwr_bldg_clktwr_b
Meshes/bldg_clktwr_bldg_clktwr_c
Meshes/bldg_clktwr_bldg_clktwr_d
Meshes/bldg_clktwr_bldg_clktwr_o
Meshes/bldg_clktwr_bldg_clktwr_p
Meshes/bldg_clktwr_bldg_clktwr_n
Meshes/bldg_clktwr_bldg_clktwr_s
Meshes/bldg_clktwr_bldg_clktwr_t
Meshes/bldg_clktwr_bldg_clktwr_w
Properties
Material Generator
Run a command

AI materials

- Diffusion models make it easy to generate textures
- Materials in Roblox are PBR materials
- A vanilla 2D texture looks bland
- Solution:
  - Need a PBR model beyond just a 2D image
  - Need textures to be tiled
- Preventing offensive content:
  - Pick a safe image generation model
  - Pre filter prompt
  - Post filter output

Code in Roblox: Lua(u) attached to objects



AI coding: powered by large language models

Language model: an overview

- Model text as a sequence of tokens
- Learn a probability distribution over the next token to output
- Called autoregressively to generate output tokens
- Stops when you hit a certain number of tokens or a stop token/sequence

AI coding: areas of focus in this talk

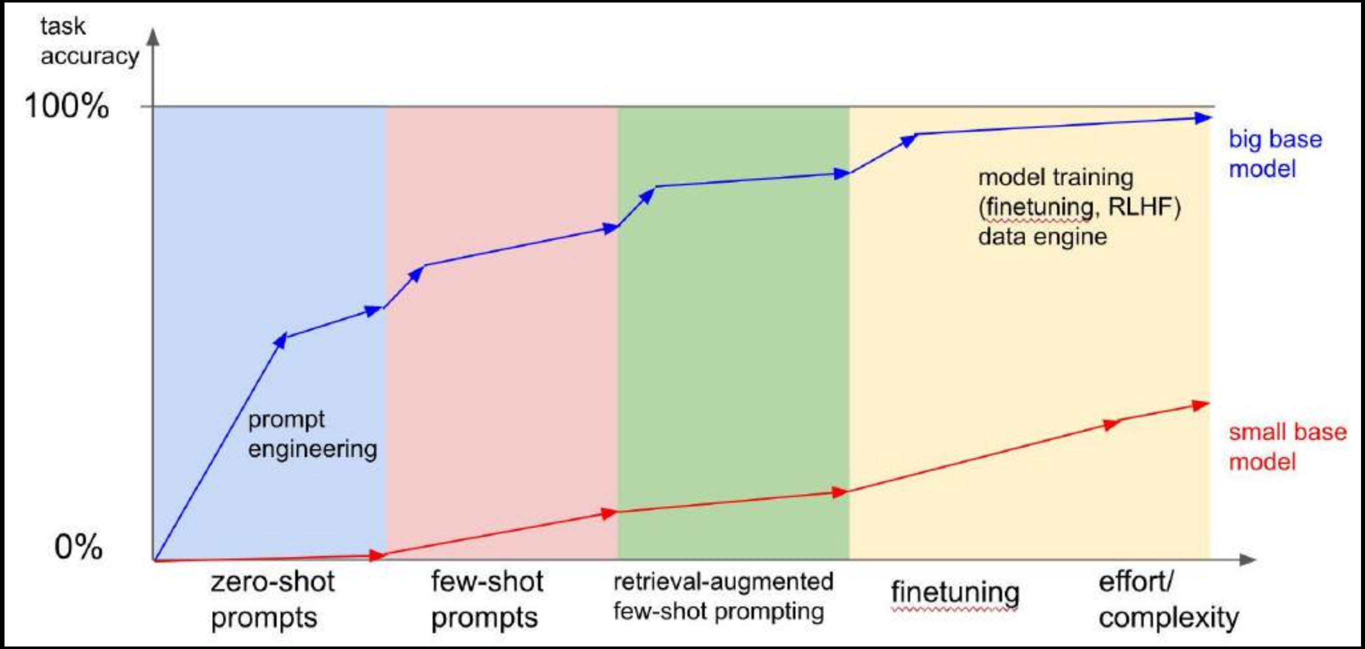
This isn't a talk about language model basics

- It's a talk about making coding LMs work well

Will focus on 3 main areas of coding language models

- Evaluation
- Prompting
- Fine tuning

The path to improving model quality
Source: [@karpathy tweet]



ROBLOX

Evaluation methodology

Two benchmarks

- Metrics like Bleu score aren't great for code
- HumanEval eval suite, translated to Lua
    - Data structures and algorithms tests
    - pass@k metric:
    - "Do any of k generations pass tests?"
    - In practice, generate n > k examples
- RobloxEval: Roblox centric benchmark
    - Physics, Simulation, games
- Online experiments: A/B testing accept rates

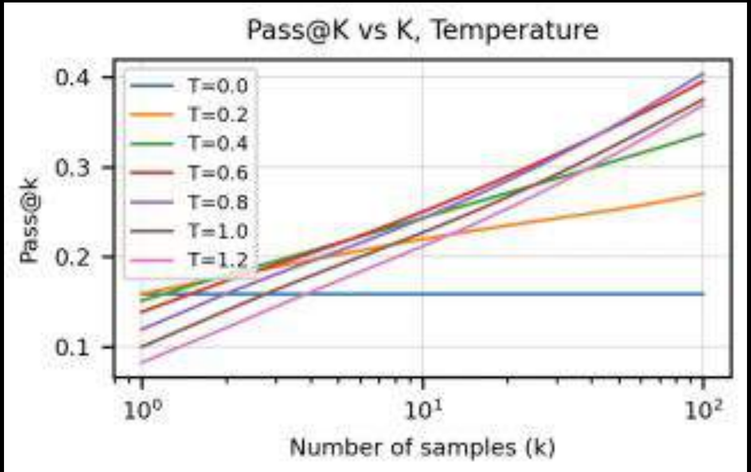
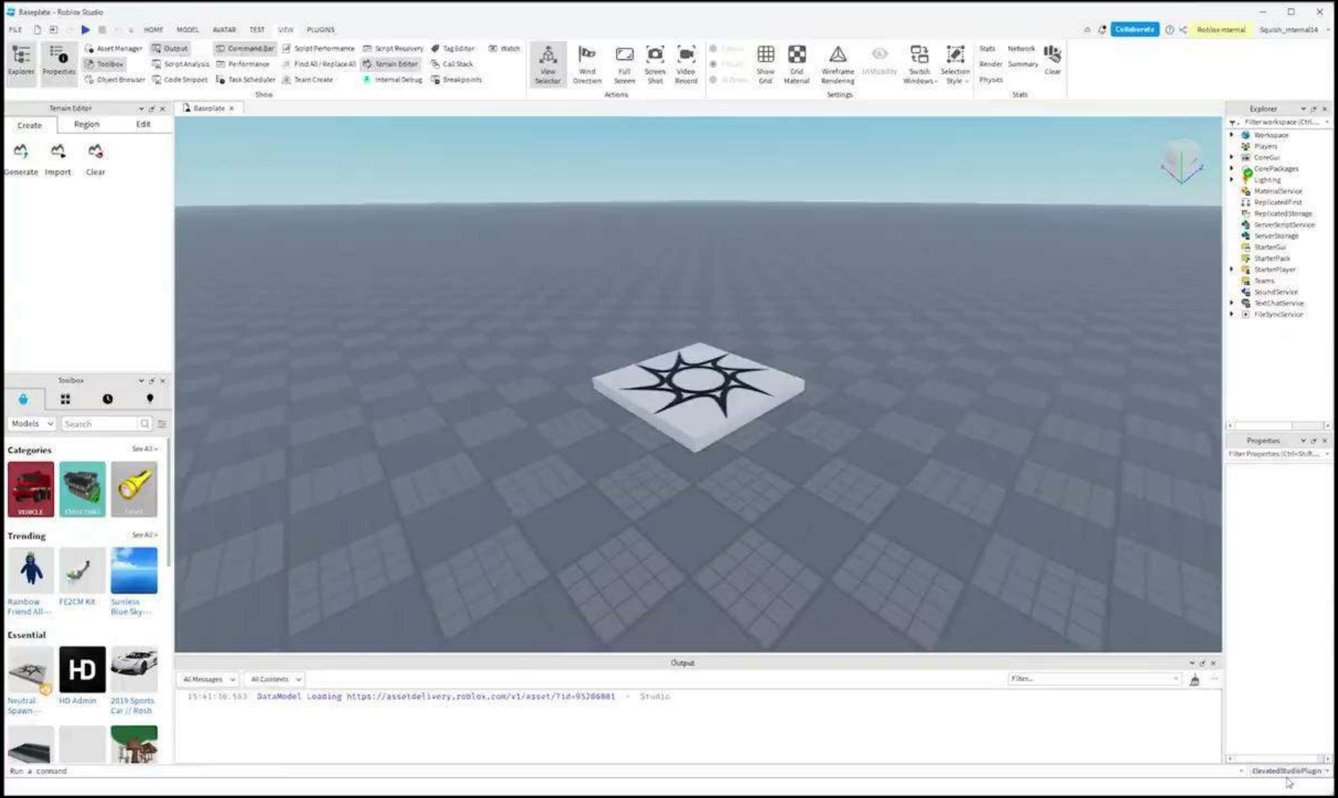
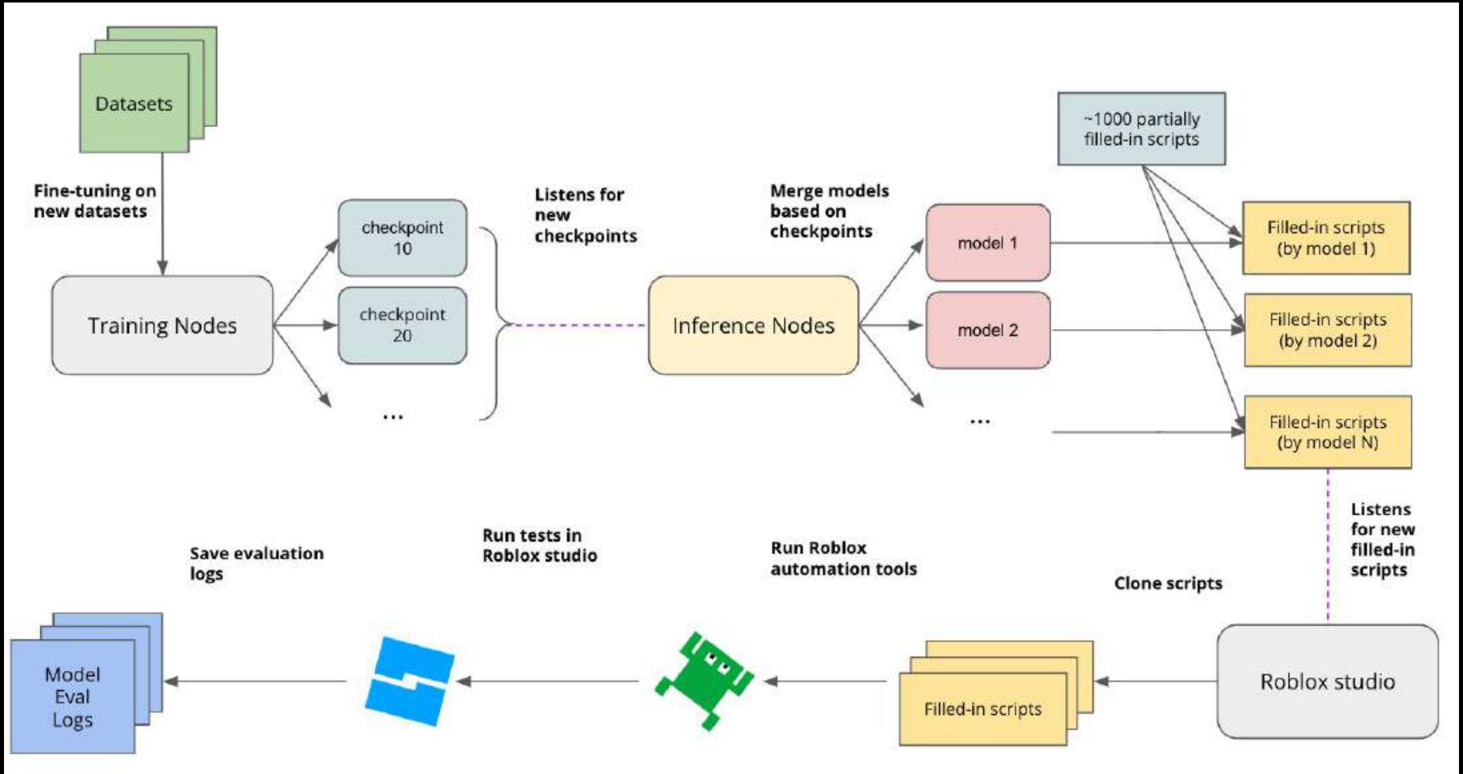


Image credit: [GPT Codex paper]()

Eval framework demo video



Evaluation while fine tuning



Some notes on quality and fine tuning methodology

1. All data used is publicly available
2. Datasets used include:
   a. Roblox marketplace
   b. The Stack
3. All quality gains are relative vs. a baseline
4. Prediction quality vs. latency / inference cost tradeoff:
   a. Inconclusive experiments
   b. We tend to err on the side of prediction quality

AI code: prompting experiments

Prompt based path names	+15% (depends on baseline model)
Fill in the middle prompting	+10% over baseline
Prompt with contents of related files	TBD

AI code: some quality experiments

Fine tune on docs examples	+2% over baseline
Fine tune on cleaned marketplace data	+4% over baseline
Fine tune on path names	+10-15%
Fine tuned on cleaned Lua Stack corpus	+4% over baseline
Fine tune with type annotation of parent	inconclusive

Future directions

Beyond just code completion

- Explaining code
- Debugging code
- Write commit messages
- Asking for coding help

Luau code model: [Luau] is Roblox's optional typed language

- Truly open (MIT Licensed)

RLHF: Reinforcement learning from human feedback

- Can we use human ranking of example pairs to learn better?
- Have a training pipeline working
- Unclear if the data quality we have will give us good results

Future directions: complex multi modal creation

“Create a block of pink lava that kills the player when they touch it.”

Audience question:

How would you solve this?

Potential approaches

1. Create a block
    a. Create a primitive object?
    b. Or fetch it from the marketplace
2. How do you interpret lava block?
    a. Is it a reference to appearance?
    b. Or functionality?
    c. If it refers to appearance:
        i. Does it mean a texture?
        ii. Or an in built material?
3. Functionality enabled by scripting: “kill the player”
    a. Create a script?
    b. Pick one from a library?
    c. What if the object you found from the marketplace already has a script

Want to learn more and work on these problems?

Stay in touch:

Mubbasir Kapadia (Email: mkapadia@roblox.com
Honglu Zhou (Email: hz289@scarletmail.rutgers.edu)
Derek Liu (Email: hsuehtiliu@roblox.com)
Daniel Ritchie (Email: daniel_ritchie@brown.edu)
Kartik Ayyar (Email: kayyar@roblox.com, Twitter: ayyar)

Careers at Roblox:

https://careers.roblox.com/jobs

ROBLOX

My goals for this talk:

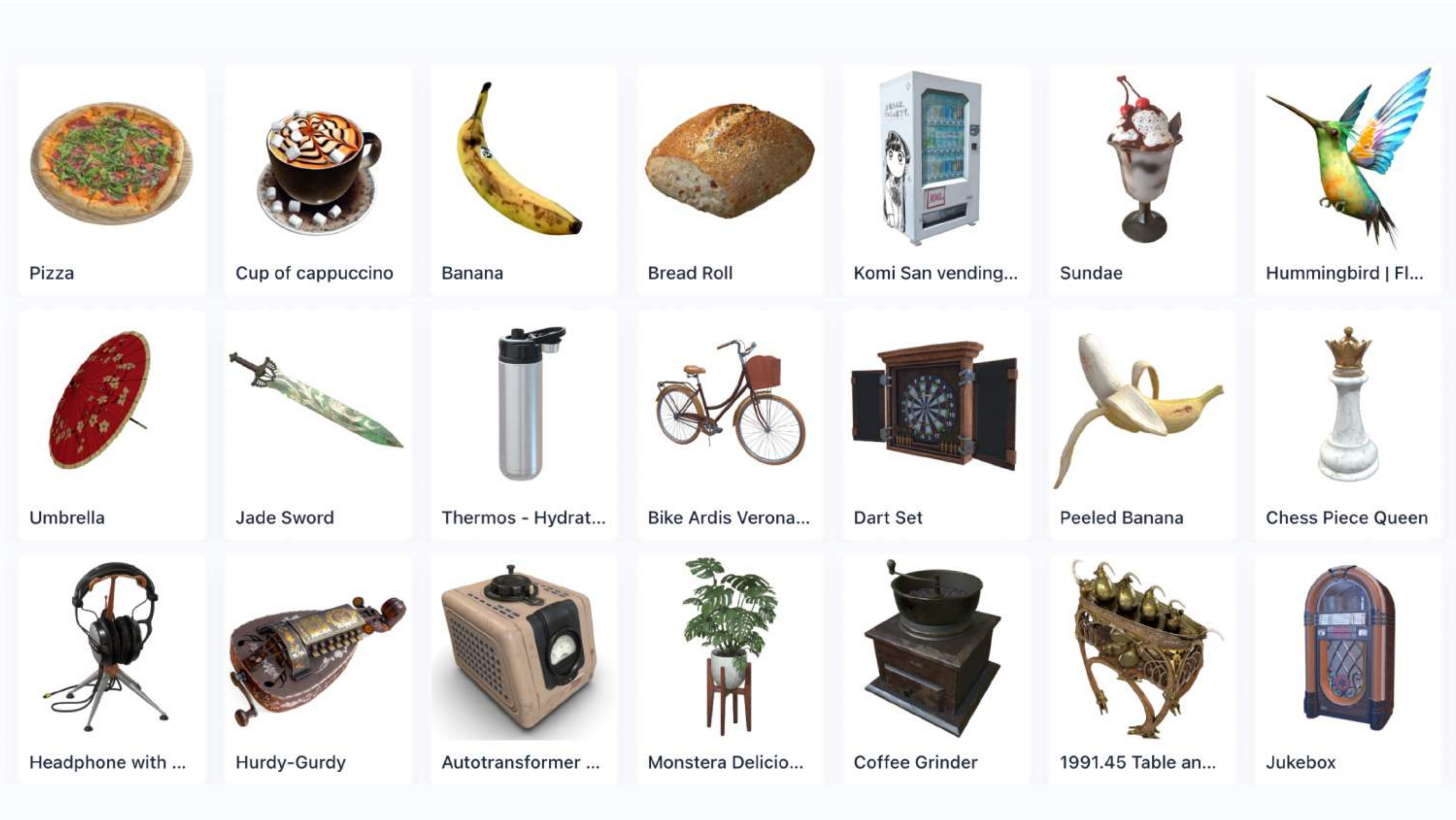
1. Introduce you to neurosymbolic models

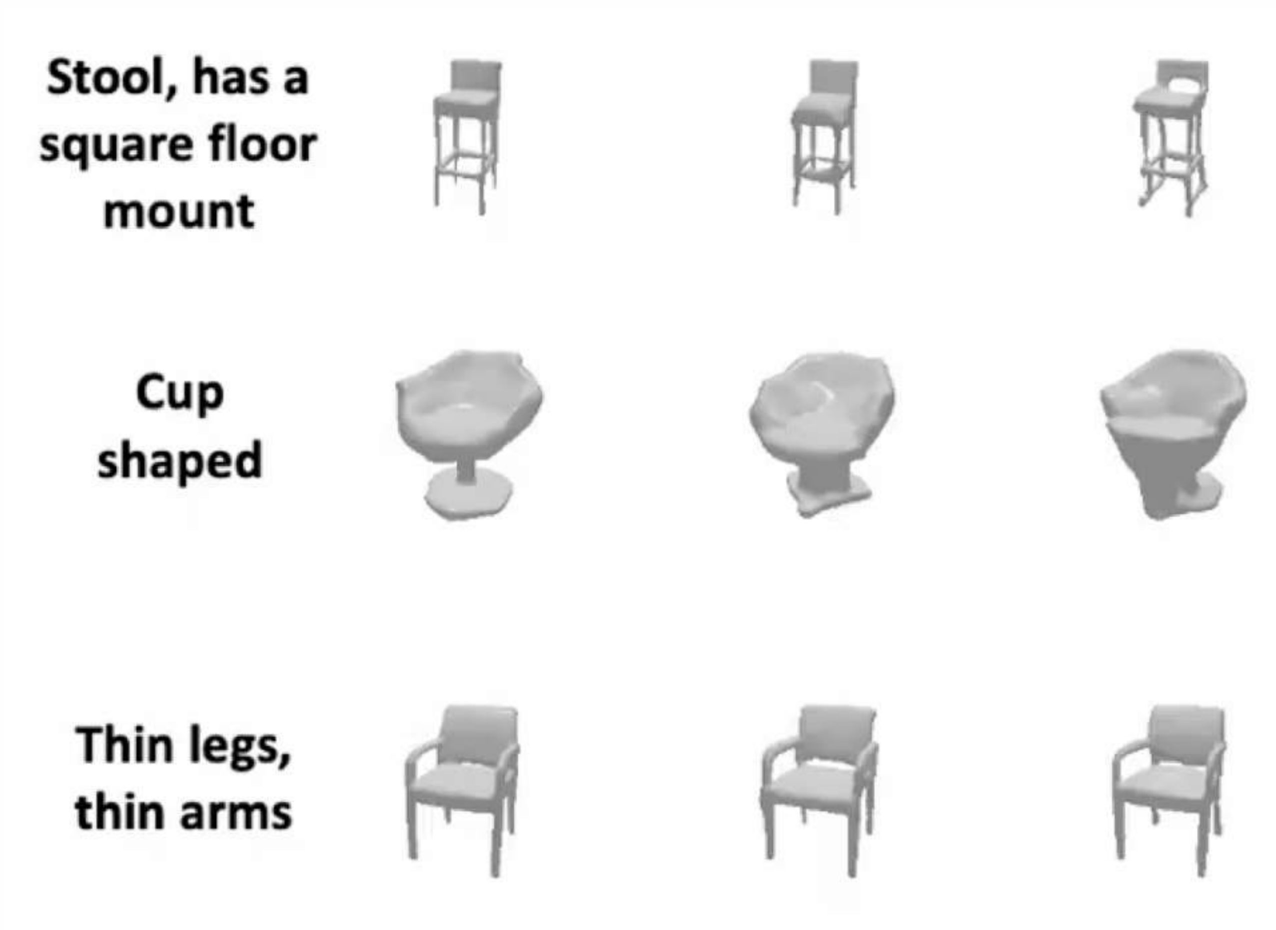
2. Convince you that...


THIS IS THE WAY

Everyone is excited about
deep generative models these days!

[Chen & Zhang '19. Learning Implicit Fields for Generative Shape Modeling]


Pizza
Cup of cappuccino
Banana
Bread Roll
Komi San vending...
Sundae
Hummingbird | Fl...
Umbrella
Jade Sword
Thermos - Hydrat...
Bike Ardis Verona...
Dart Set
Peeled Banana
Chess Piece Queen
Headphone with ...
Hurdy-Gurdy
Autotransformer ...
Monstera Delicio...
Coffee Grinder
1991.45 Table an...
Jukebox



[Mittal et al. ’22. AutoSDF]

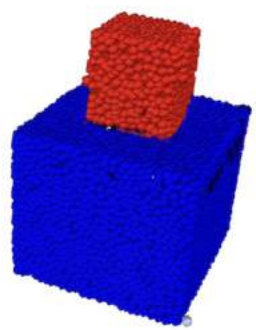
"a corgi wearing a
red santa hat"

"a multicolored rainbow
pumpkin"

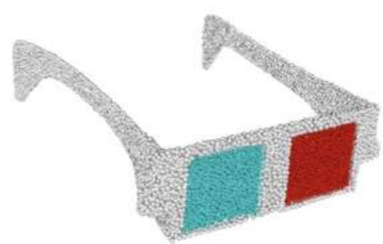
"an elaborate fountain"

"a traffic cone"

"a vase of purple flowers"



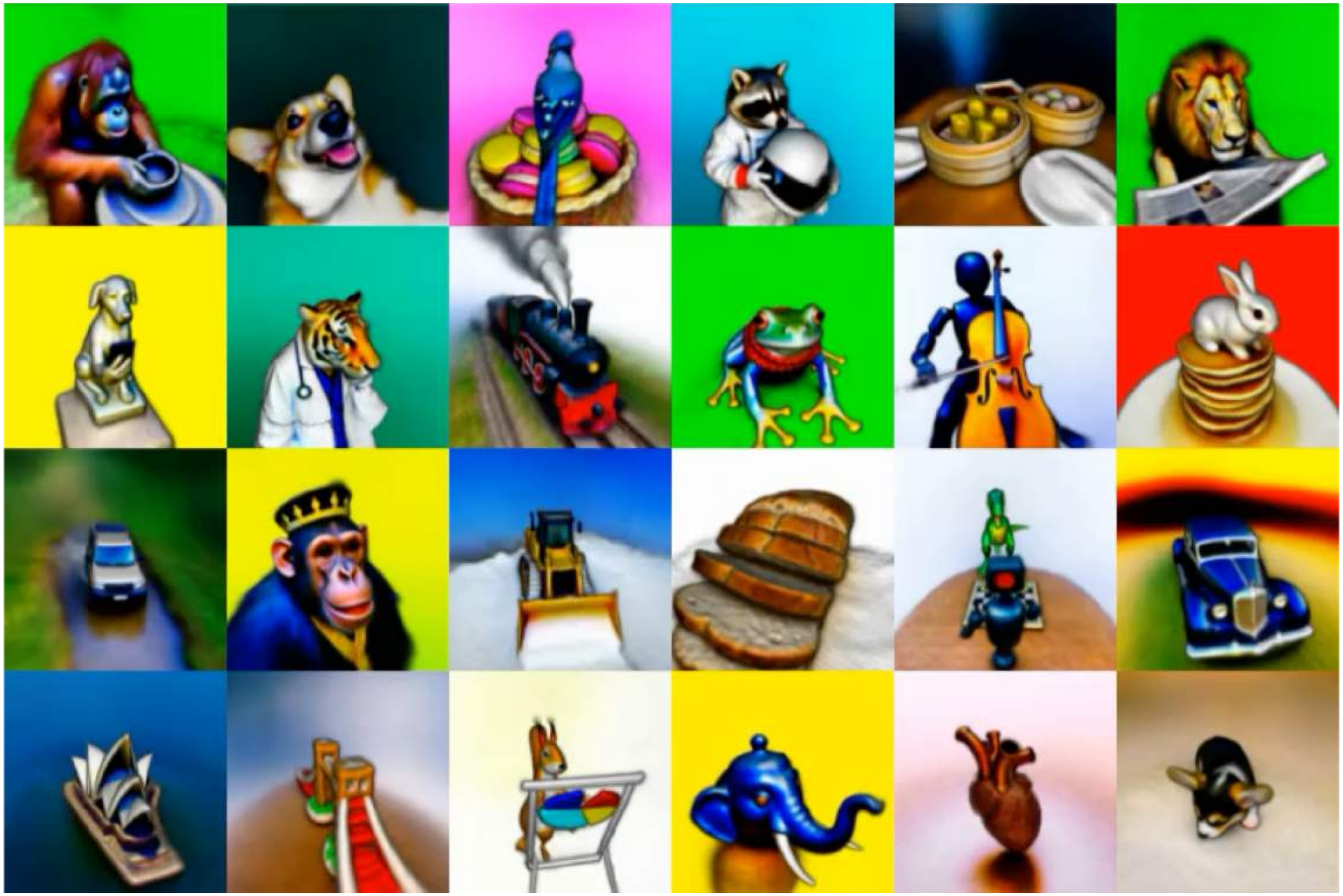
"a small red cube is sitting
on top of a large blue cube.
red on top, blue on bottom"



"a pair of 3d glasses,
left lens is red right
is blue"

"an avocado chair, a chair
imitating an avocado"

[Nichol et al. '22. Point-E]



[Poole et al. ’22. DreamFusion]

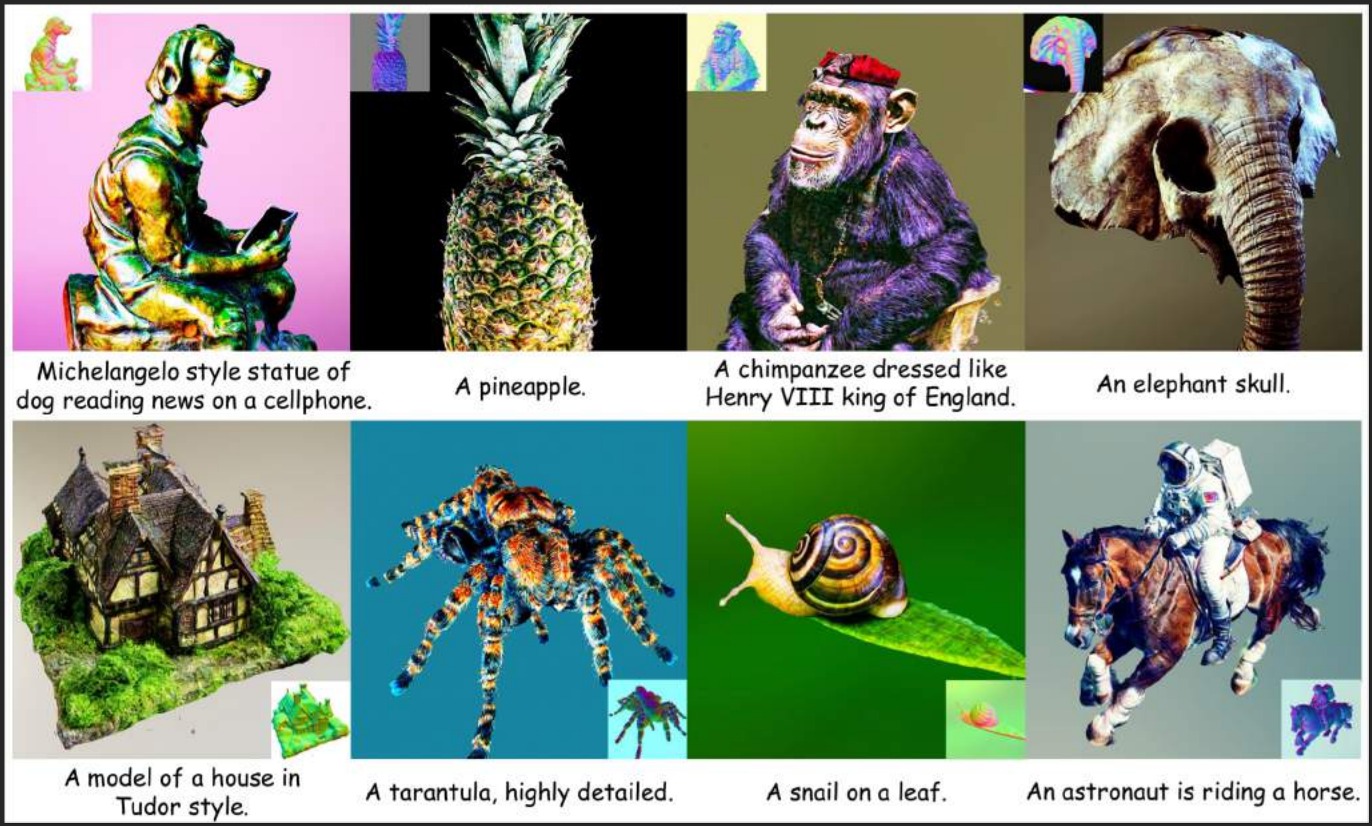
What Makes Deep Generative Models Great?

Detail

[Zhang et al. ’23. Locally Attentional SDF Diffusion]

What Makes Deep Generative Models Great?

Detail



[Wang et al. '23. ProlificDreamer]

What Makes Deep Generative Models Great?

Detail

Variety

...

[Hui et al. ’22. Neural Wavelet-Domain Diffusion]

What Makes Deep Generative Models Great?

Detail

Variety

Ease

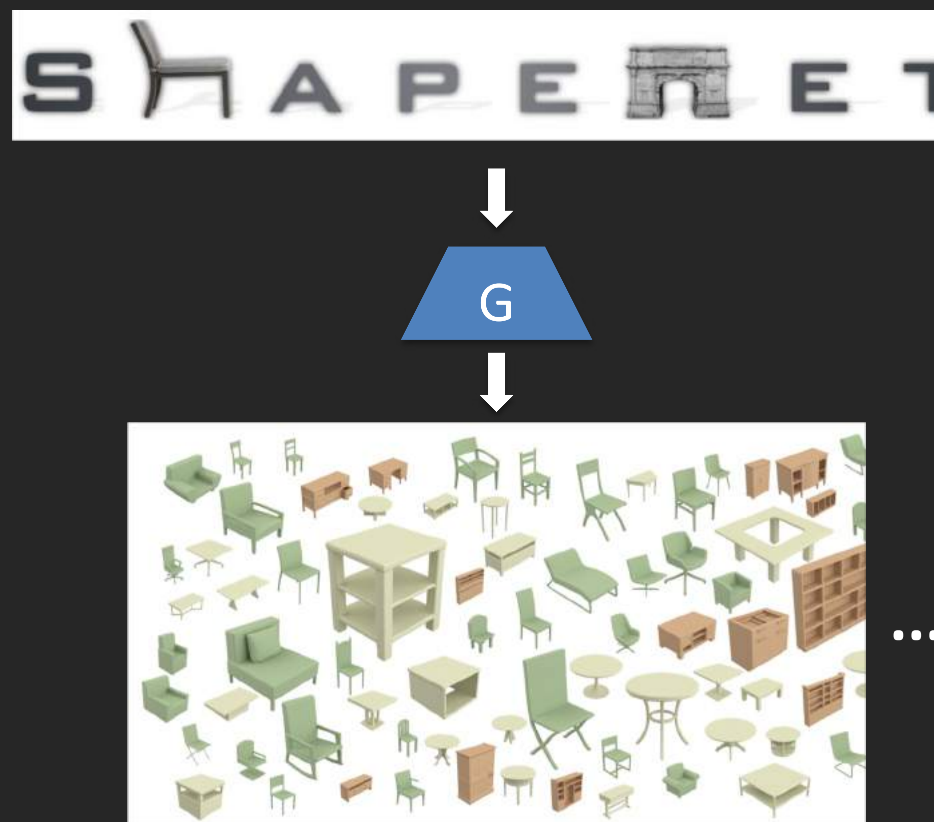
Autodesk Maya

What Makes Deep Generative Models Great?

Detail

Variety

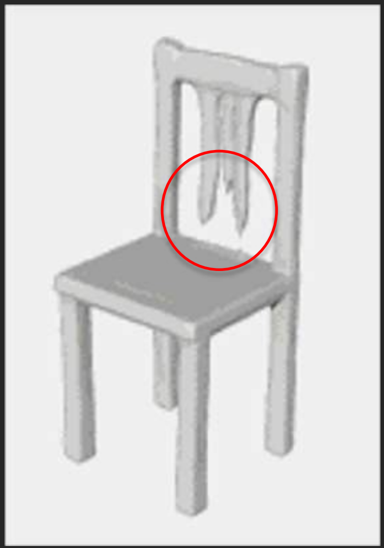
Ease



Everyone is excited about
deep generative models these days!

...so are we done?

What Makes Deep Generative Models ***Not*** Great?



Quality Control

What Makes Deep Generative Models *Not* Great?

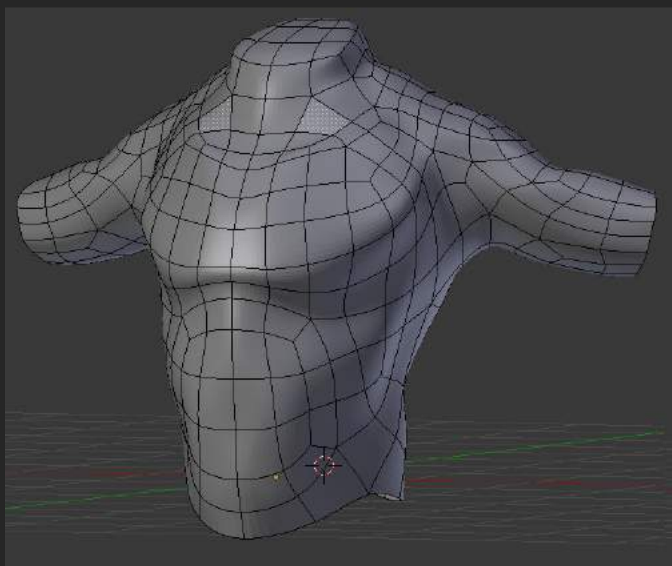
[Maneesh Agrawala '23. Unpredictable Black Boxes are Terrible User Interfaces]

Quality Control

Interpretability

"Prompt Engineering Hell"

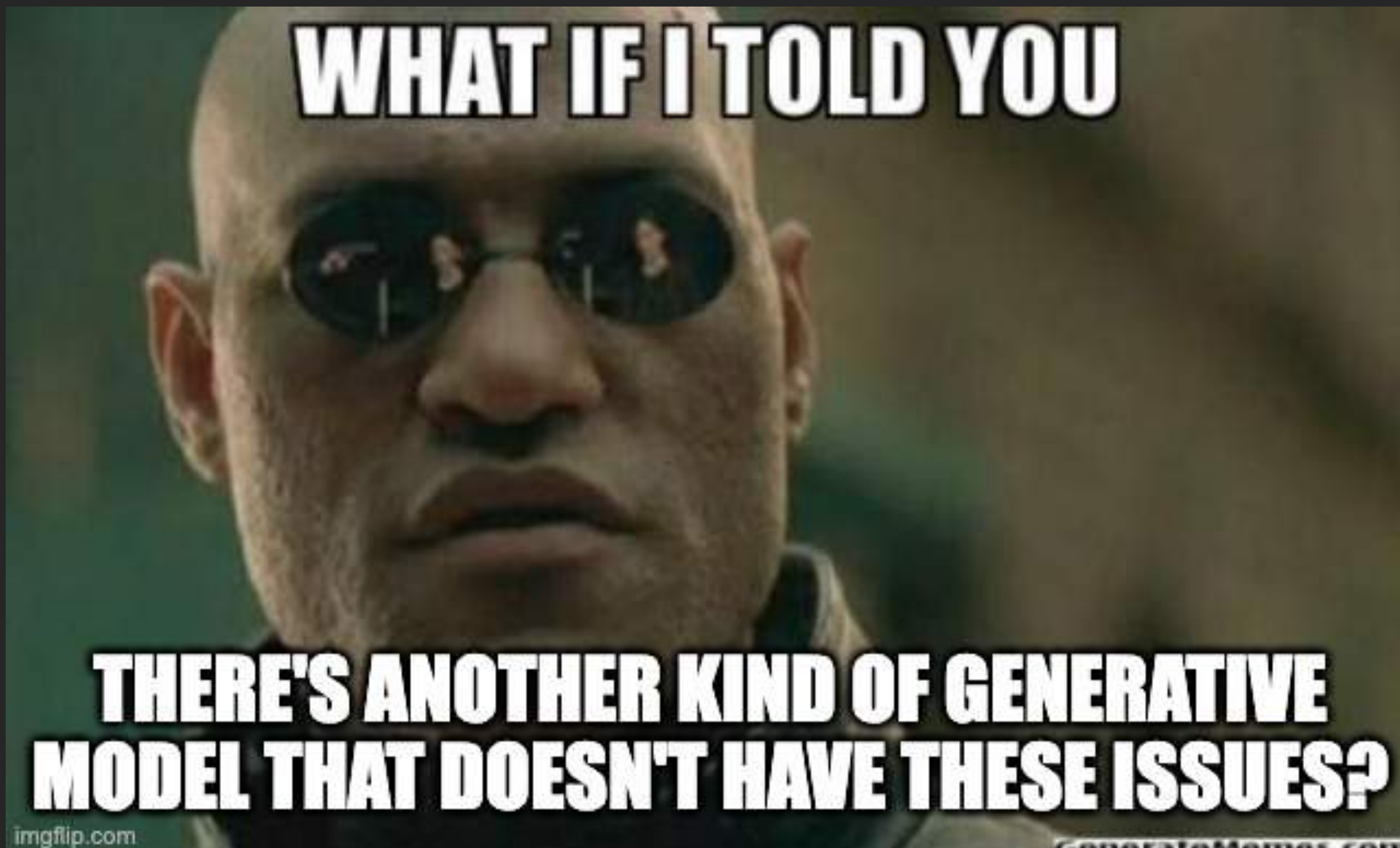
What Makes Deep Generative Models ***Not*** Great?



Quality Control

Interpretability

Manipulability


WHAT IF I TOLD YOU
THERE'S ANOTHER KIND OF GENERATIVE MODEL THAT DOESN'T HAVE THESE ISSUES?
imgflip.com

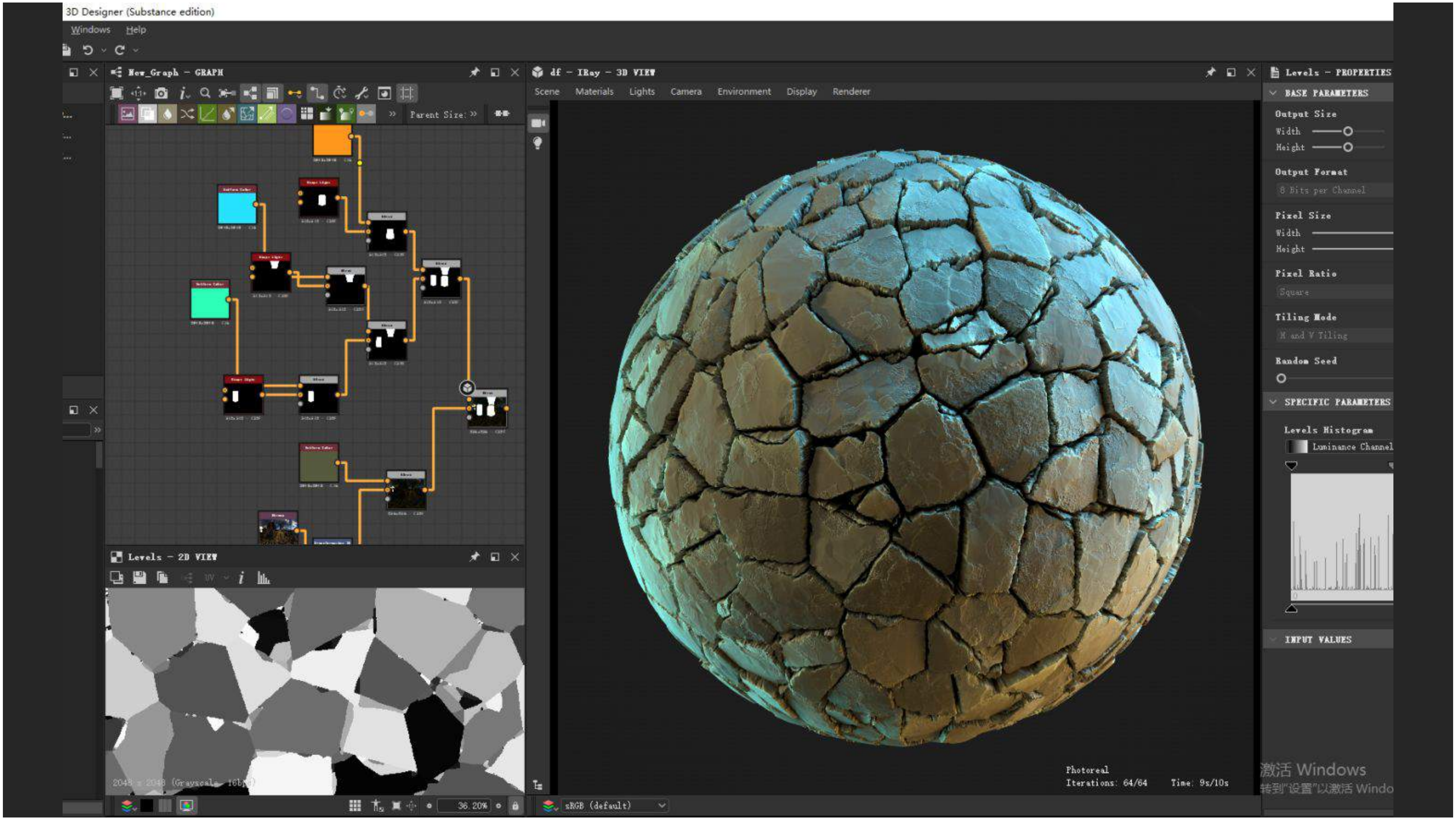
It’s not some ultra-new, top-secret, stealth-mode technology

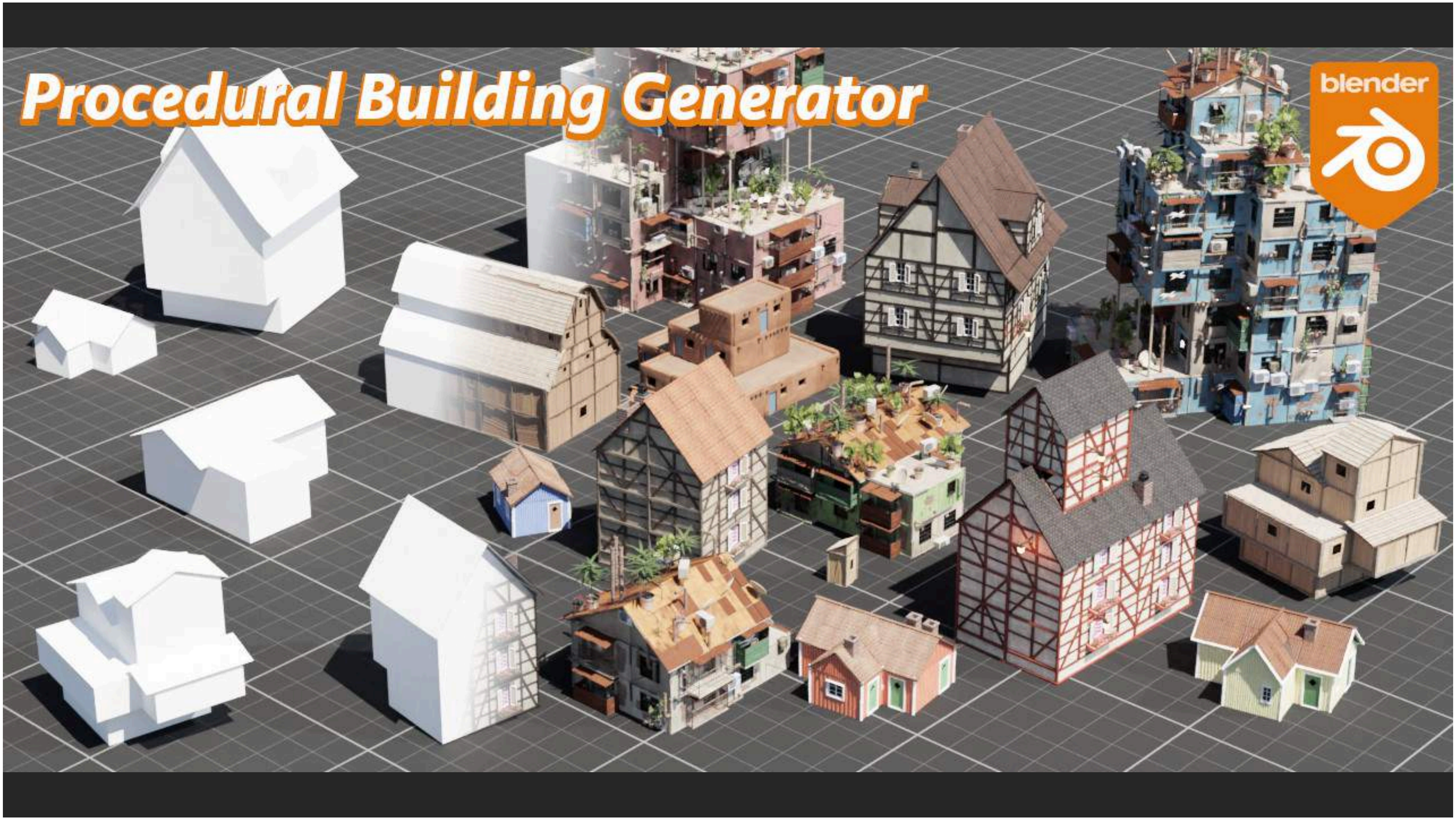
It’s actually something we graphics folks have been using for decades

PROCEDURAL MODELING

Graphics jargon for:

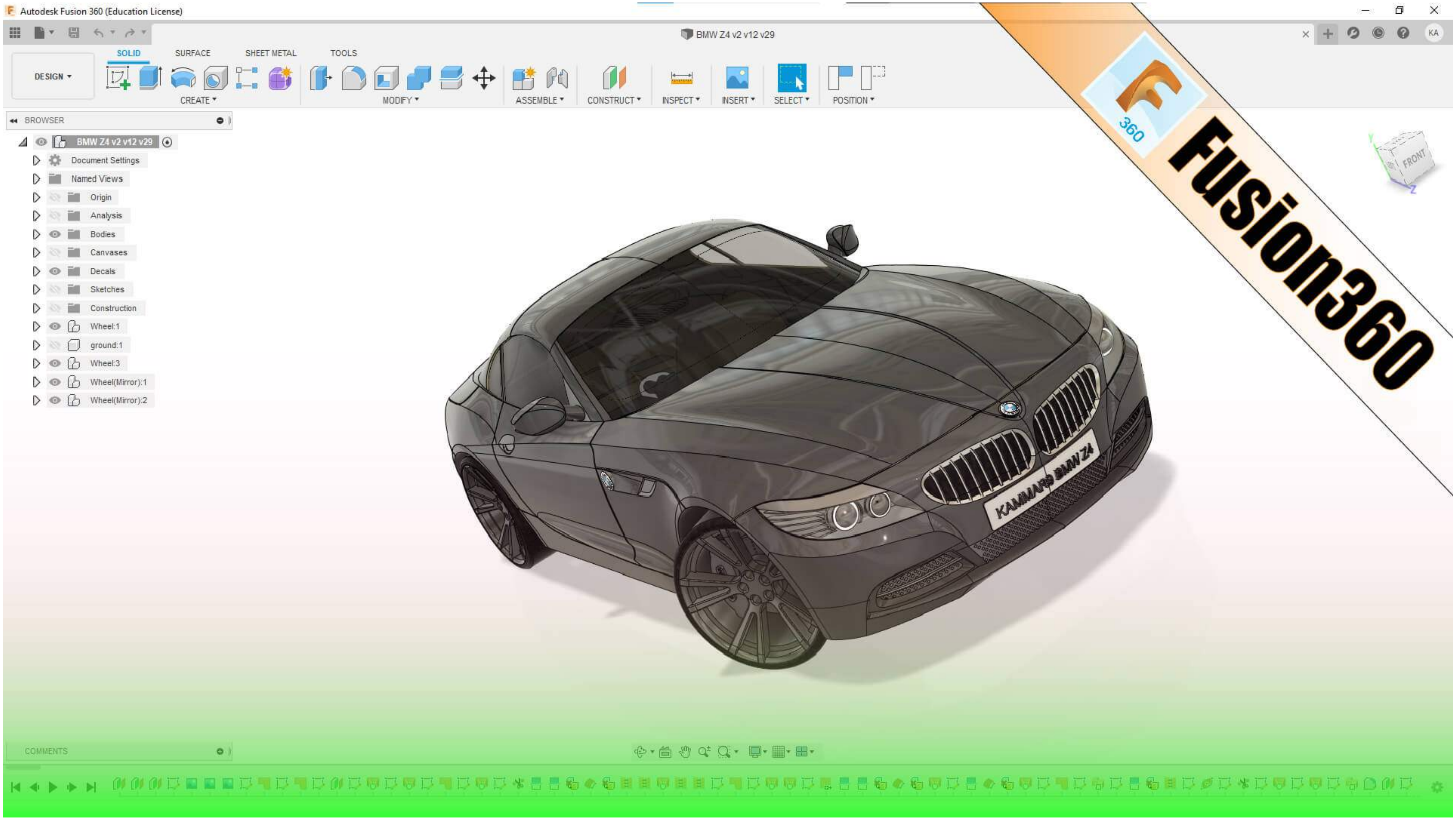
"writing a (potentially pseudorandom) program that outputs graphics assets"


3D Designer (Substance edition)
Windows
Help
New_Graph - GRAPH
Parent Size:
df - IRay - 3D VIEW
Scene
Materials
Lights
Camera
Environment
Display
Renderer
Levels - PROPERTIES
BASE PARAMETERS
Output Size
Width
Height
Output Format
8 Bits per Channel
Pixel Size
Width
Height
Pixel Ratio
Square
Tiling Mode
H and V Tiling
Random Seed
SPECIFIC PARAMETERS
Levels Histogram
Luminance Channel
INPUT VALUES
Levels - 2D VIEW
Photoreal
Iterations: 64/64
Time: 9s/10s
sRGB (default)
36.20%
激活 Windows



Procedural Building Generator
blender


MVRsimulation



Autodesk Fusion 360 (Education License)
BMW Z4 v2 v12 v29
SOLID
SURFACE
SHEET METAL
TOOLS
DESIGN
CREATE
MODIFY
ASSEMBLE
CONSTRUCT
INSPECT
INSERT
SELECT
POSITION
BROWSER
BMW Z4 v2 v12 v29
Document Settings
Named Views
Origin
Analysis
Bodies
Canvases
Decals
Sketches
Construction
Wheel:1
ground:1
Wheel:3
Wheel(Mirror):1
Wheel(Mirror):2
FRONT
360
Fusion360
KAMMARS BMW Z4
COMMENTS

Procedural Models Don’t Have the Issues that Deep Generative Models Have

Quality Control

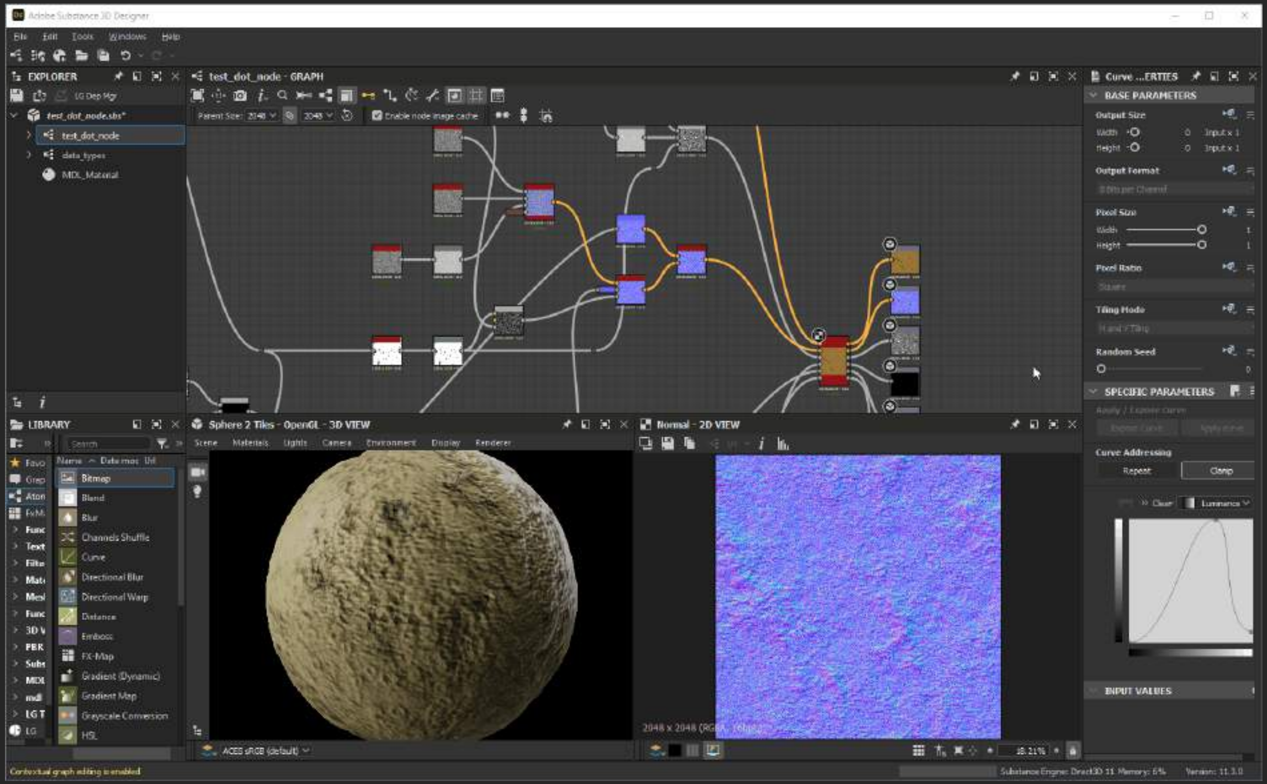
Code has:
- Well-defined structure
- Meaningful parameters with range bounds

Many types of “bad” outputs aren’t possible, by construction
- In some languages, you could even *prove* this with static analysis

Procedural Models Don't Have the Issues that Deep Generative Models Have

Quality Control

Interpretability

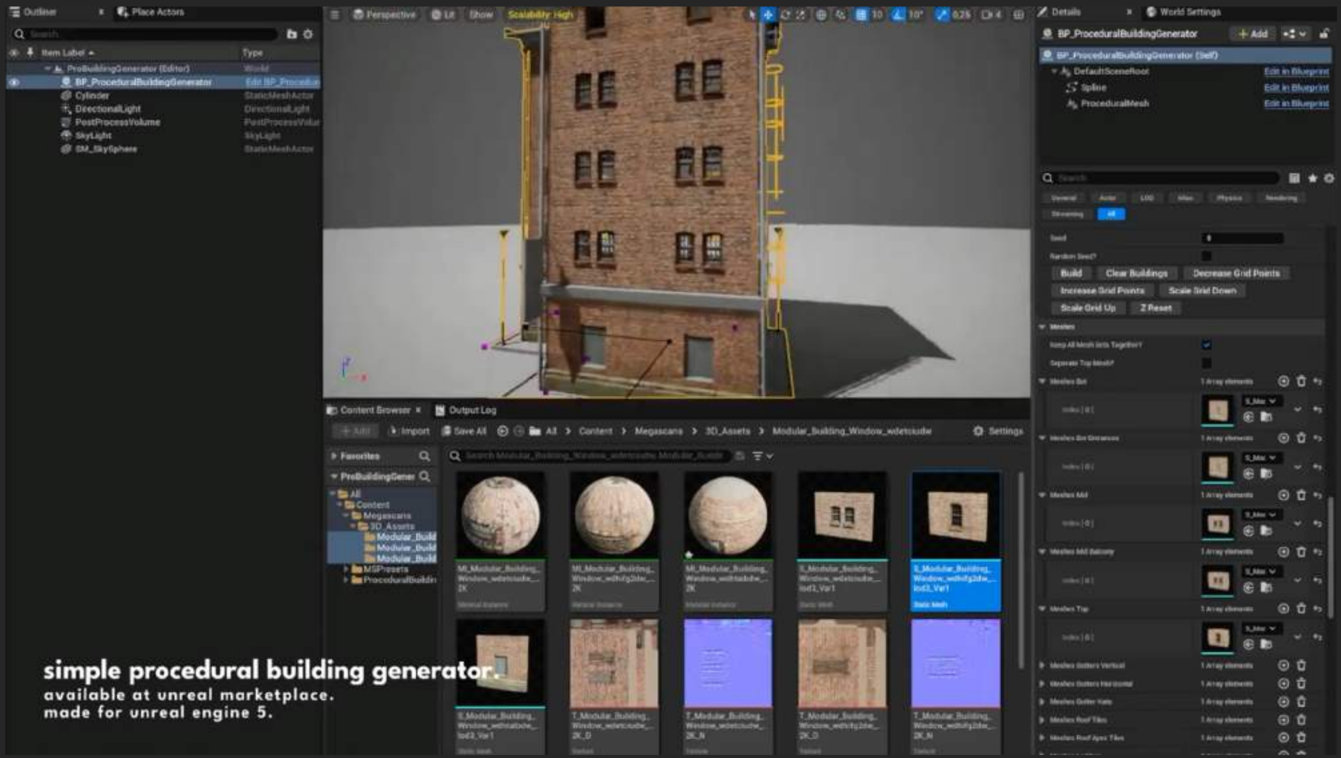


Procedural Models Don't Have the Issues that Deep Generative Models Have

Quality Control

Interpretability

Manipulability



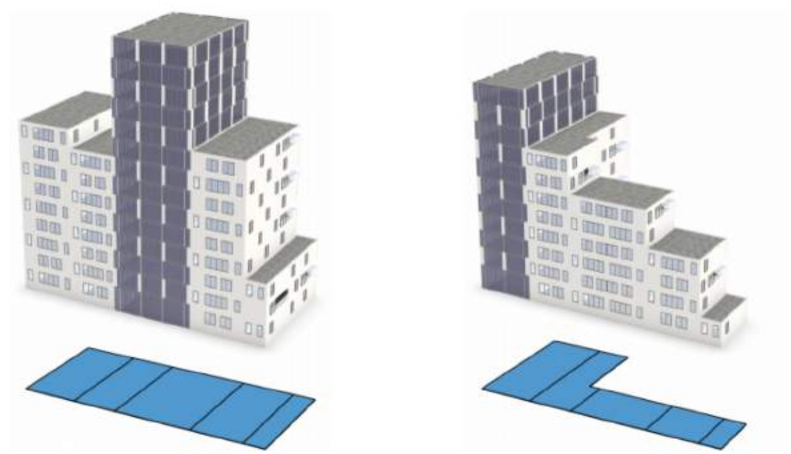
Wait a minute...

If procedural models are so great, why isn't 3D content creation "solved" already?

Problems with Procedural Models

```
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) } )
```



Ease

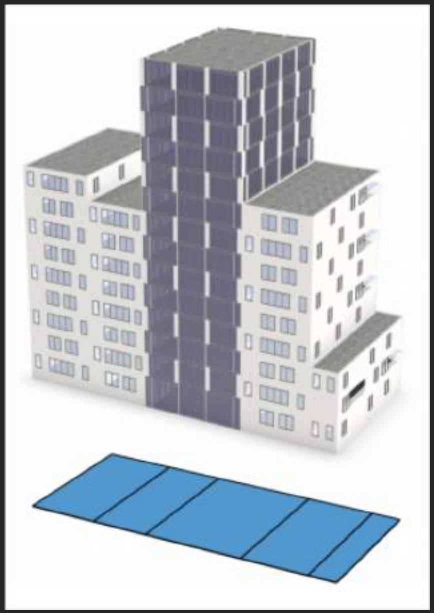
Problems with Procedural Models

Ease

Variety

…

Problems with Procedural Models



Ease

Variety

Detail

Let’s recap...

Pros & Cons
Deep Generative Models

Pros	Cons
Detail	Quality Control
Variety	Interpretability
Ease	Manipulability

Pros & Cons
Procedural Models

Pros	Cons
Quality Control	Detail
Interpretability	Variety
Manipulability	Ease

Pros & Cons **Deep Generative Models**		Pros & Cons **Procedural Models**	
Detail	Quality Control	Quality Control	Detail
Variety	Interpretability	Interpretability	Variety
Ease	Manipulability	Manipulability	Ease

Inverses of each other!

Can we get the best of both worlds?

Pros & Cons
Deep Generative Models

Detail	Quality Control
Variety	Interpretability
Ease	Manipulability

Pros & Cons
Procedural Models

Quality Control	Detail
Interpretability	Variety
Manipulability	Ease

Pros & Cons
Deep Generative Models

Detail

Variety

Ease

Pros & Cons
Procedural Models

Quality Control

Interpretability

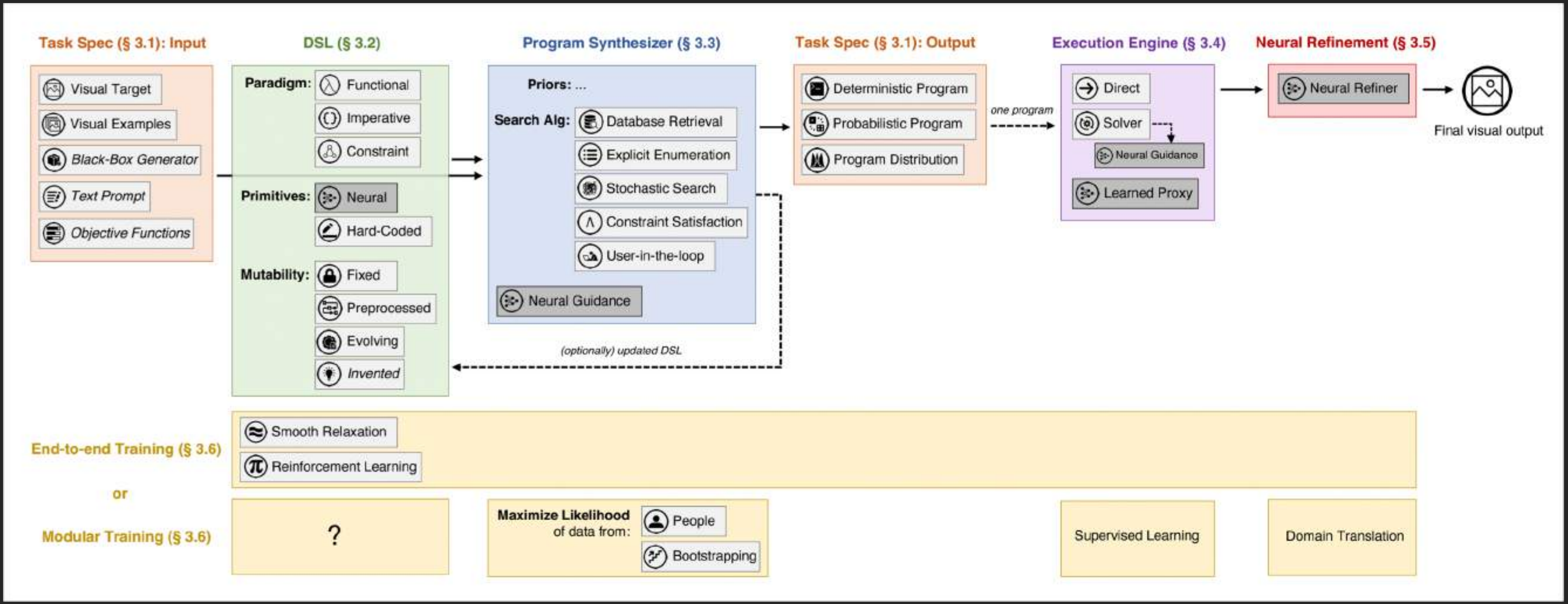
Manipulability

Neurosymbolic Models

Detail	Quality Control
Variety	Interpretability
Ease	Manipulability

Many, Many Ways to Combine Them...

Design space diagram from our Eurographics '23 State-of-the-Art Report (STAR)



[Ritchie et al. '23. Neurosymbolic Models for Computer Graphics]

Two Important Types of Combination

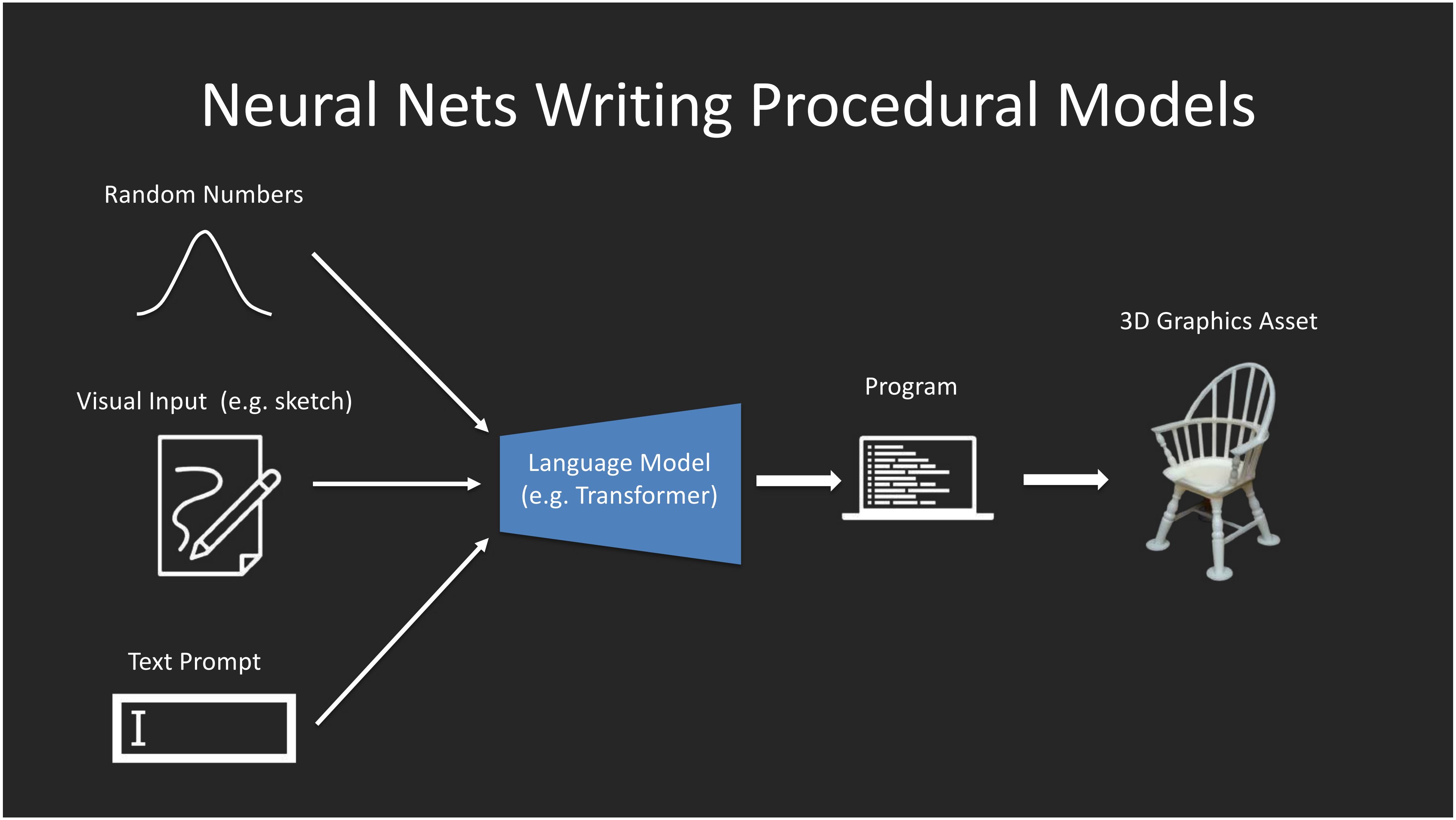
1. Using neural networks to write procedural models

2. Adding neural elements/details to procedural models

Two Important Types of Combination

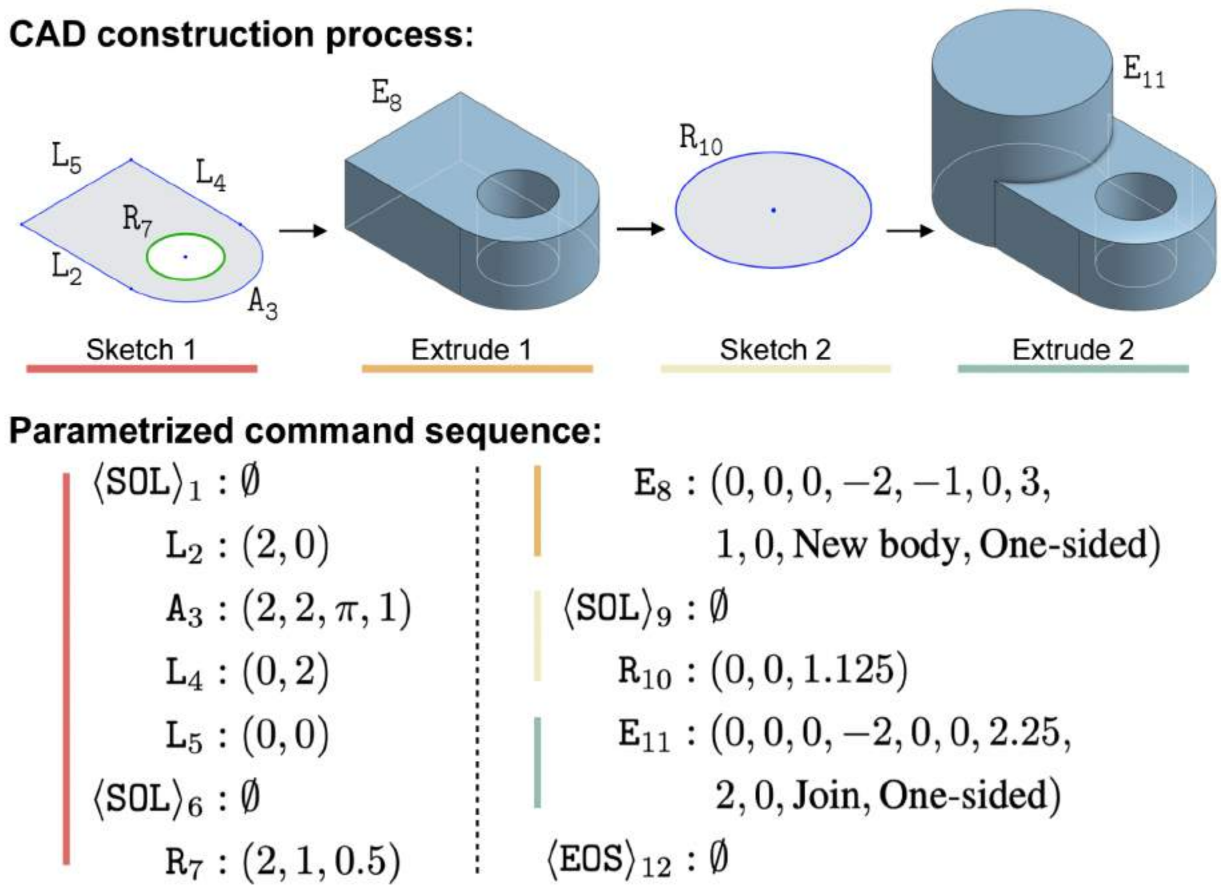
1. **Using neural networks to write procedural models**

2. Adding neural elements/details to procedural models


Neural Nets Writing Procedural Models
Random Numbers
Visual Input (e.g. sketch)
Text Prompt
Language Model
(e.g. Transformer)
Program
3D Graphics Asset

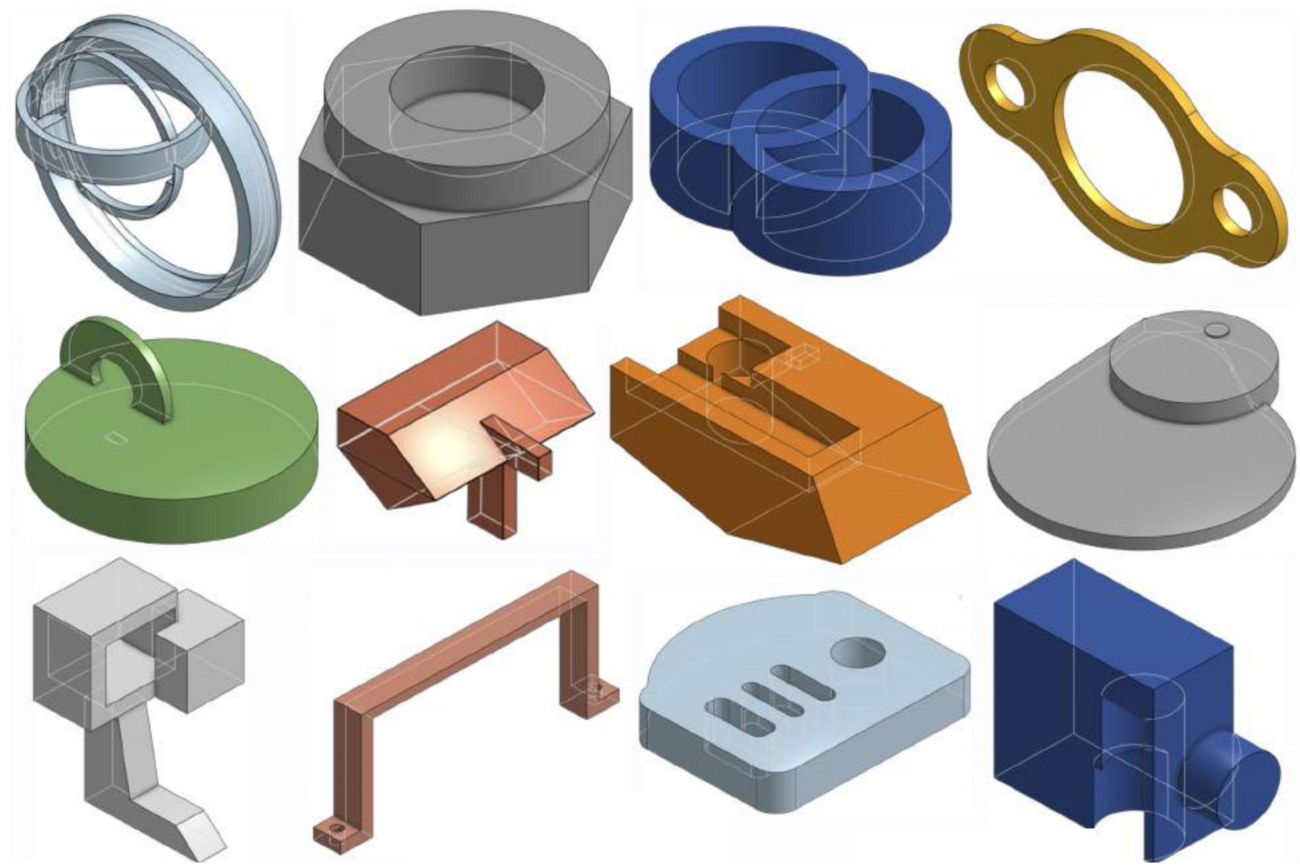
CAD Modeling

CAD Models as Programs



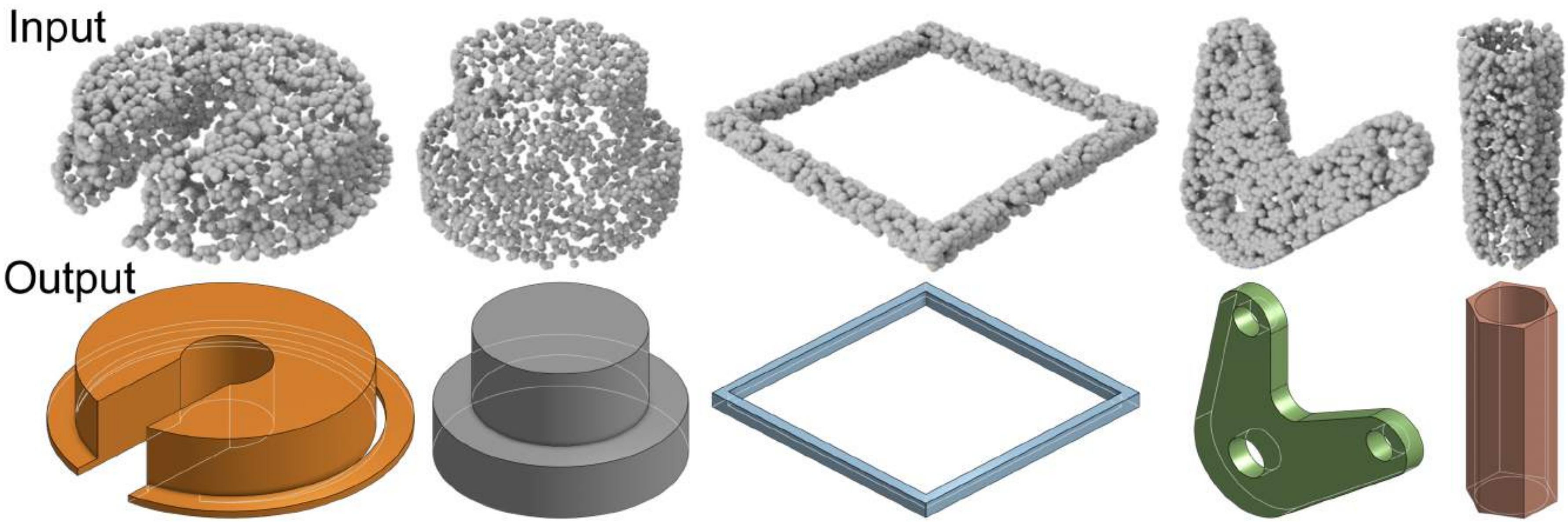
[Wu et al. ’21. DeepCAD]

Randomly Sampling CAD Programs



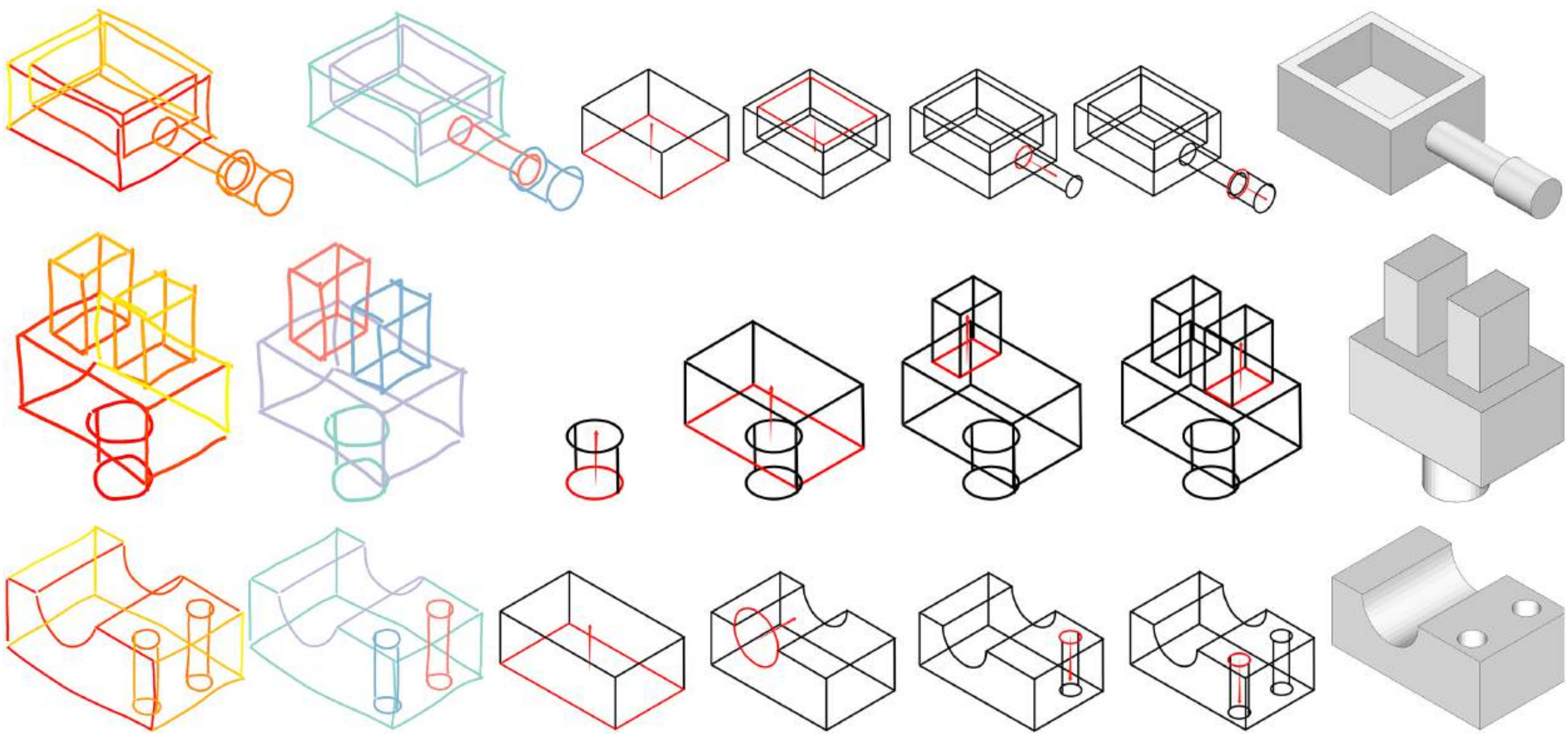
[Wu et al. '21. DeepCAD]

Inferring CAD Programs from Point Clouds



[Wu et al. ’21. DeepCAD]

Inferring CAD Programs from Sketches



[Li et al. ’22. Free2CAD]

Shape Part Structures

Kenny Jones

Paul Guerrero

Niloy Mitra

Me

…

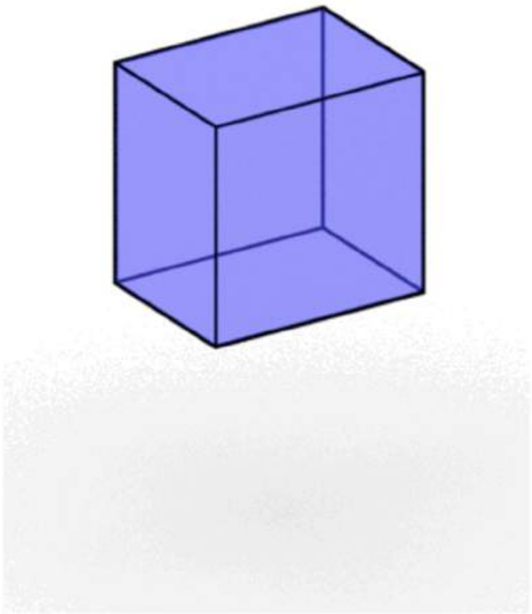
[Jones et al. ‘20. ShapeAssembly]

[Jones et al. ‘21. ShapeMOD]

[Jones et al. ‘23. ShapeCoder]

The ShapeAssembly Modeling Language

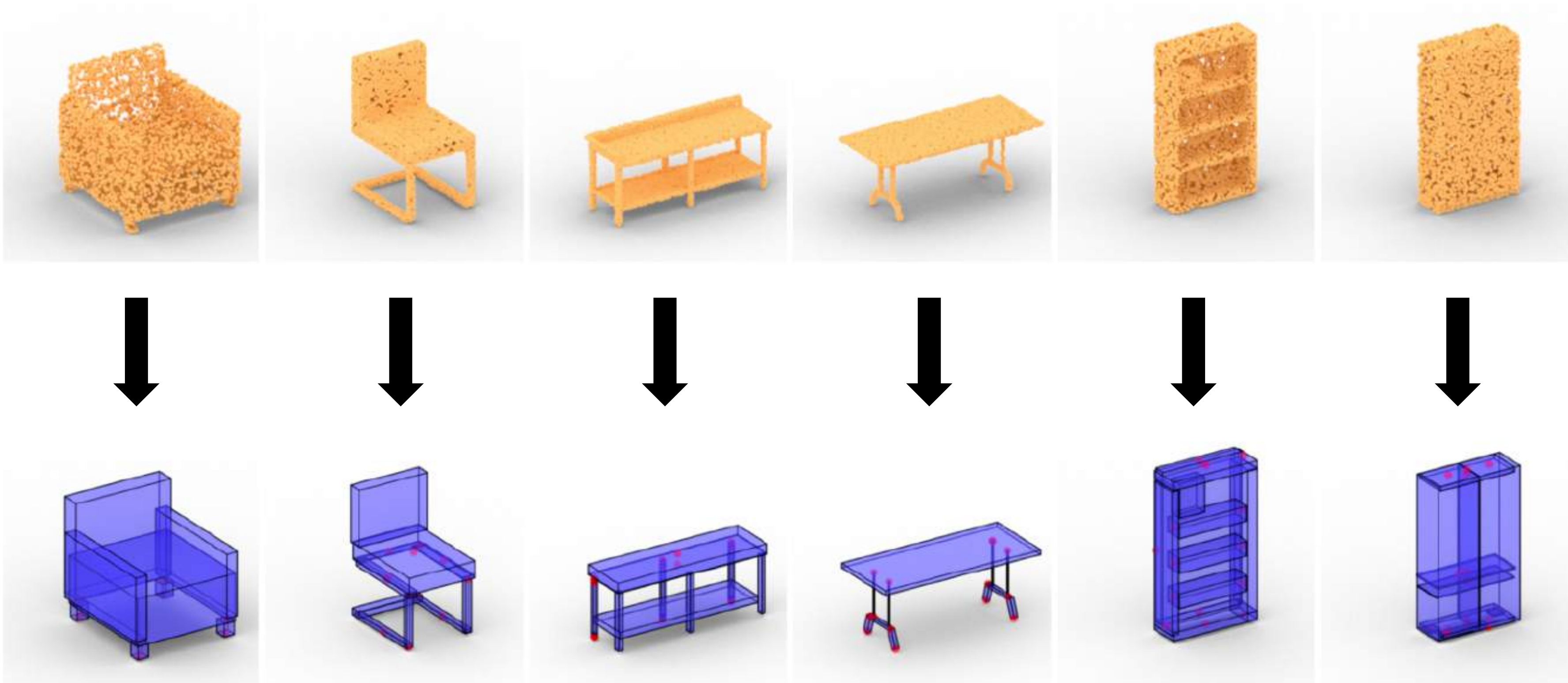
```
Assembly Program_0 {
    bbox = Cuboid(0.732, 1.742, 0.559, True)
    Program_1 = Cuboid(0.689, 0.672, 0.517, True)
}
```



Generating & Editing ShapeAssembly Code

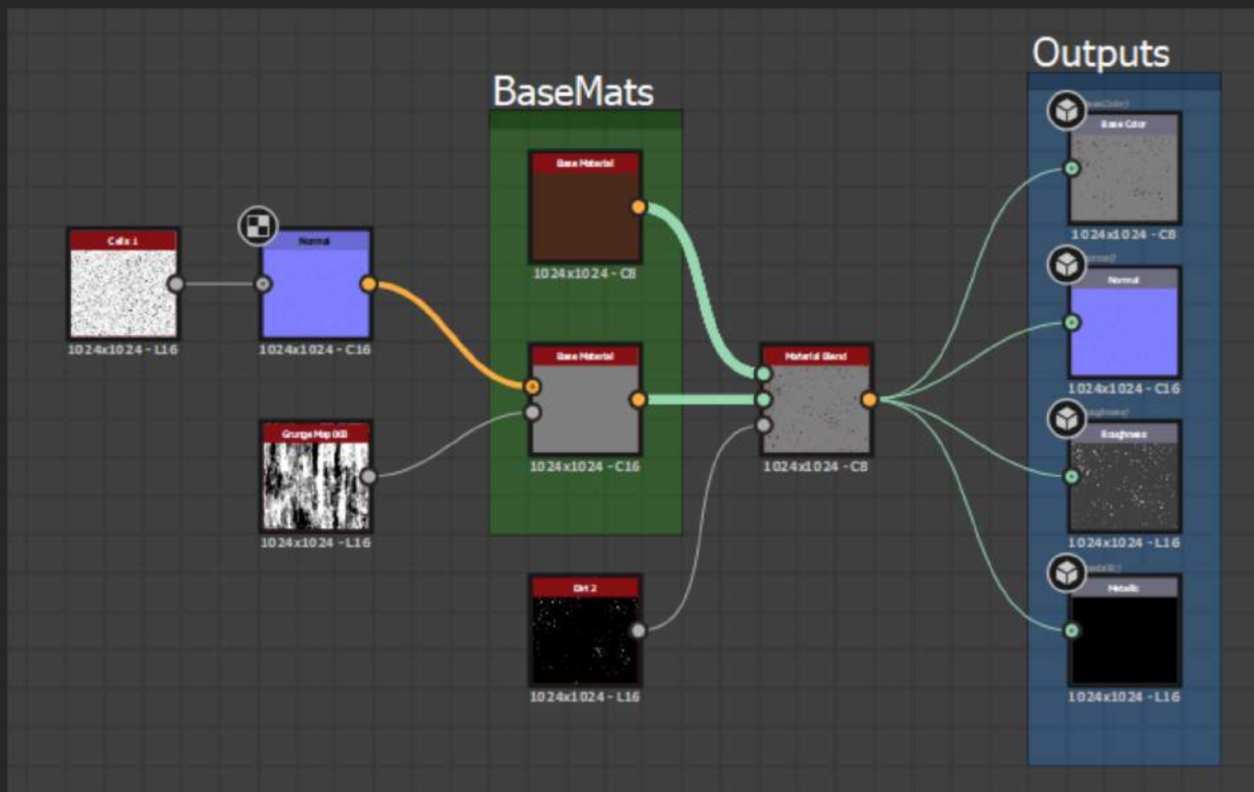
$\mathcal{N}(\mu,\sigma)$

Generating Programs from Point Clouds

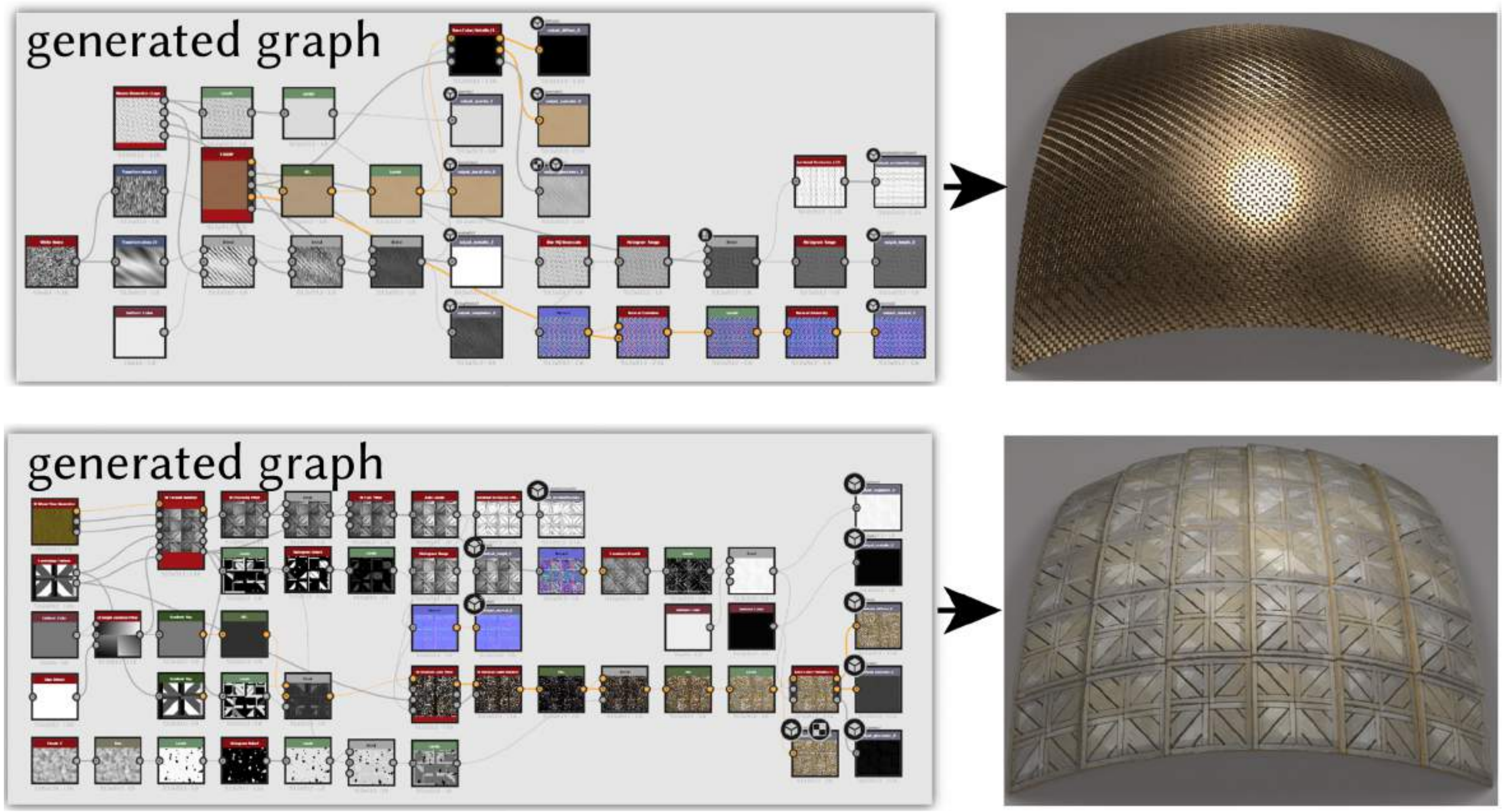


Procedural Materials

Materials Can Be Specified w/ Dataflow Graphs

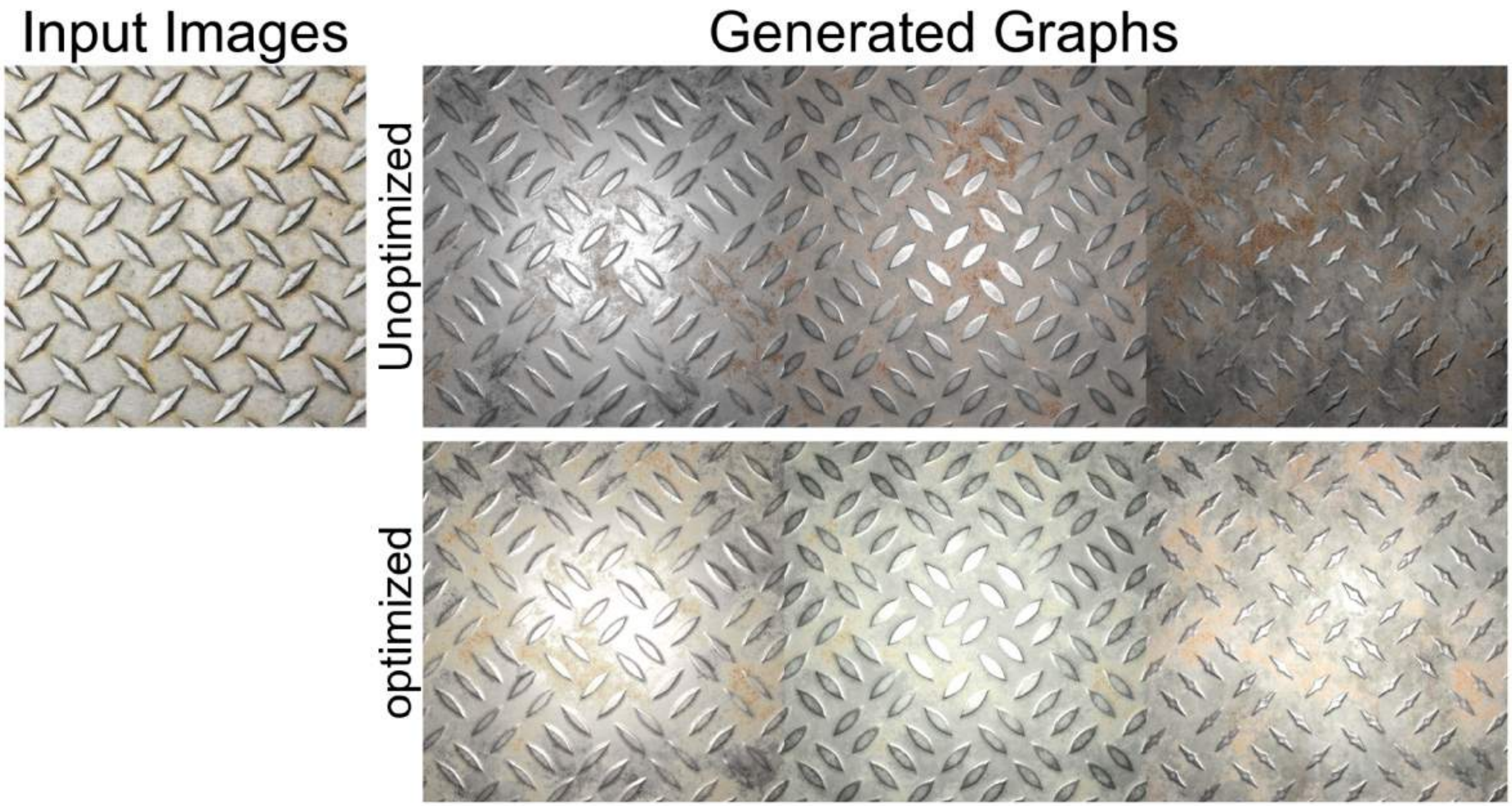


MatFormer Learns to Generate Graphs



[Guerrero et al. ’22. MatFormer]

MatFormer Generate Graphs *from Images*



[Hu et al. ’23. Generating Procedural Materials from Text or Image Prompts]

MatFormer Generate Graphs ***from Text***

"holiday wrapping paper"

"aged wood planks"

[Hu et al. ’23. Generating Procedural Materials from Text or Image Prompts]

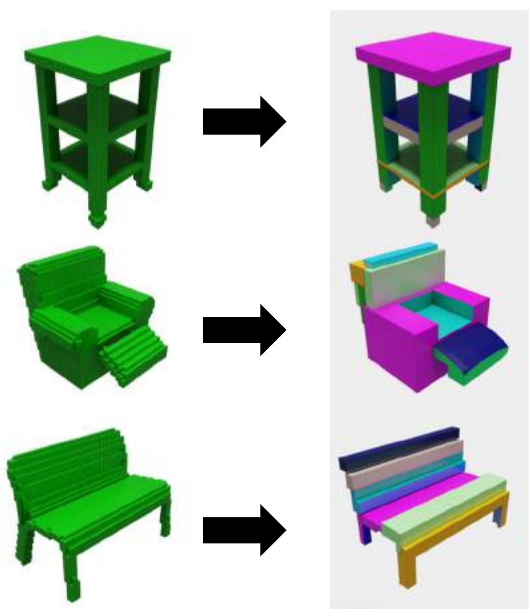
Wait a minute...

Procedural models are hard to write...

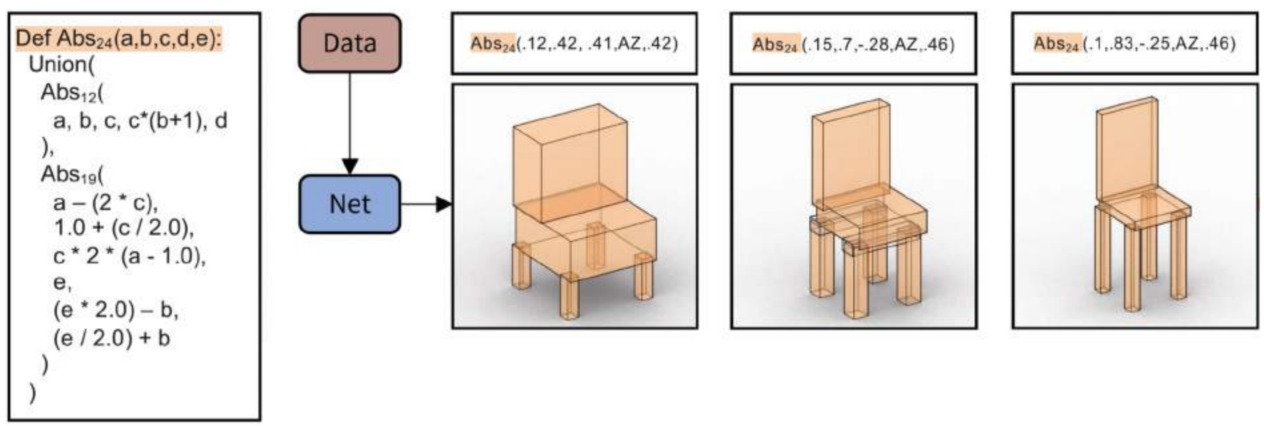
...so am I just kicking the can down the road by requiring large amounts of them as training data?

Important Ongoing Direction:
Learning ***without*** ground-truth programs

Bootstrapping on synthetic data



+ abstraction discovery (library learning)



[Ganeshan et al. ’23. Improving Unsupervised Visual Program Inference with Code Rewriting Families]

[Jones et al. ’23. ShapeCoder]

Exciting Future Work Direction:
Can LLMs help us write procedural models?

LLMs can write image editing programs...


```
Instruction: Hide Daniel Craig with 8) and Sean Connery with ;)
Program:
OBJ0=FaceDet(image=IMAGE)
OBJ1=Select(image=IMAGE, object=OBJ0, query='Daniel Craig', category=None)
IMAGE0=Emoji(image=IMAGE, object=OBJ1, emoji='smiling_face_with_sunglasses')
OBJ2=Select(image=IMAGE, object=OBJ0, query='Sean Connery', category: None)
IMAGE1=Emoji(image=IMAGE0, object=OBJ2, emoji='winking_face')
RESULT=IMAGE1
```

...could they also write shape-generating programs?

[Gupta & Kembhavi et al. '23. Visual Programming]

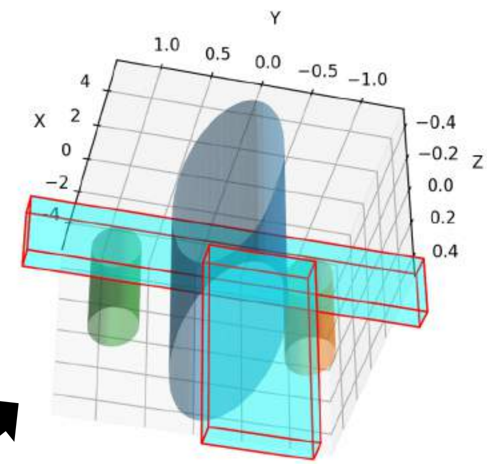
Exciting Future Work Direction:
Can LLMs help us write procedural models?

DA

You are an AI agent tasked with writing a procedural model which generates different kinds of airplanes. You are going to write the code in Python. The functions that you write should have parameters which control the most important geometric attributes of the airplane shape (e.g. body length, number of engines, wing size, etc.) The program should output geometry in the form of parameterized cuboids, cylinders, or other types of primitive shapes.

(+ a couple corrections...)

Far from perfect...but there's useful structure here!




```python
class Airplane:
    def __init__(self, body_length, body_radius, wing_width, tail_size, engine_radius):
        self.fuselage = Cylinder(np.array([body_length, body_radius, body_radius]), np.array([0, 0, 0]))
        self.wings = Cuboid(np.array([body_length / 10, wing_width, body_length / 30]), np.array([0, 0, 0]))
        self.tail = Cuboid(np.array([tail_size, tail_size, tail_size]), np.array([body_length/2, 0, body_radius]))
        self.engines = [Cylinder(np.array([body_length / 4, engine_radius, engine_radius]), np.array([-(body_length / 4), -body_radius, 0])),
                        Cylinder(np.array([body_length / 4, engine_radius, engine_radius]), np.array([-(body_length / 4), body_radius, 0]))]
```

Two Important Types of Combination

1. **Using neural networks to write procedural models**

2. Adding neural elements/details to procedural models

Two Important Types of Combination

1. Using neural networks to write procedural models

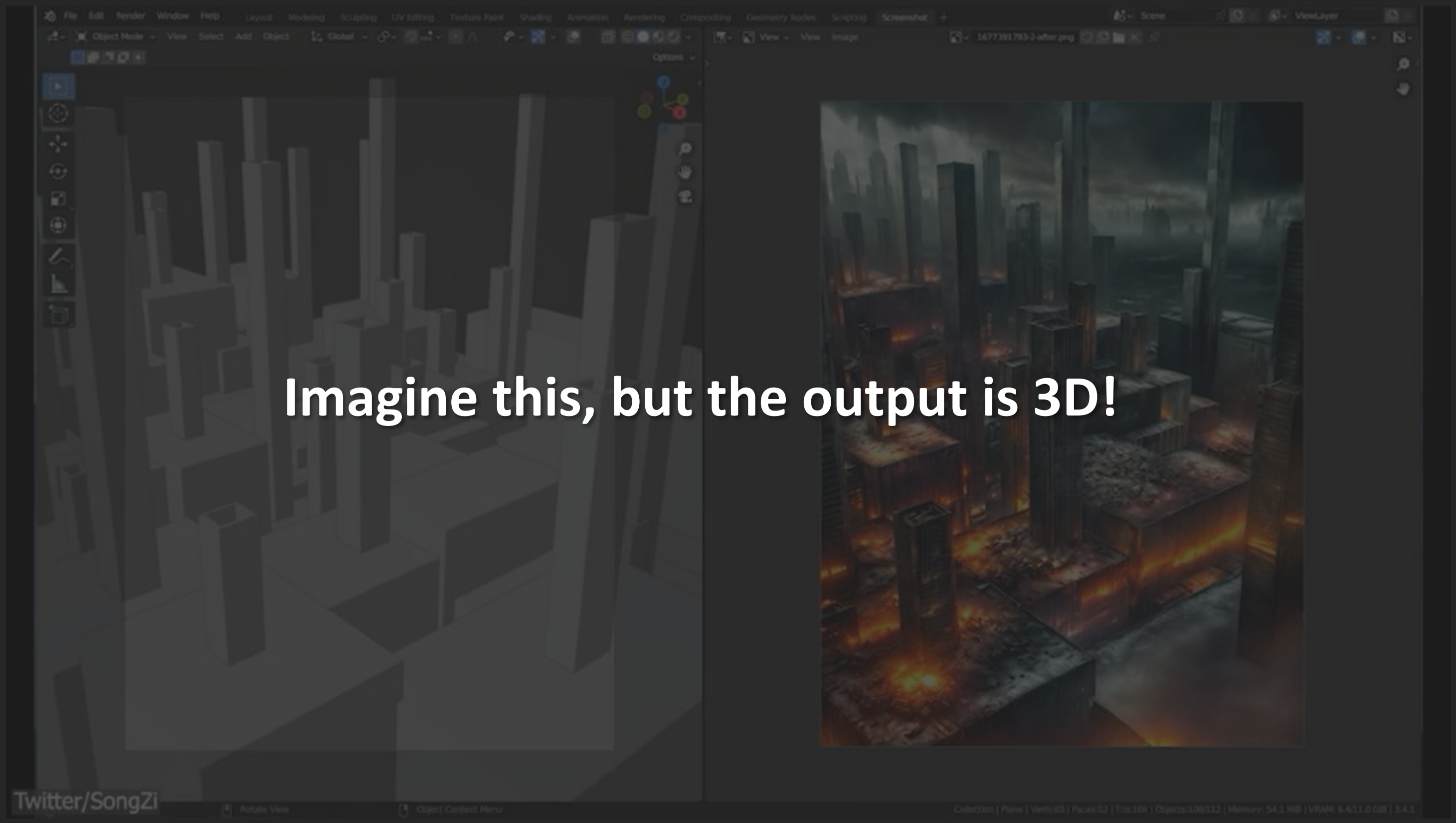
2. **Adding neural elements/details to procedural models**

Learning to Write Programs w/ Neural Primitives

[Deng et al. ’22. Unsupervised Learning of Shape Programs with Repeatable Implicit Parts]

Exciting Future Work Direction:
Can we learn *parametric* neural primitives?

Exciting Future Work Direction:
Neural details as (guided) post-process



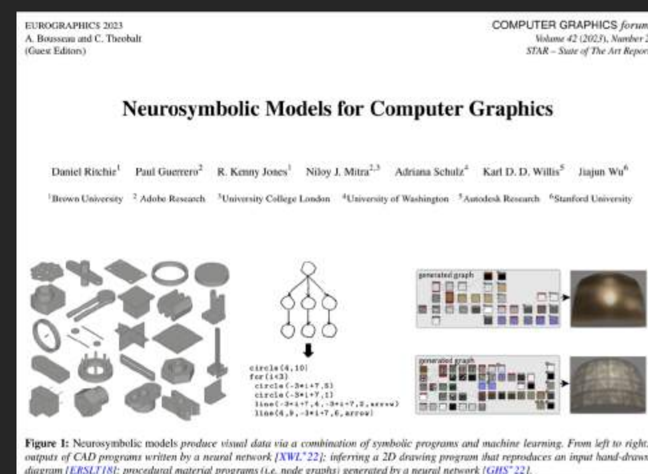
Thanks!

Want to talk more about this stuff? Collaborate? Feel free to reach out :)

https://dritchie.github.io
daniel_ritchie@brown.edu

Link to our state-of-the-art report on neurosymbolic models for graphics:

https://tinyurl.com/neurosymbolicstar



EUROGRAPHICS 2023
A. Bousseau and C. Theobalt
(Guest Editors)

COMPUTER GRAPHICS *forum*
Volume 42 (2023), Number 2
STAR – State of The Art Report

Neurosymbolic Models for Computer Graphics

Daniel Ritchie[1] Paul Guerrero[2] R. Kenny Jones[1] Niloy J. Mitra[2,3] Adriana Schulz[4] Karl D. D. Willis[5] Jiajun Wu[6]

[1]Brown University [2]Adobe Research [3]University College London [4]University of Washington [5]Autodesk Research [6]Stanford University

Figure 1: Neurosymbolic models *produce visual data via a combination of symbolic programs and machine learning. From left to right: outputs of CAD programs written by a neural network [XWL*22]; inferring a 2D drawing program that reproduces an input hand-drawn diagram [ERSLT18]; procedural material programs (i.e. node graphs) generated by a neural network [GHS*22].*

Geometric Learning on Discrete Surface Meshes

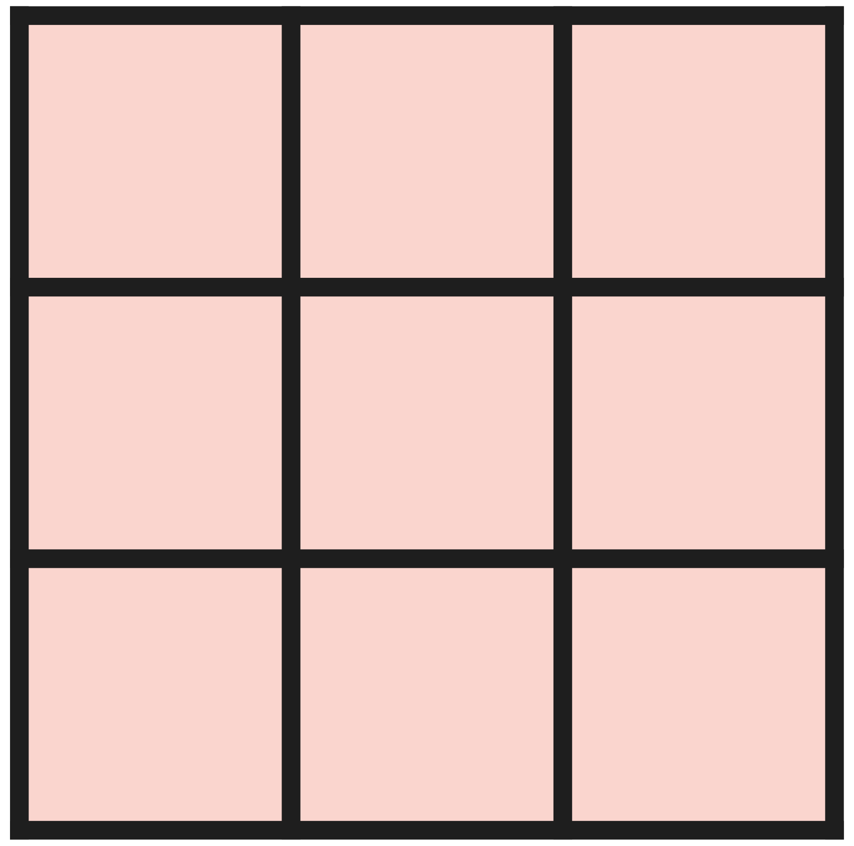
Surface Convolutions

Hsueh-Ti Derek Liu

ROBLOX

Image Convolution

Image Convolution



filter

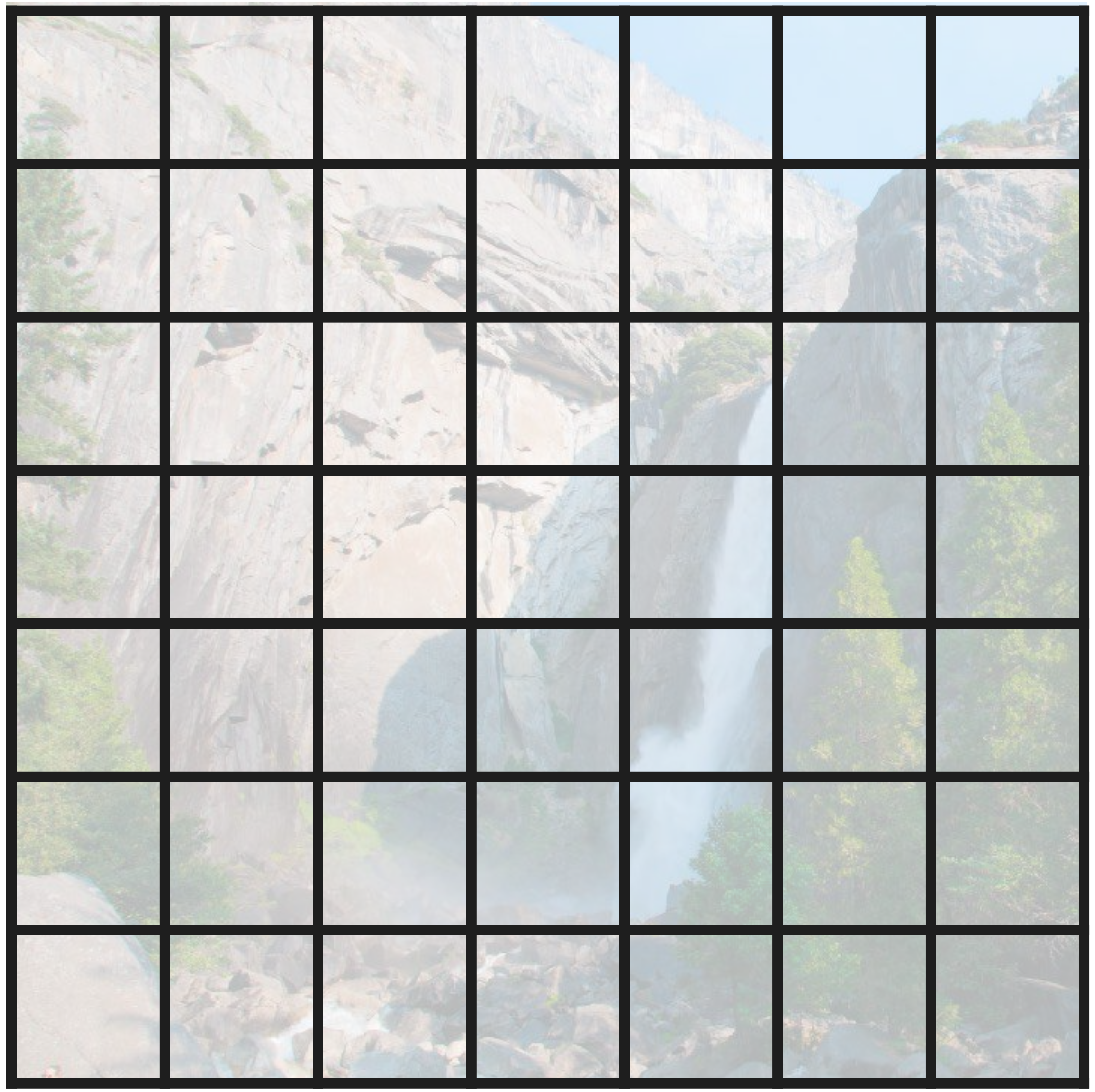


Image Convolution

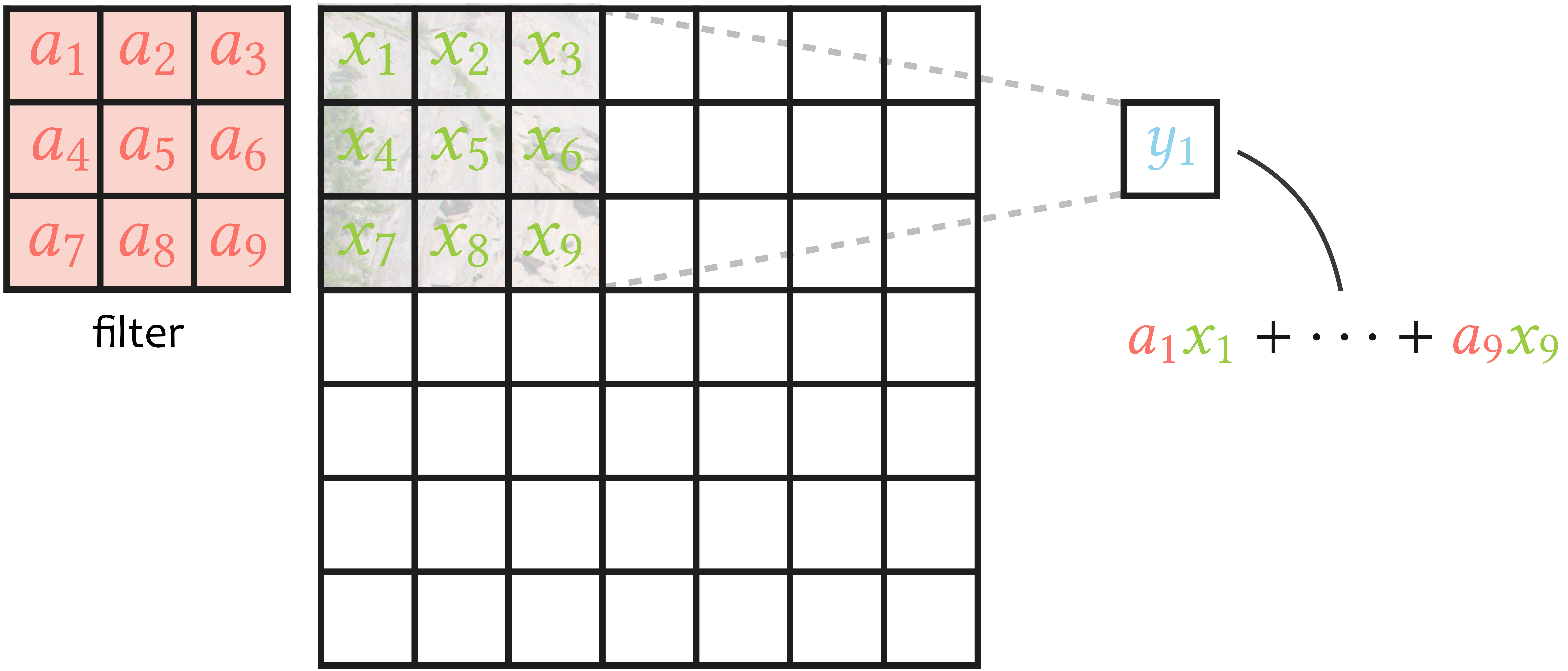


Image Convolution

a_1	a_2	a_3
a_4	a_5	a_6
a_7	a_8	a_9

filter

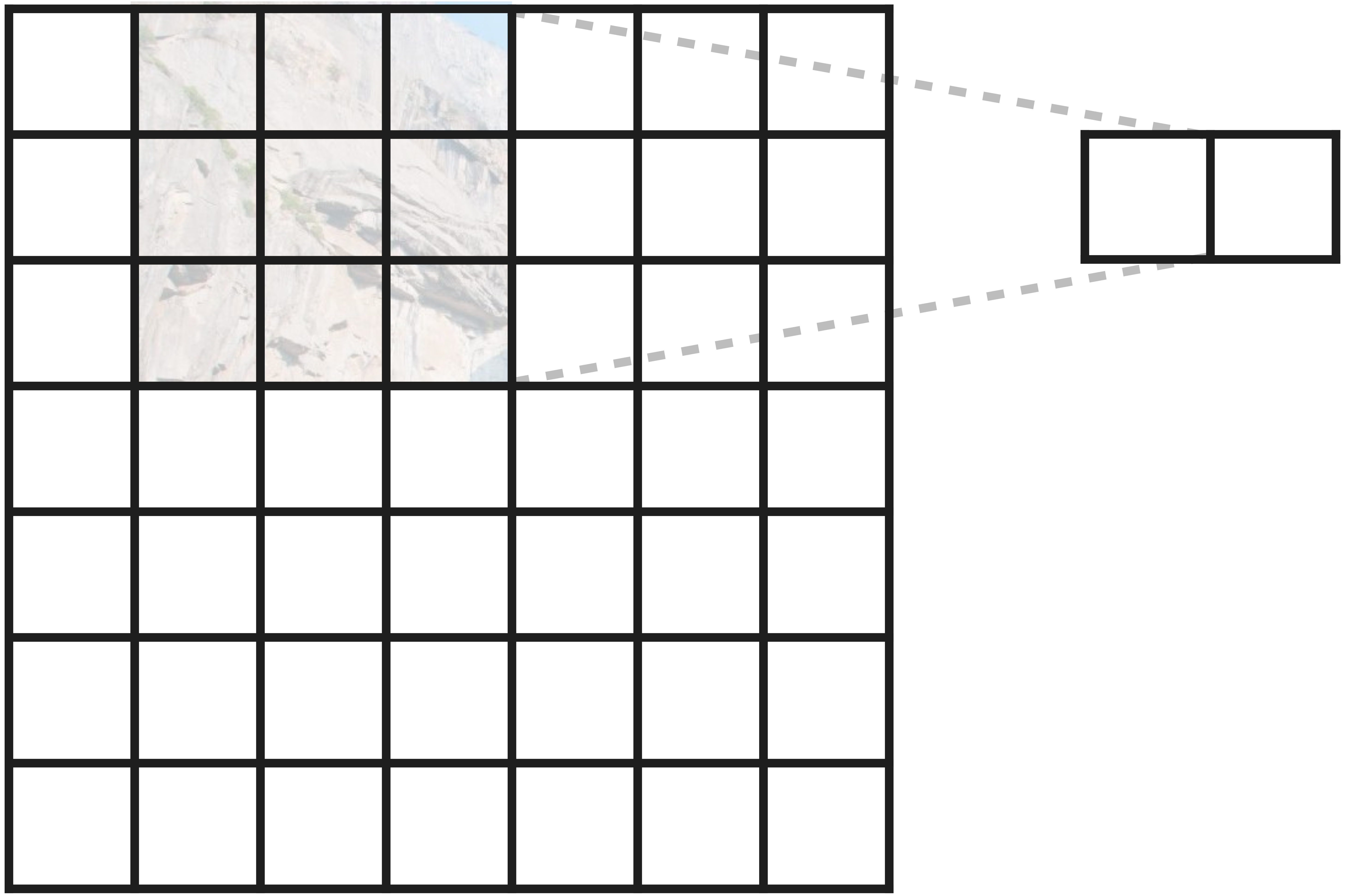


Image Convolution

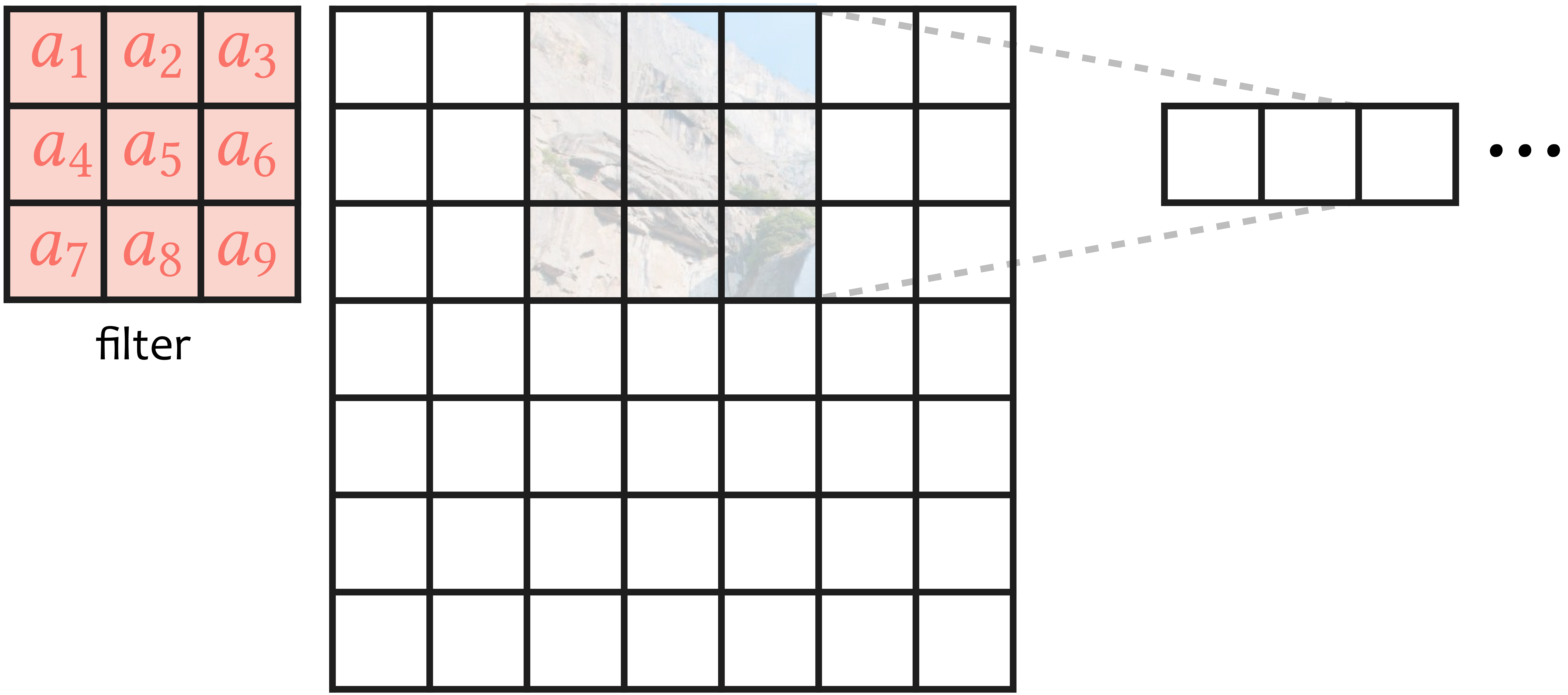
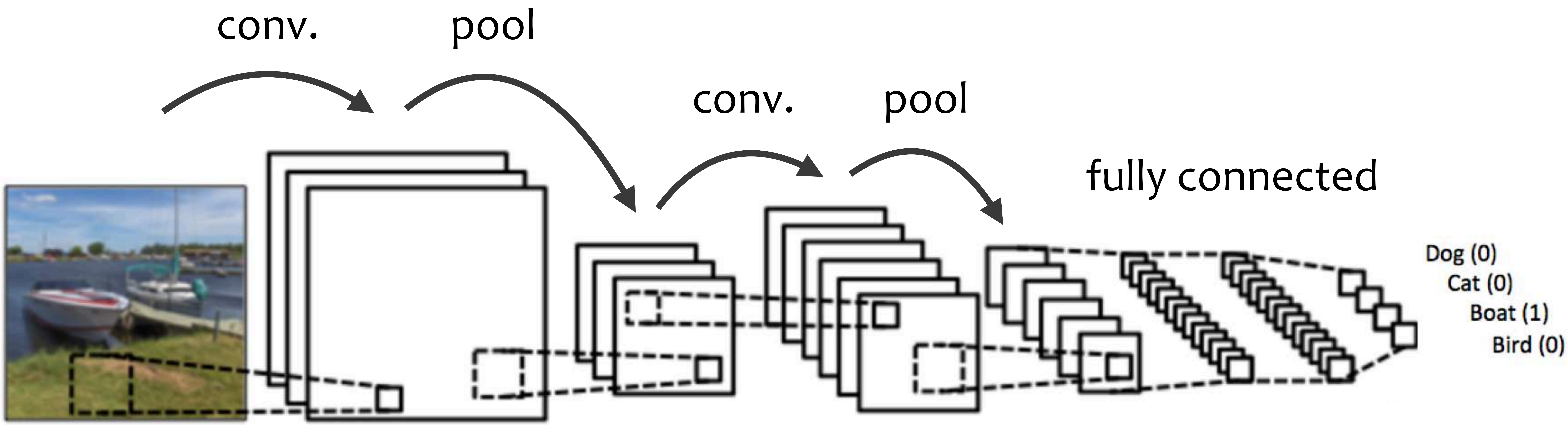
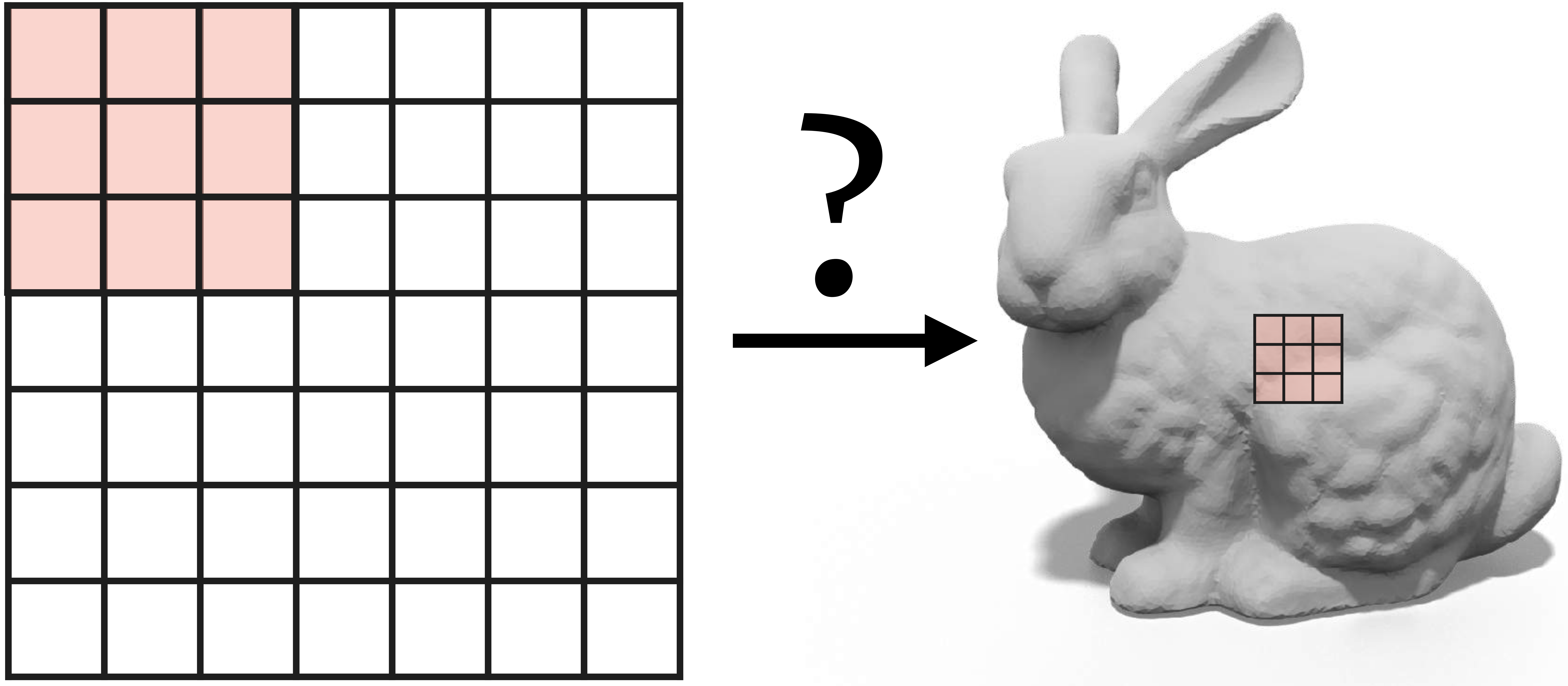


Image Convolutional Neural Networks

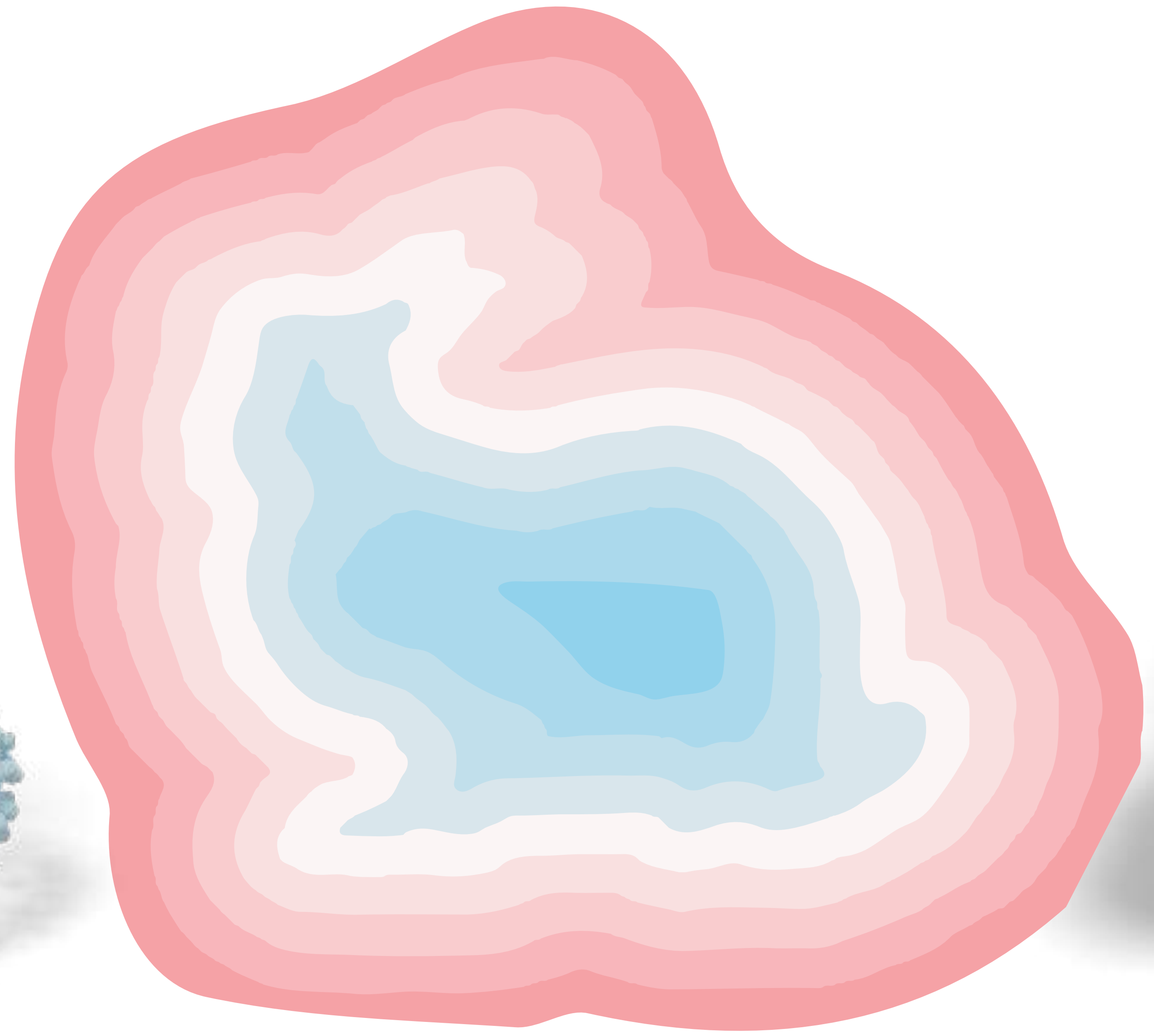


Convolution on Surface Meshes

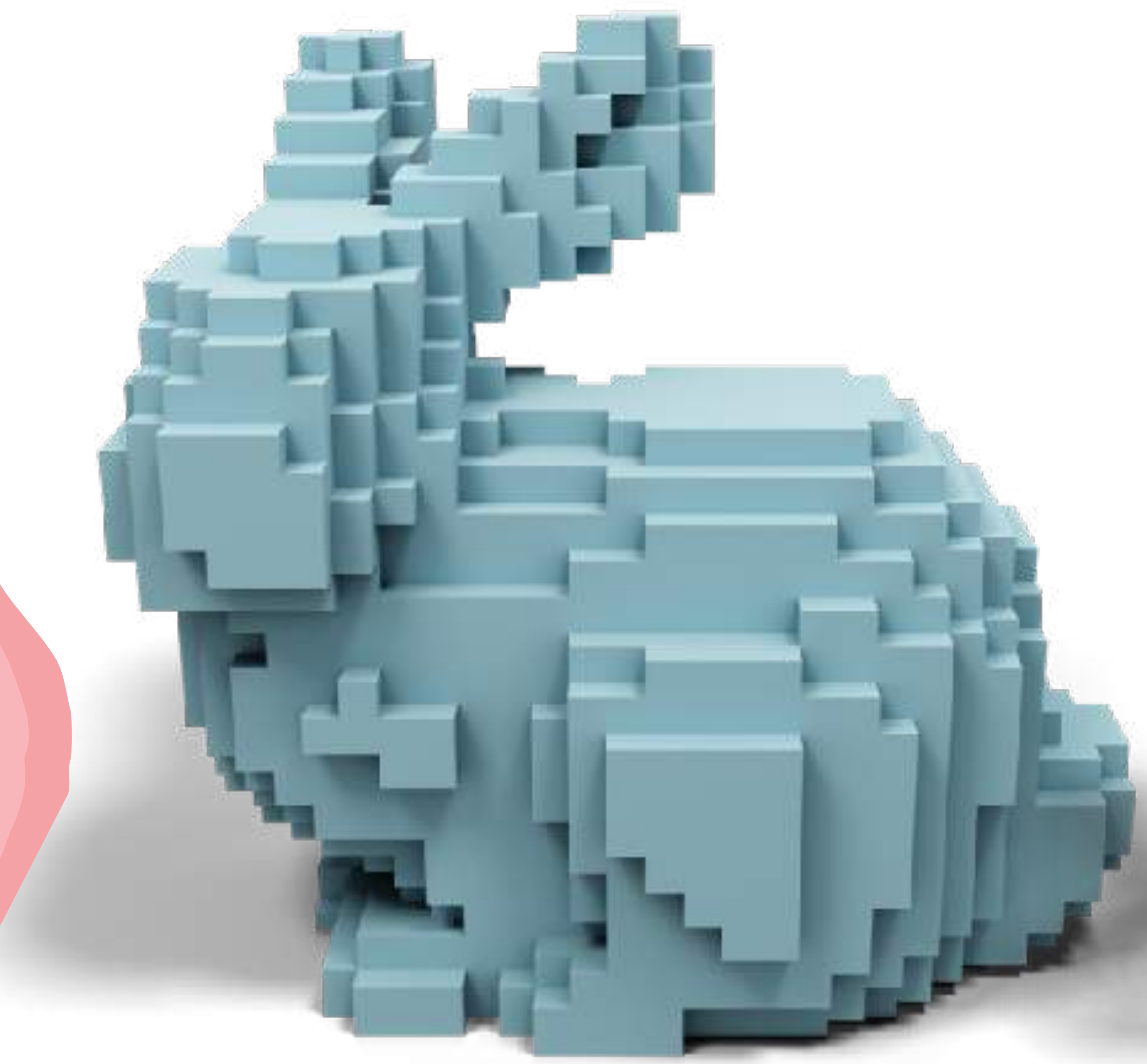


Other Shape Representations

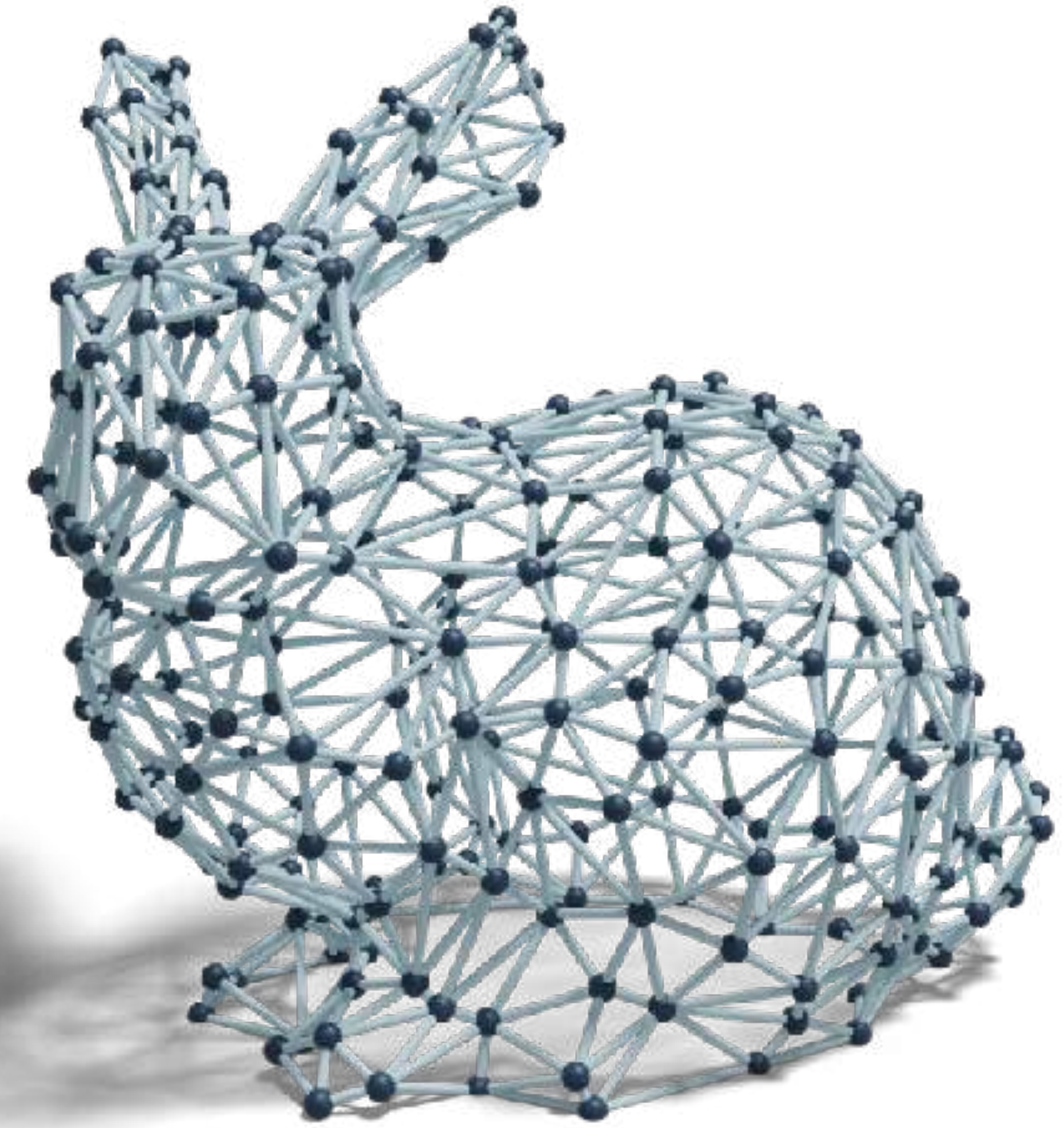
point cloud



implicits

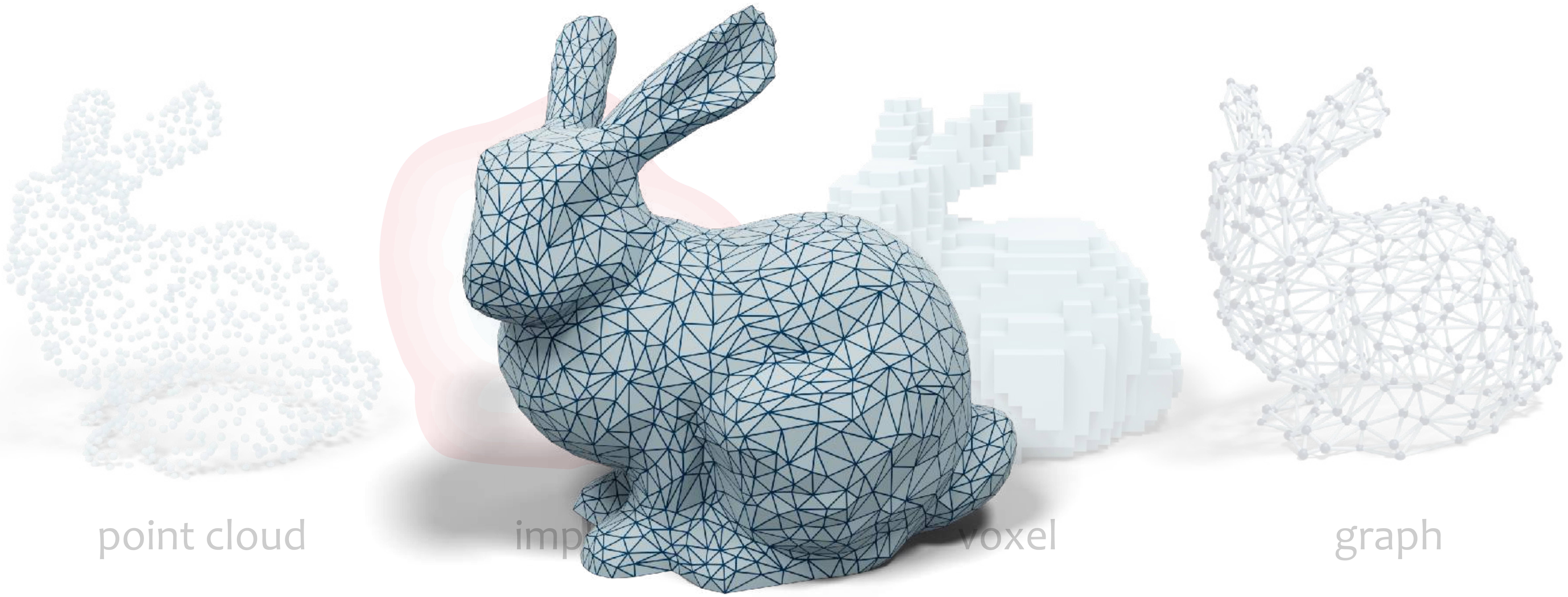


voxel



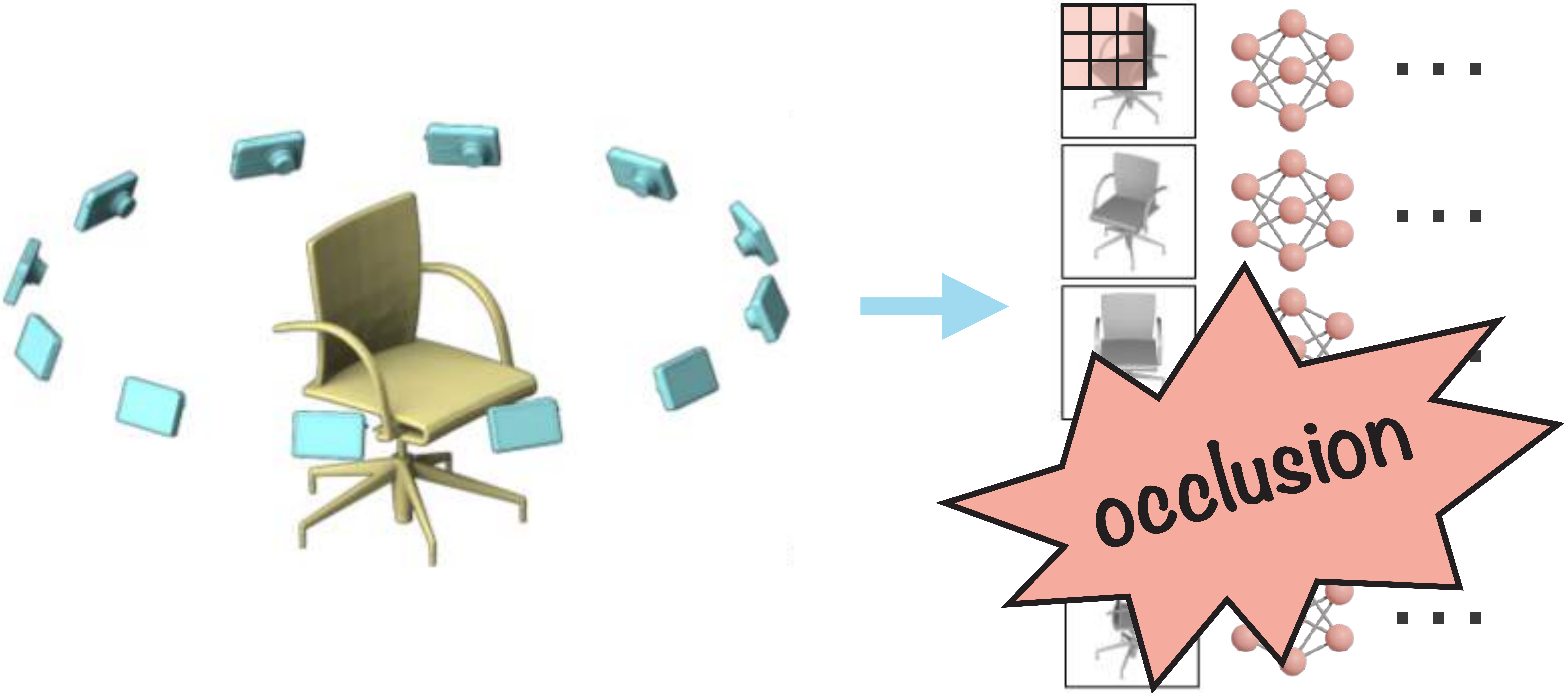
graph

Triangle Meshes



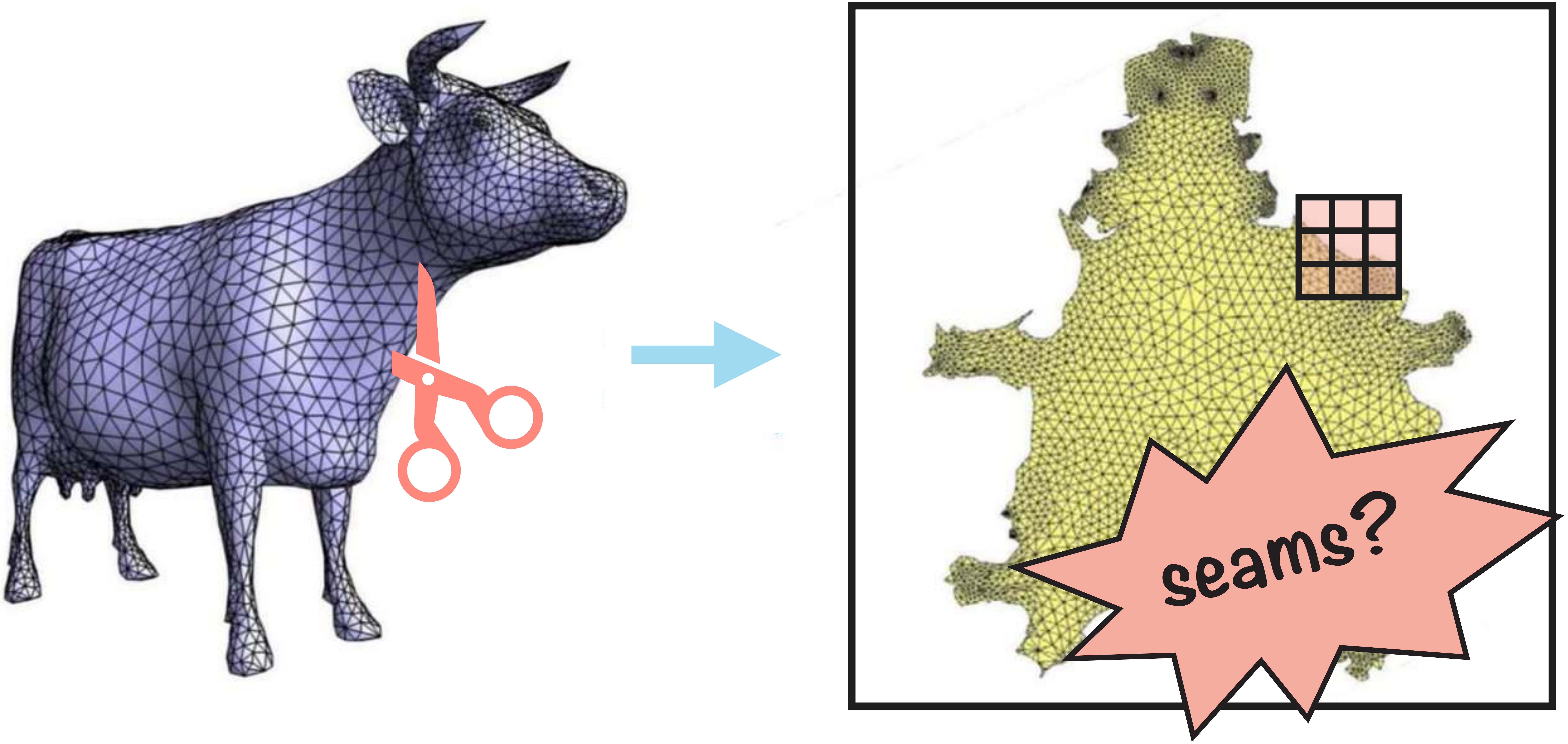
History on Surface Mesh Convolution

One of the first ideas: Image Convolution

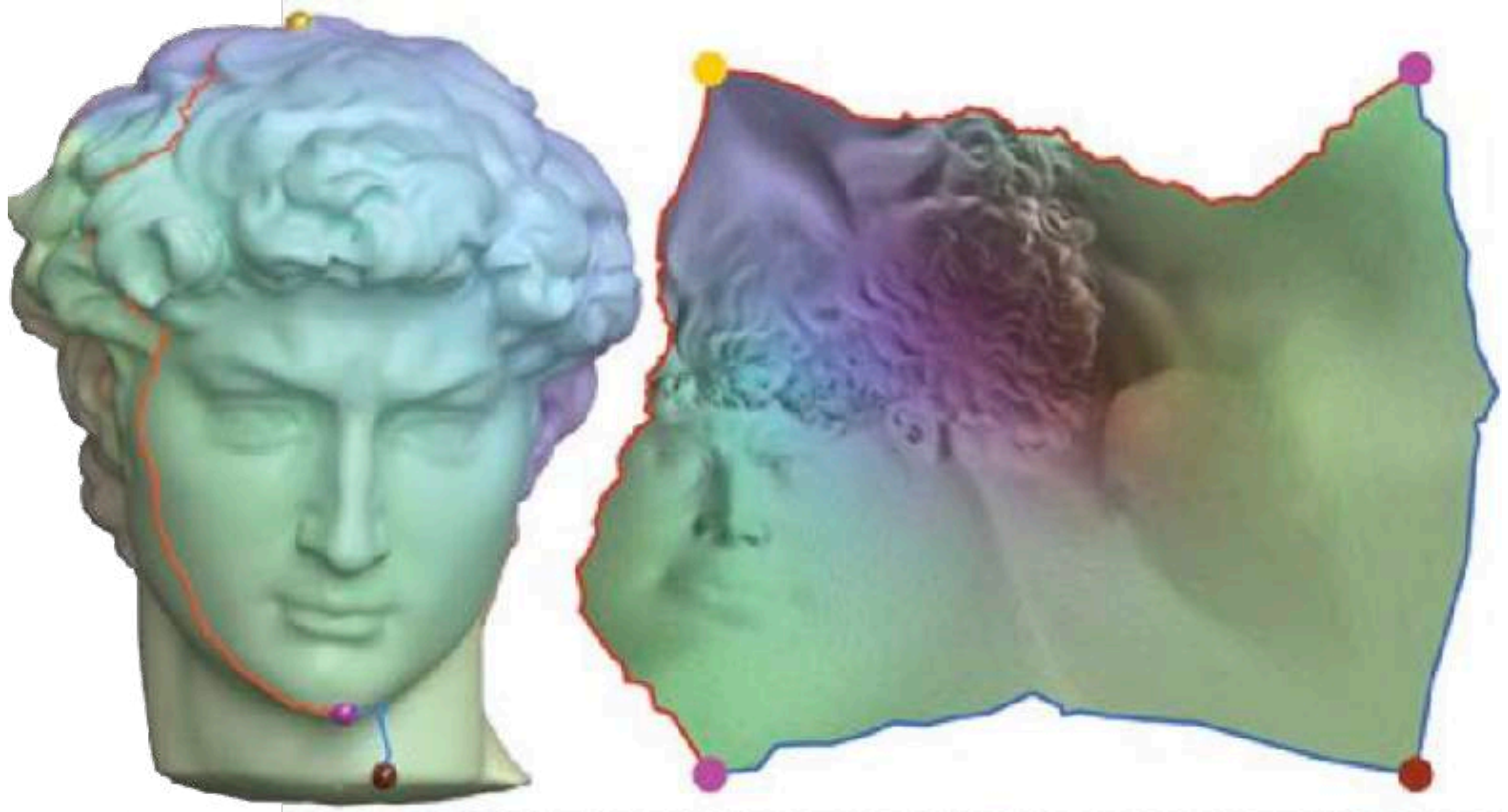


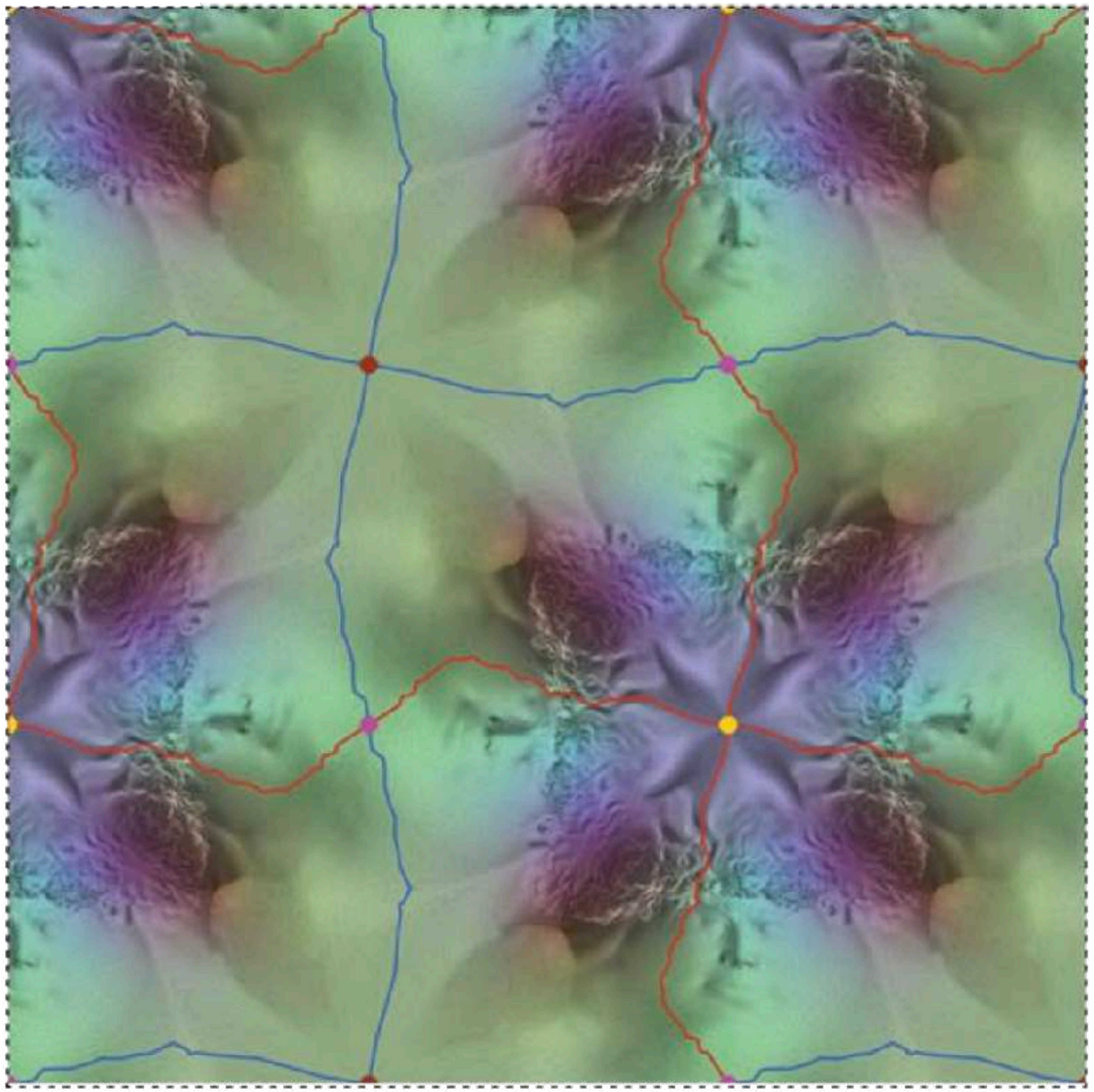
Su et al. 2015

Global Parameterization



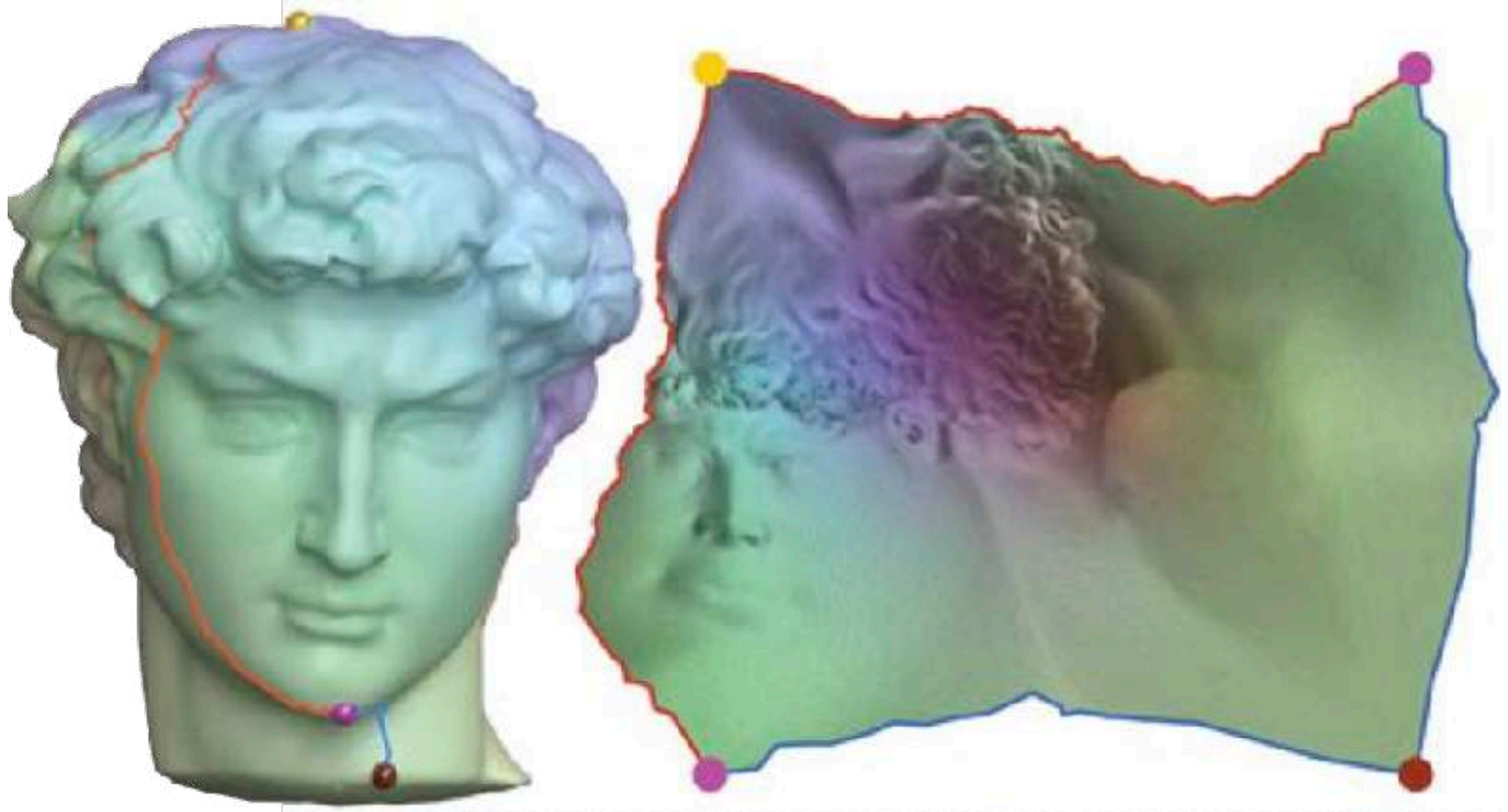
Global Seamless Parameterization

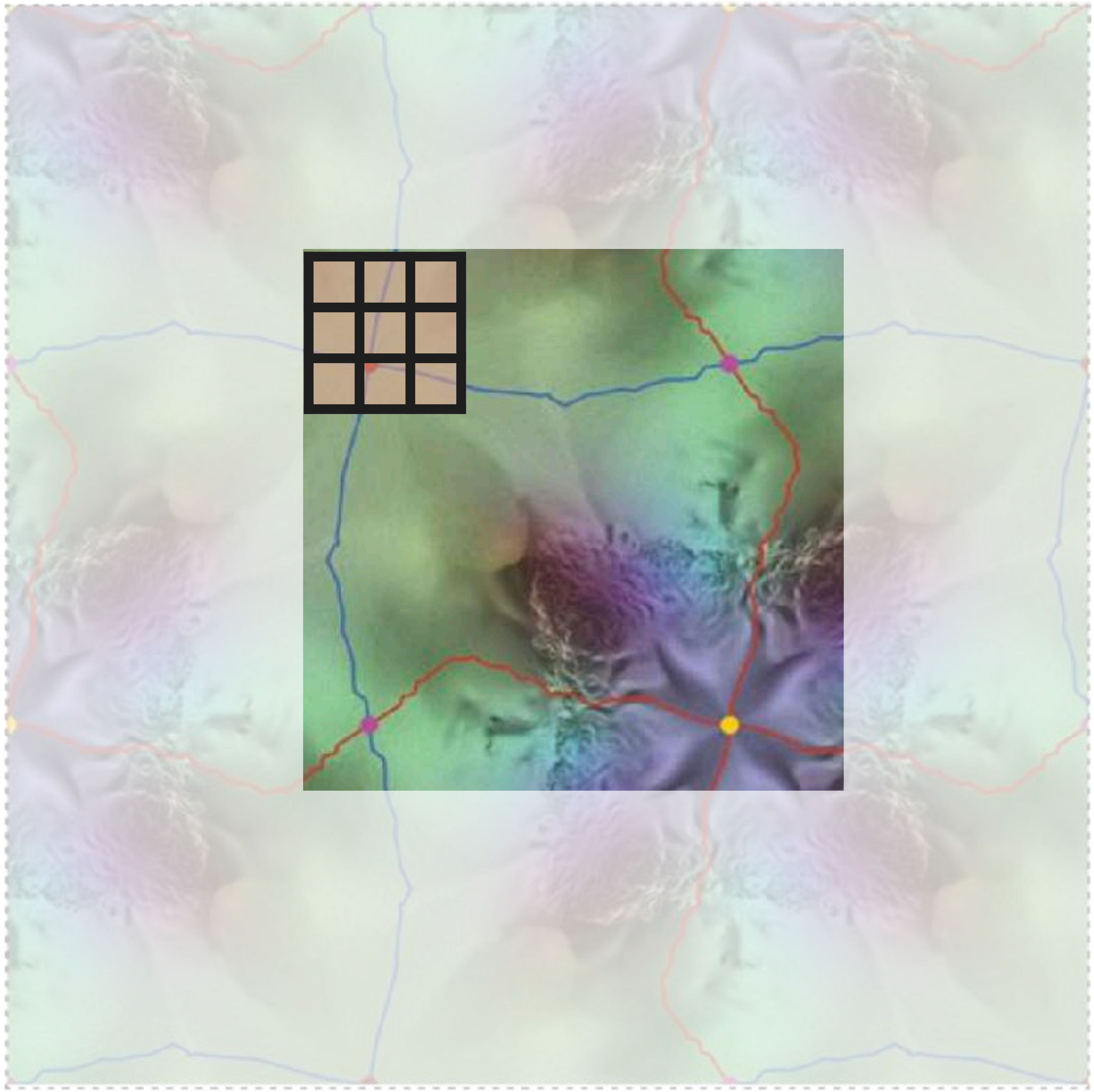




Aigerman & Lipman 2015

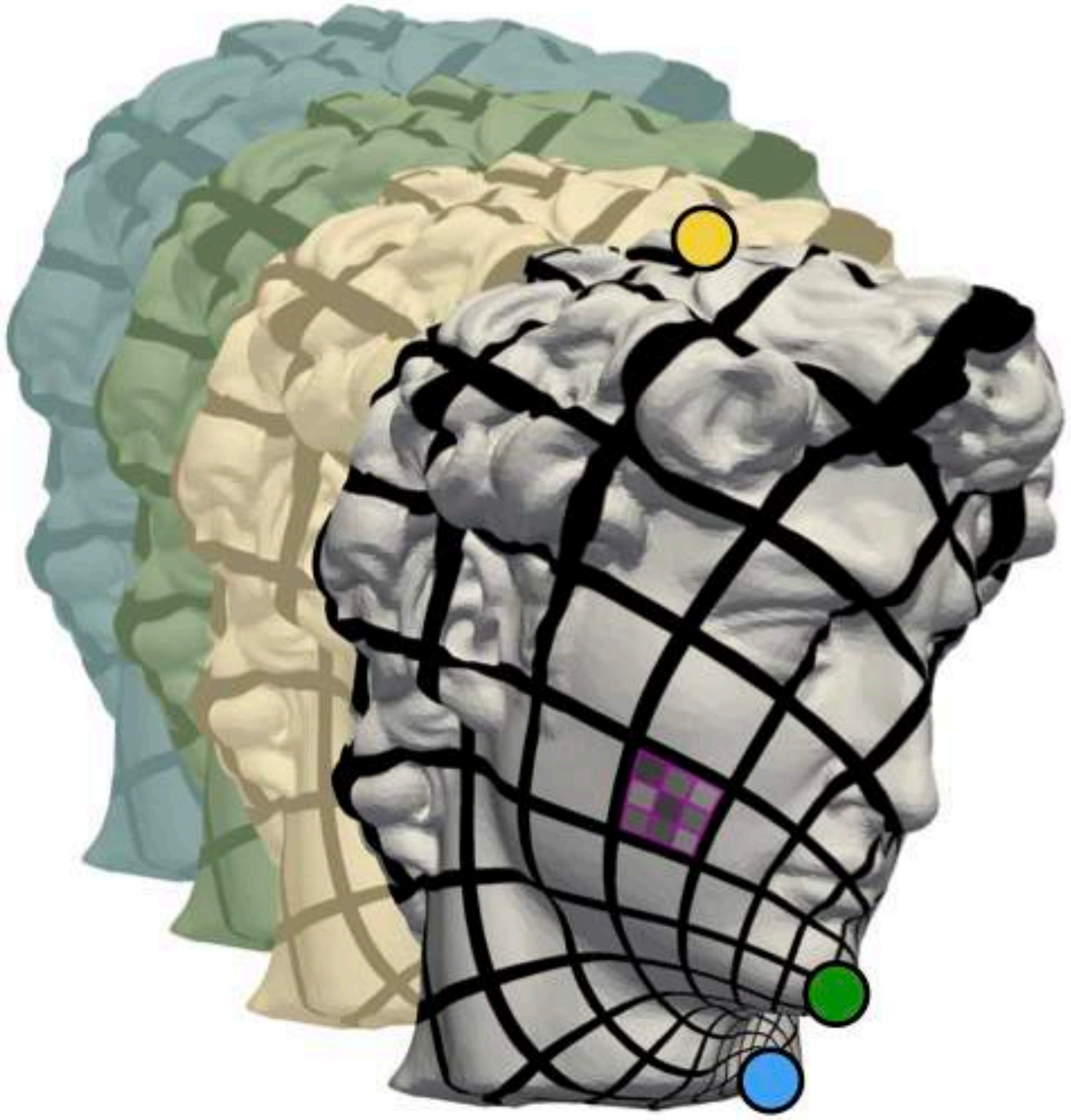
Global Seamless Parameterization





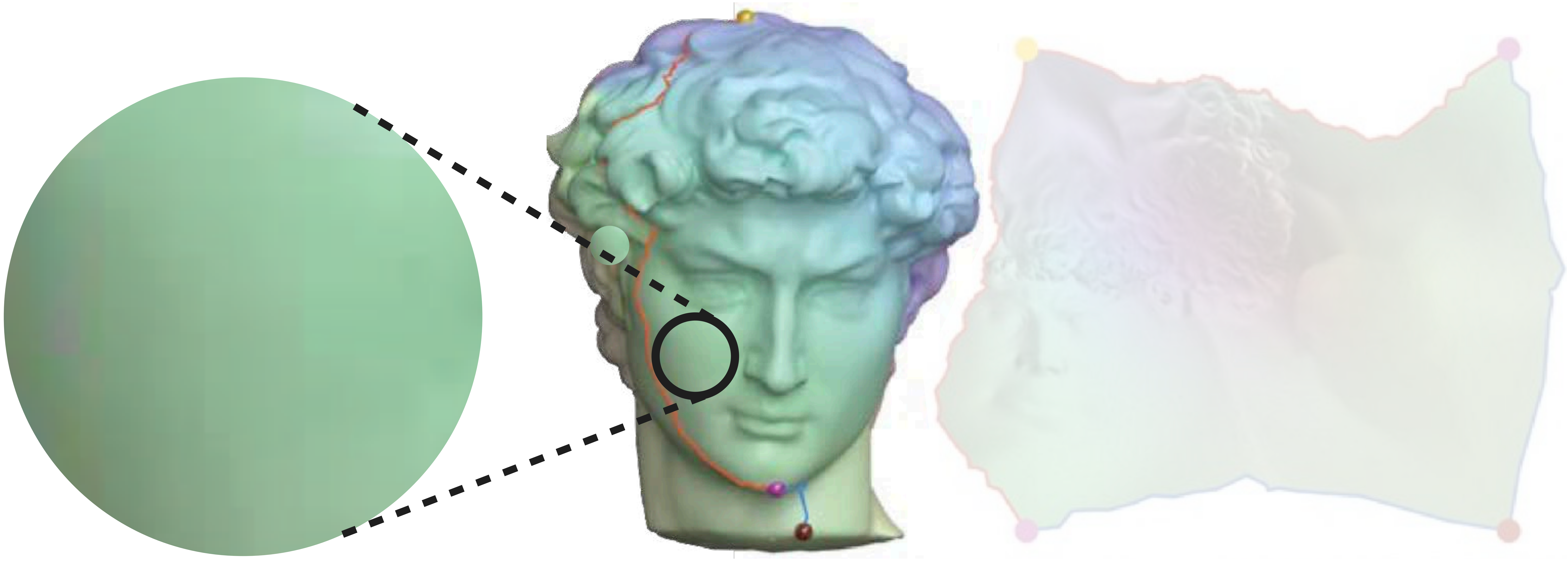
Aigerman & Lipman 2015

Global Seamless Parameterization

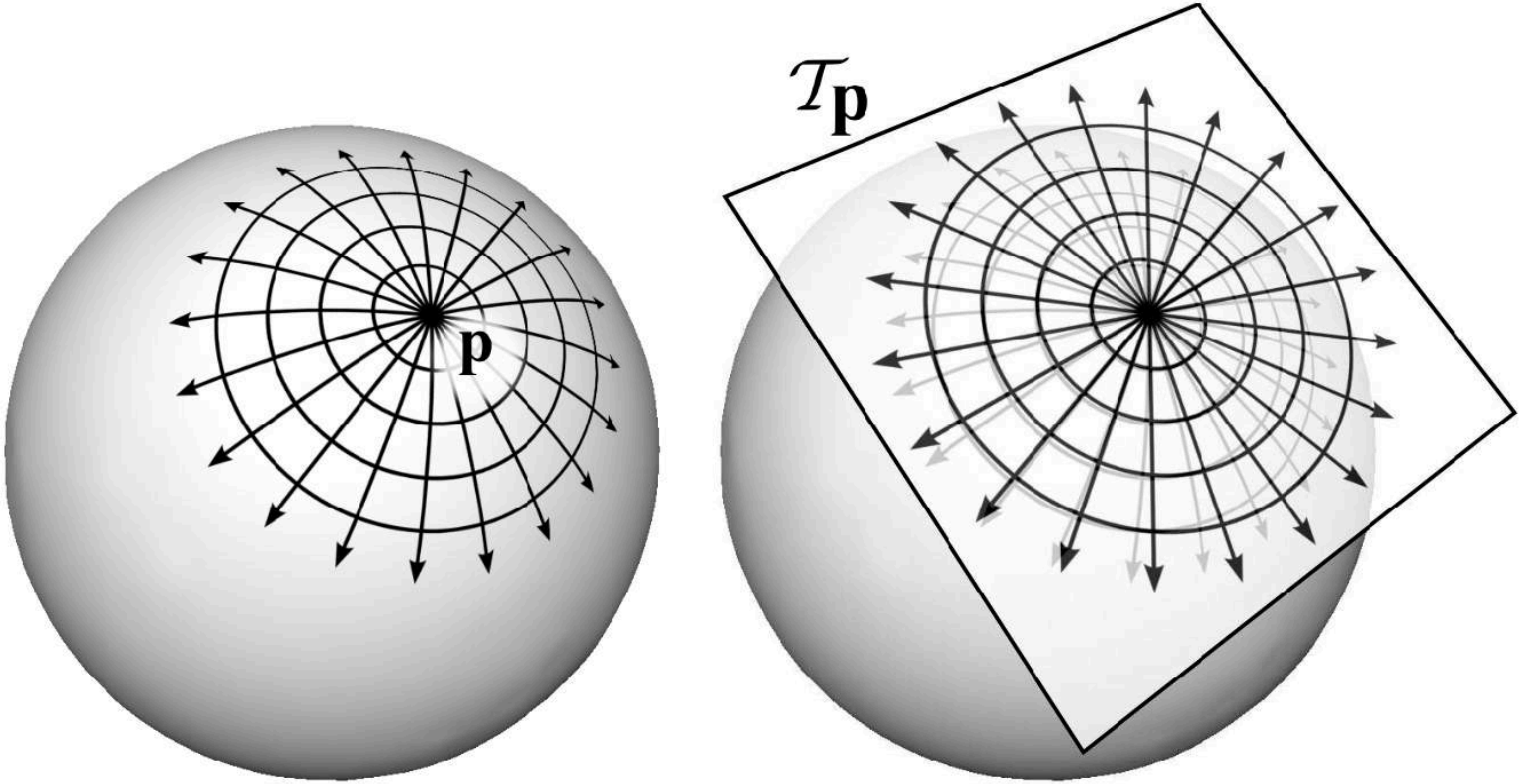


- Not unique
- Distortion
- Other issues (e.g., orientation)

Local Flattening

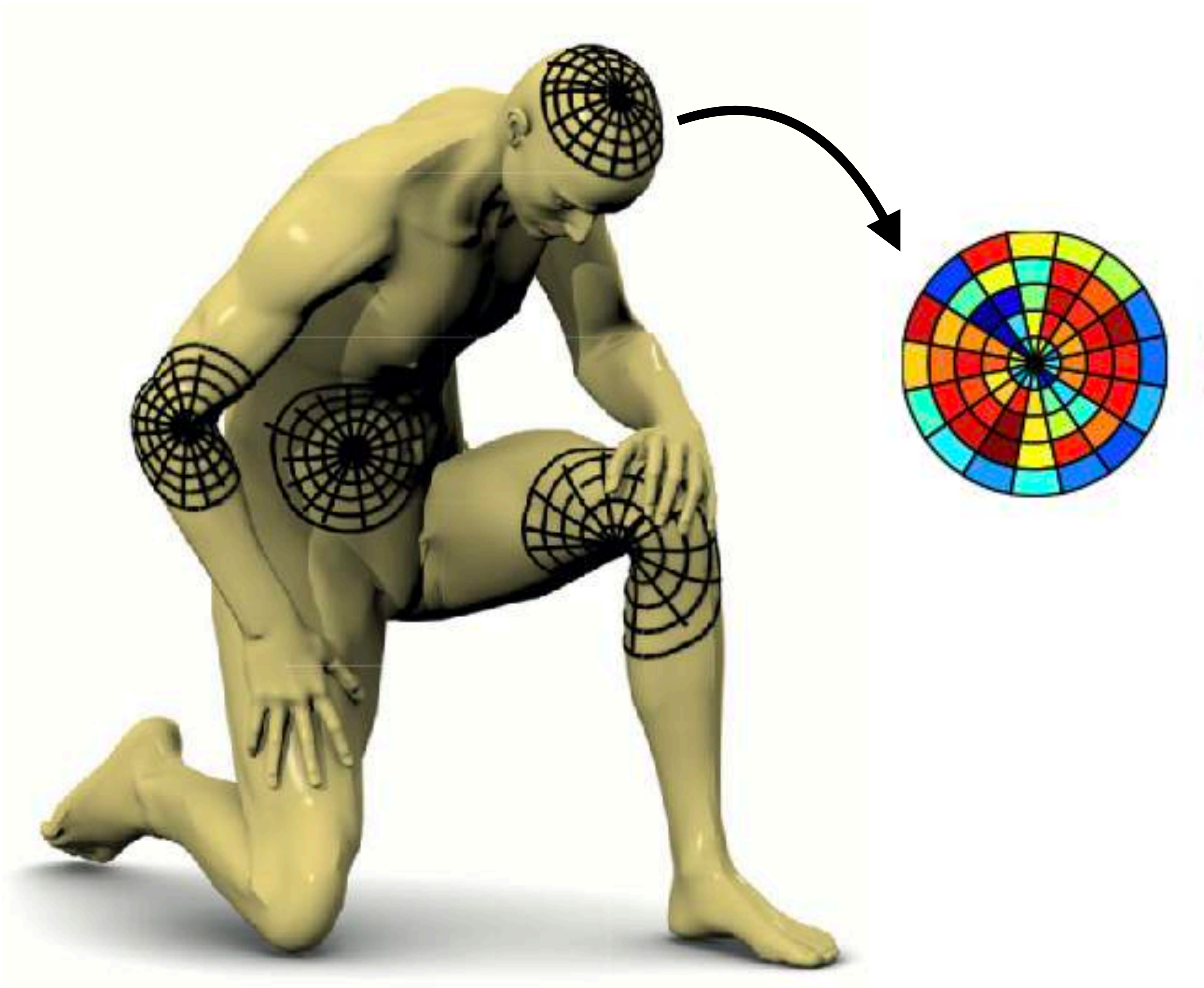


Logarithmic Maps (a.k.a. Exponential Maps)



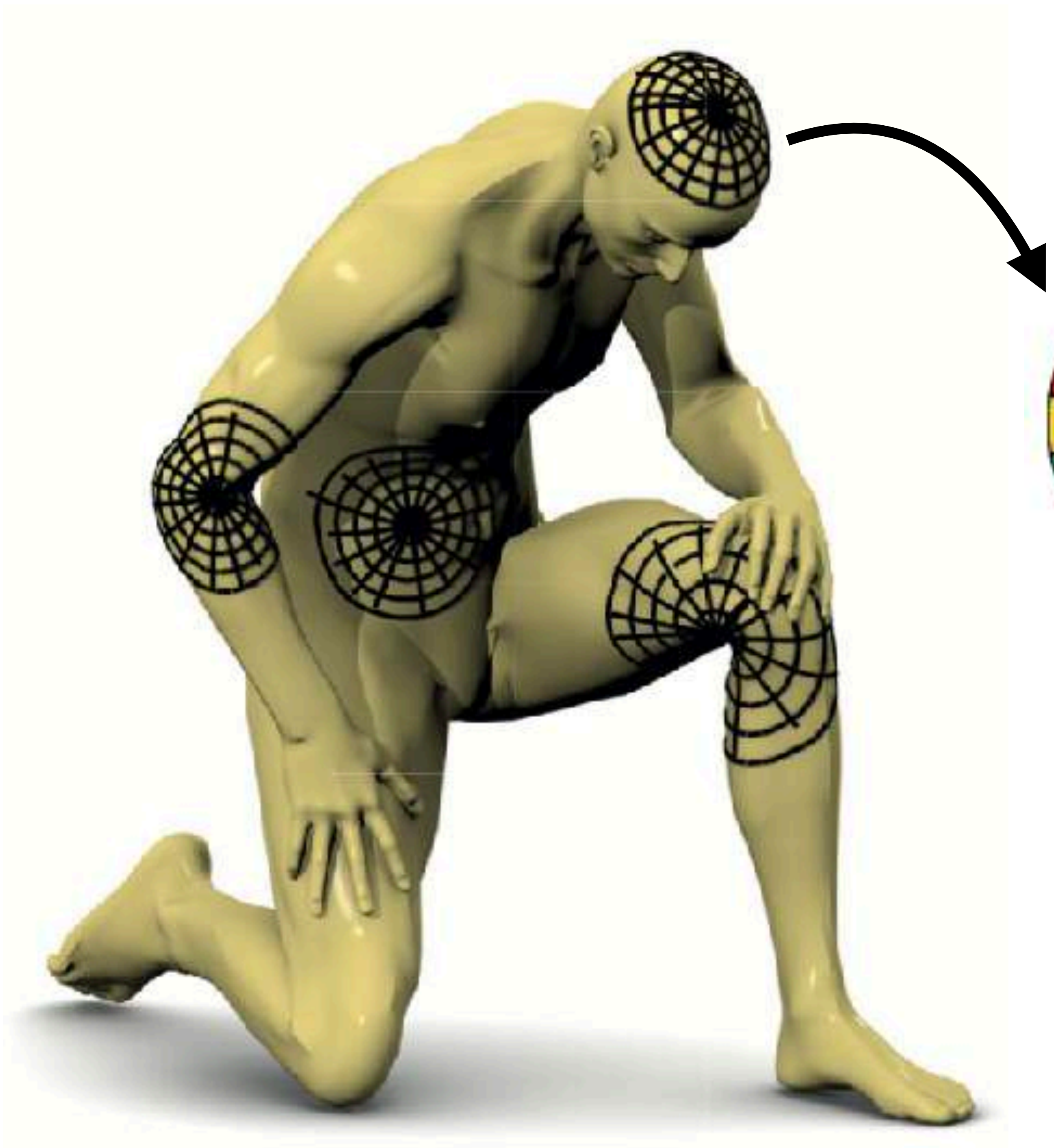
Schmidt et al. 2006

e.g., Geodesic Convolution

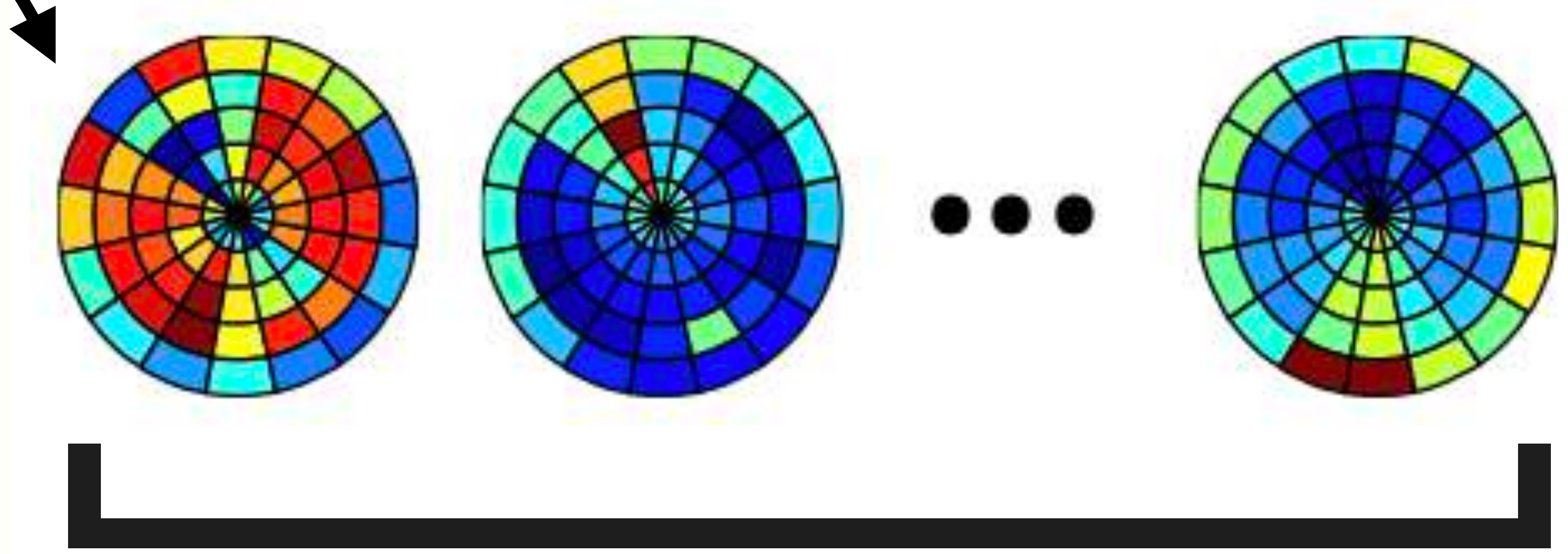


Sharp et al. 2019

e.g., Geodesic Convolution

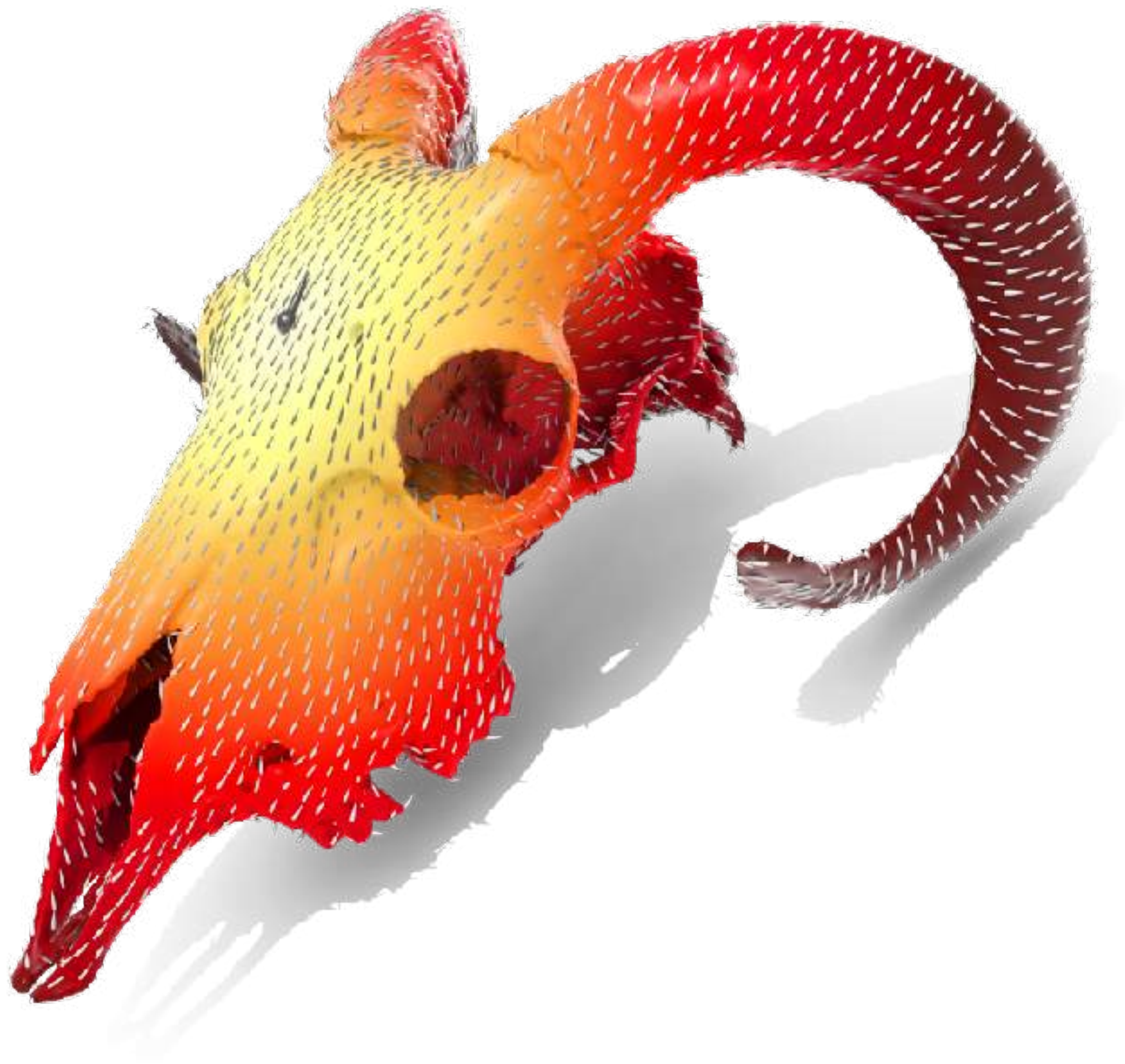


Consider all directions
[Masci et al. 2015]



Max. / Avg.

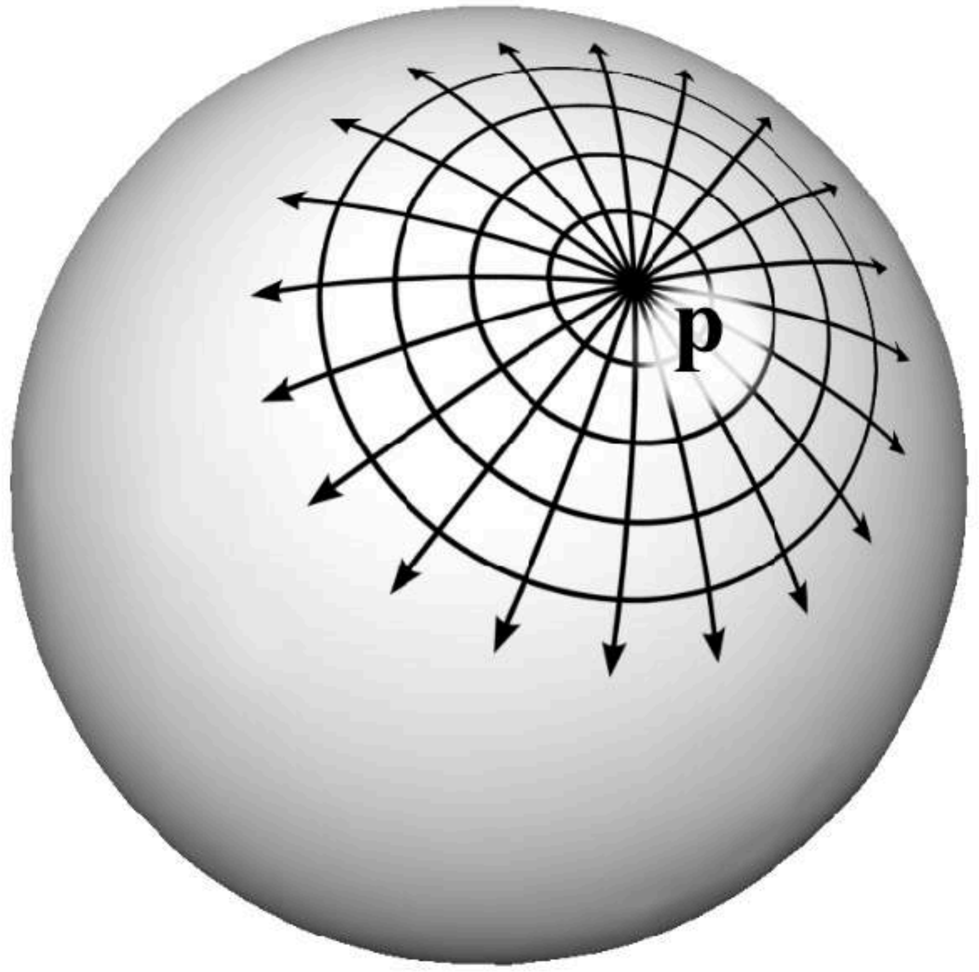
Pick one direction at a time
[Poulenard & Ovsjanikov 2018]



Sharp et al. 2019

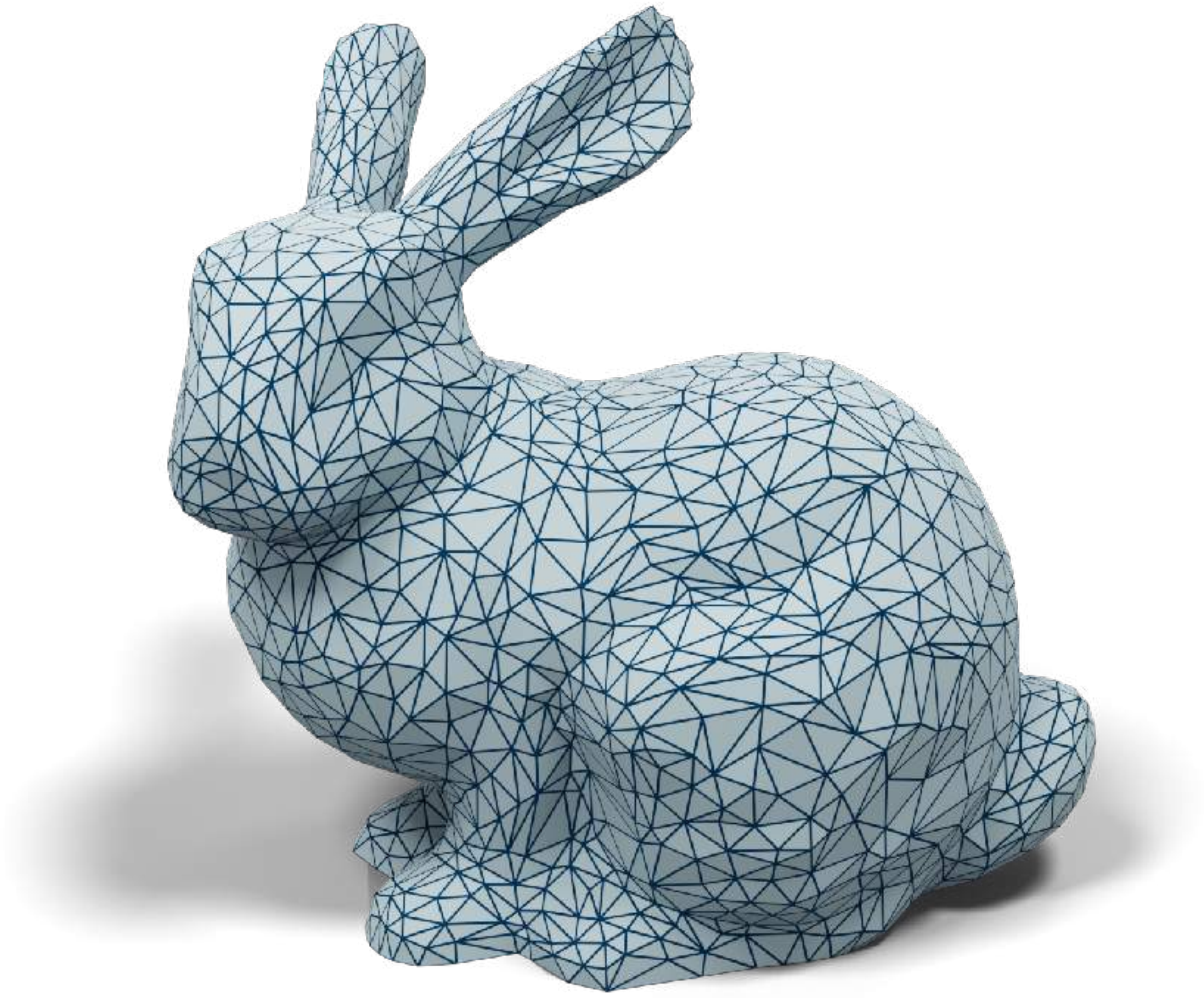
Summary of Parameterization-Based Convolution

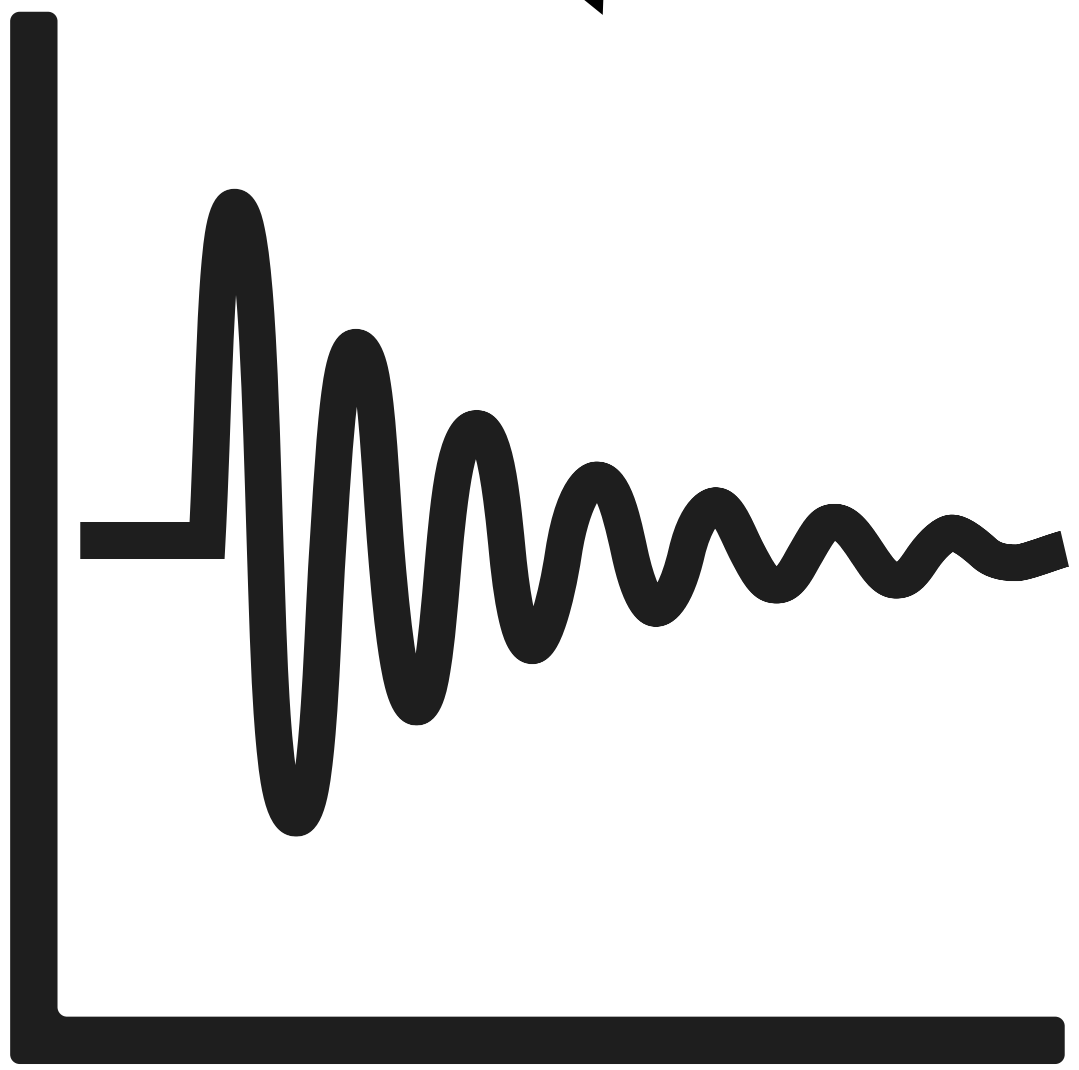
- Flatten meshes to 2D and use 2D convolution
- “Resample” the flattened mesh -> robust to discretization
- Suffer from orientation ambiguity, distortion, expensive



Go back to the first principle

Convolution theorem:
Convolution in the **spatial** domain is the pointwise product in the **spectral** domain.

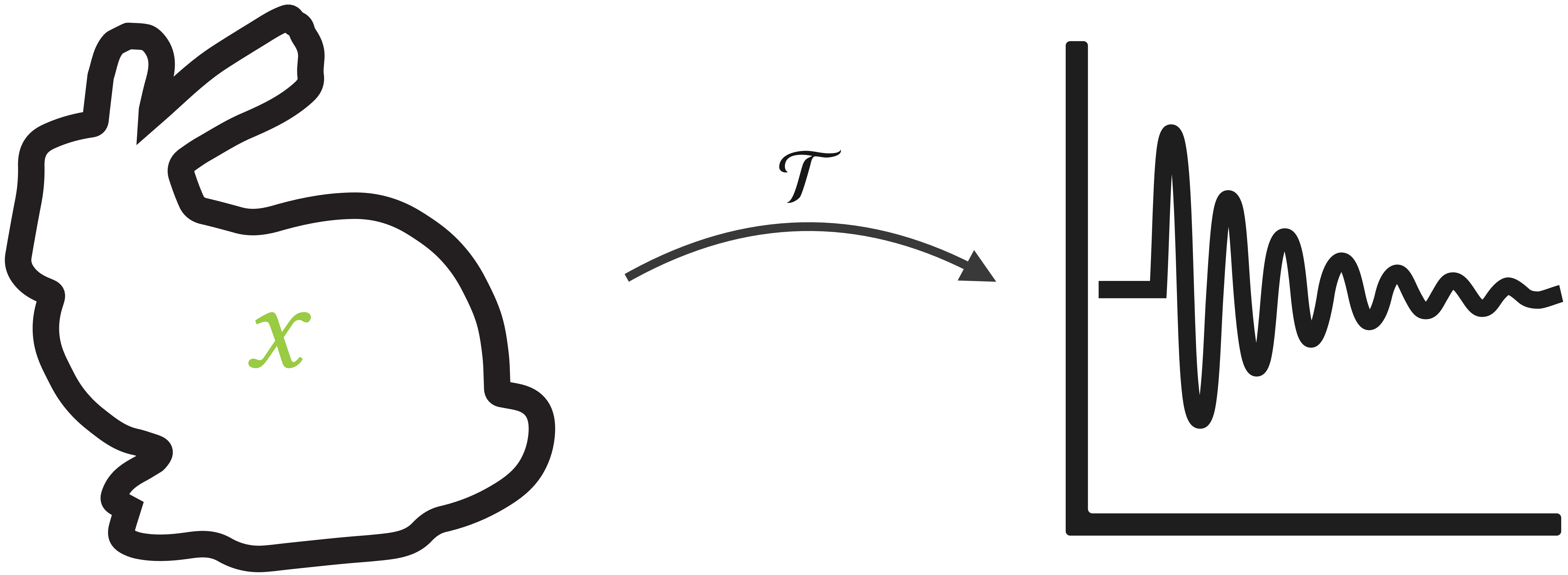




Spectral Convolution

Conv. filter weights

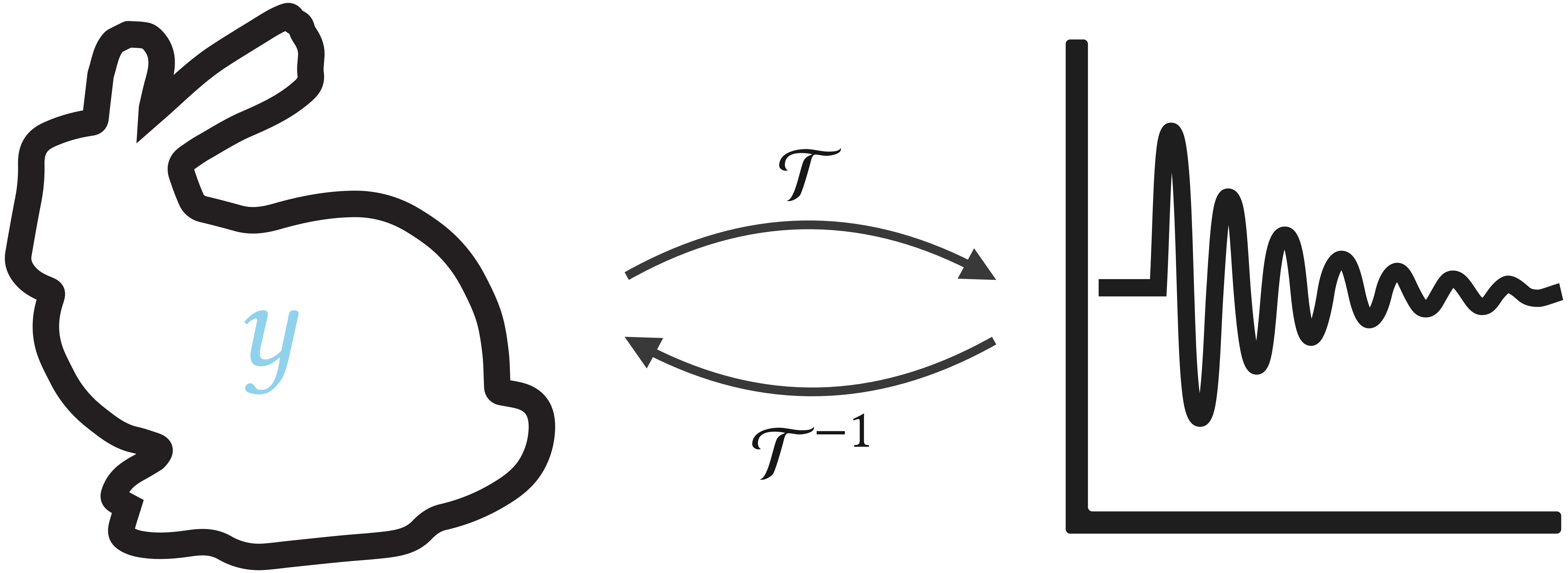
$$y = \mathcal{T}^{-1}\left(w \odot \mathcal{T}(x)\right)$$



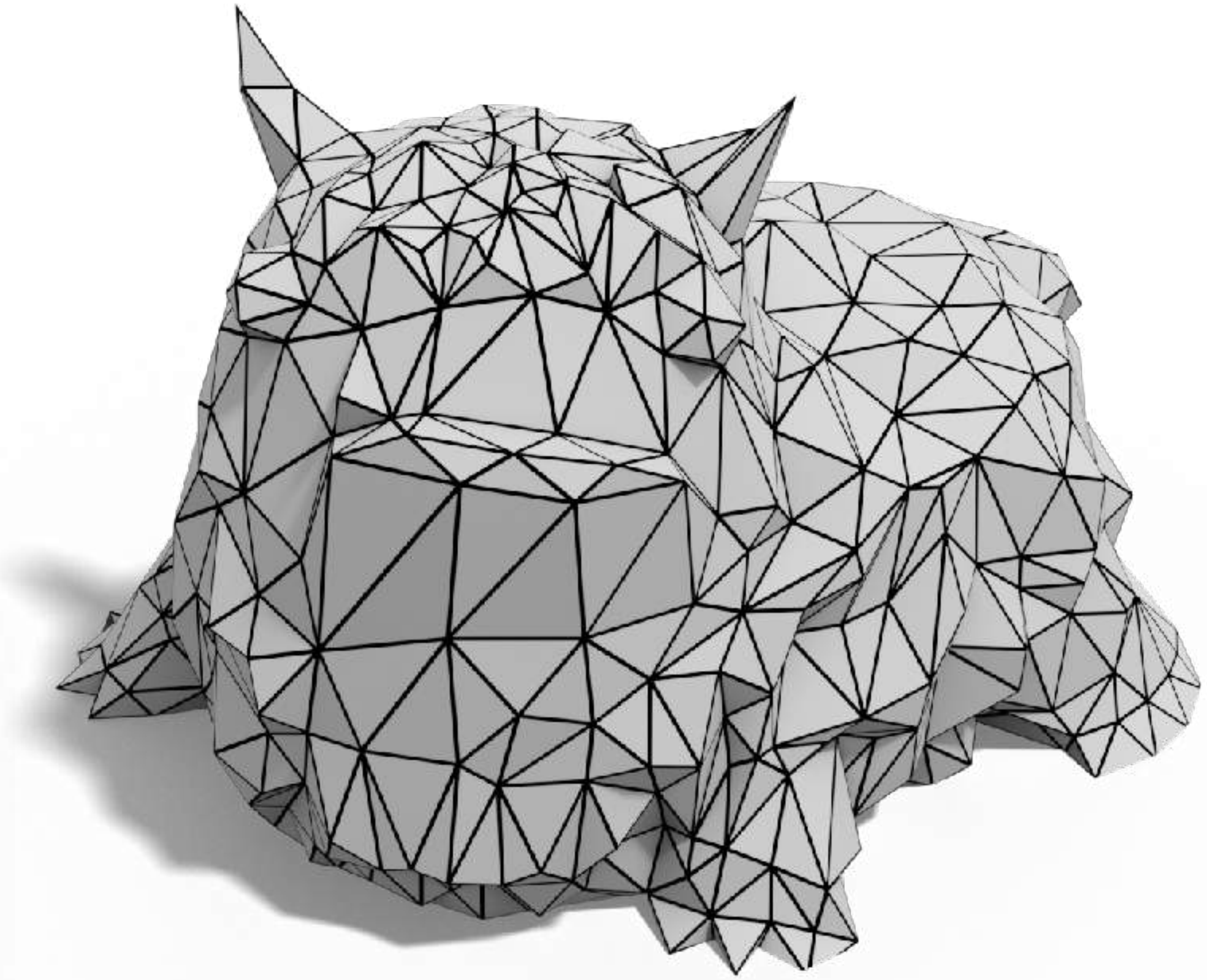
Spectral Convolution

Conv. filter weights

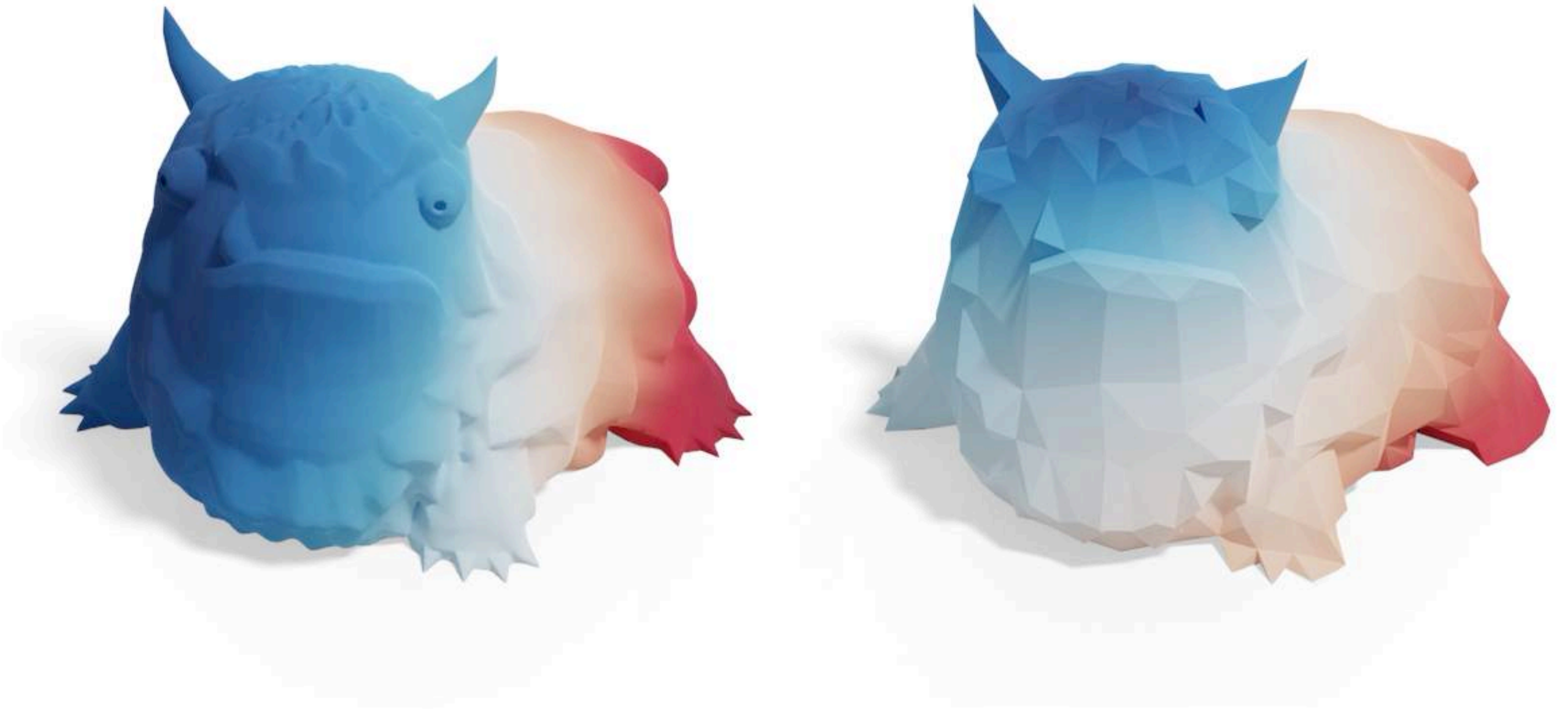
$$y = \mathcal{T}^{-1}\left(w \odot \mathcal{T}(x)\right)$$



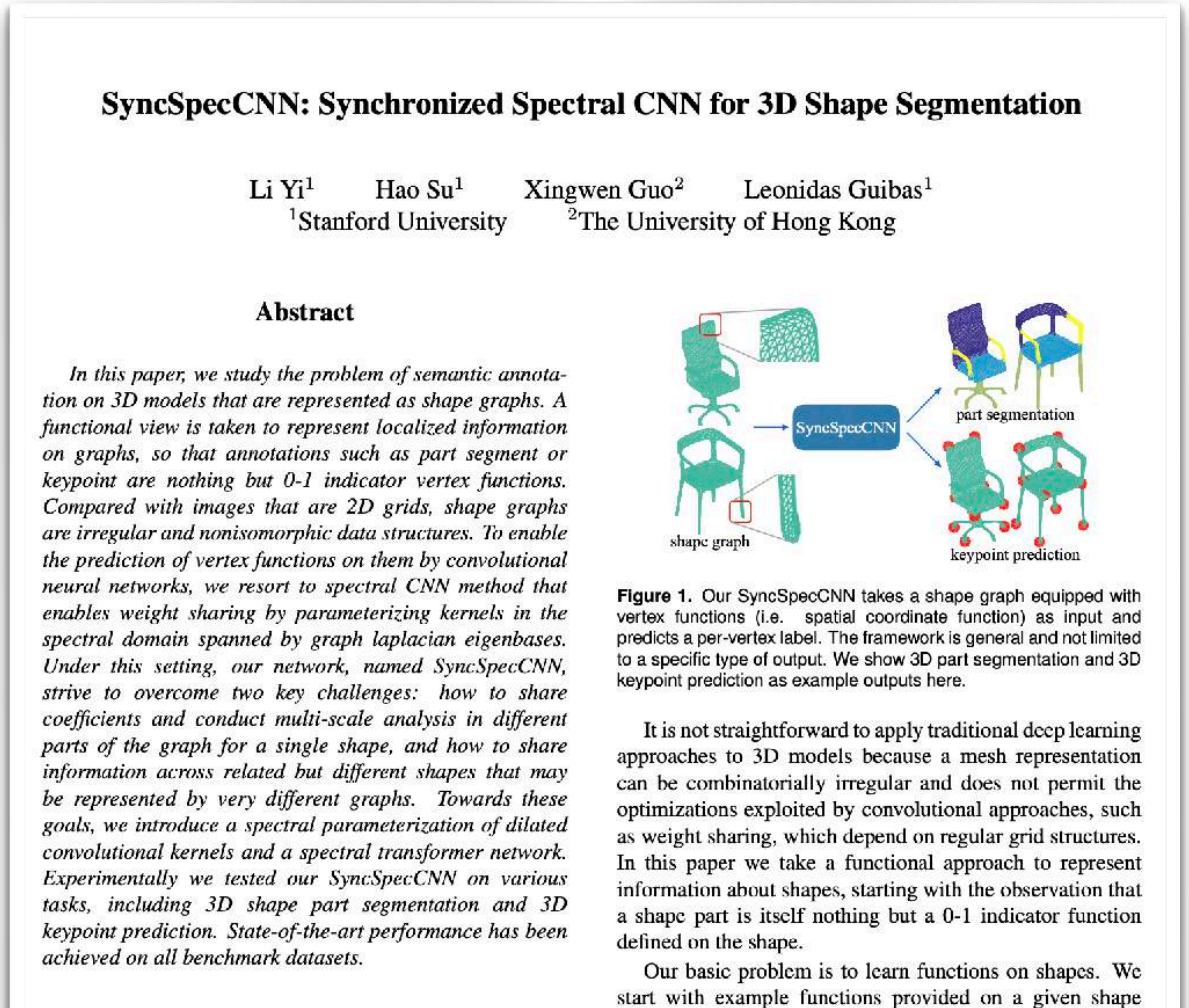
Different shapes have different spectral spaces



Different shapes have different spectral spaces



Some attempts



SyncSpecCNN: Synchronized Spectral CNN for 3D Shape Segmentation

Li Yi[1] Hao Su[1] Xingwen Guo[2] Leonidas Guibas[1]
[1]Stanford University [2]The University of Hong Kong

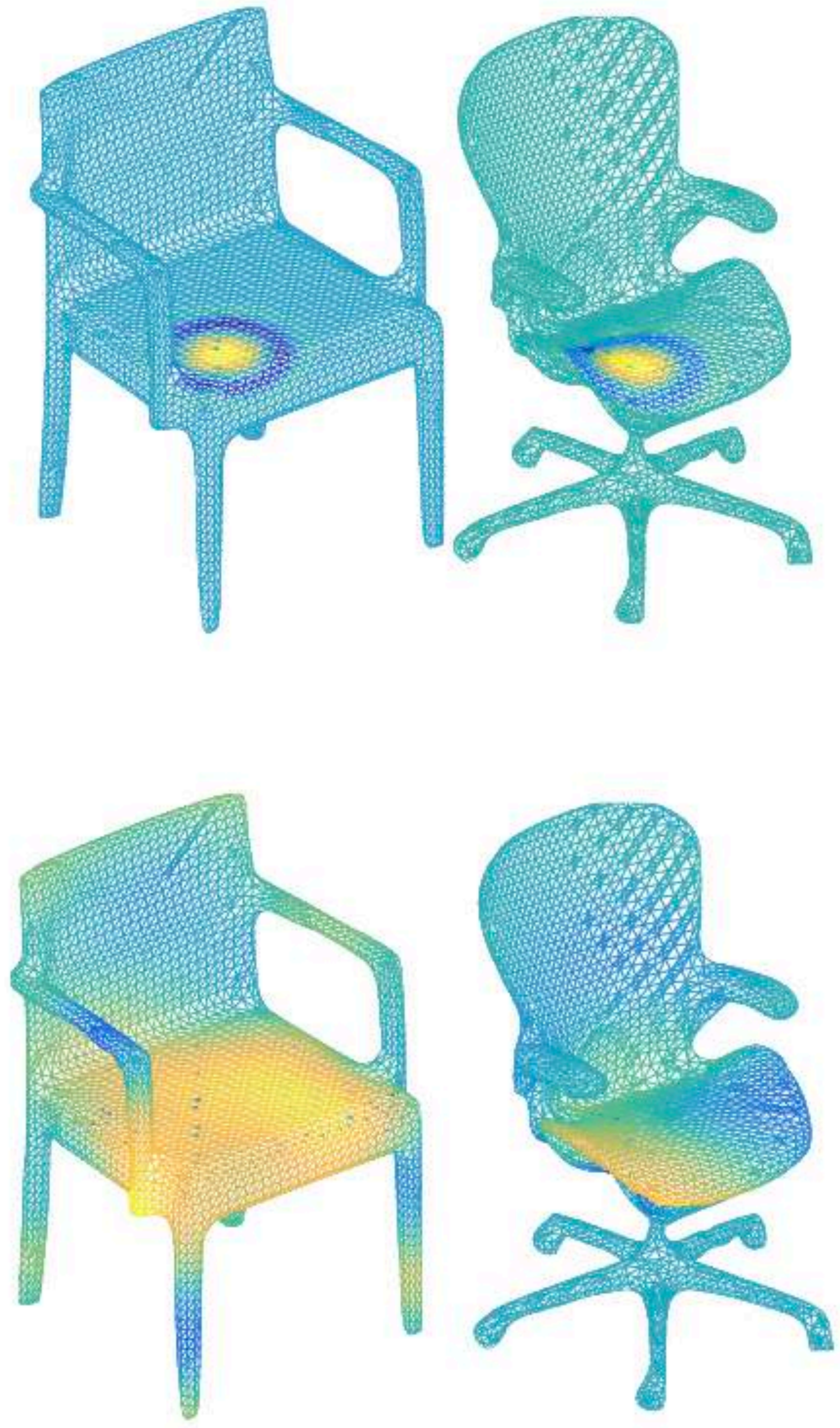
Abstract

In this paper, we study the problem of semantic annotation on 3D models that are represented as shape graphs. A functional view is taken to represent localized information on graphs, so that annotations such as part segment or keypoint are nothing but 0-1 indicator vertex functions. Compared with images that are 2D grids, shape graphs are irregular and nonisomorphic data structures. To enable the prediction of vertex functions on them by convolutional neural networks, we resort to spectral CNN method that enables weight sharing by parameterizing kernels in the spectral domain spanned by graph laplacian eigenbases. Under this setting, our network, named SyncSpecCNN, strive to overcome two key challenges: how to share coefficients and conduct multi-scale analysis in different parts of the graph for a single shape, and how to share information across related but different shapes that may be represented by very different graphs. Towards these goals, we introduce a spectral parameterization of dilated convolutional kernels and a spectral transformer network. Experimentally we tested our SyncSpecCNN on various tasks, including 3D shape part segmentation and 3D keypoint prediction. State-of-the-art performance has been achieved on all benchmark datasets.

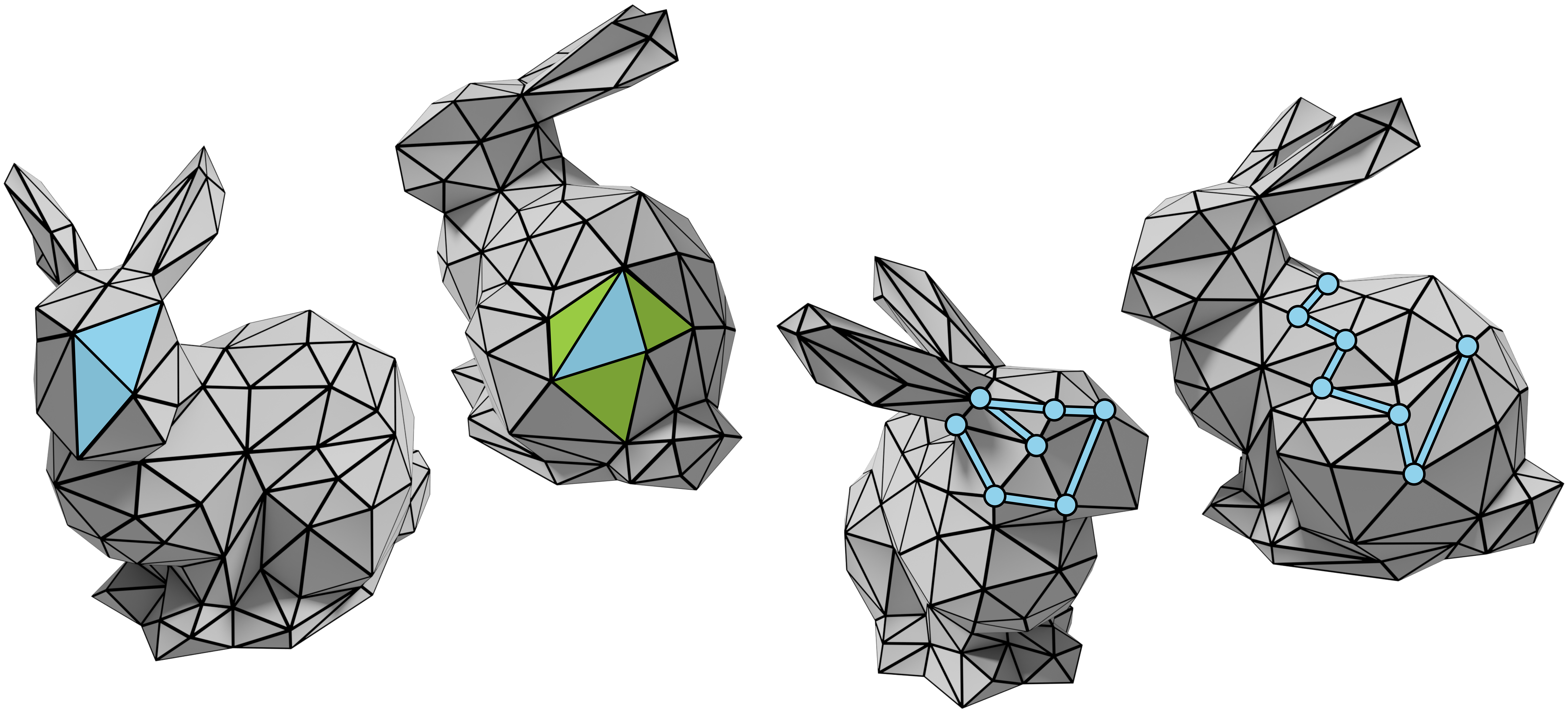
Figure 1. Our SyncSpecCNN takes a shape graph equipped with vertex functions (i.e. spatial coordinate function) as input and predicts a per-vertex label. The framework is general and not limited to a specific type of output. We show 3D part segmentation and 3D keypoint prediction as example outputs here.

It is not straightforward to apply traditional deep learning approaches to 3D models because a mesh representation can be combinatorially irregular and does not permit the optimizations exploited by convolutional approaches, such as weight sharing, which depend on regular grid structures. In this paper we take a functional approach to represent information about shapes, starting with the observation that a shape part is itself nothing but a 0-1 indicator function defined on the shape.

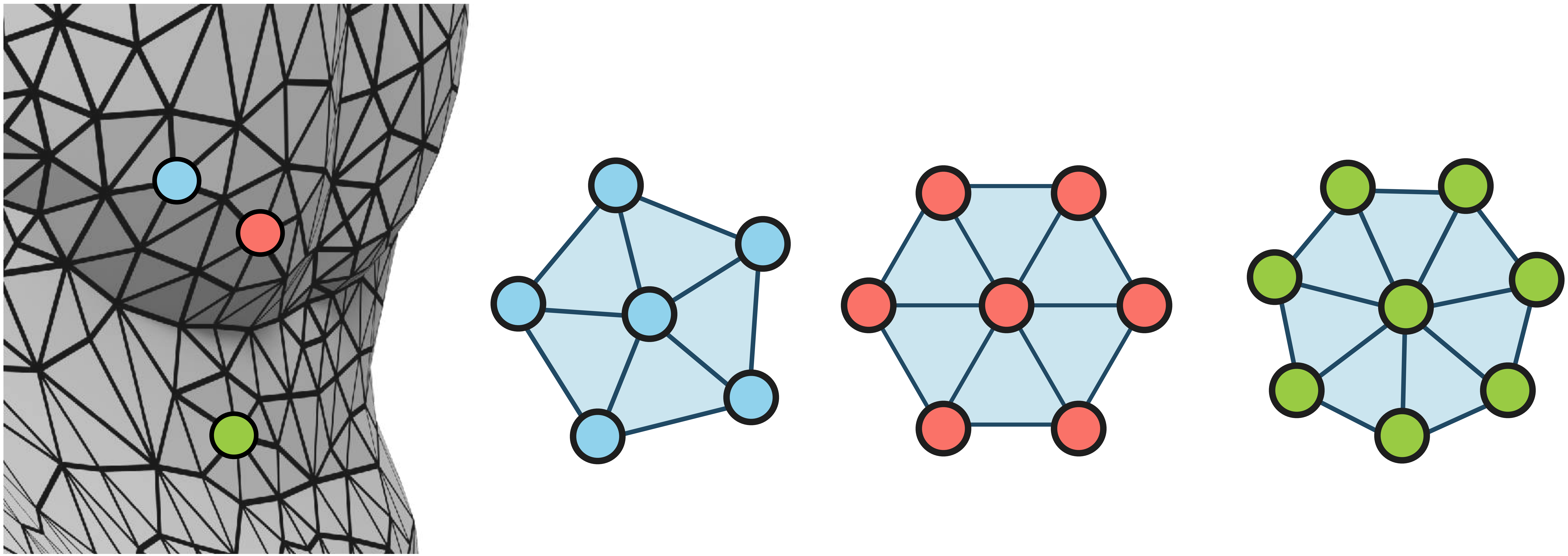
Our basic problem is to learn functions on shapes. We start with example functions provided on a given shape



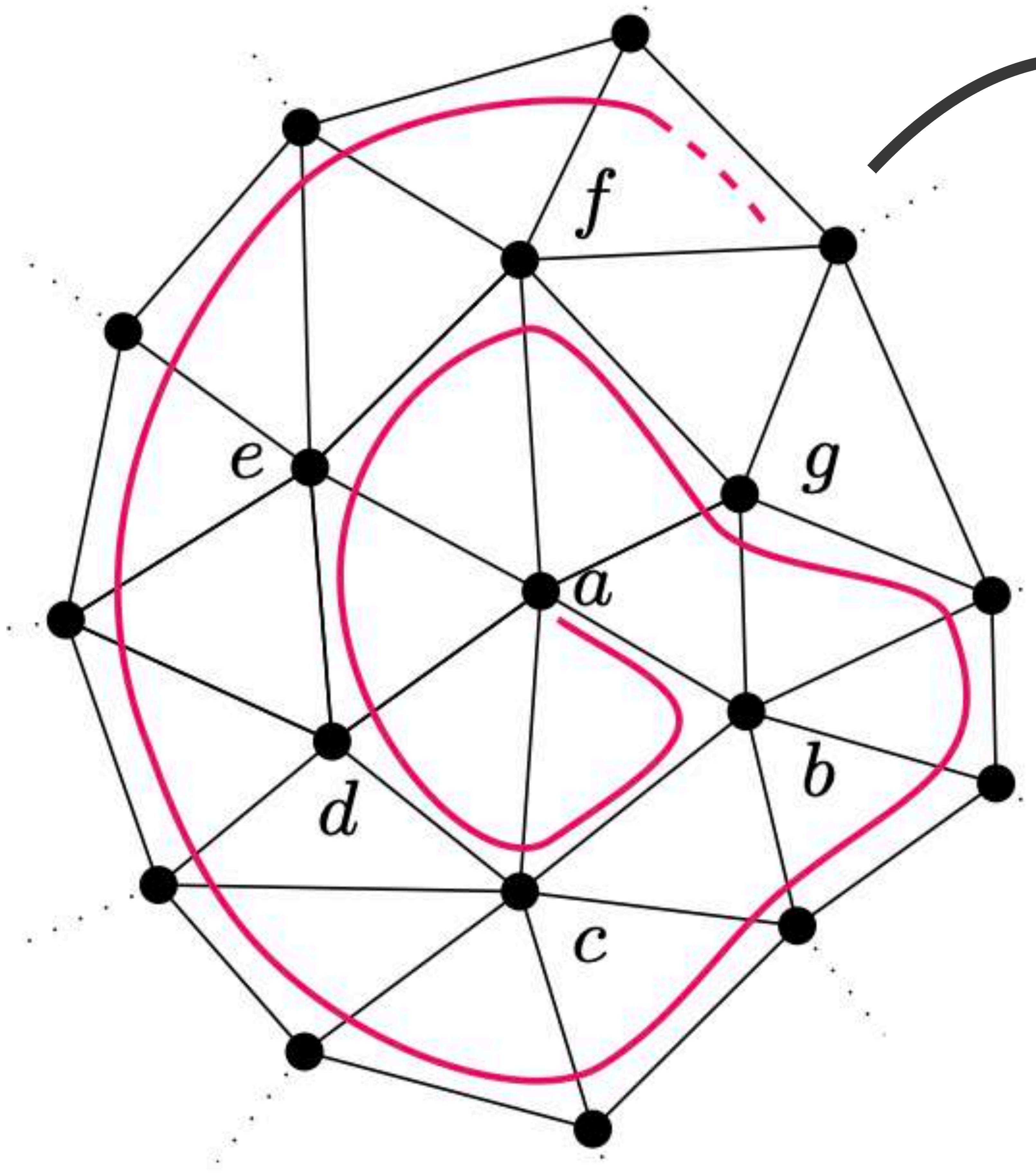
Direct Discrete Mesh Convolutions



Irregular Structure



Spiral Convolution



vertex features

$$[v_a \; v_b \; v_c \; \cdots \; v_k]$$

1D filter

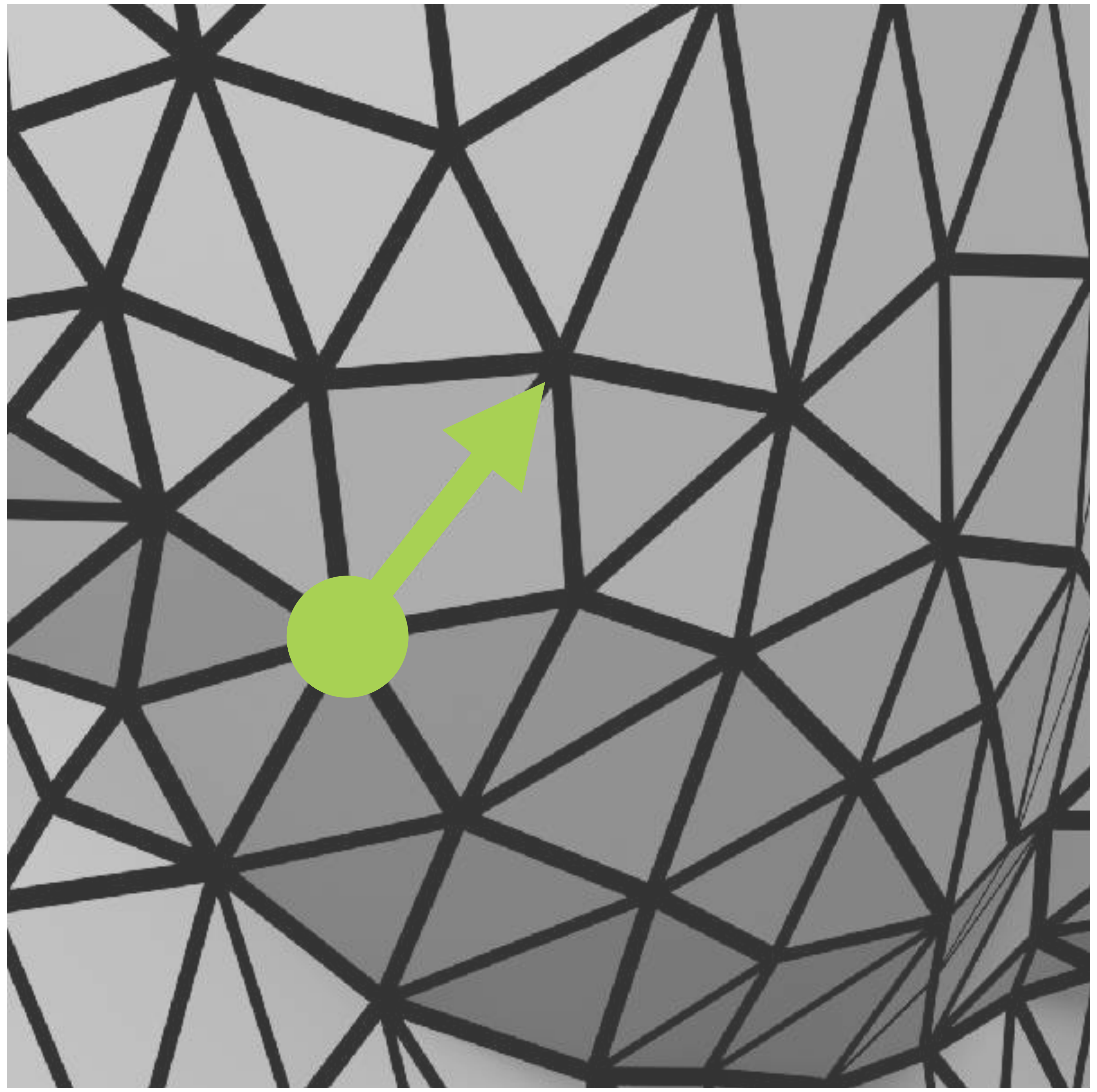
$$[w_1 \; w_2 \; w_3 \; \cdots \; w_k]$$

convolution

$$y = w^\top v$$

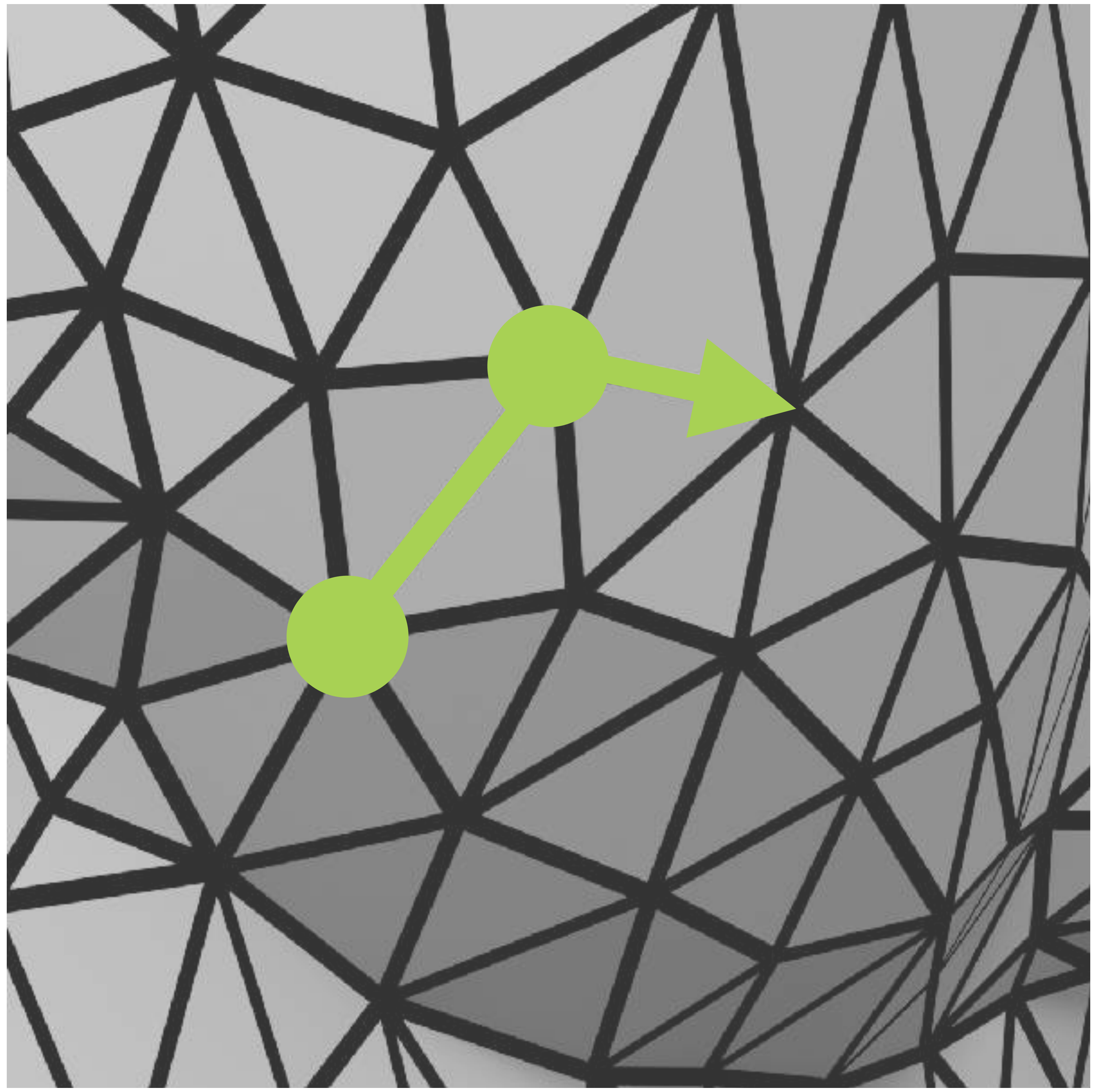
Lim et al. 2018, Gong et al. 2018, Bouritsas et al. 2019

Random Walks

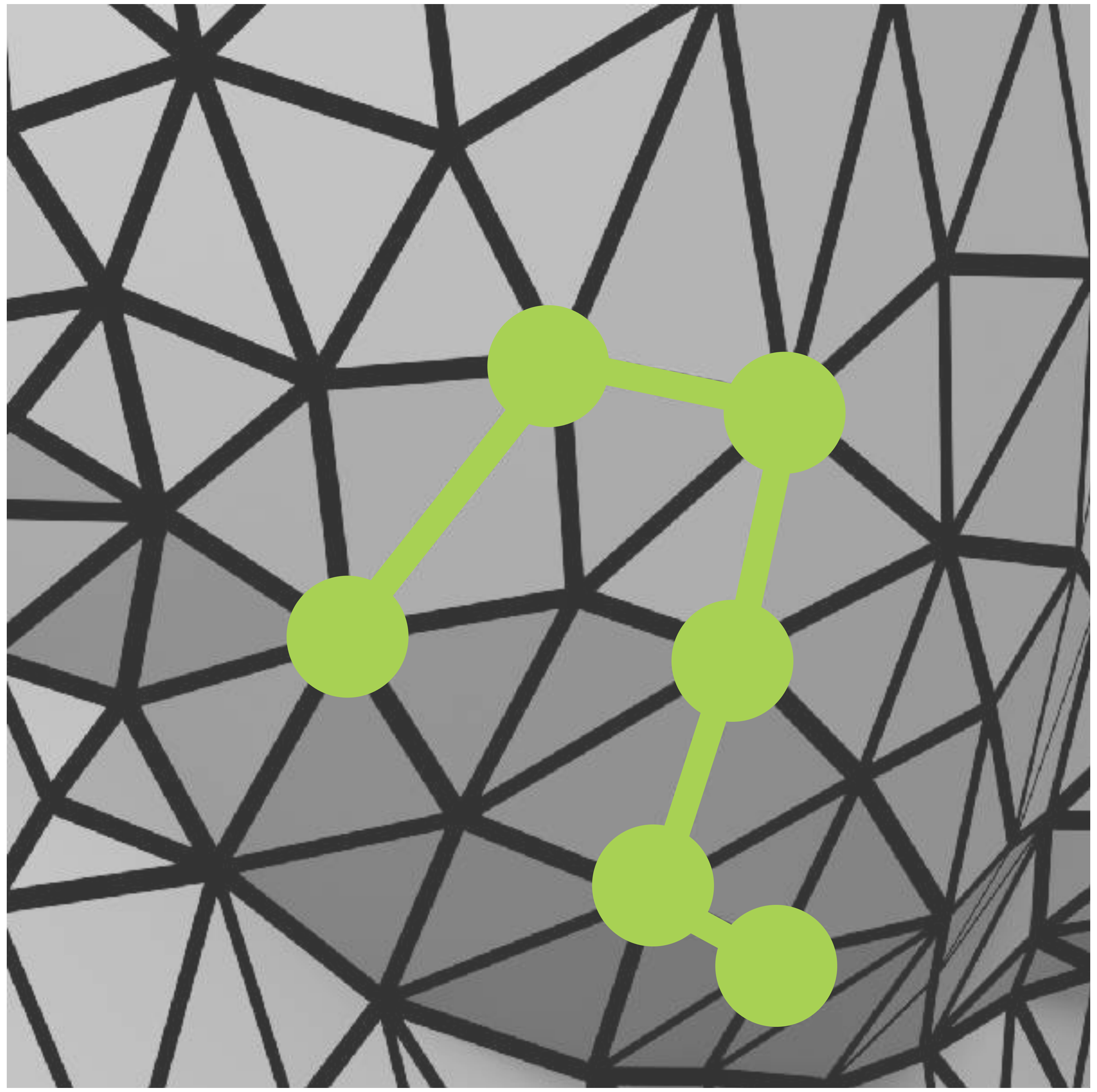


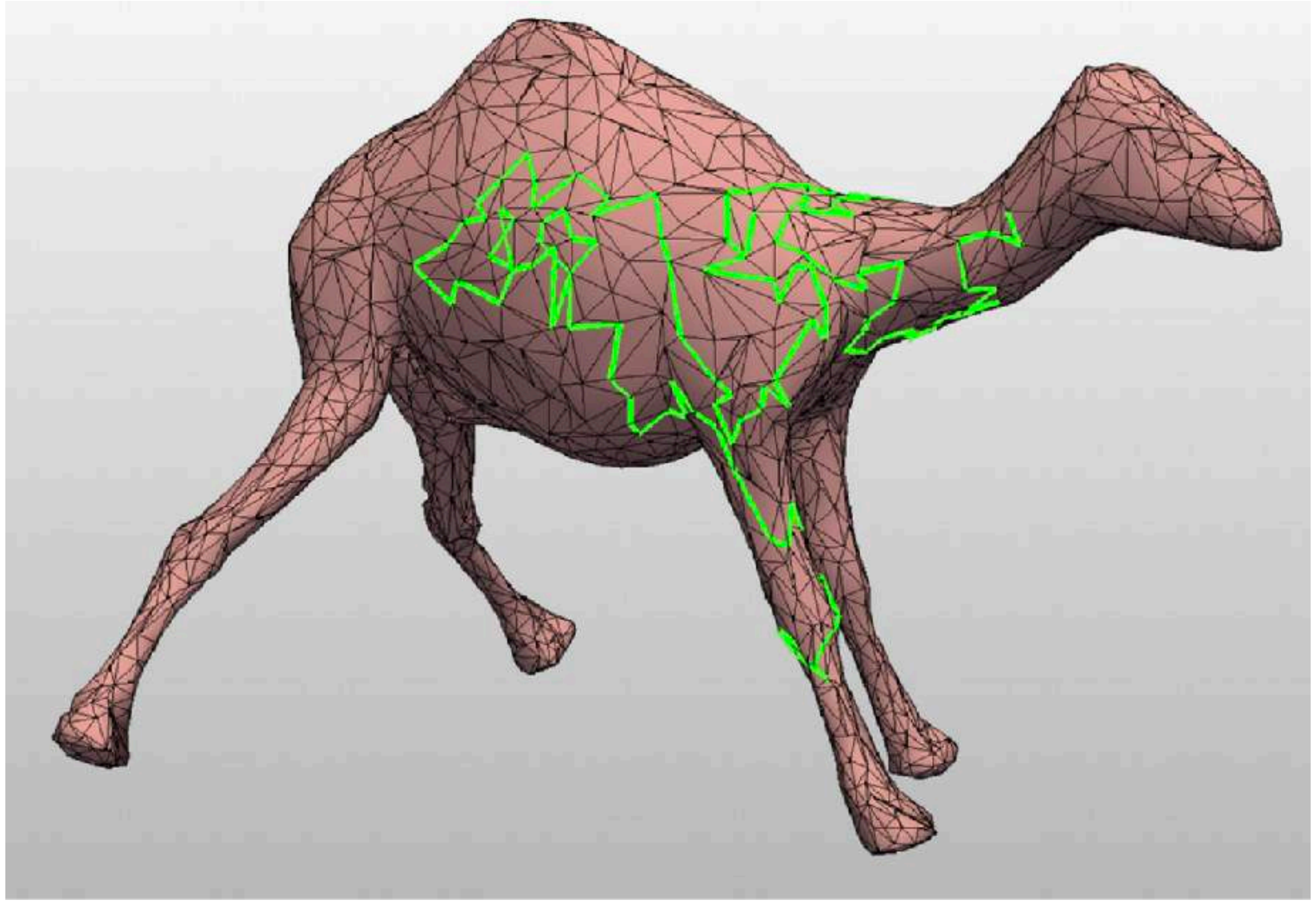
Lahav & Tal 2020

Random Walks



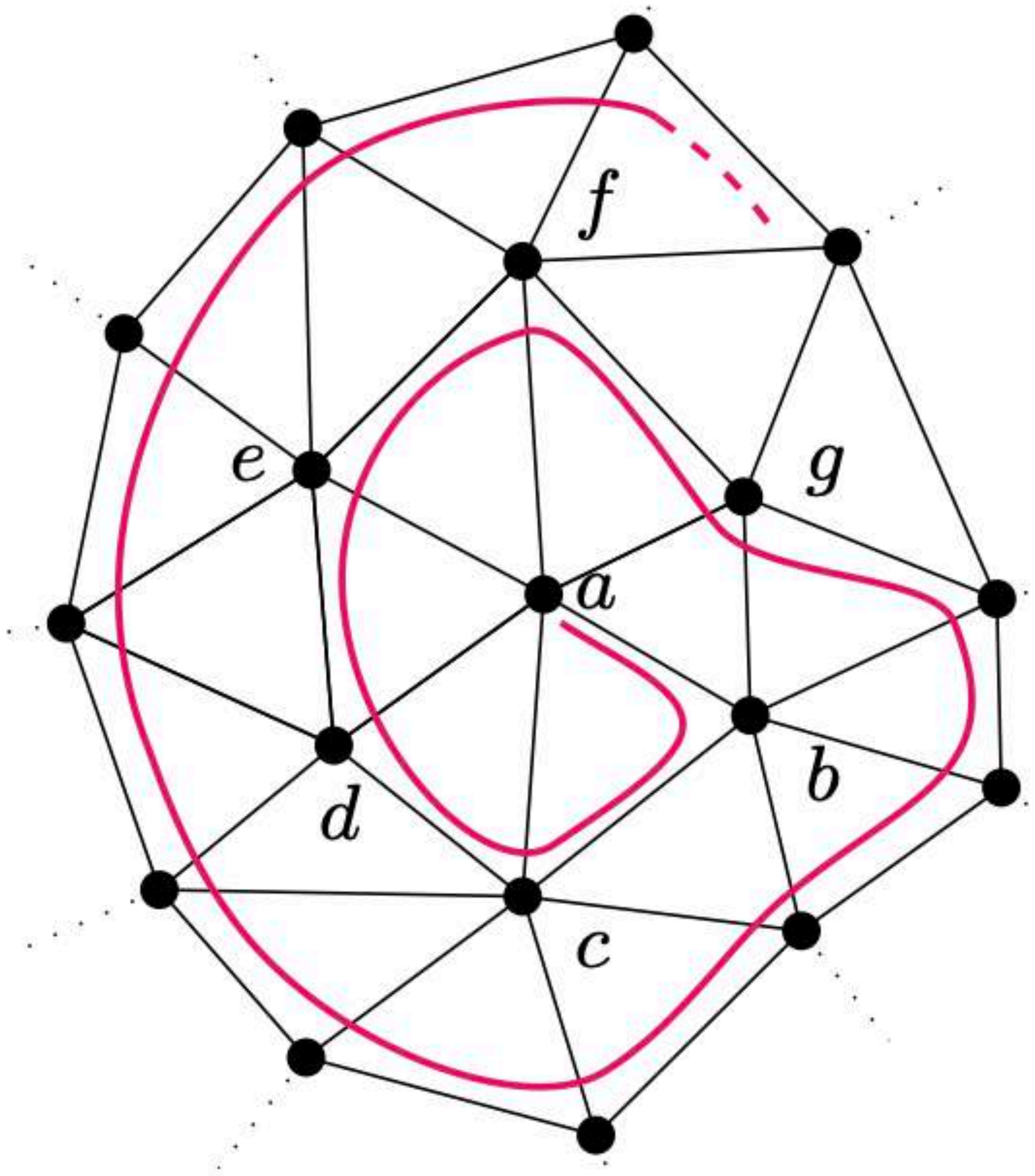
Random Walks

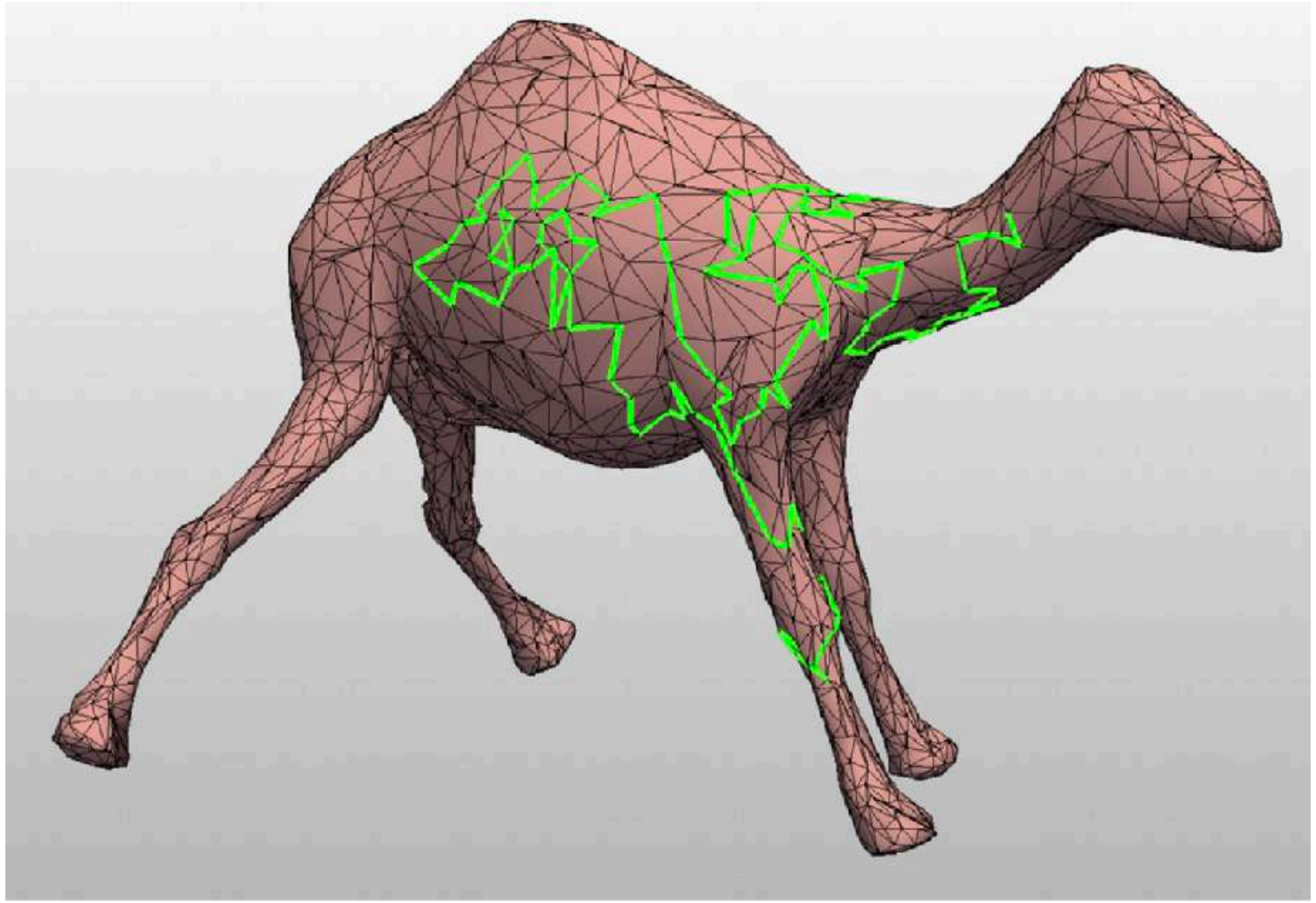




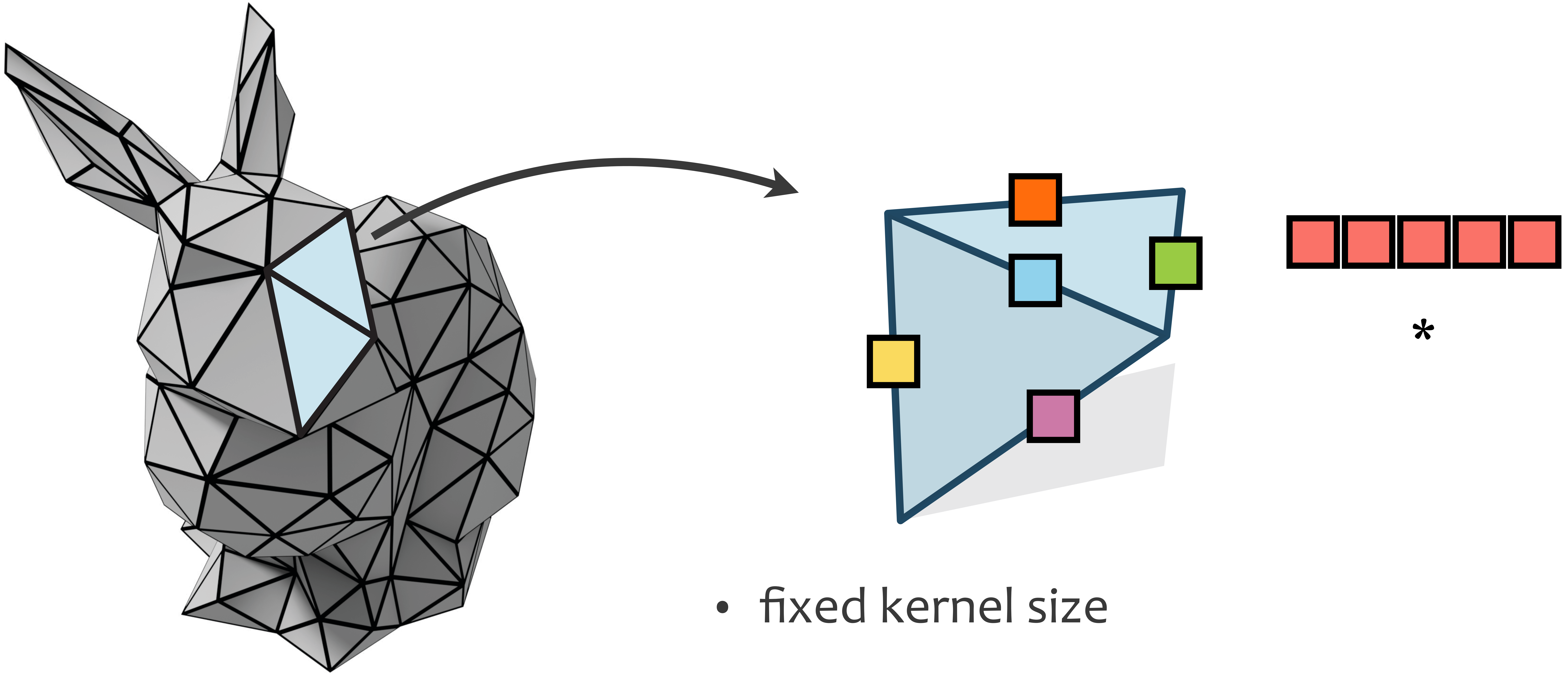
Lahav & Tal 2020

Ambiguities in how to pick vertex orders





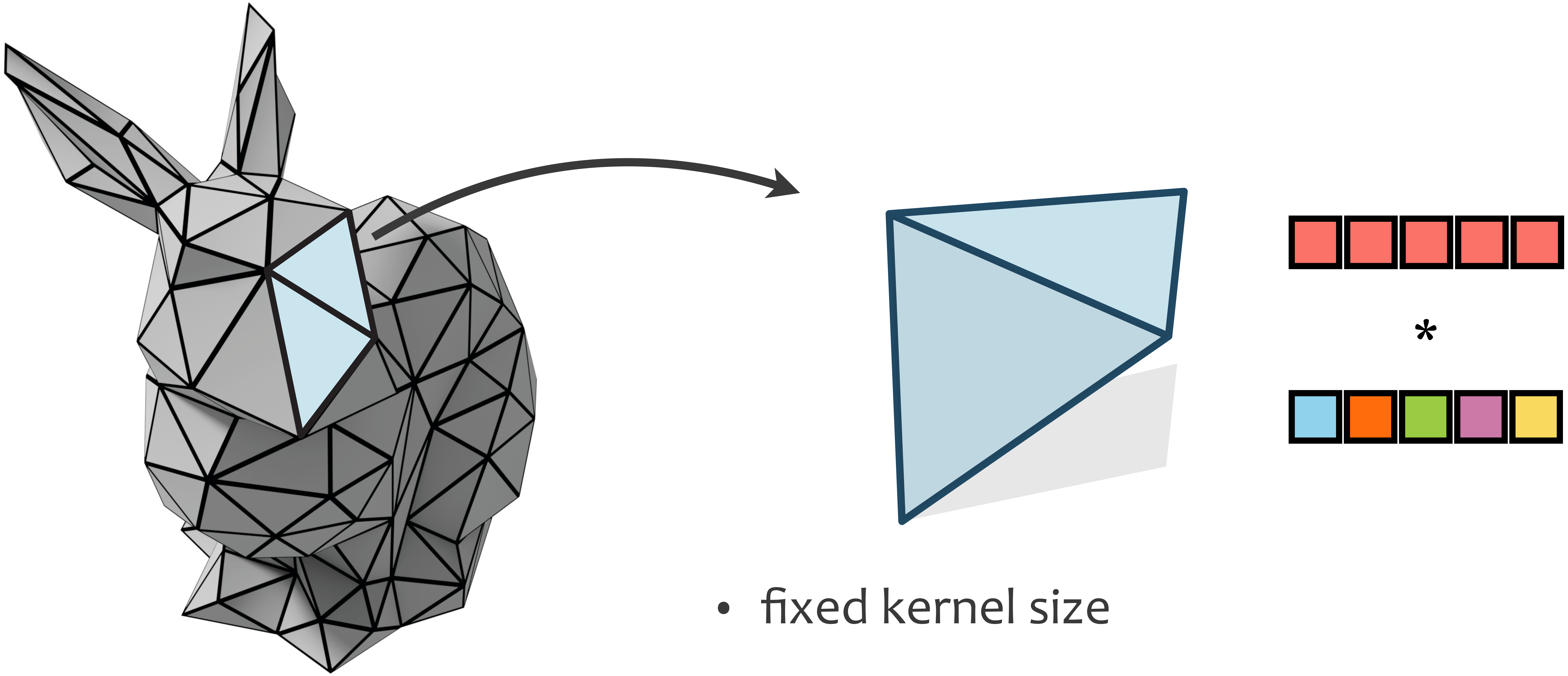
Edge Convolution



- fixed kernel size

Hanocka et al. 2019, Liu et al. 2020

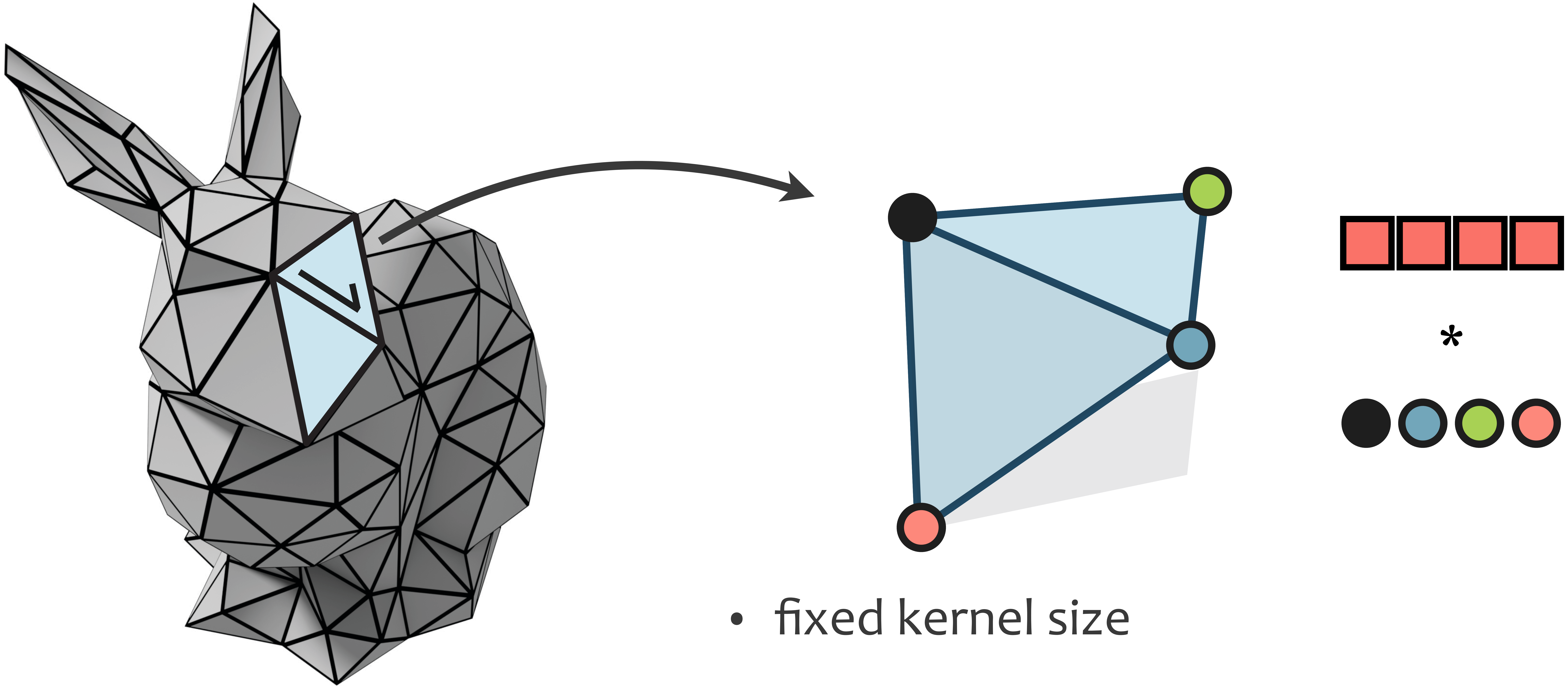
Edge Convolution



- fixed kernel size

Hanocka et al. 2019

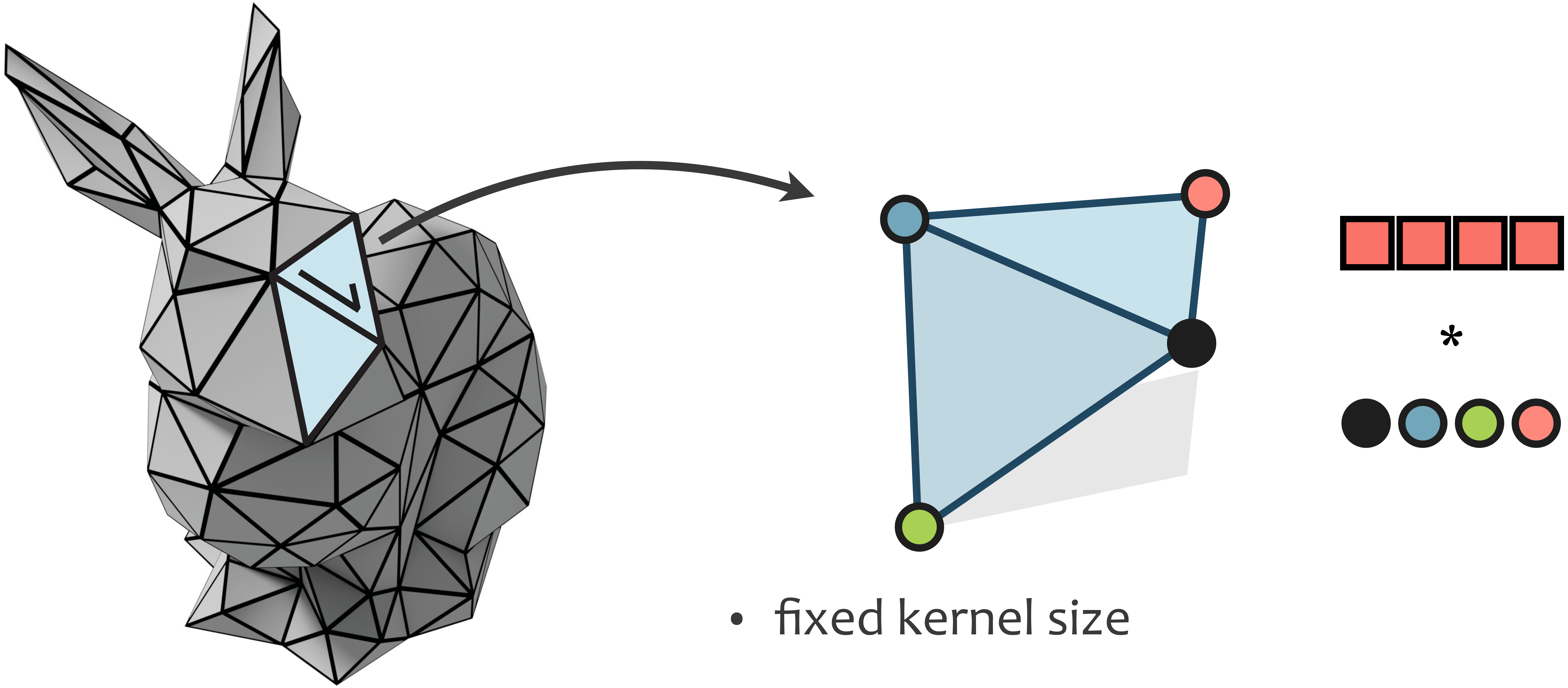
Half-Edge Convolution



- fixed kernel size

Hanocka et al. 2019

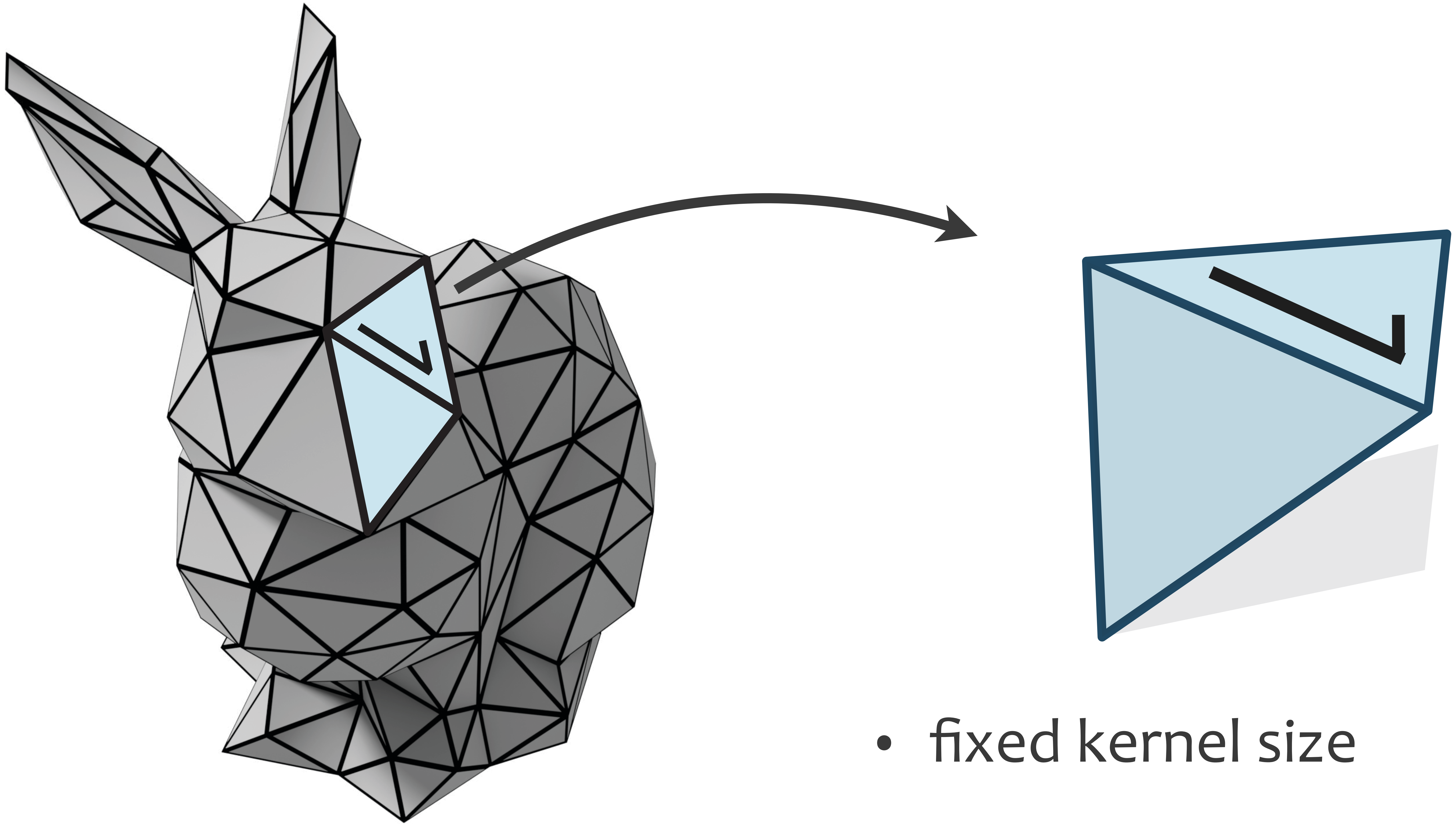
Half-Edge Convolution



- fixed kernel size

Hanocka et al. 2019

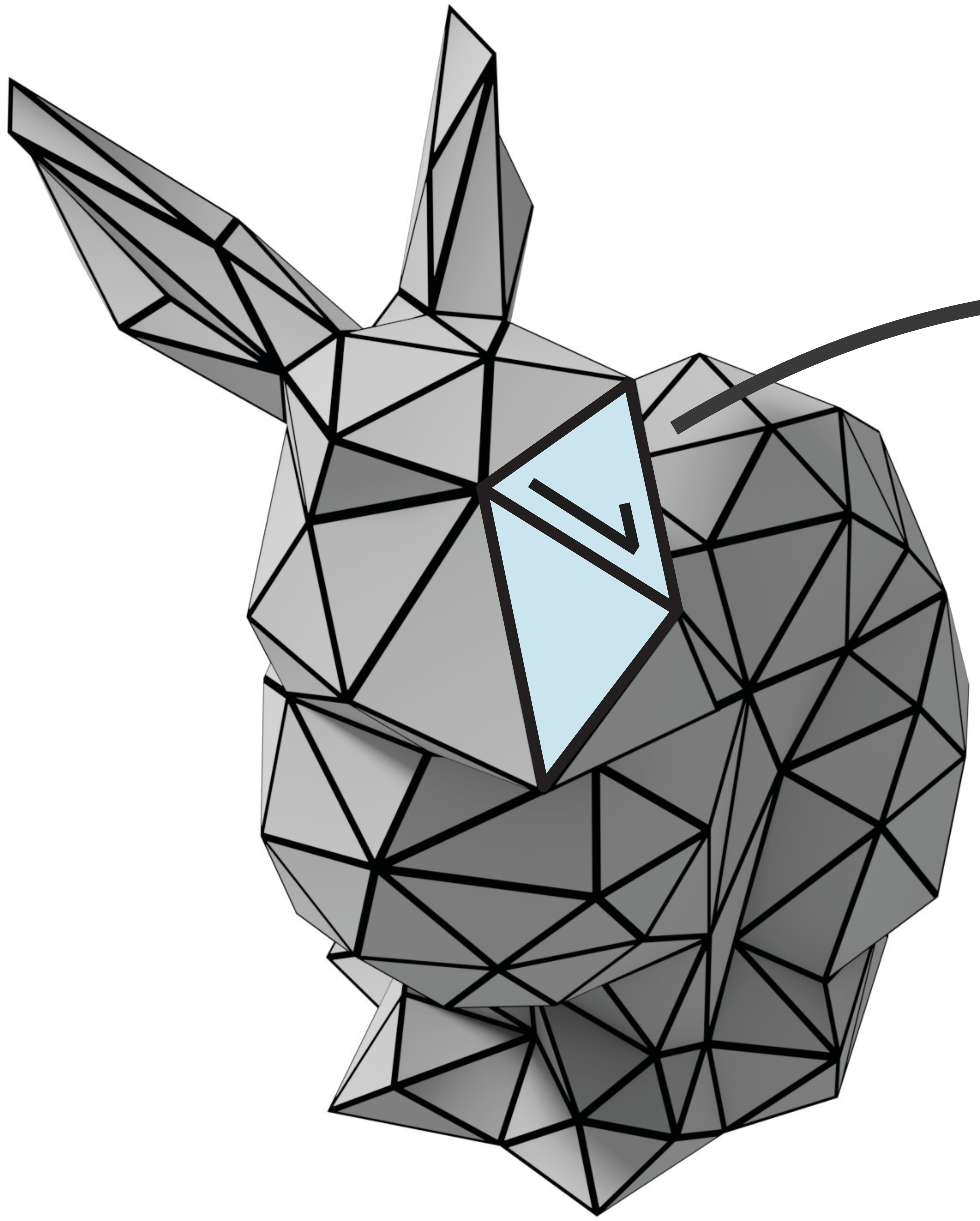
Half-Edge Convolution

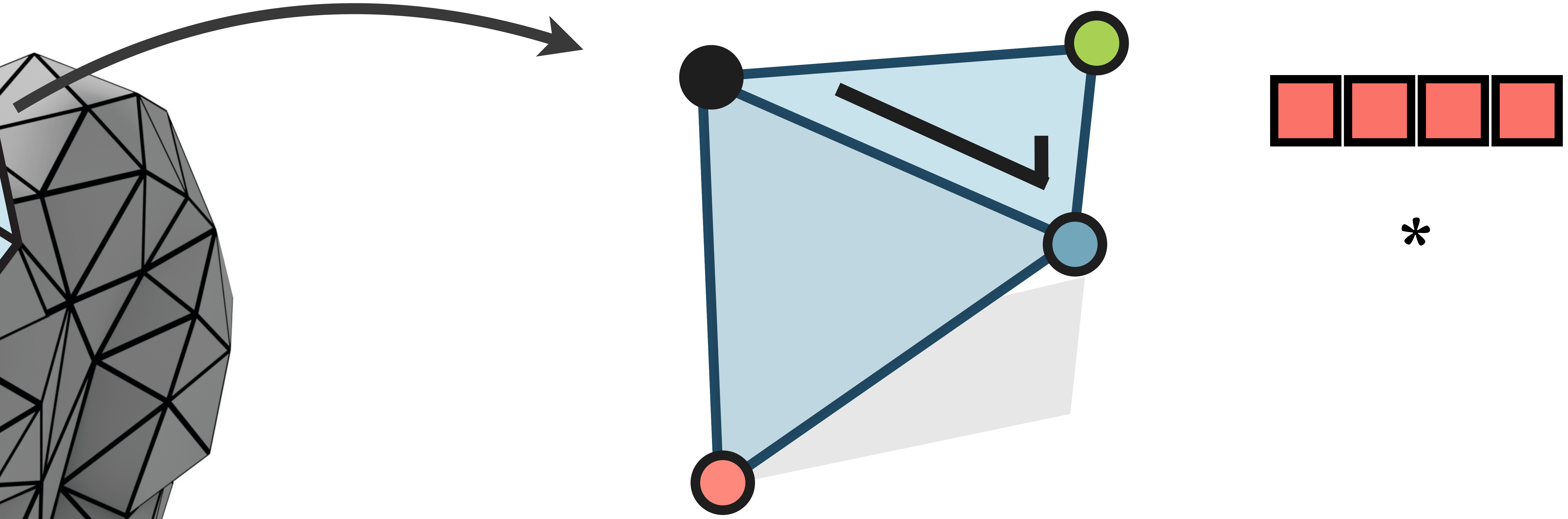


- fixed kernel size

Hanocka et al. 2019, Liu et al. 2020

Half-Edge Convolution

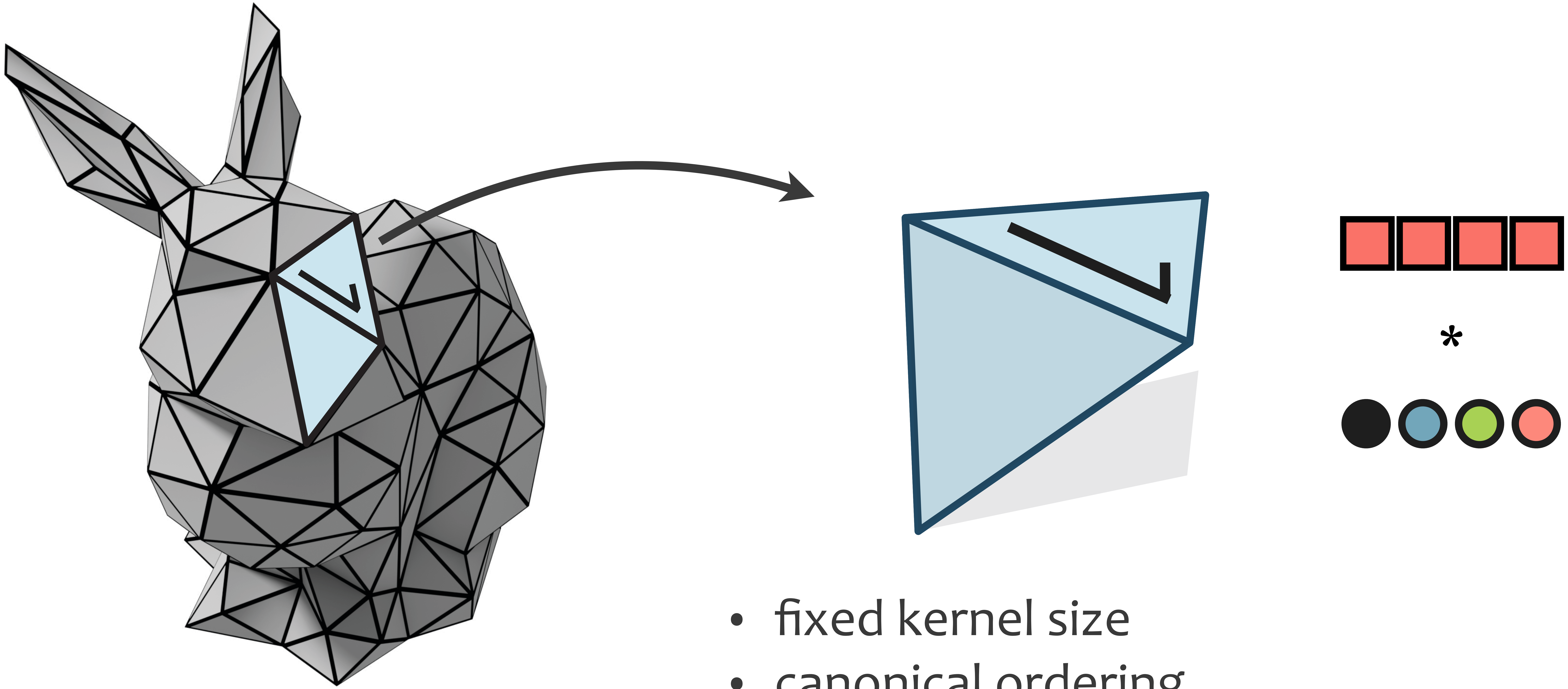




- fixed kernel size
- canonical ordering

Hanocka et al. 2019, Liu et al. 2020

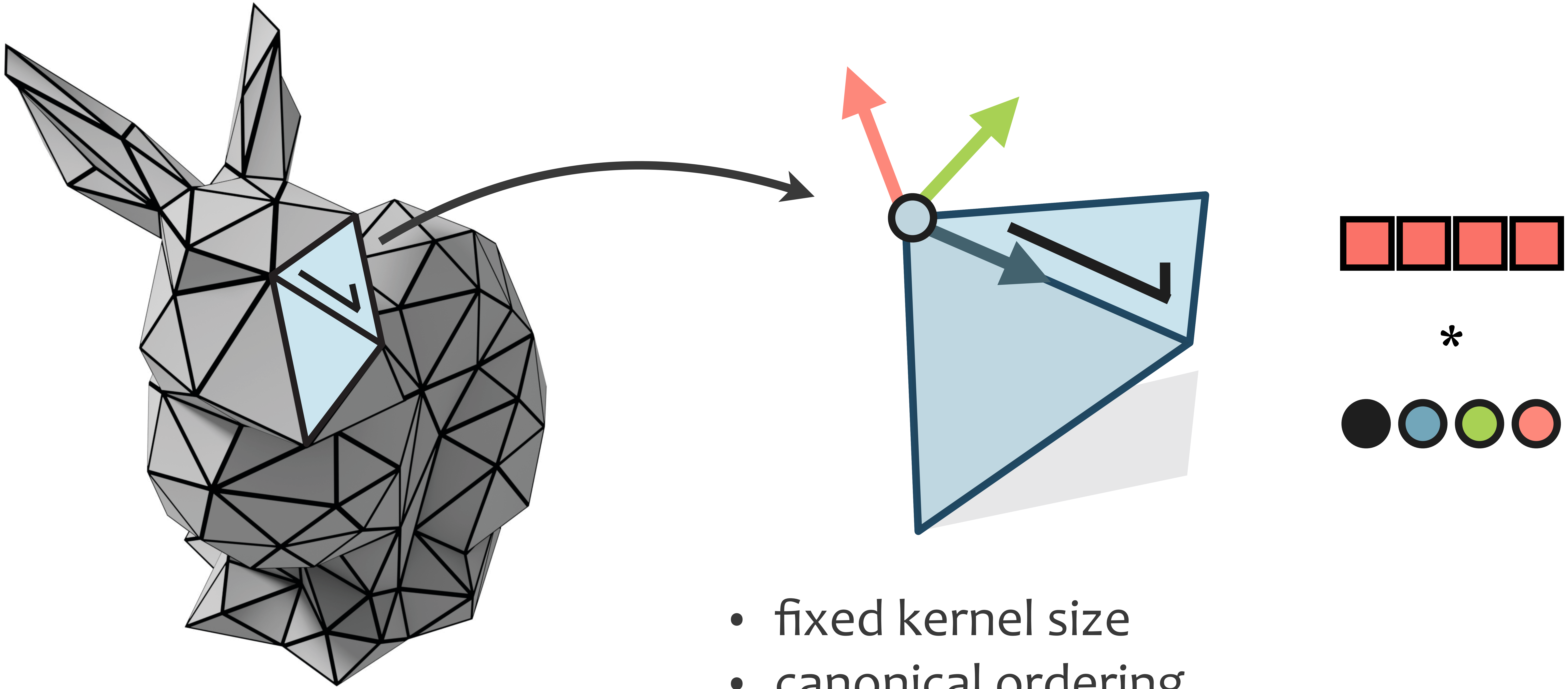
Half-Edge Convolution



- fixed kernel size
- canonical ordering

Hanocka et al. 2019, Liu et al. 2020

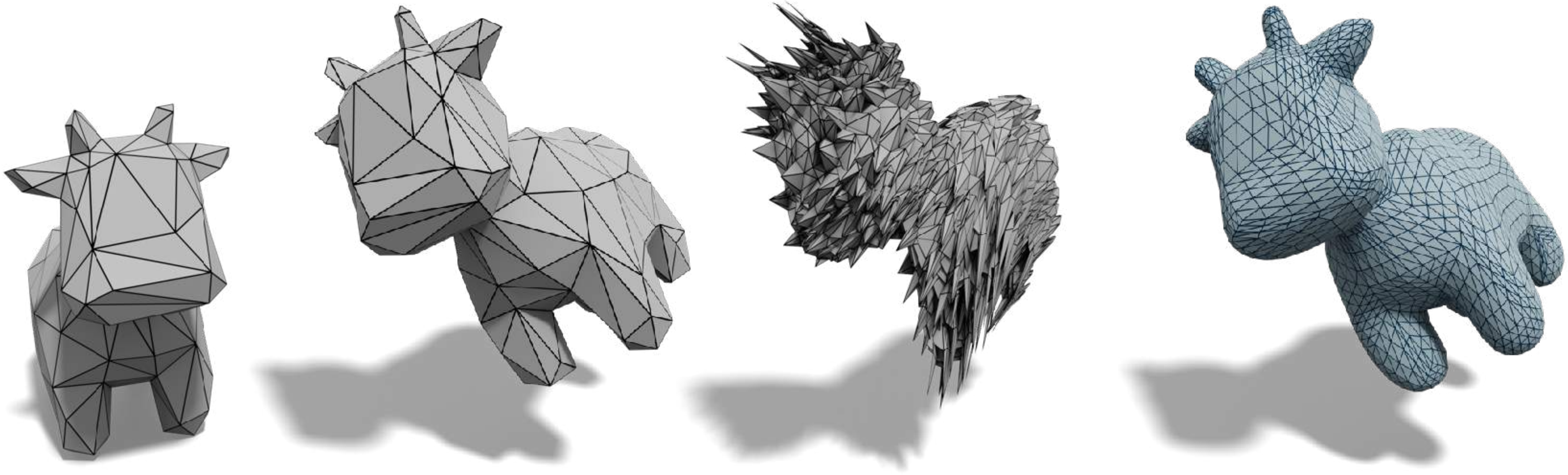
Half-Edge Convolution



- fixed kernel size
- canonical ordering
- local coordinates (rigid motion invariant)

Hanocka et al. 2019, Liu et al. 2020

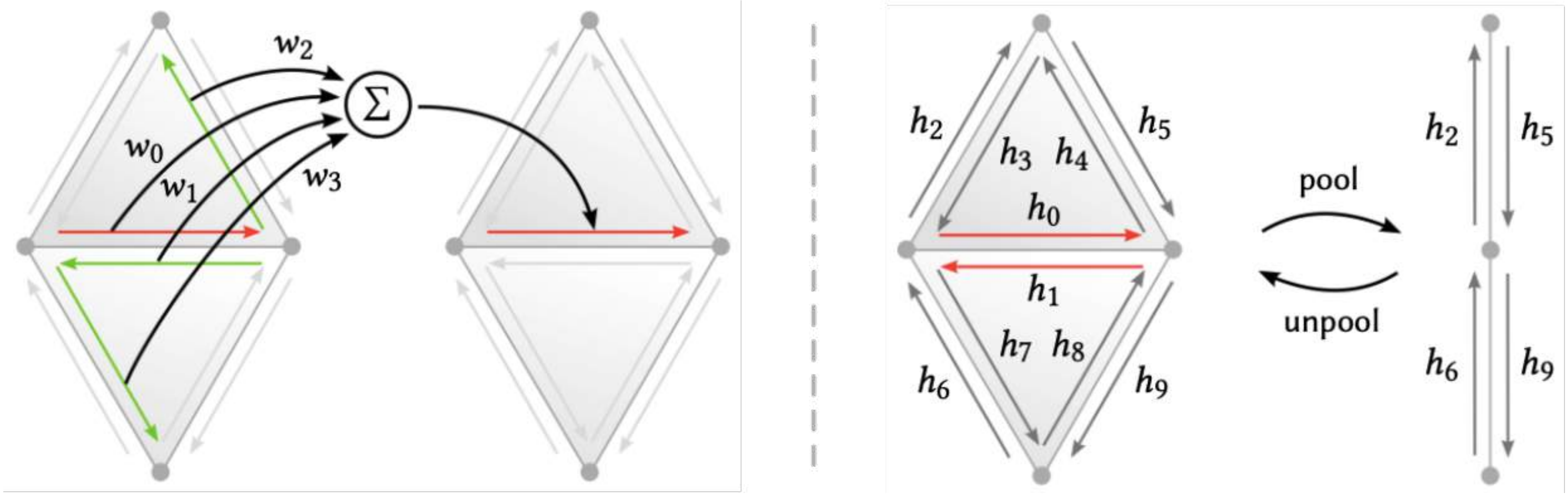
Rigid Motion Invariant



train mesh | test mesh | trained upsampling w/o half-edge | trained upsampling with half-edge

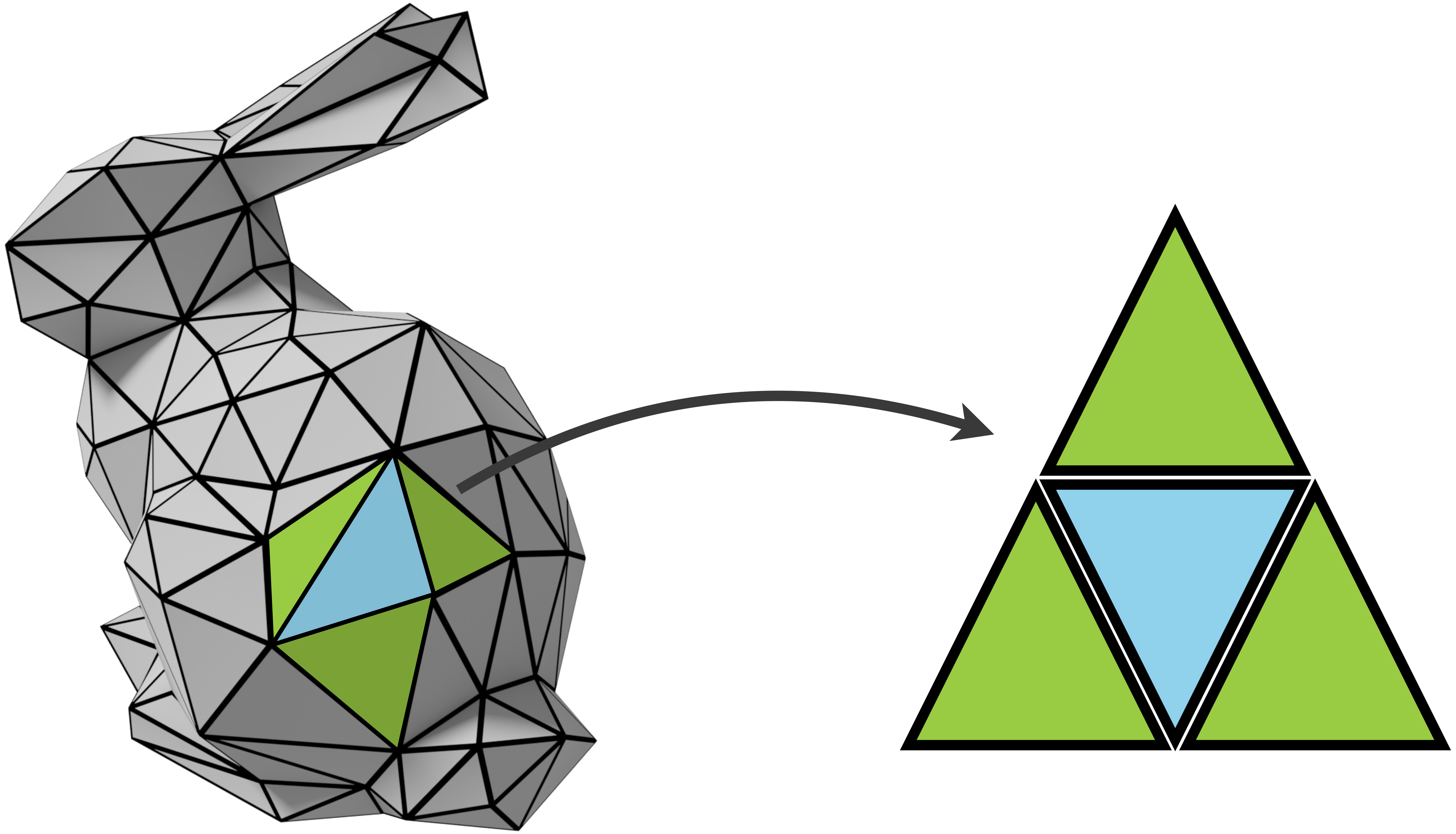
Liu et al. 2020

HalfedgeCNN



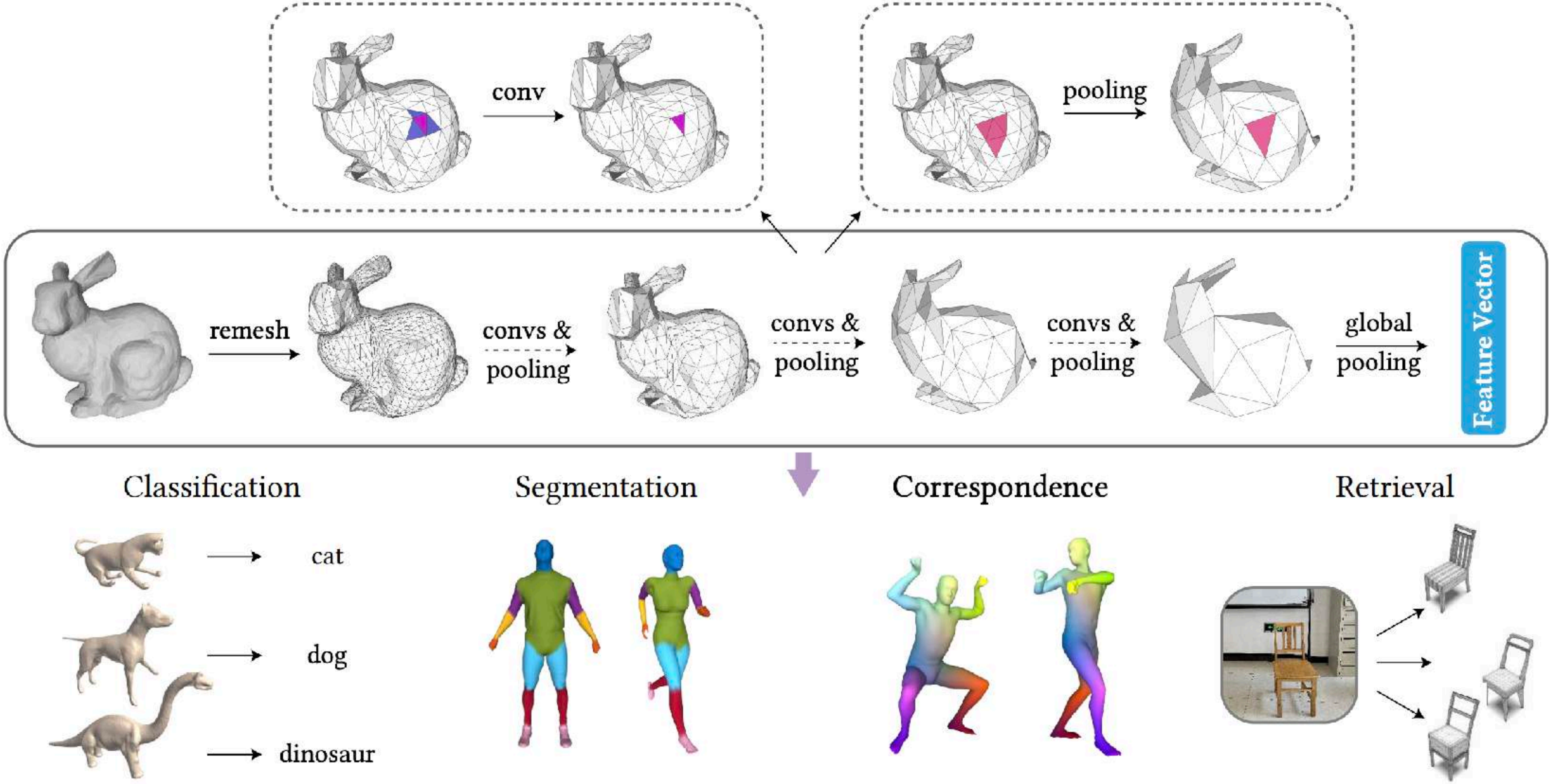
Ludwig et al. 2023

Face Convolution



Hertz et al. 2020, Hu et al. 2022

SubdivNet



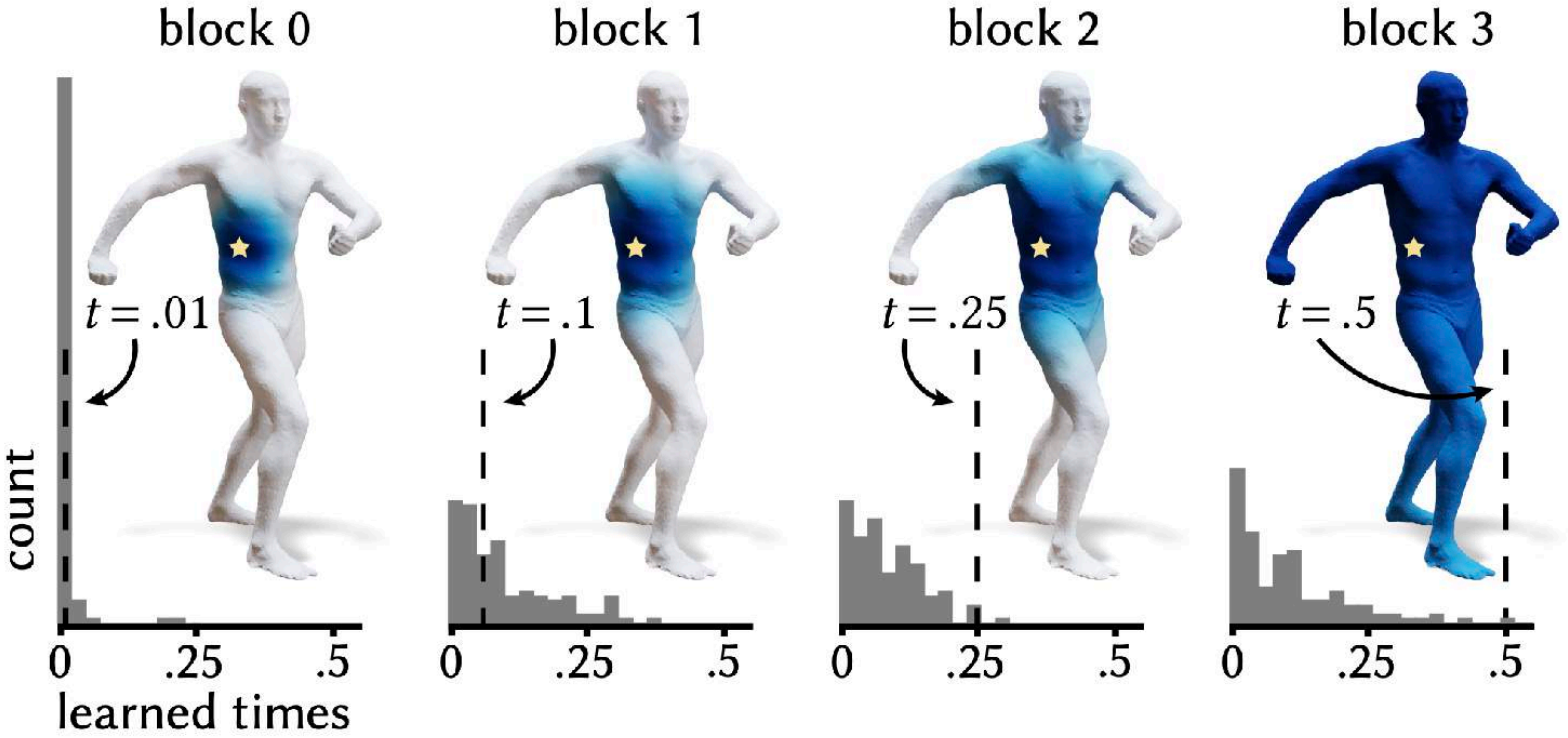
Hu et al. 2022

Discrete Mesh Convolution

- Becomes more popular recently
- Dependent to the discretization

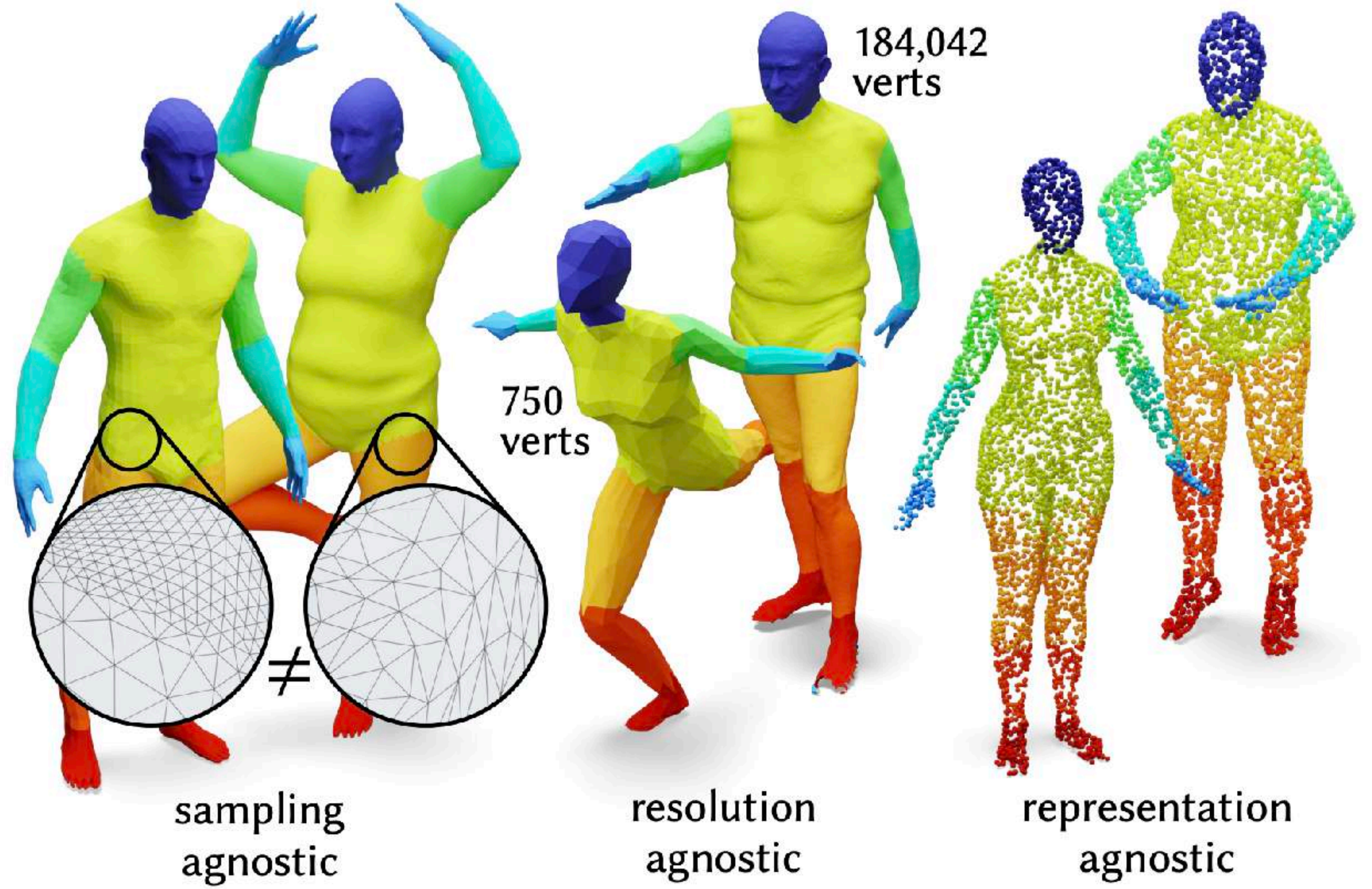
PDE-based Convolution

- (Heat) DiffusionNet != Diffusion Models



Sharp et al. 2022

More Robust to Discretization

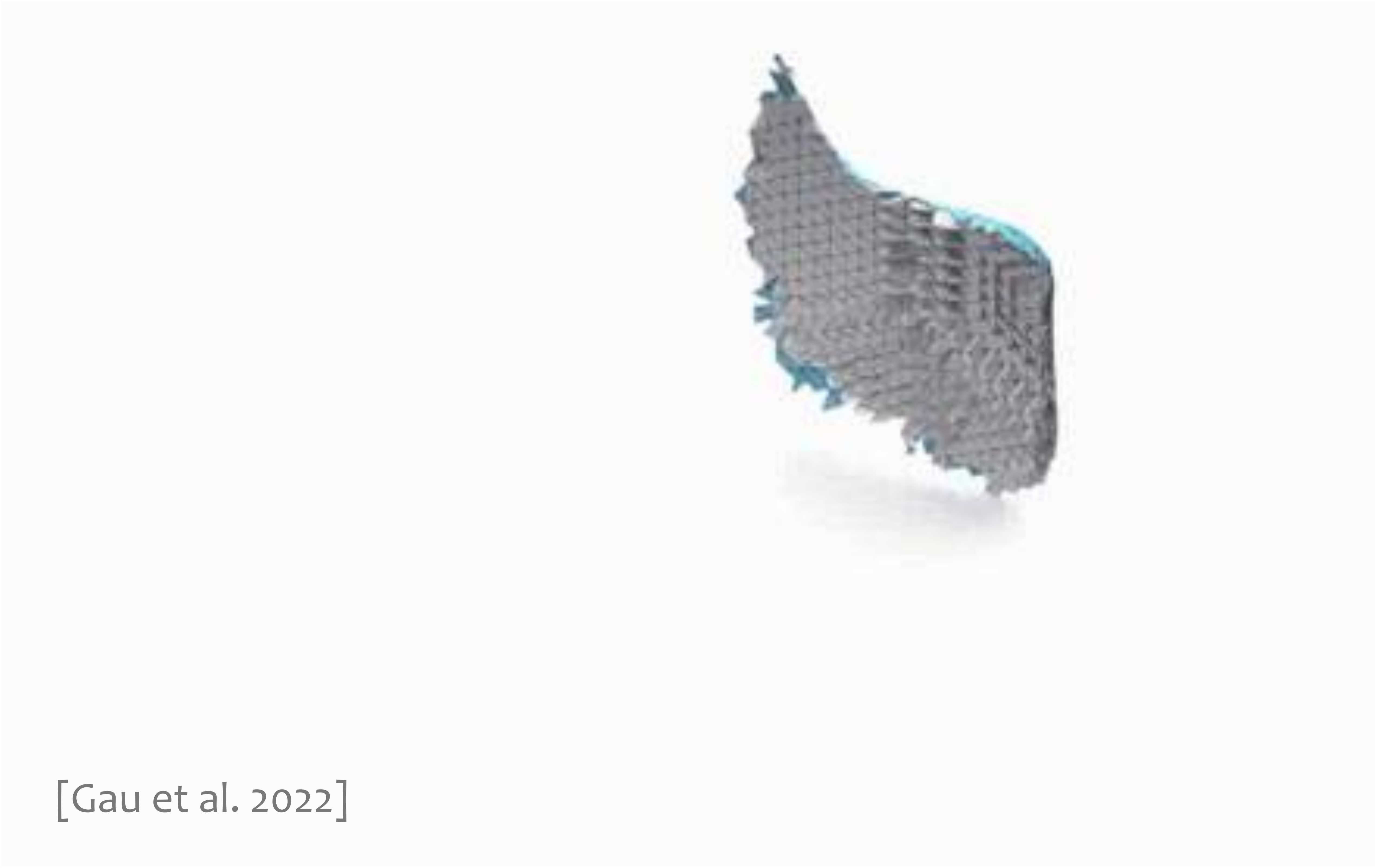


Sharp et al. 2022

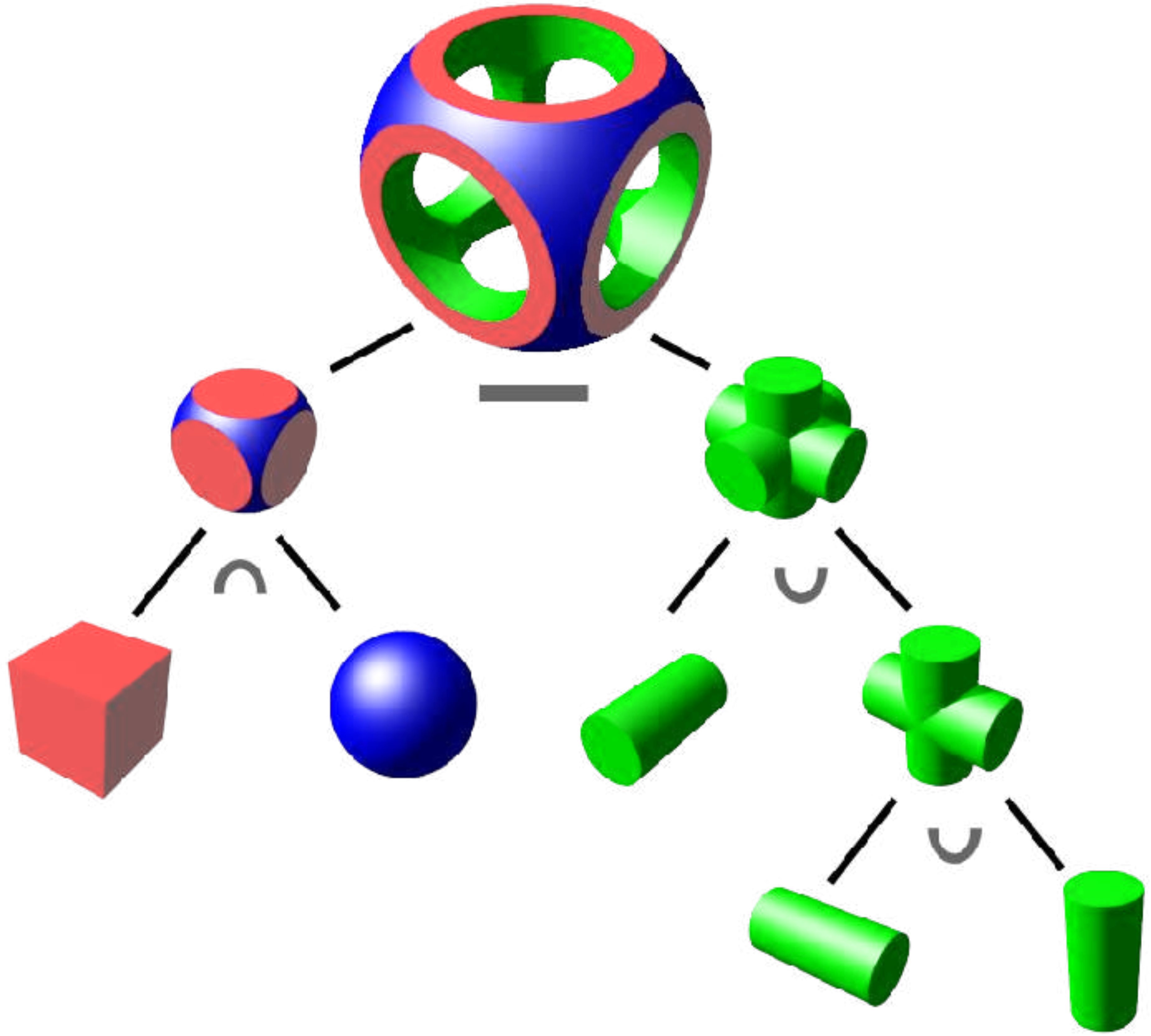
Forward Looking

Extending to Solid Geometry (instead of Surface)

- Tet / Hex meshes
- Constructive Solid Geometry

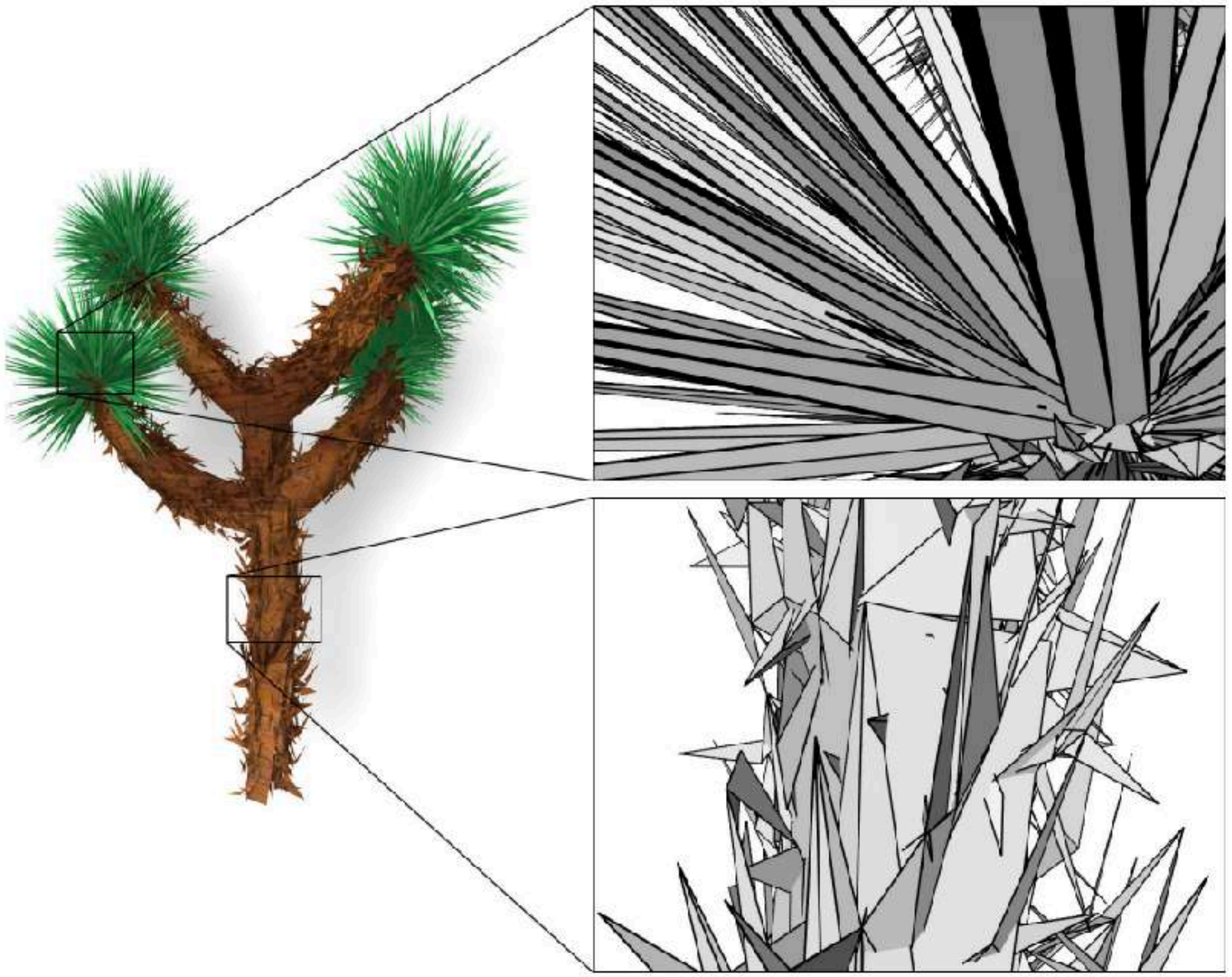


[Gau et al. 2022]

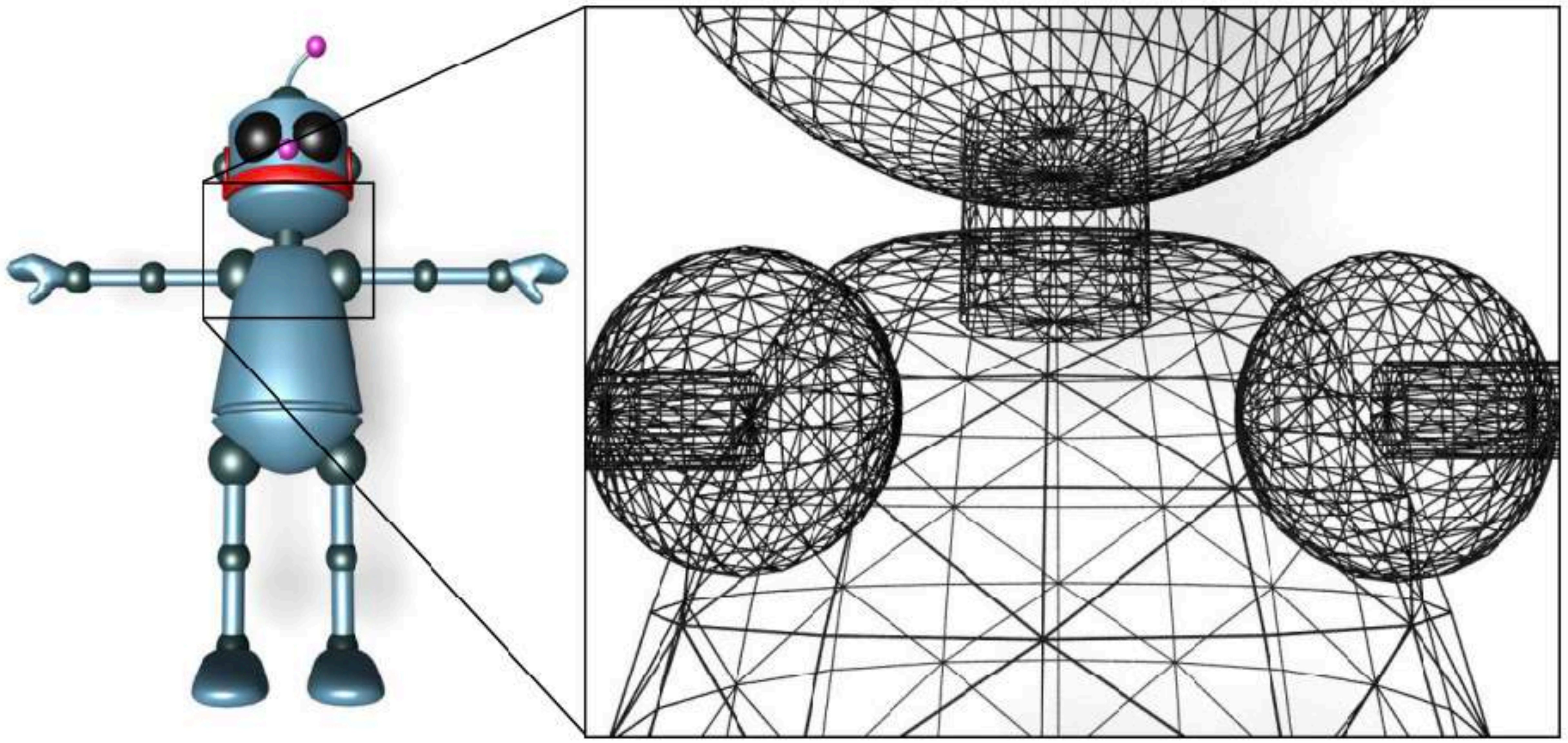


Representation Agnostic Convolution

Robust to Mesh Defects

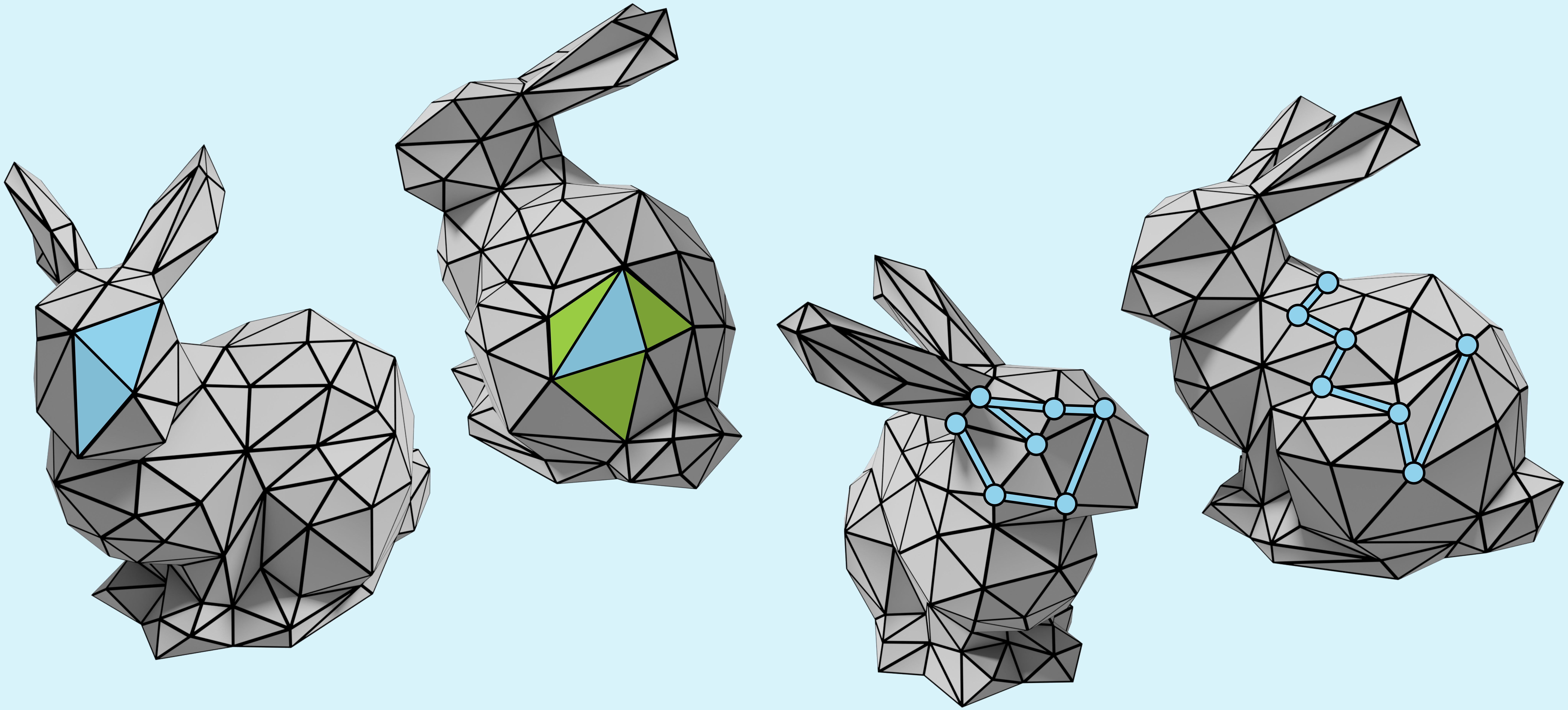


triangle soups



multiple components

source: Spillmann et al. 2006



Geometric Learning on Discrete Surface Meshes

Hsueh-Ti Derek Liu
hsuehtil@gmail.com