Lecture 26:

3D Reconstruction and Generation

Computer Graphics and Imaging

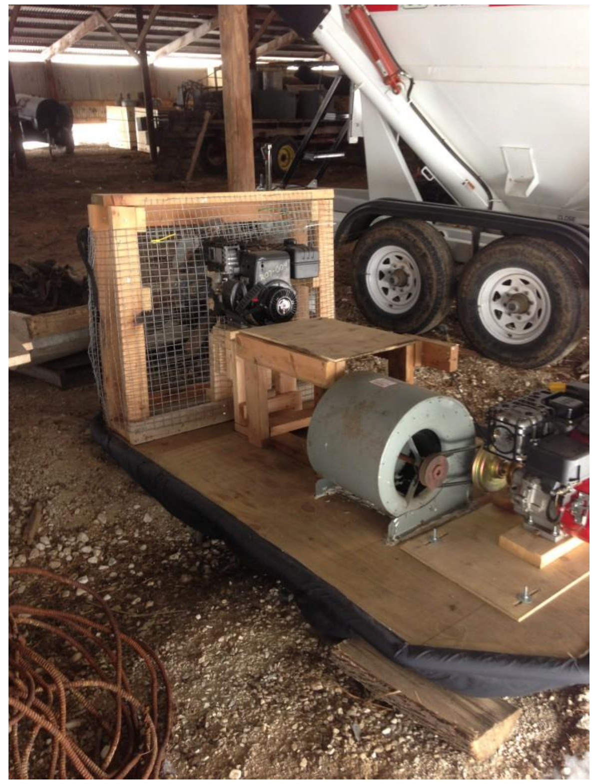
UC Berkeley CS184

Ethan Weber

advised by
Professor Angjoo Kanazawa
at UC Berkeley

Wisconsin

-2016



MIT

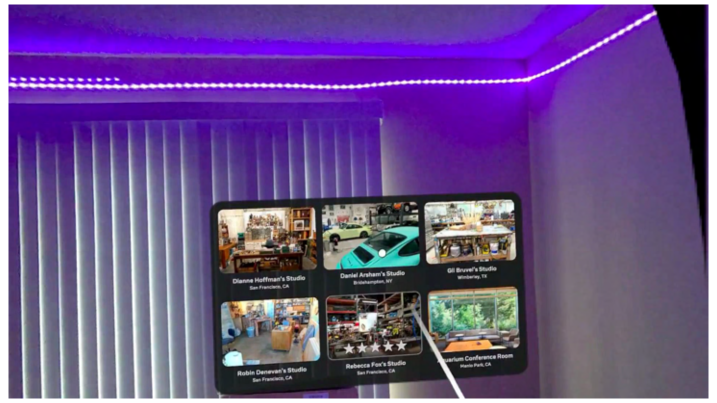
2016-2021

Berkeley

2021-2025

Meta Reality Labs

2025-



Acknowledgements

Thanks to Ethan Webber and Matthew Tancik the slides!

Overview

1

3D Reconstruction

- Structure-from-Motion
- Novel-View Synthesis
- Open-Source Tools
- Limitations

2

3D Generation

- Text-Only Conditioned
- Image & Pose Conditioned
- Scene Completion
- Large-Scale Datasets

3

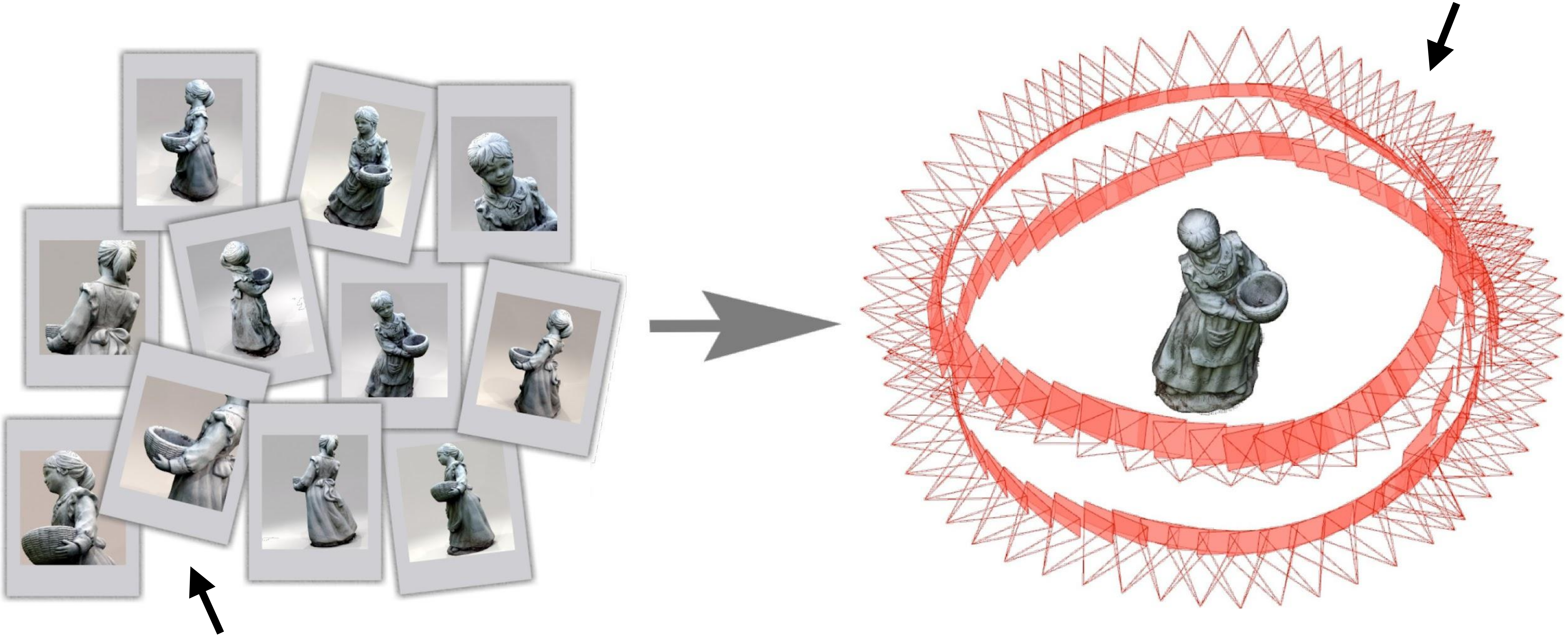
NeRF

- Large-Scale Datasets
- Scene Understanding
- Robotics Manipulation

3D Reconstruction

Structure-from-Motion

Recovered Camera Parameters



Input Images

https://www.mdpi.com/2313-433X/4/8/98

3D Reconstruction

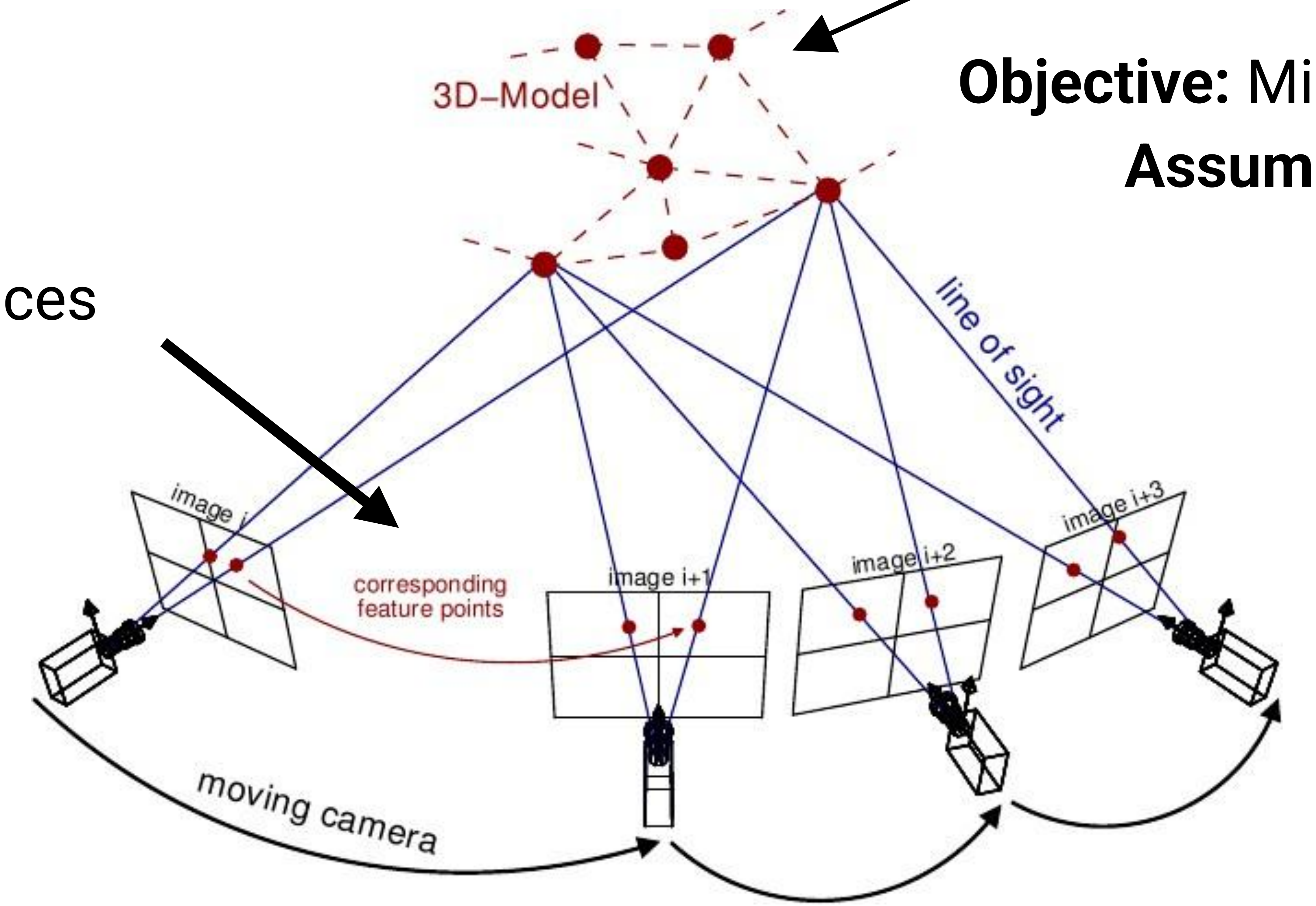
Structure-from-Motion

Backprojected 3D Points

Objective: Minimize reprojection error

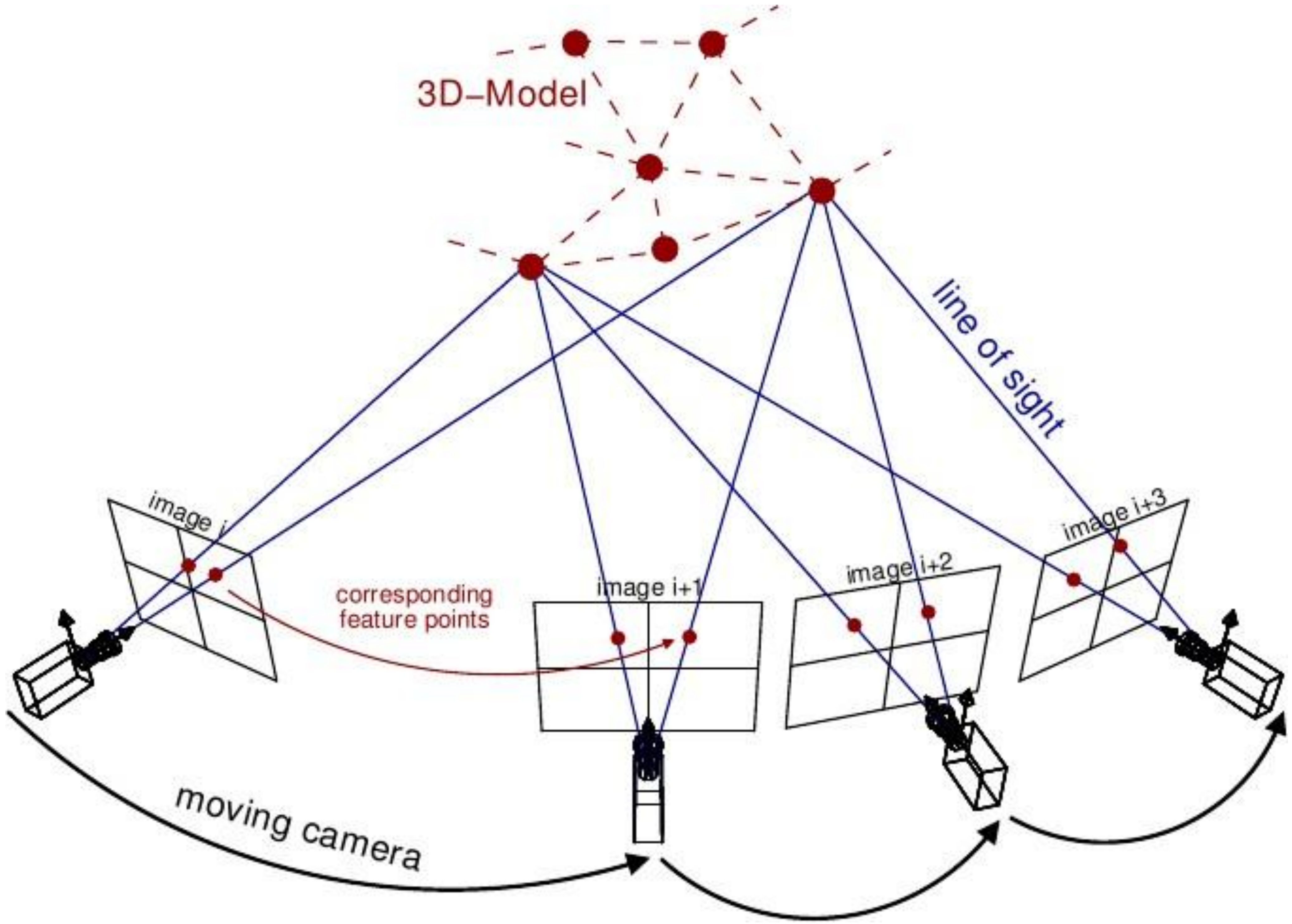
Assumptions: Camera type

2D Correspondences



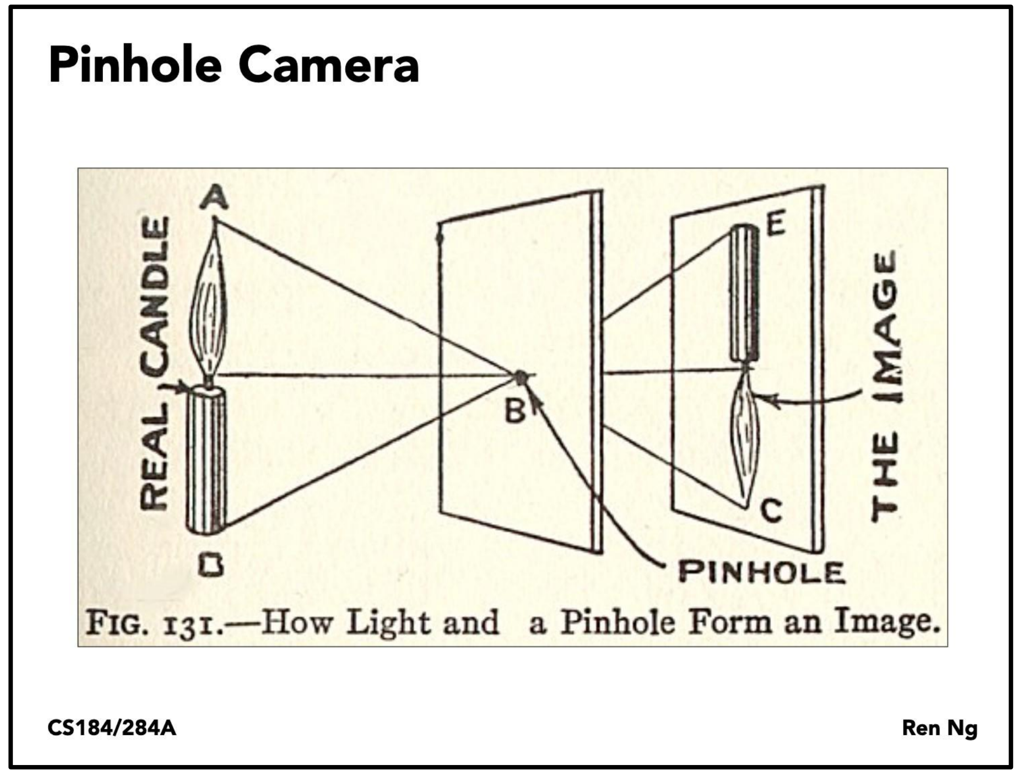
3D Reconstruction

Structure-from-Motion



Objective: Minimize reprojection error

Assumptions: Camera type



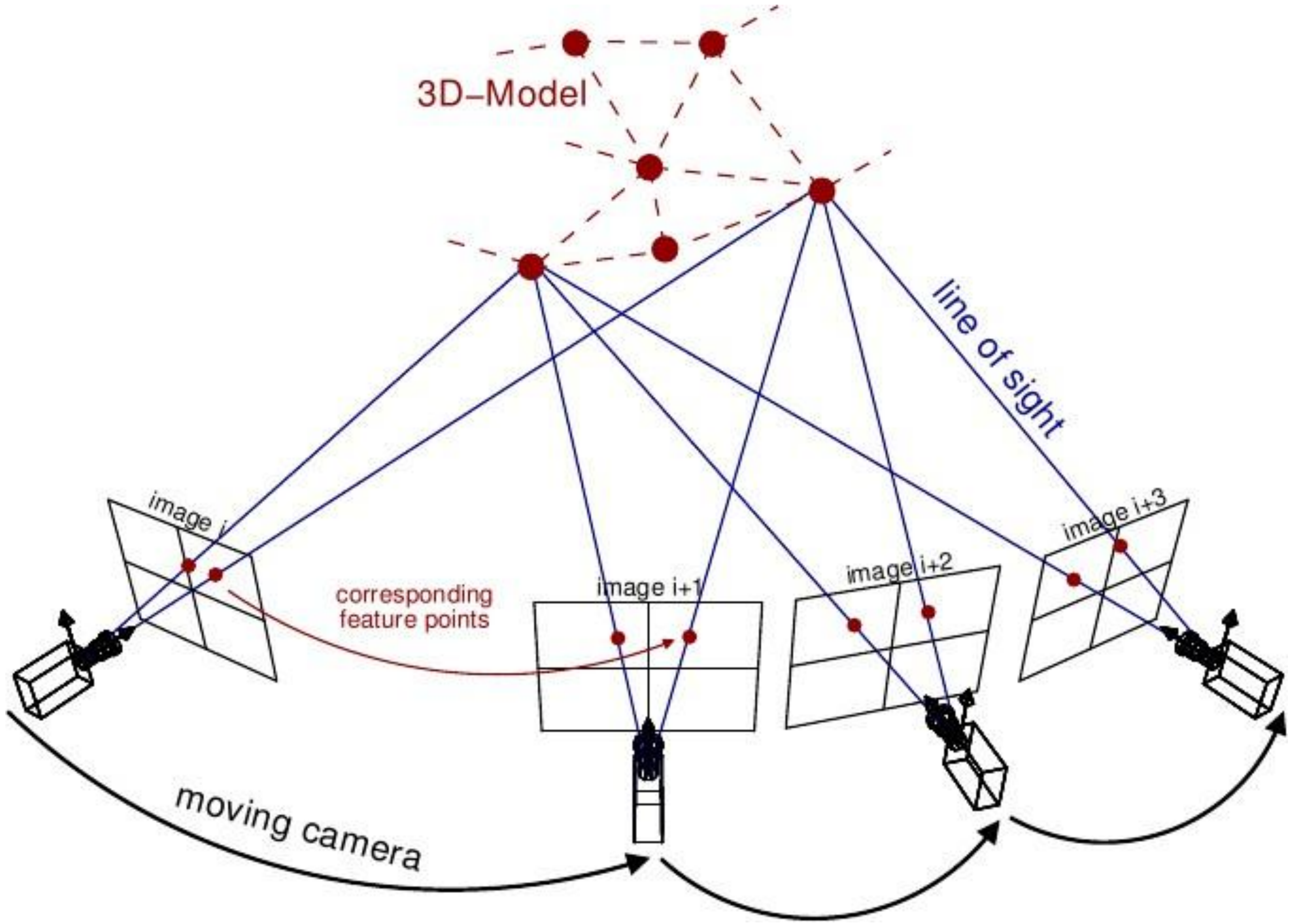
for pinhole cameras, we solve for focal length(s) and distortion parameters

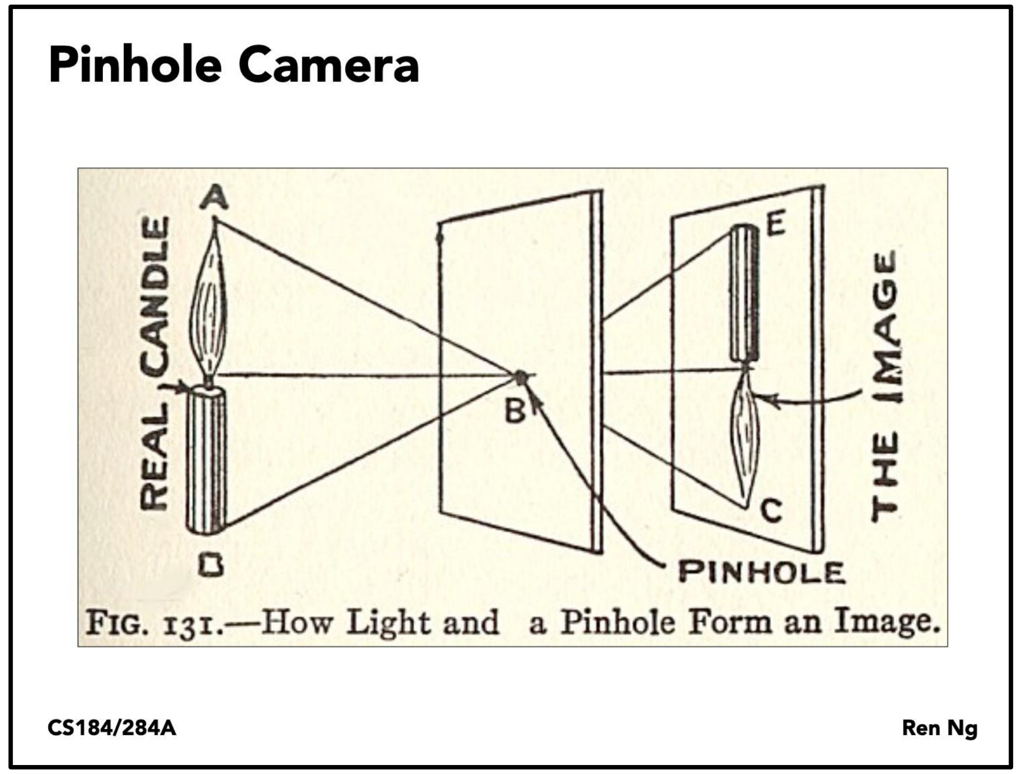
3D Reconstruction

Structure-from-Motion

Objective: Minimize reprojection error

Assumptions: Camera type





for pinhole cameras, we solve for focal length(s) and distortion parameters

and we adjust the transforms to achieve a low error

3D Reconstruction

Structure-from-Motion

Structure-from-Motion Revisited 6885
JL Schönberger, JM Frahm
Conference on Computer Vision and Pattern Recognition (CVPR), 2016

COLMAP

3D Reconstruction on Challenging Data

“The One Where They Reconstructed 3D Humans and Environments in TV Shows”

Georgios Pavlakos*, Ethan Weber*, Matthew Tancik, Angjoo Kanazawa

University of California, Berkeley

BAIR
BERKELEY ARTIFICIAL INTELLIGENCE RESEARCH


CAPTAIN
FUTURE
WIZARD OF SCIENCE

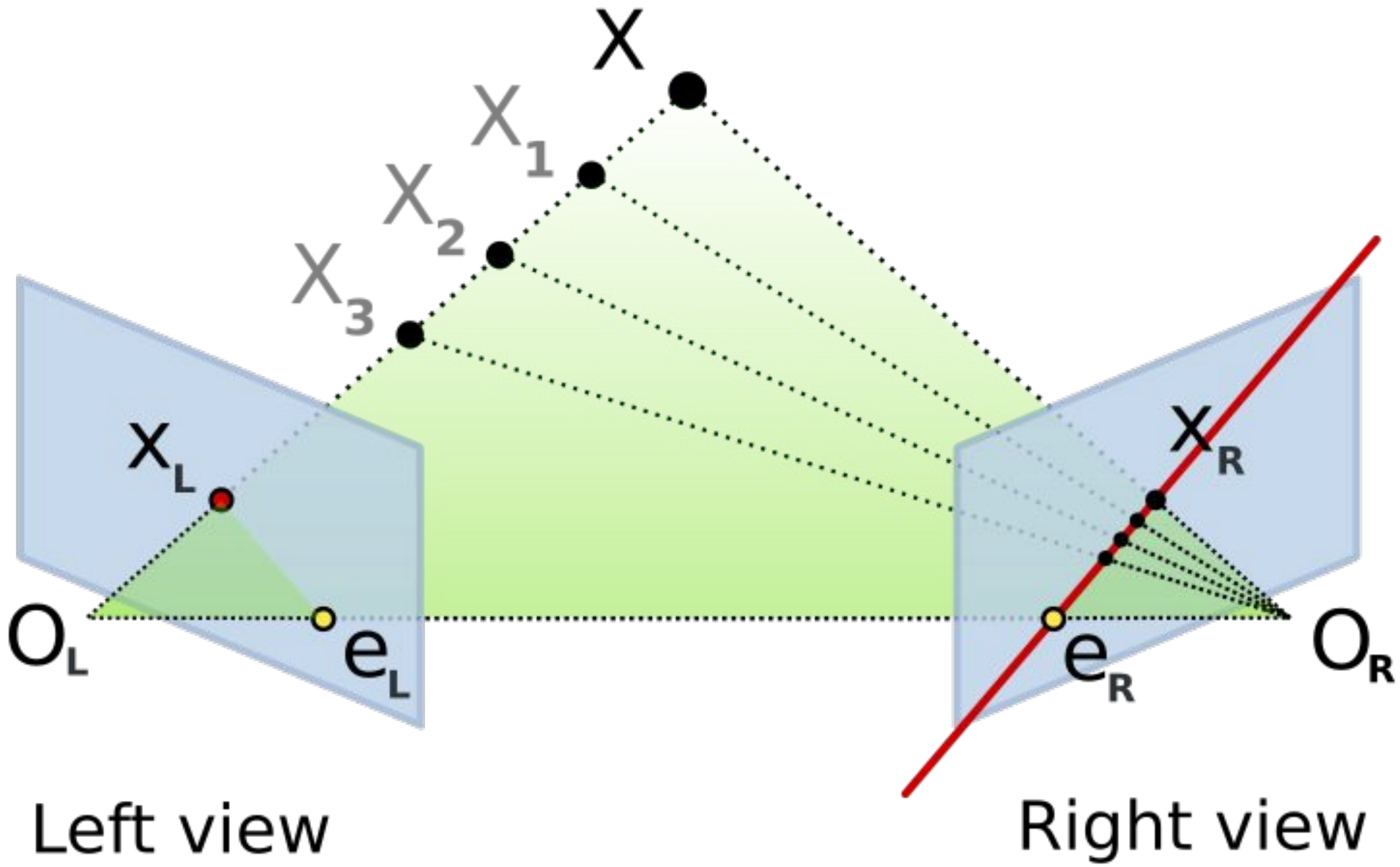

FUTURE
WIZARD OF SCIENCE


FORTUNE COOKIES
INDIVIDUALLY WRAPPED

Structure-from-Motion is robust to this data

Why is SfM so robust?

Epipolar Geometry and Ransac



Works for perspective cameras!

What if you don’t have perspective cameras?

These are all hand-drawn!

Toon3D: Seeing Cartoons from New Perspectives

Ethan Weber*, Riley Peterlinz*, Rohan Mathur, Frederik Warburg, Alexei A. Efros, and Angjoo Kanazawa

Input Images

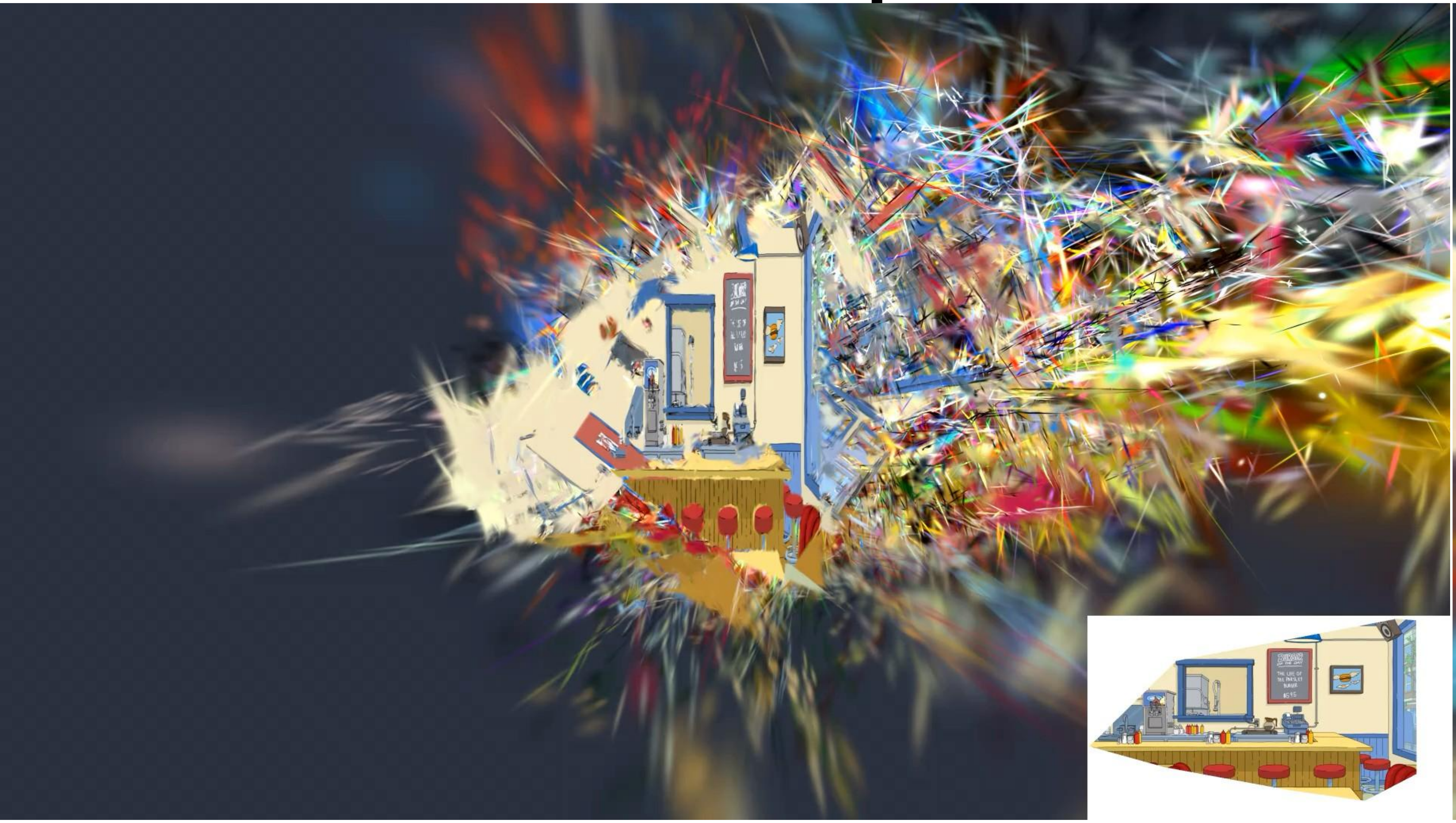
COLMAP

COLMAP with Manual Correspondences

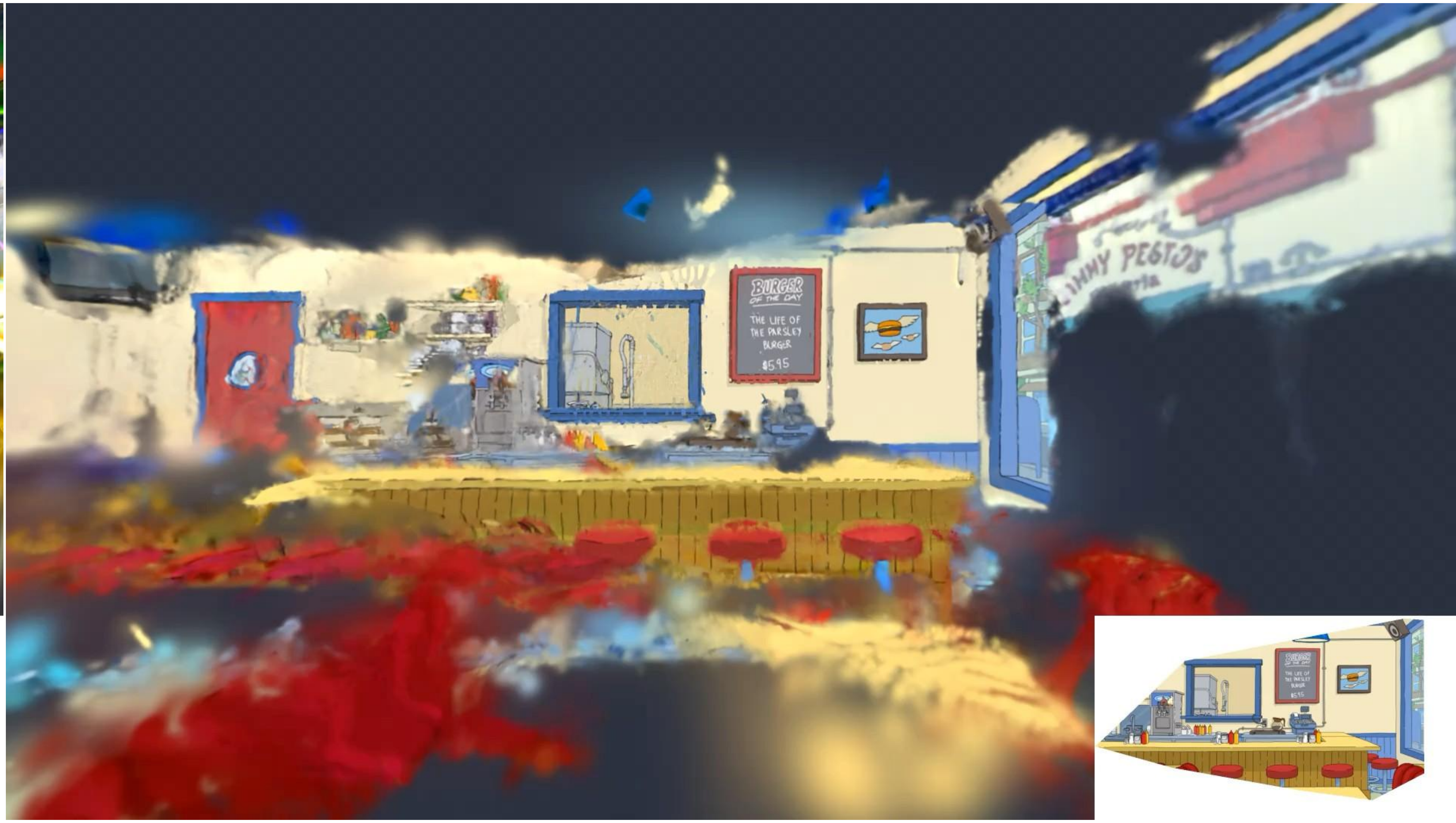
Bundle Adjustment with Manual Correspondences (no outlier rejection)

Toon3D (our method)

Bundle Adjustment with Manual Correspondences

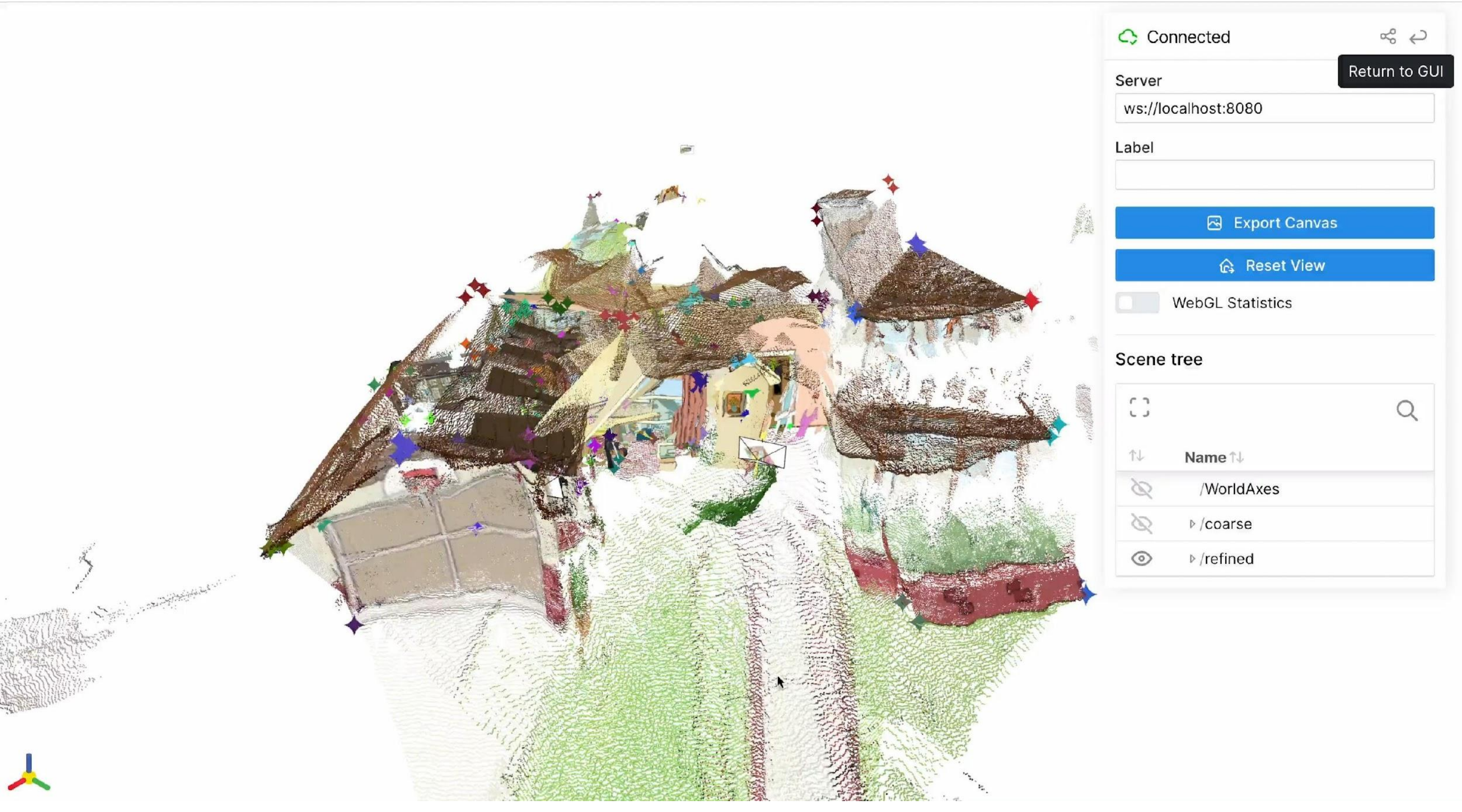


Toon3D (our method)



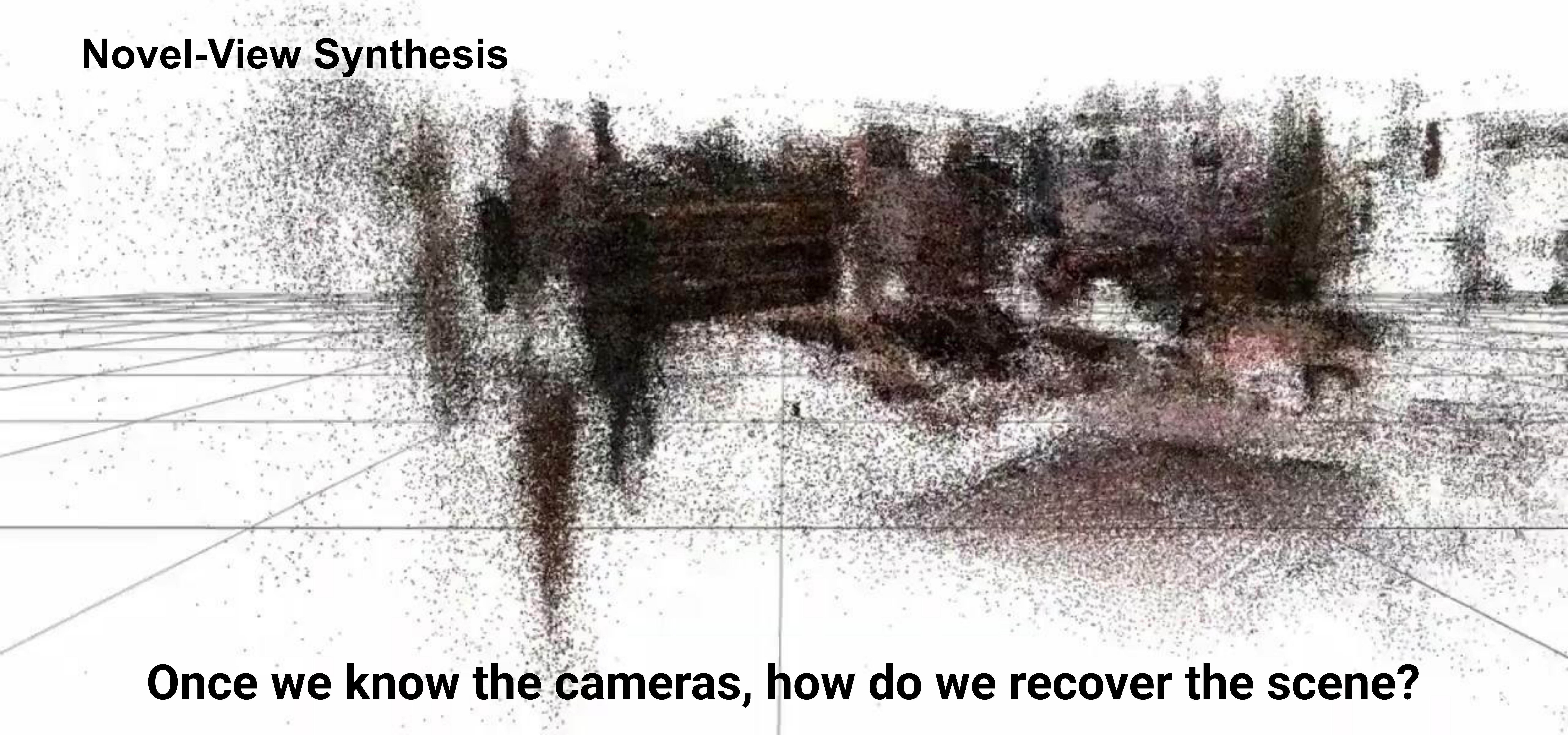
How far can we push the limits of SfM on cartoons?

How far can we push the limits of SfM on cartoons?


Connected
Return to GUI
Server
ws://localhost:8080
Label
Export Canvas
Reset View
WebGL Statistics
Scene tree
Name
/WorldAxes
/coarse
/refined

3D Reconstruction

Novel-View Synthesis

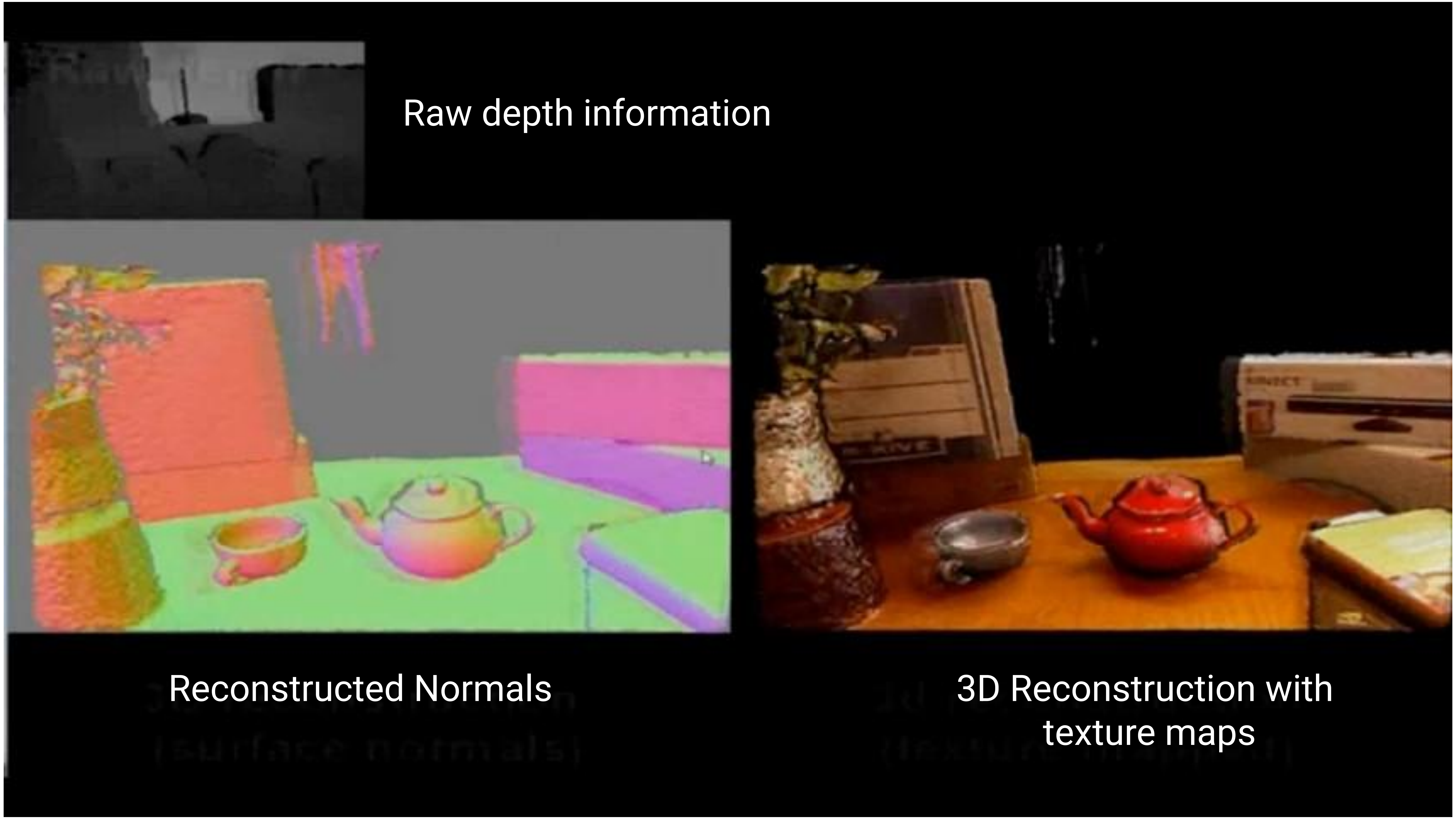


Once we know the cameras, how do we recover the scene?

If we knew the depth, that would be helpful…

Microsoft Kinect

If we knew the depth, that would be helpful…



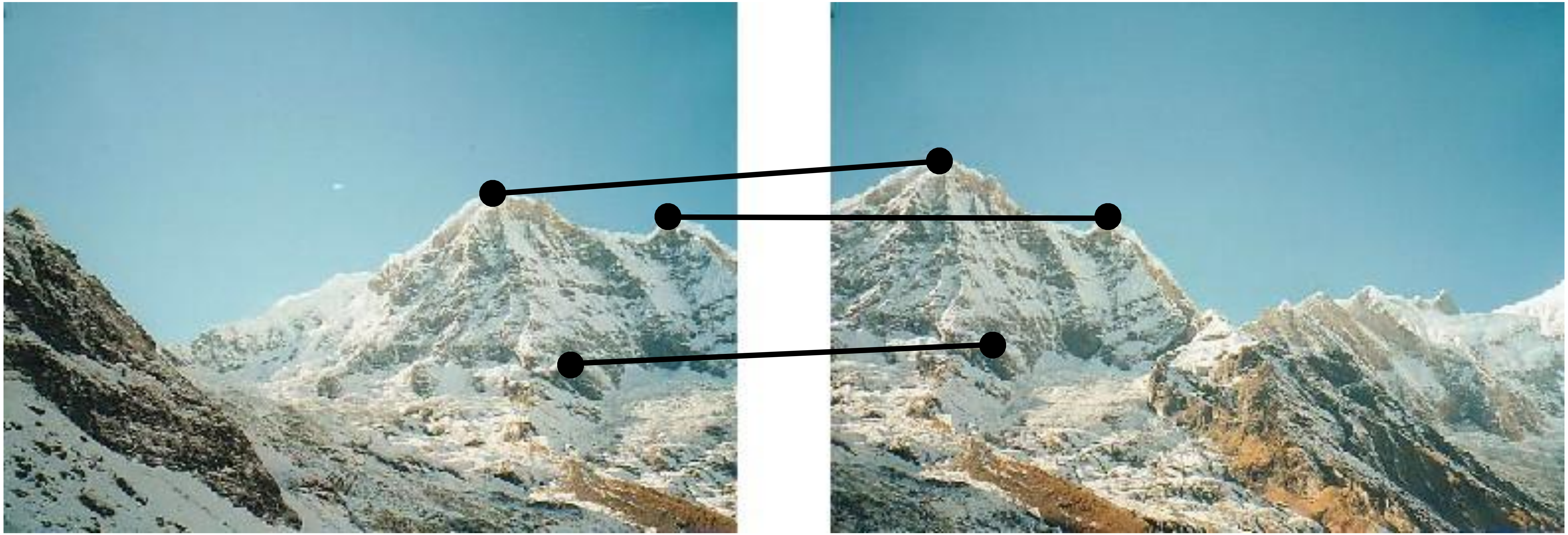
If we knew the depth, that would be helpful…

What if we don’t know the depth?

Point Cloud from Images

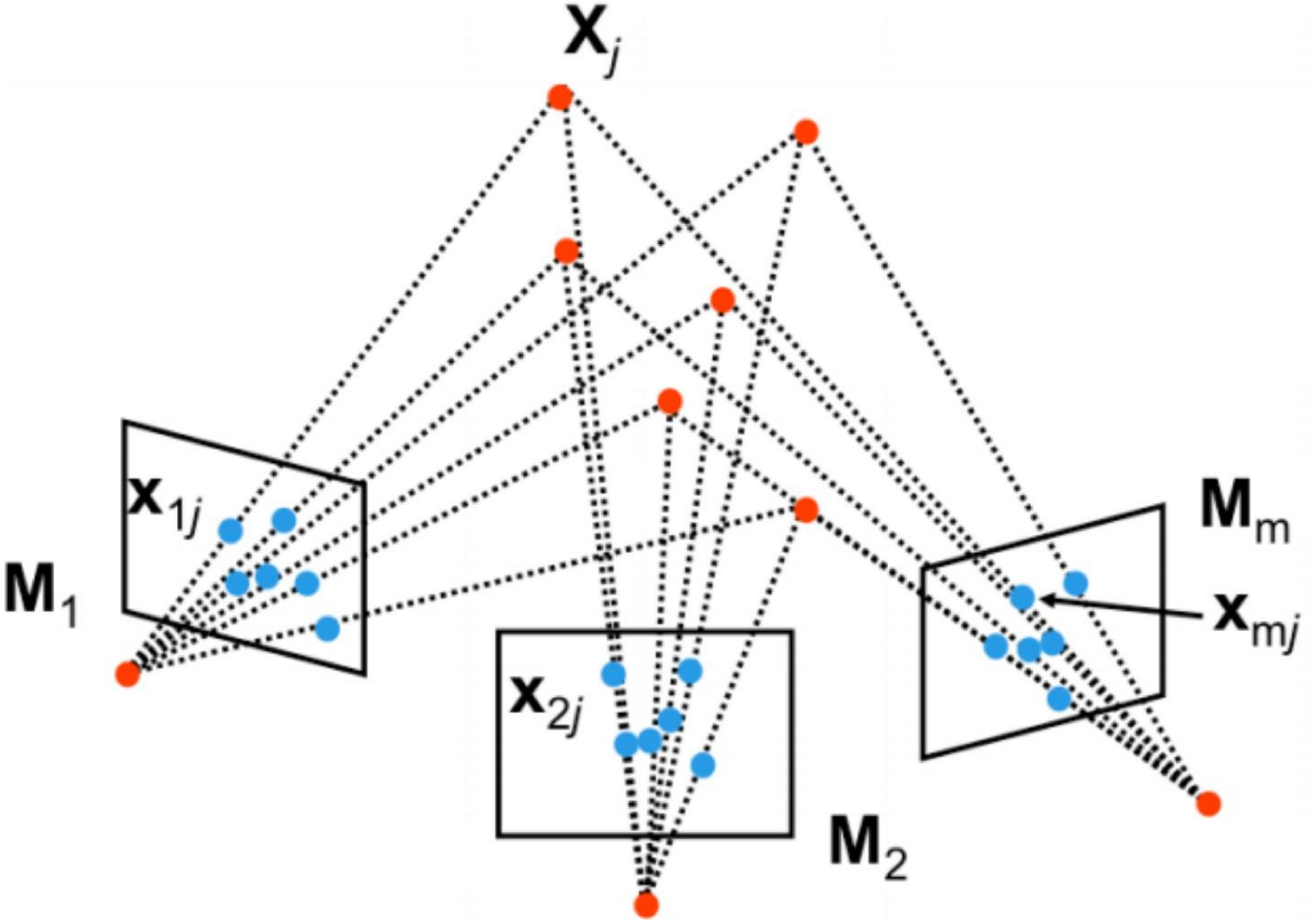
Brown and Lowe (2007)

Point Cloud from Images



Brown and Lowe (2007)

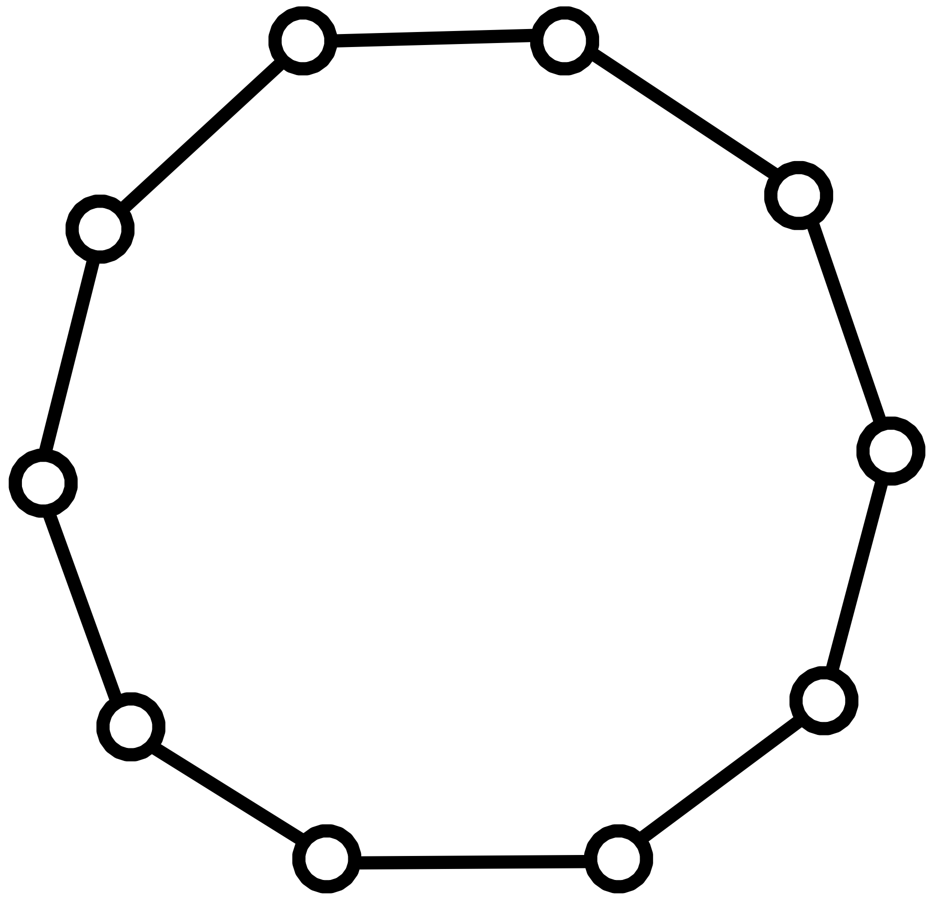
Point Cloud from Images



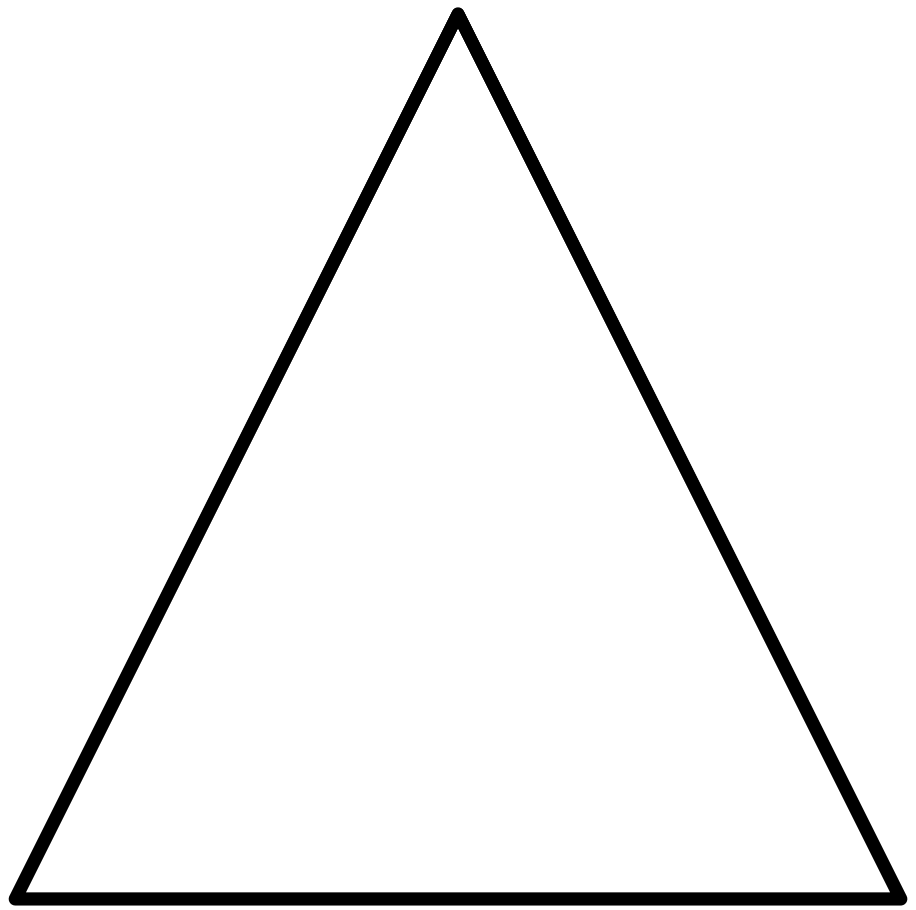
Hata and Savarese CS231a

Can we operate on the geometry directly?

Gradient Based Optimization

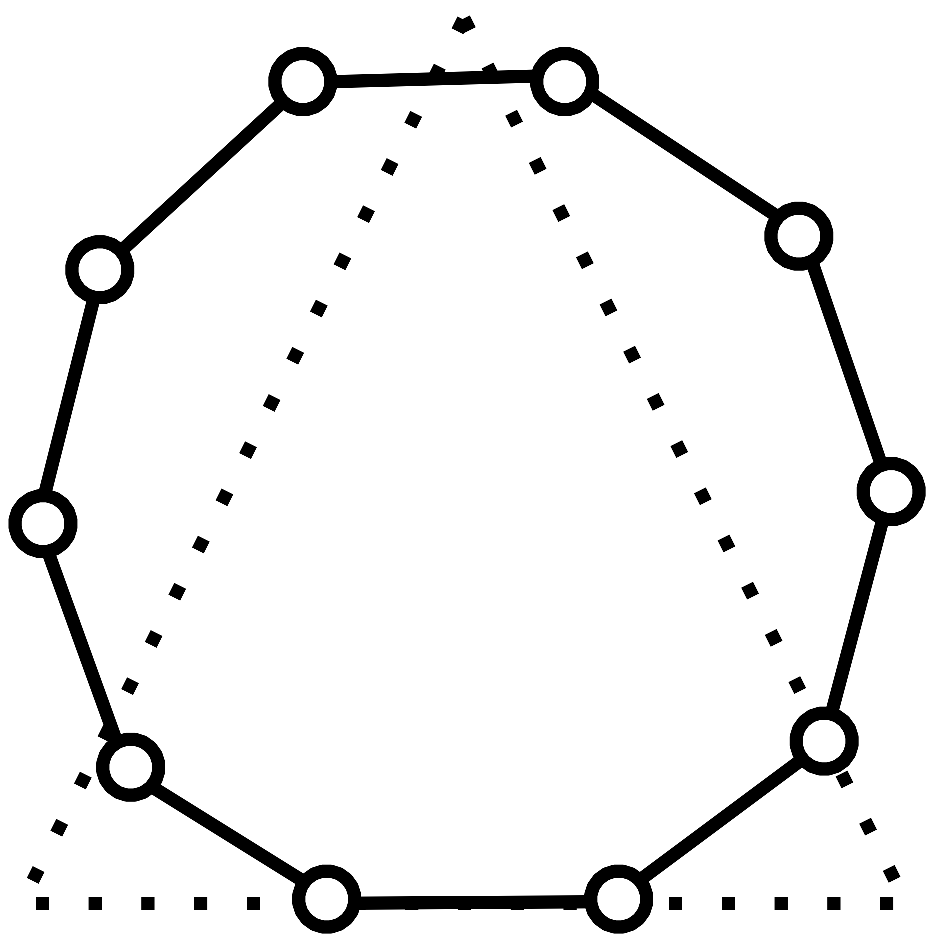


Initial Geometry

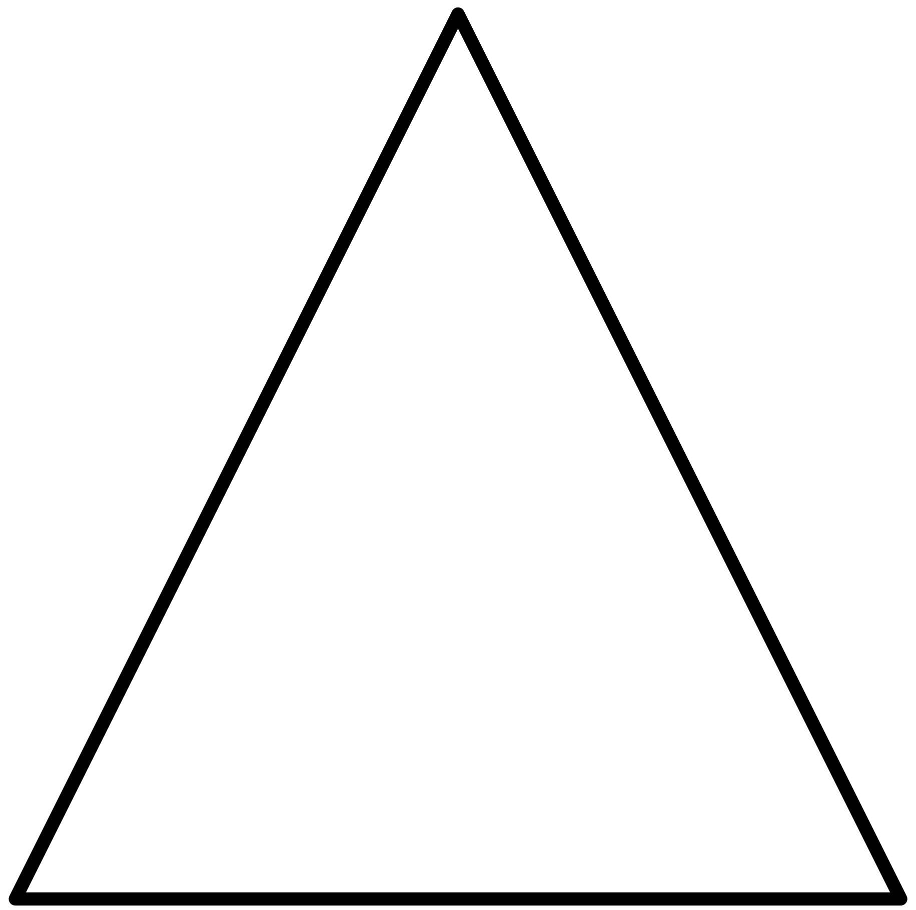


Target Geometry

Gradient Based Optimization

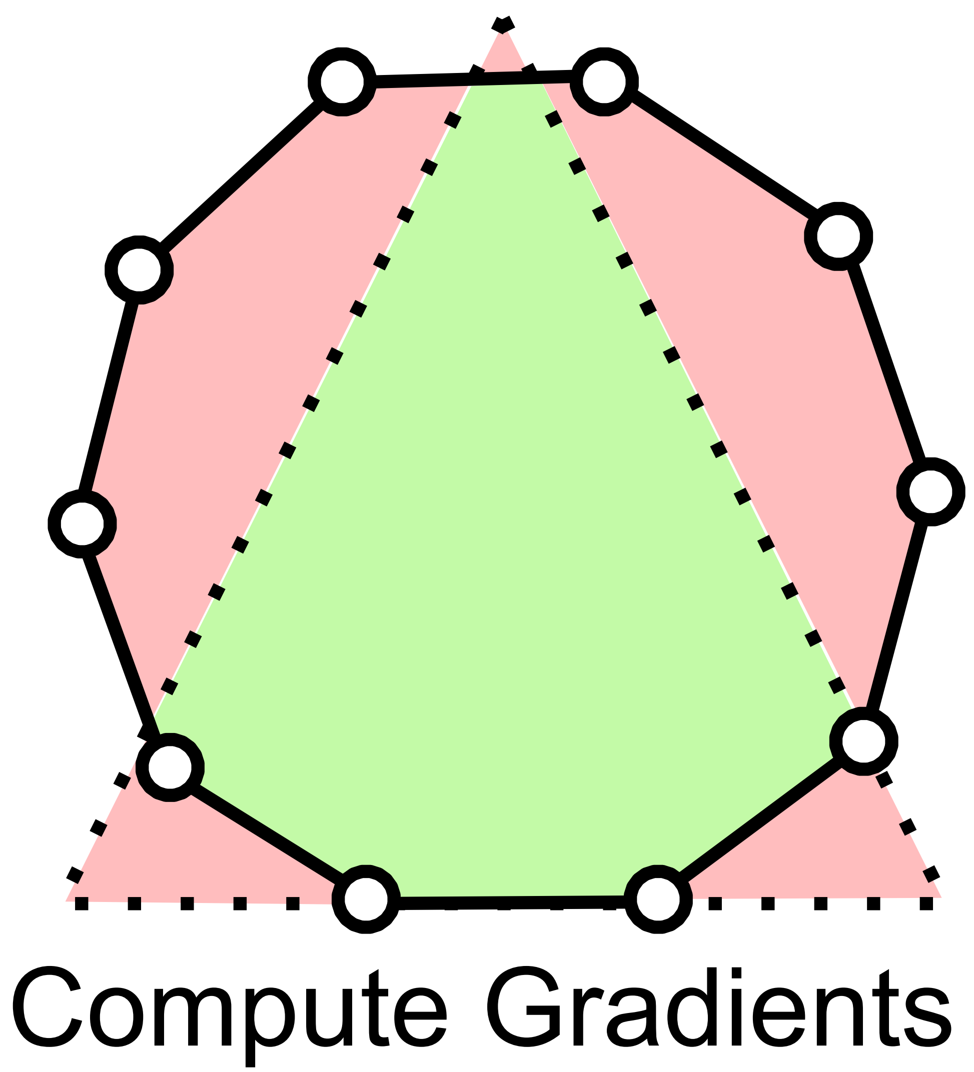


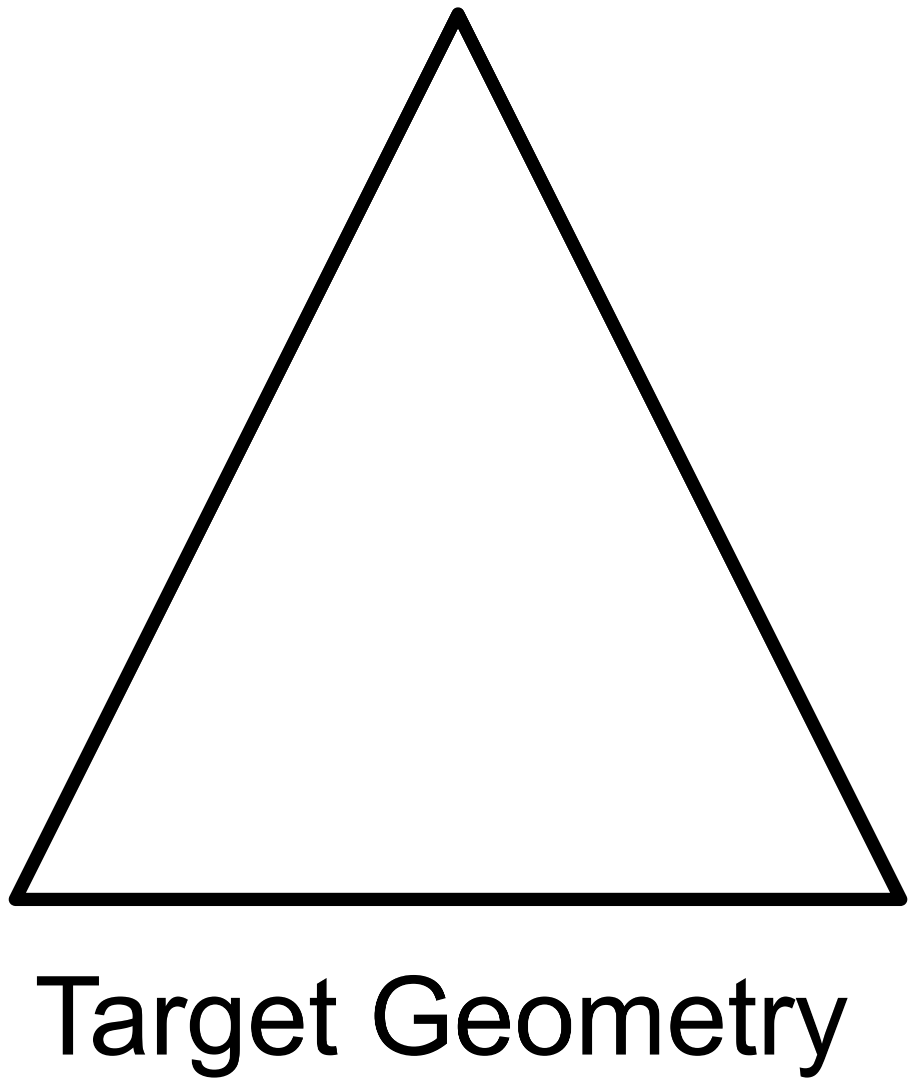
Initial Geometry



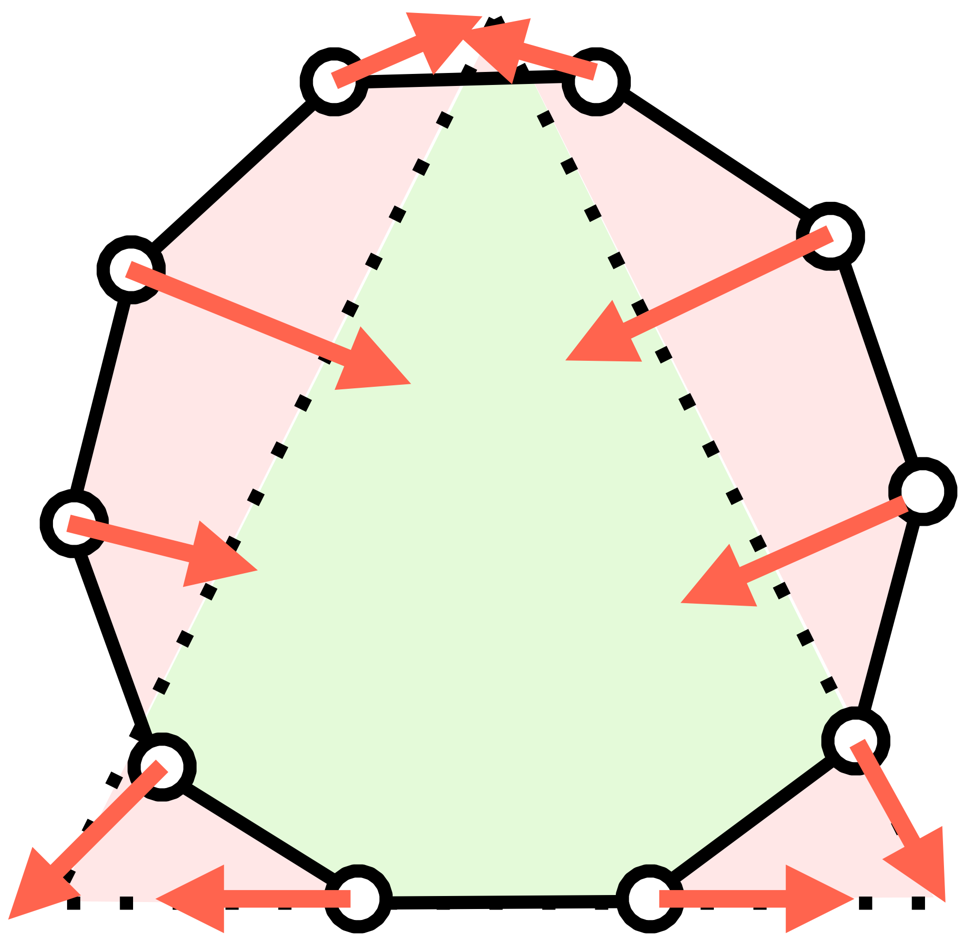
Target Geometry

Gradient Based Optimization

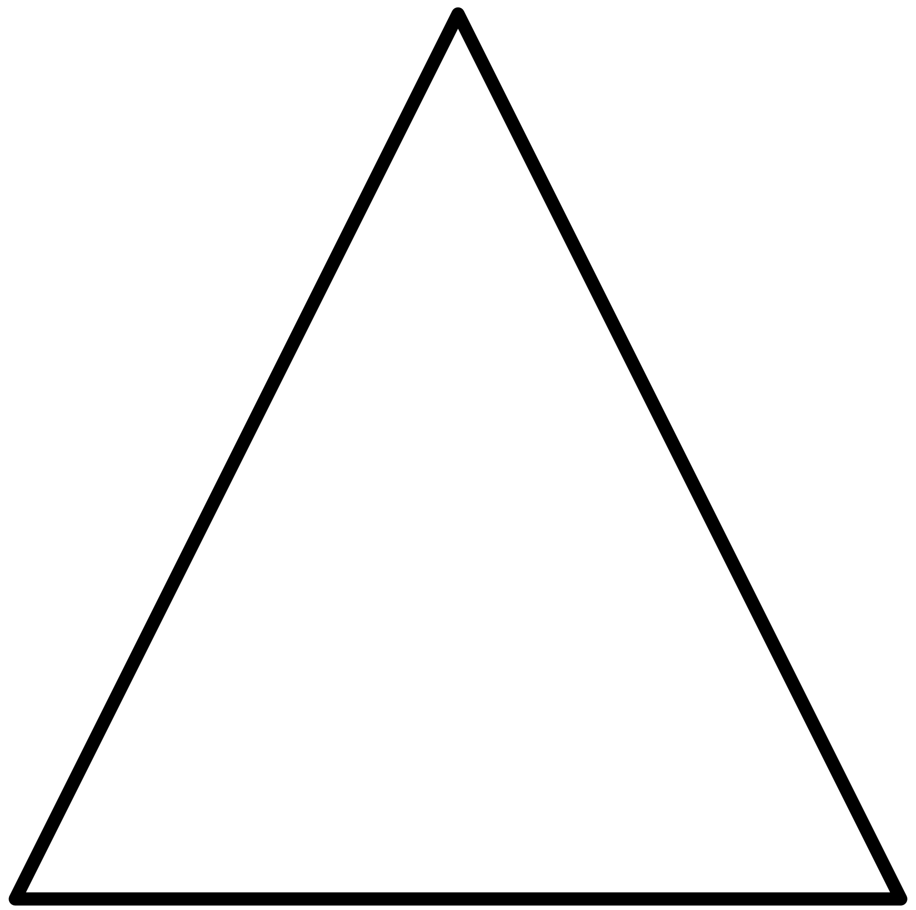




Gradient Based Optimization

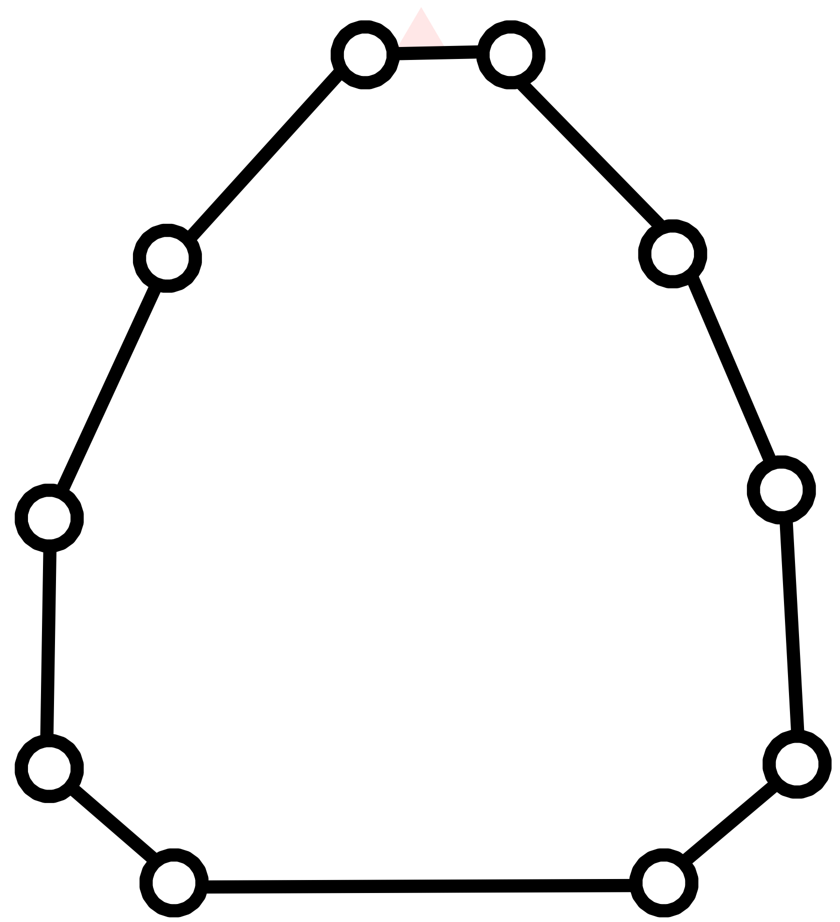


Compute Gradients

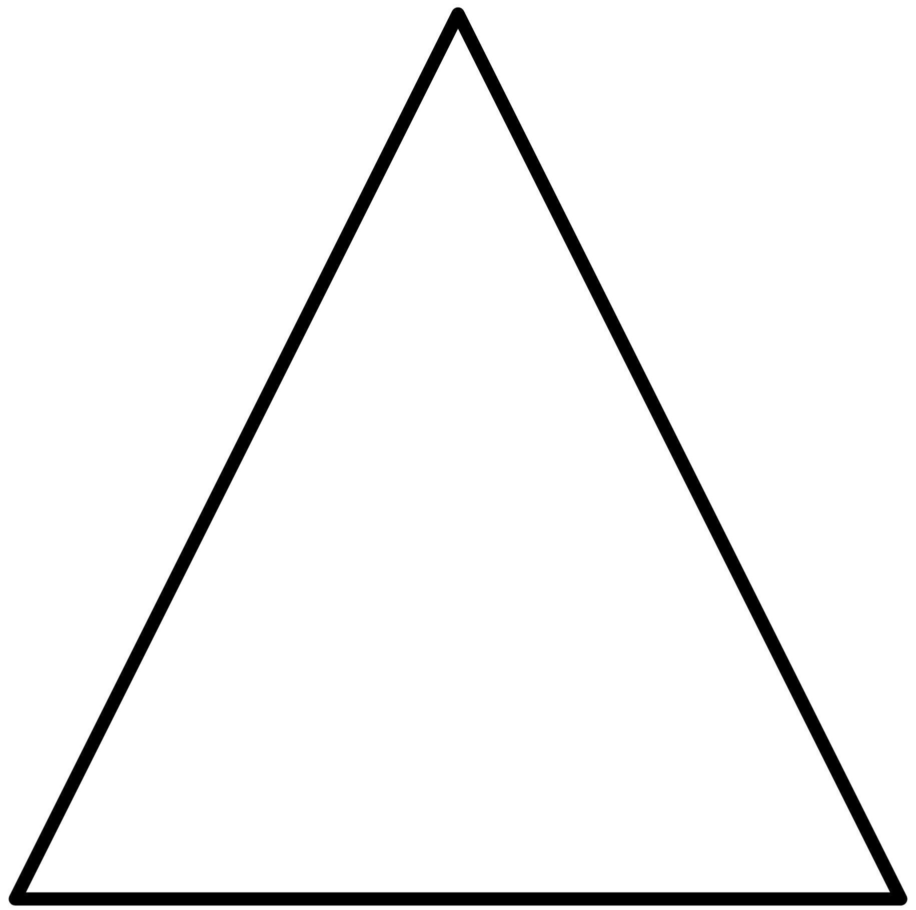


Target Geometry

Gradient Based Optimization

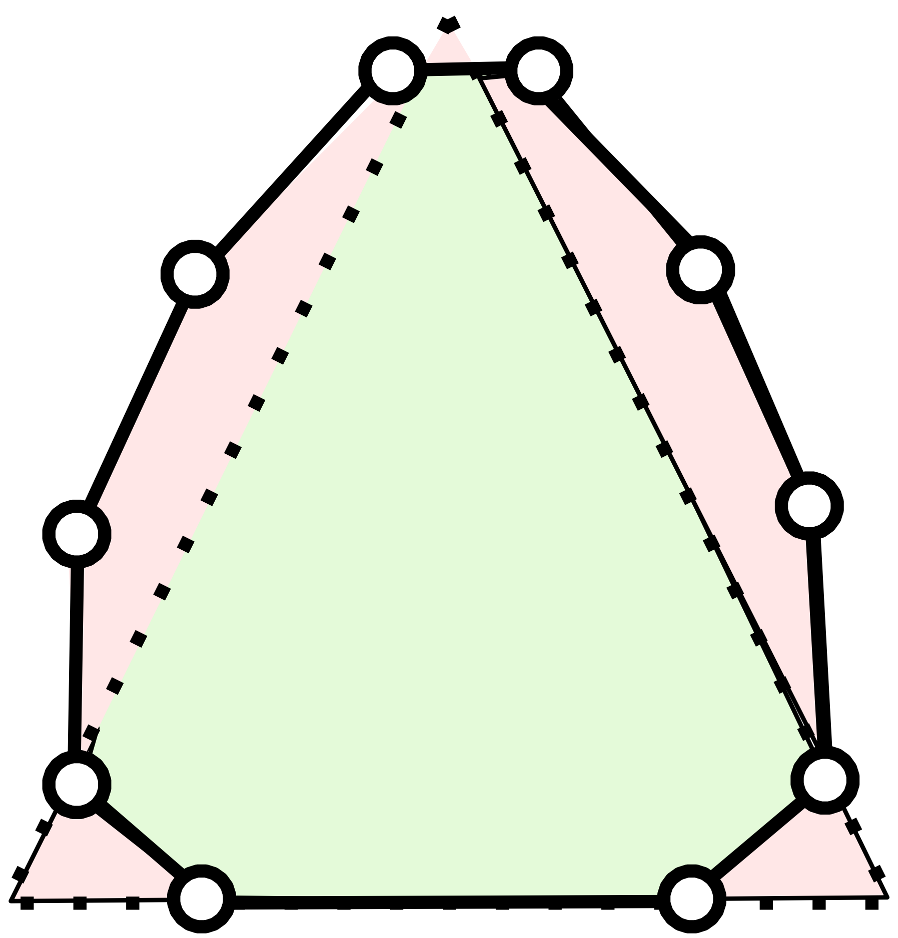


Update positions

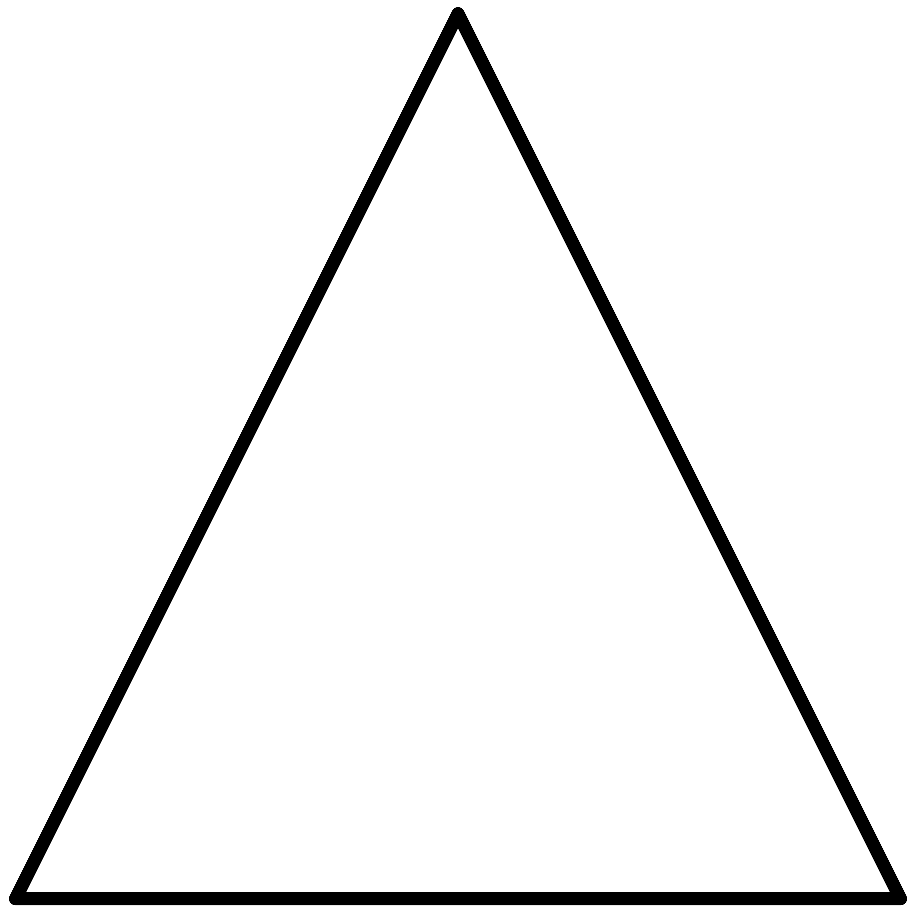


Target Geometry

Gradient Based Optimization

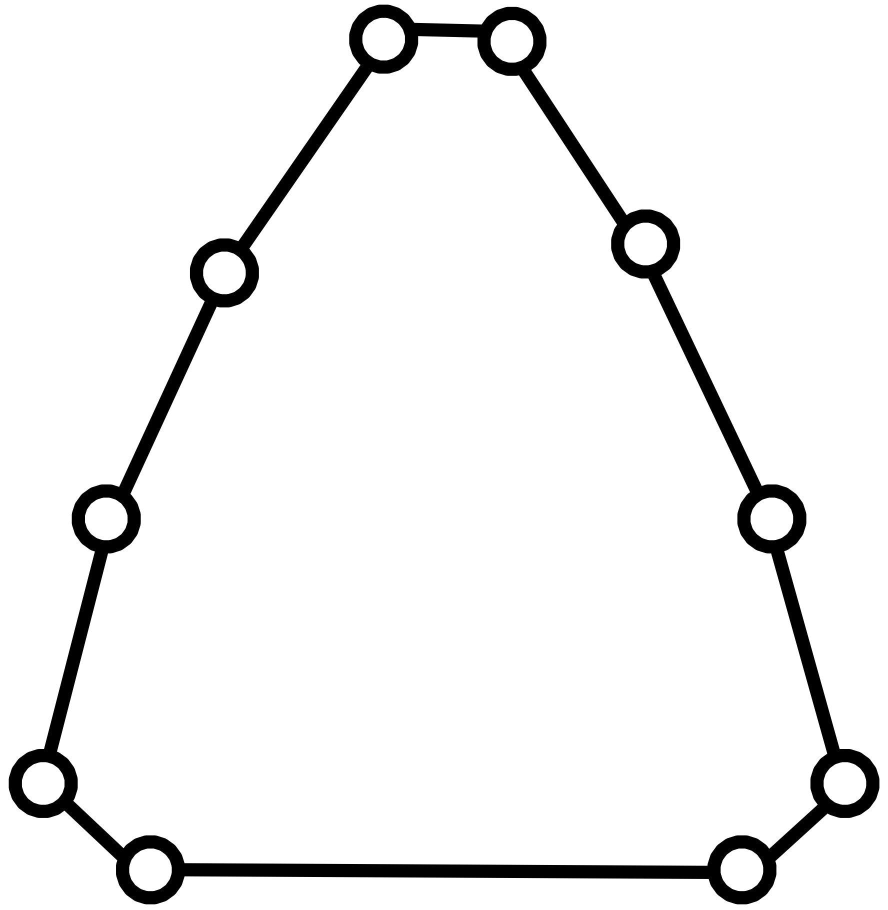


Compute New Error

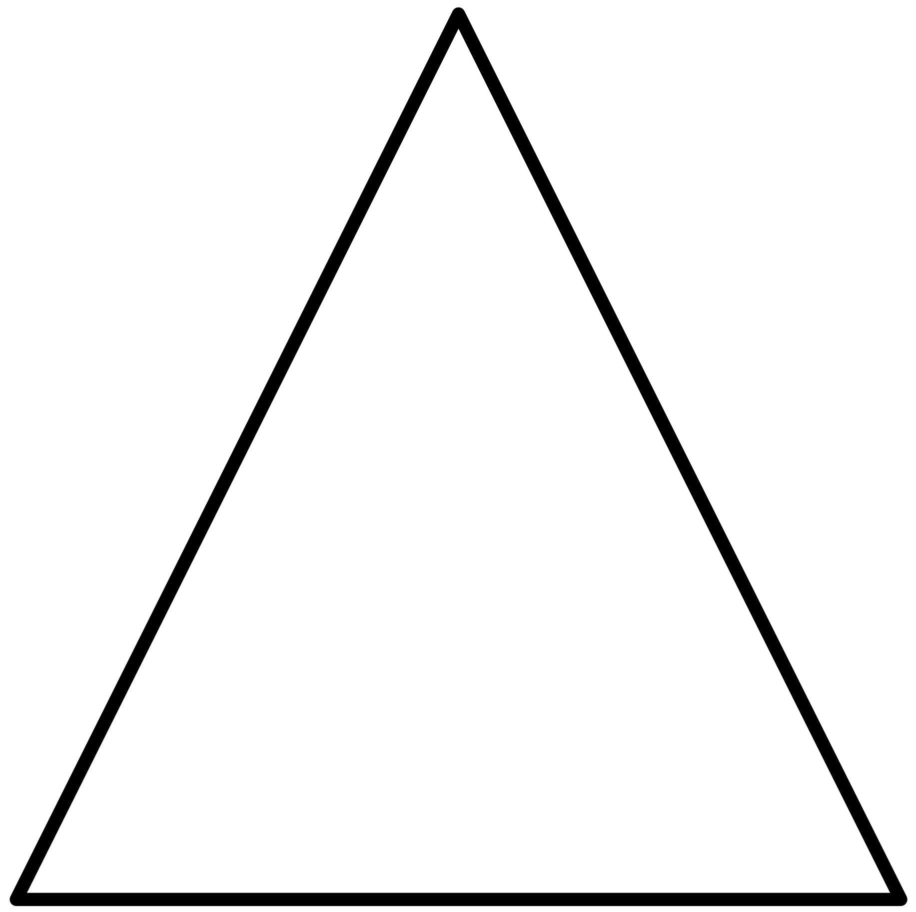


Target Geometry

Gradient Based Optimization

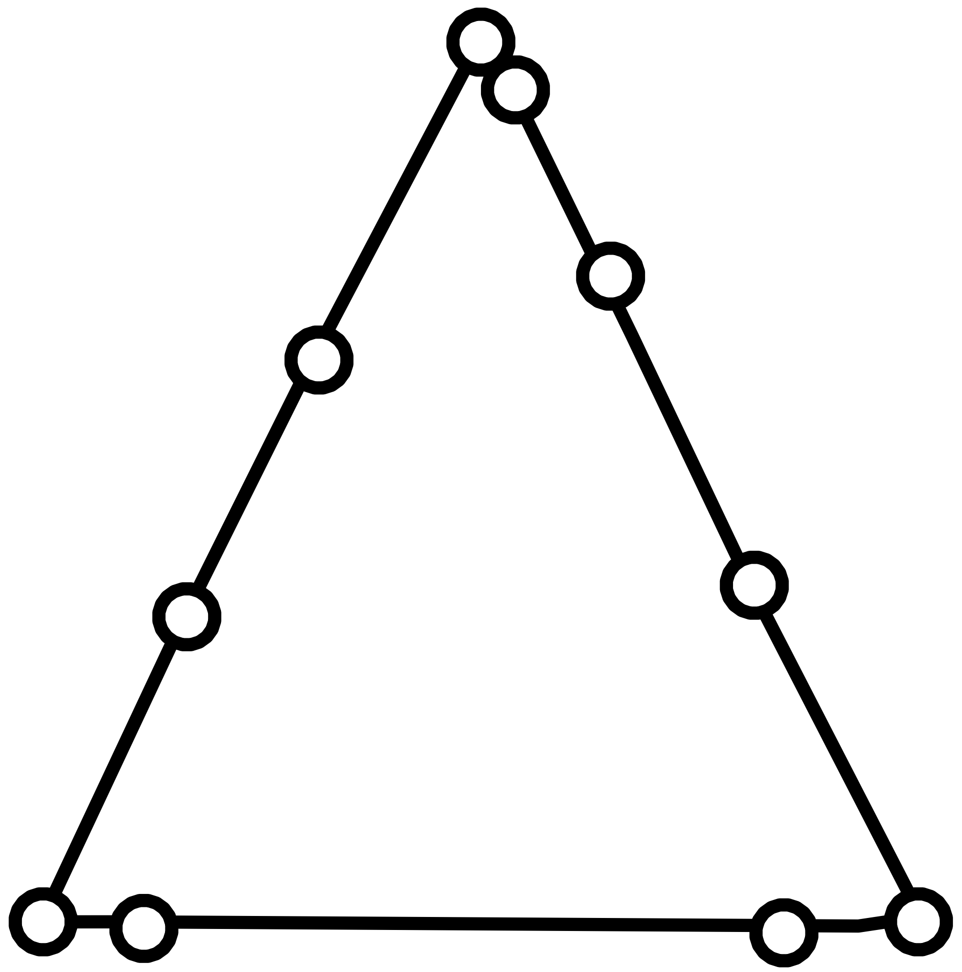


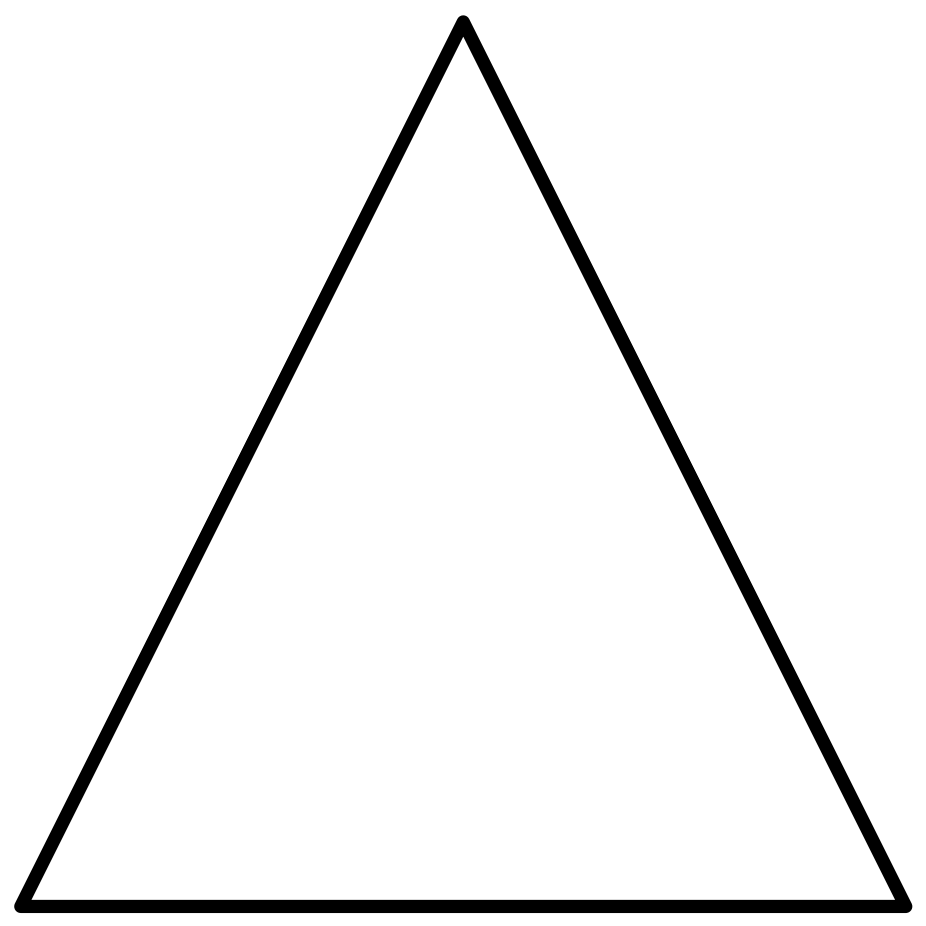
Repeat



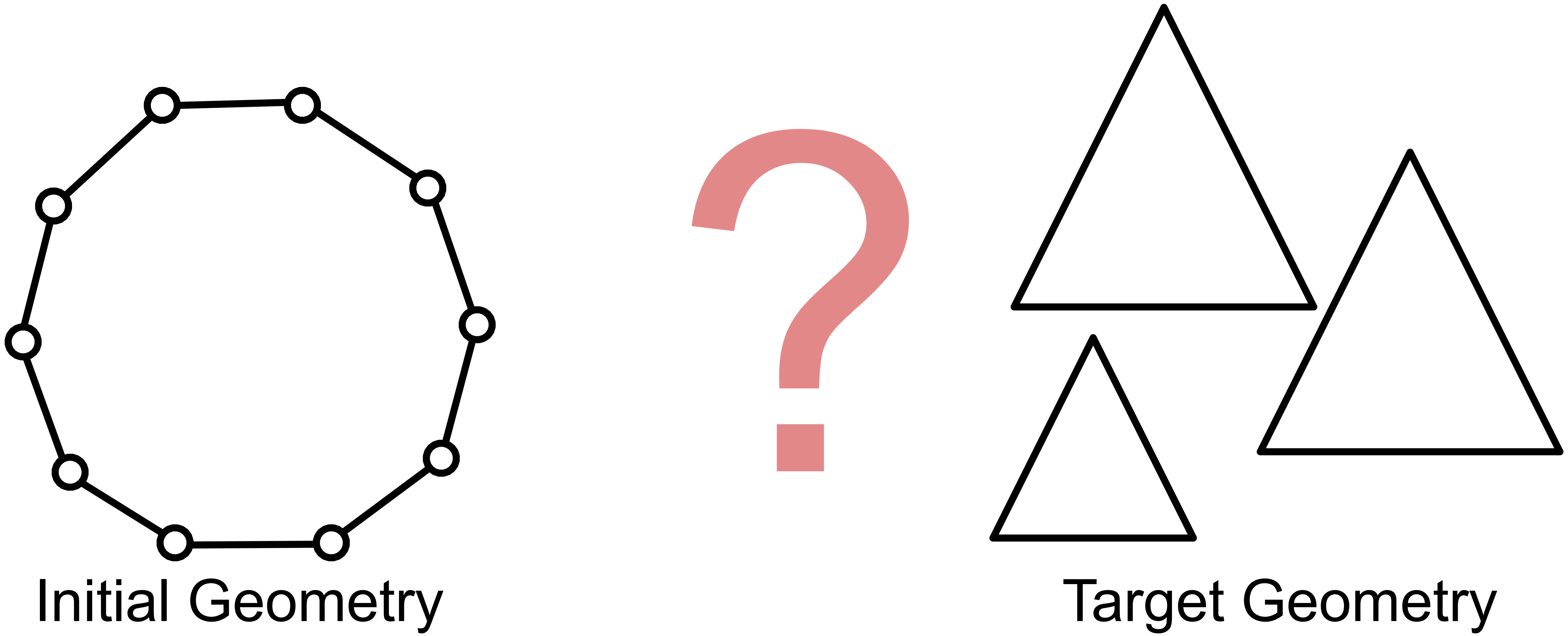
Target Geometry

Gradient Based Optimization





Gradient Based Optimization

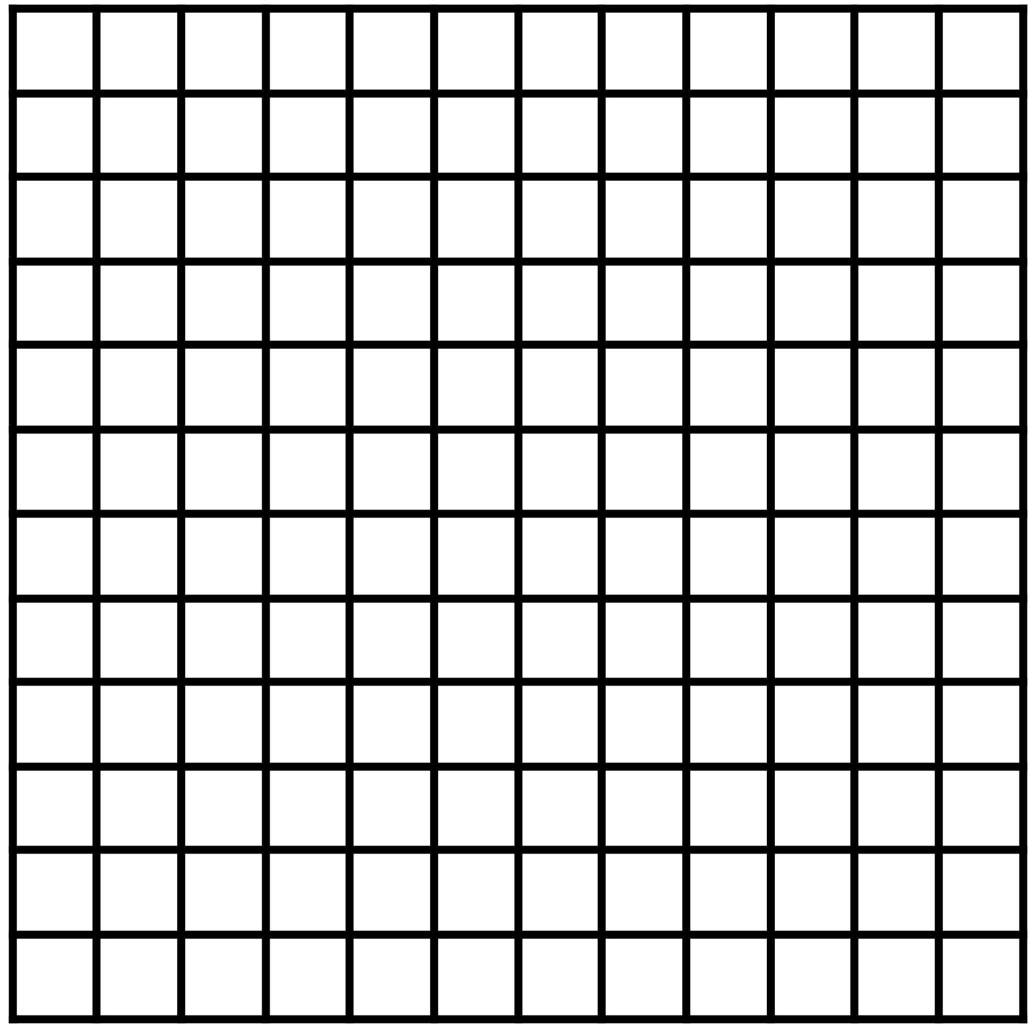


Gradient Based Optimization

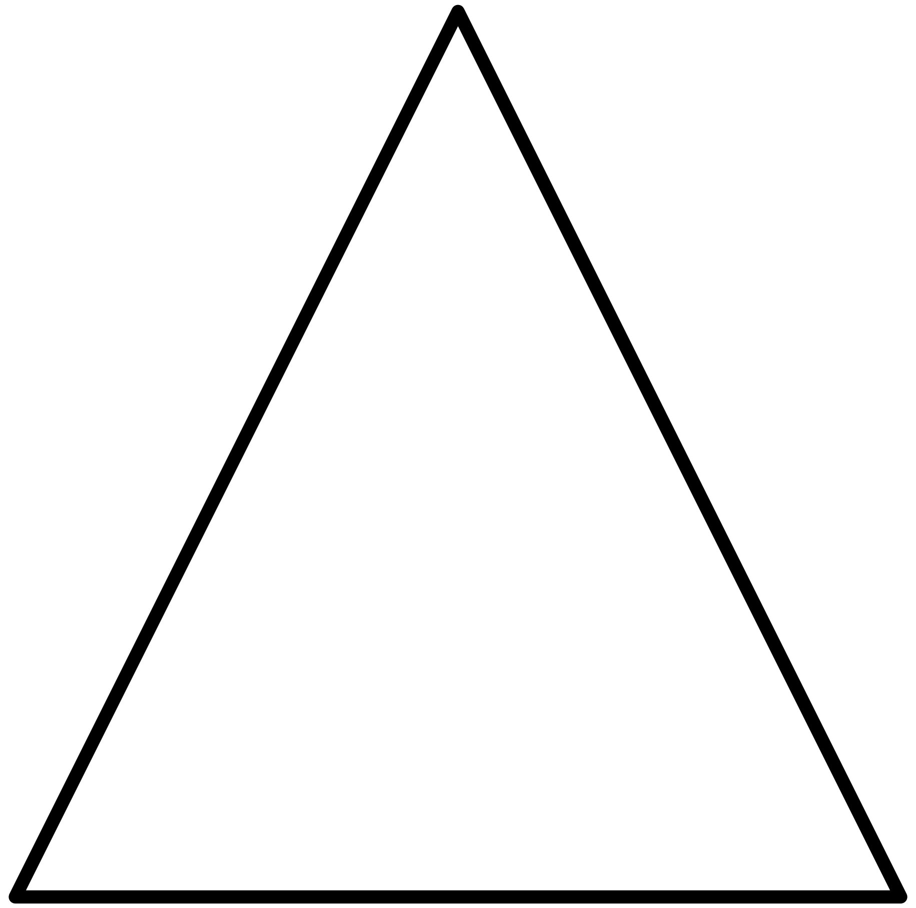
Gradient Based Optimization

Voxel Representation

Gradient Based Optimization

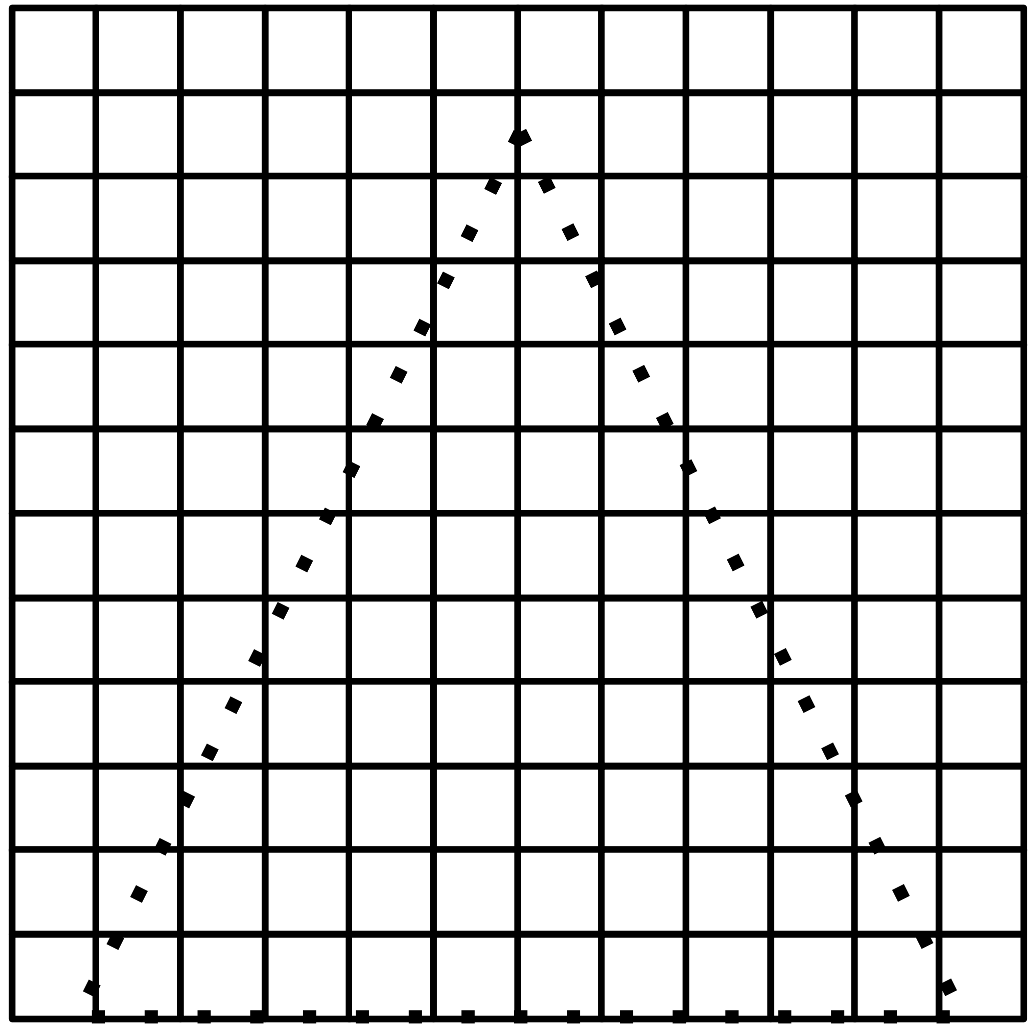


Initialized Grid

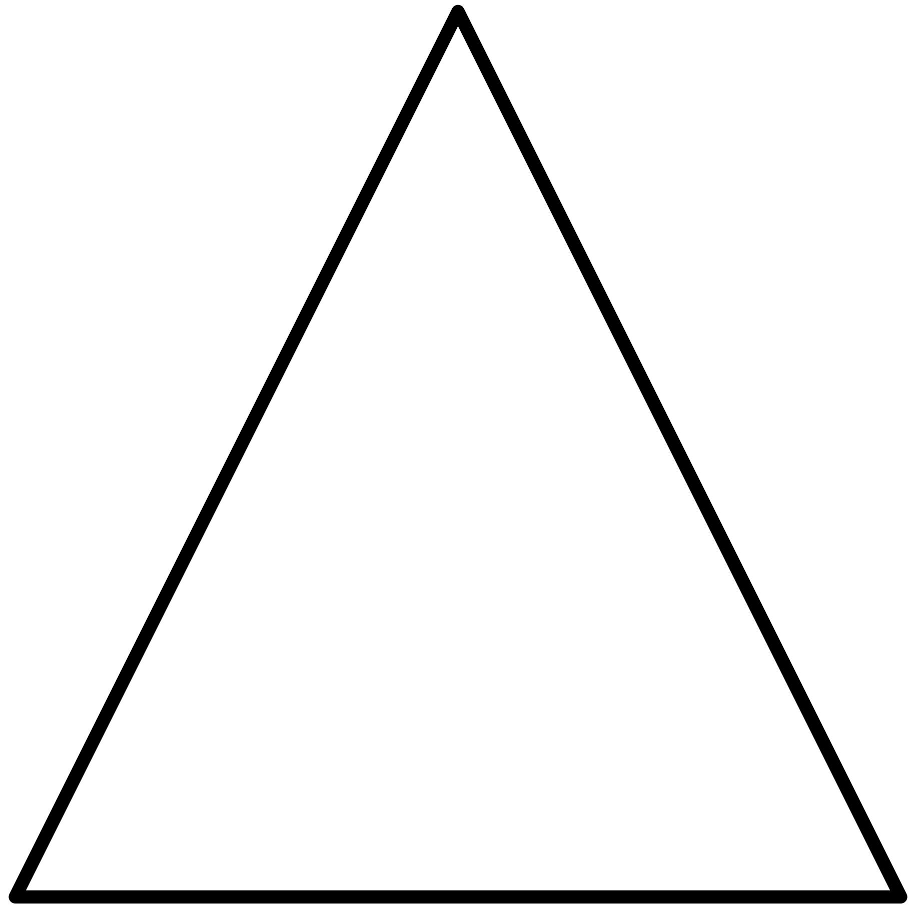


Target Geometry

Gradient Based Optimization

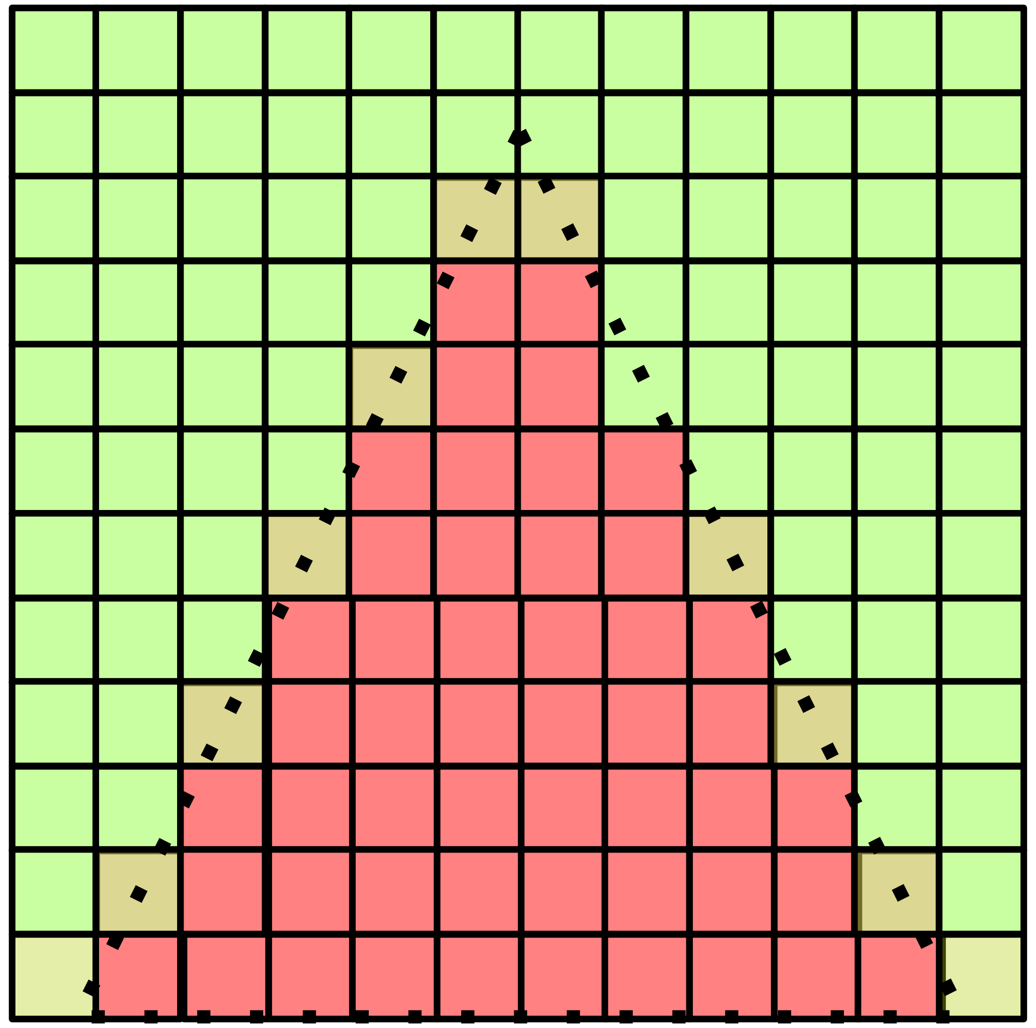


Initialized Grid

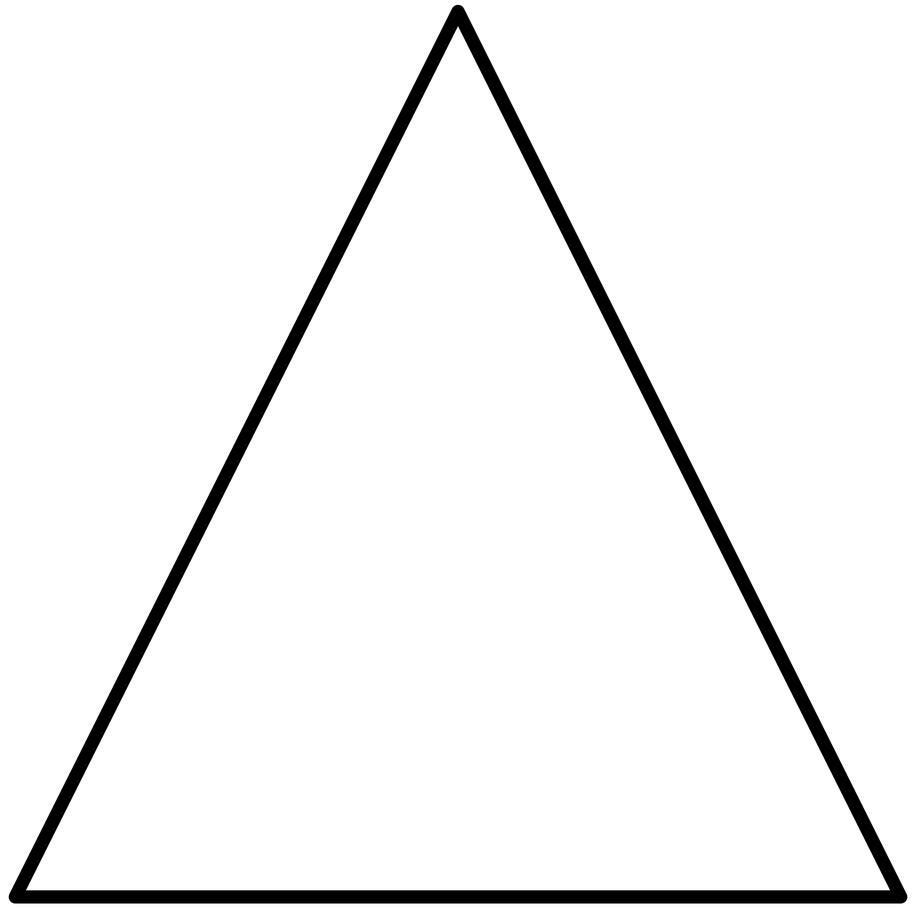


Target Geometry

Gradient Based Optimization

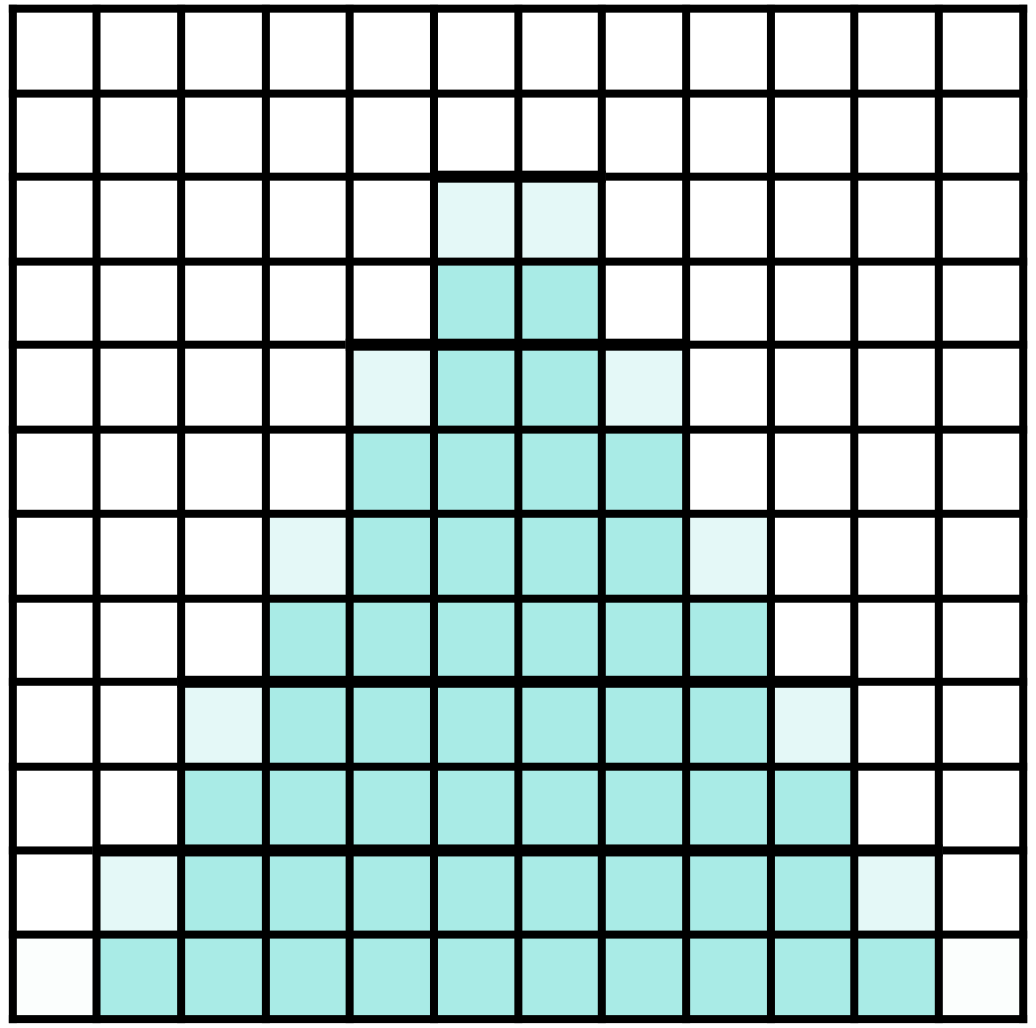


Loss

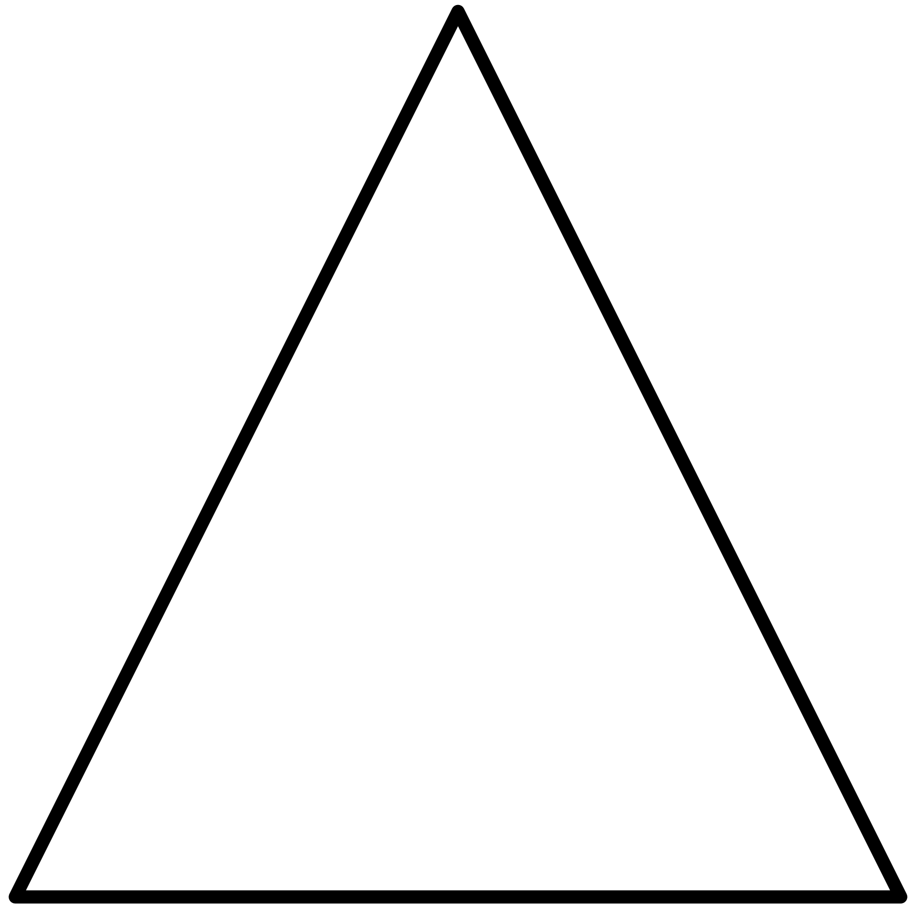


Target Geometry

Gradient Based Optimization

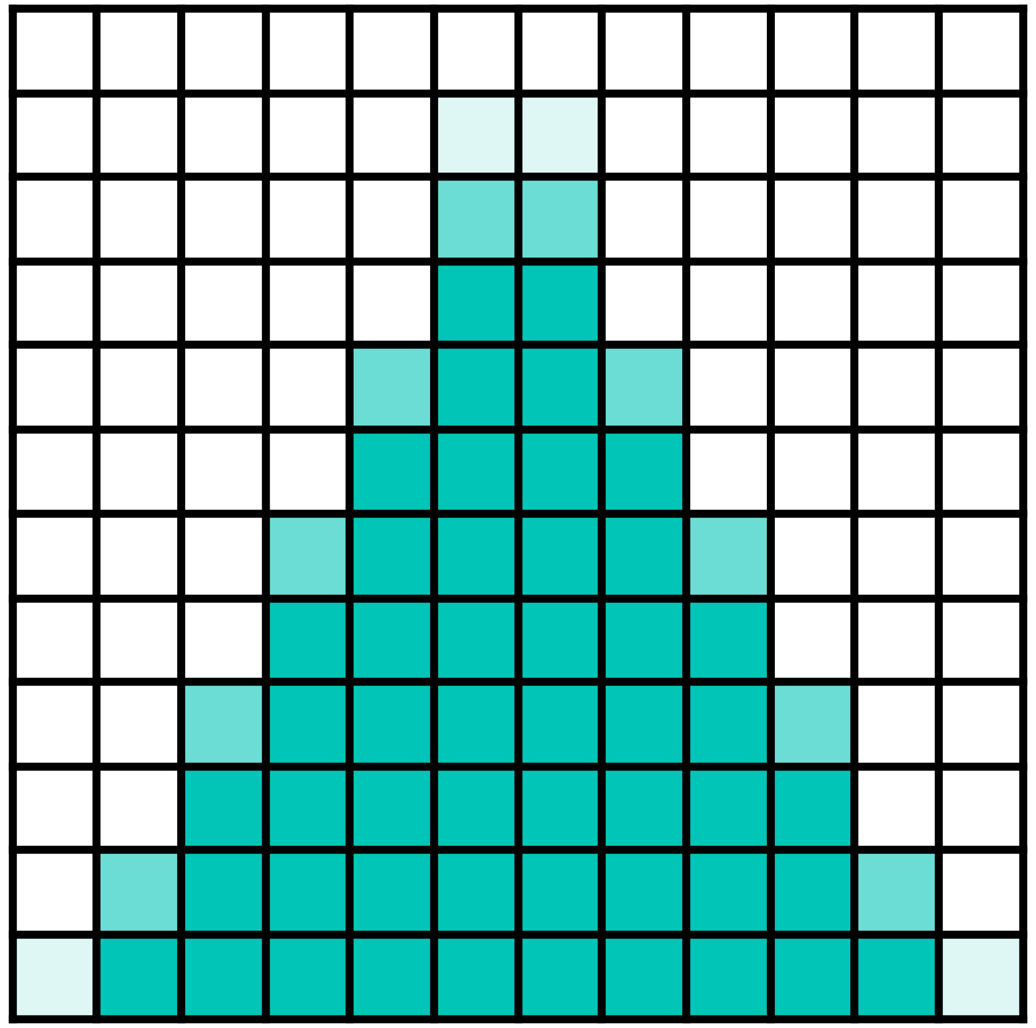


Gradient Step

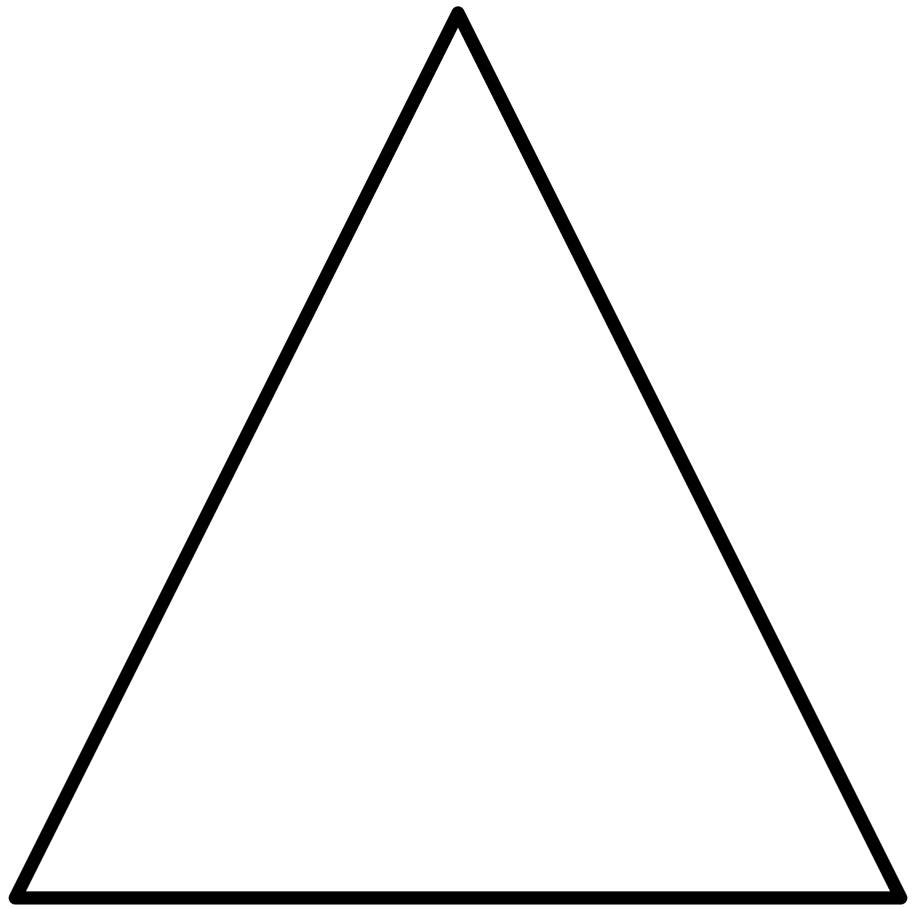


Target Geometry

Gradient Based Optimization

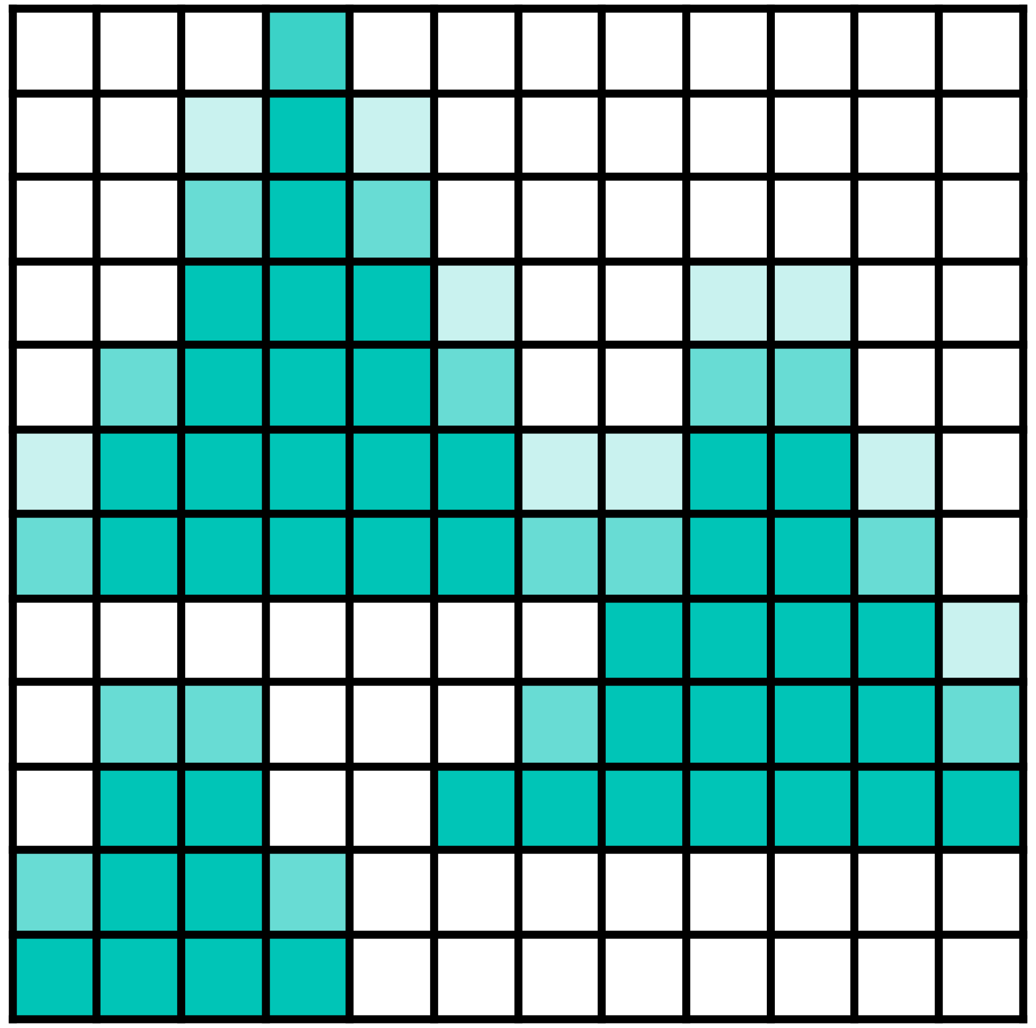


Repeat

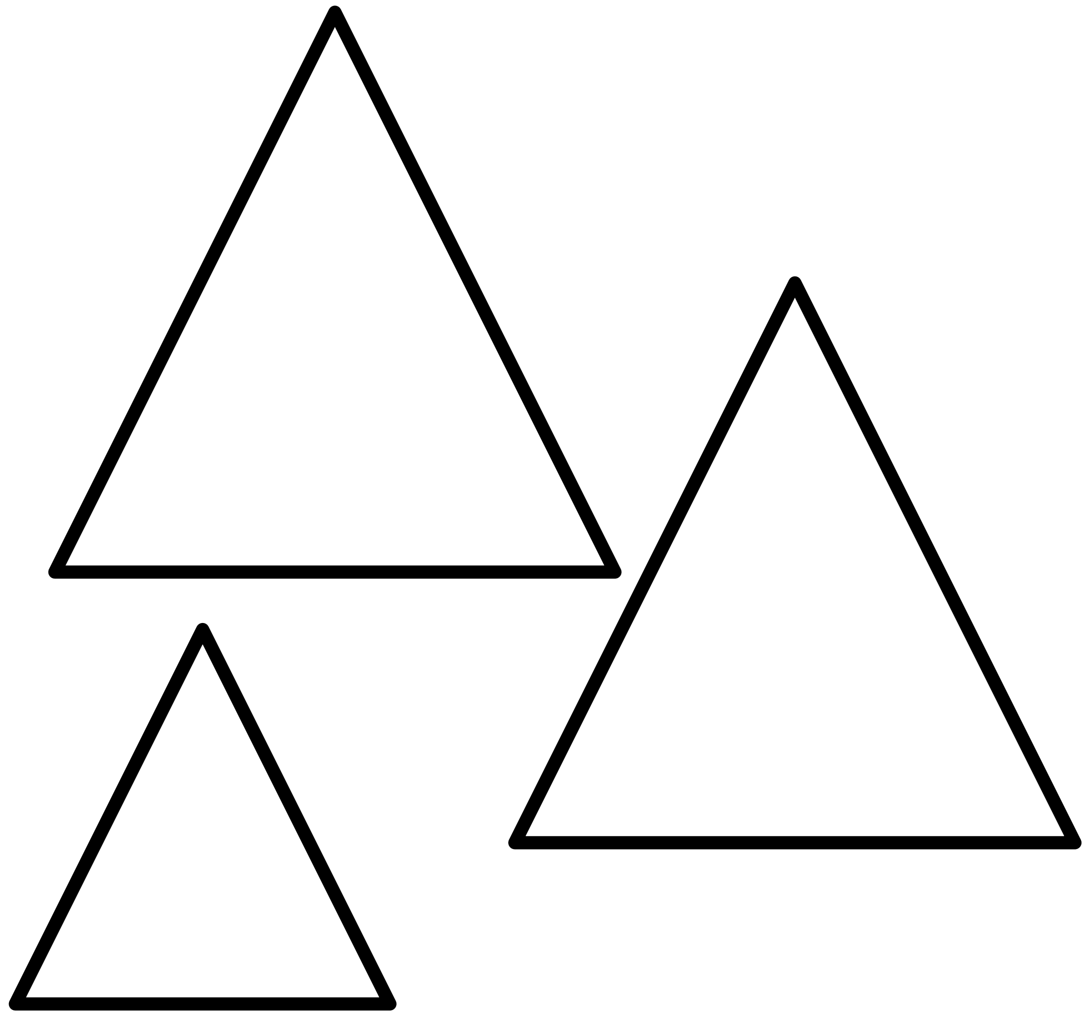


Target Geometry

Gradient Based Optimization

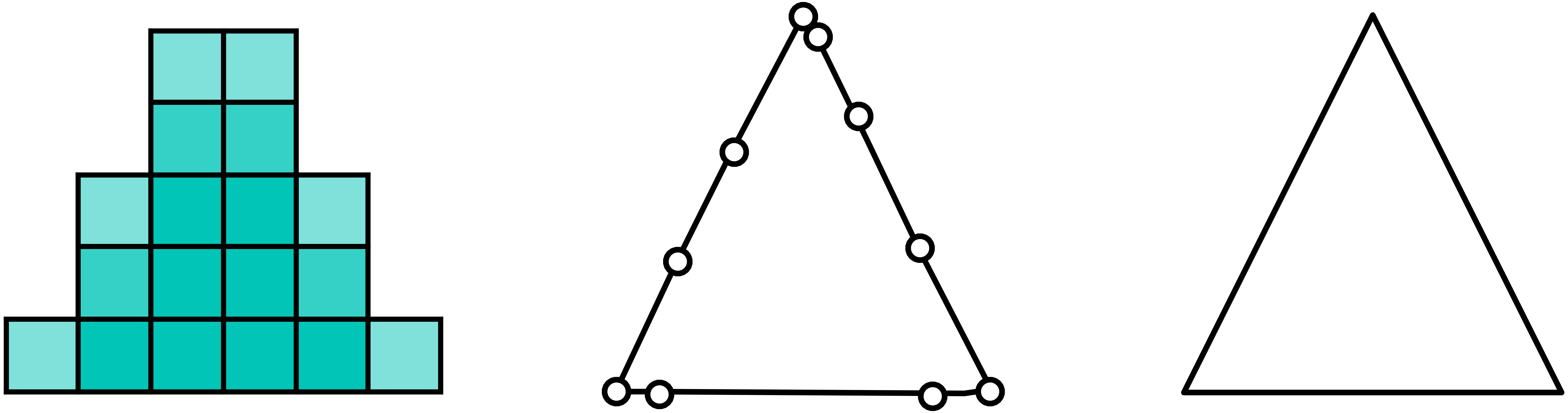


Reconstruction



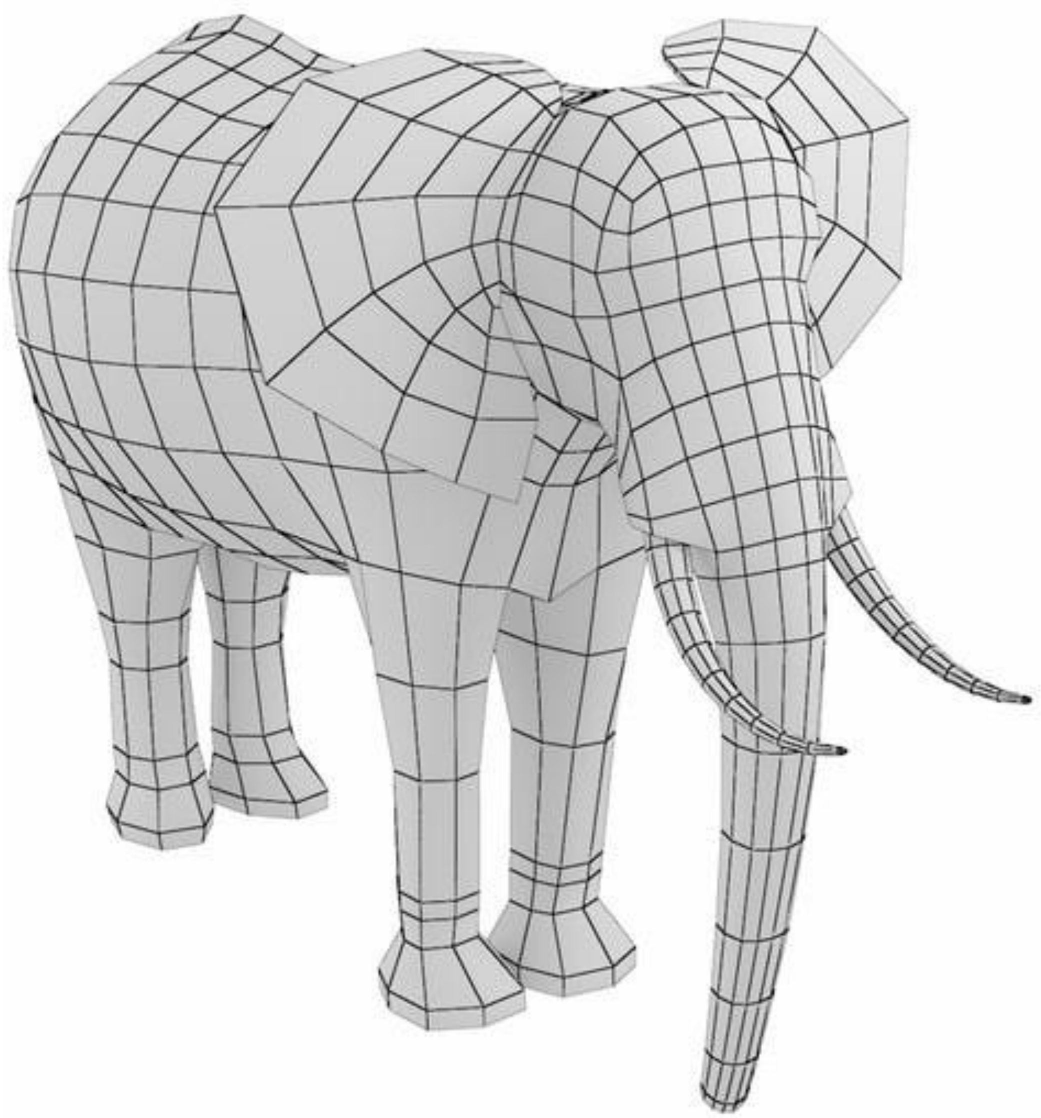
Target Geometry

Gradient Based Optimization



Voxel Mesh Target Geometry

Geometry Representations



Mesh Representation

Small memory footprint

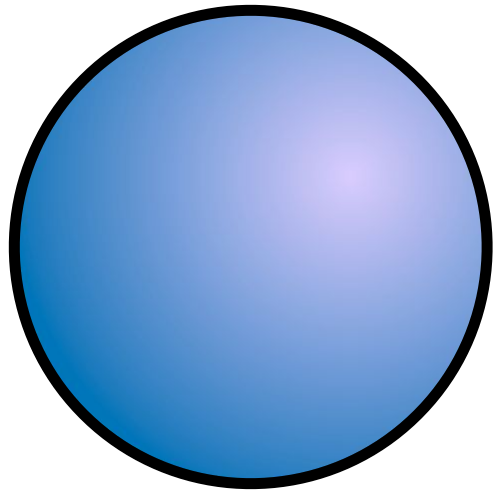
Hard to optimize

Voxel Representation

Easy to optimize

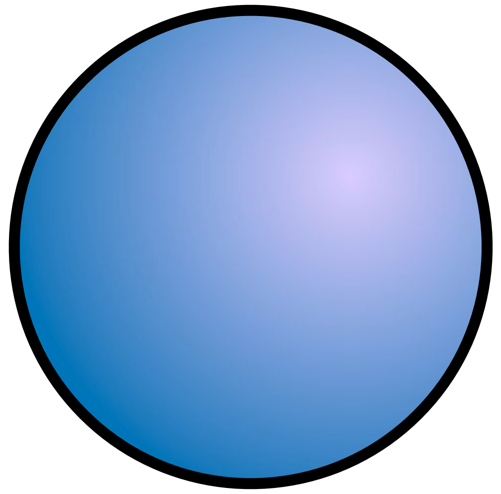
Large memory footprint

Implicit Functions

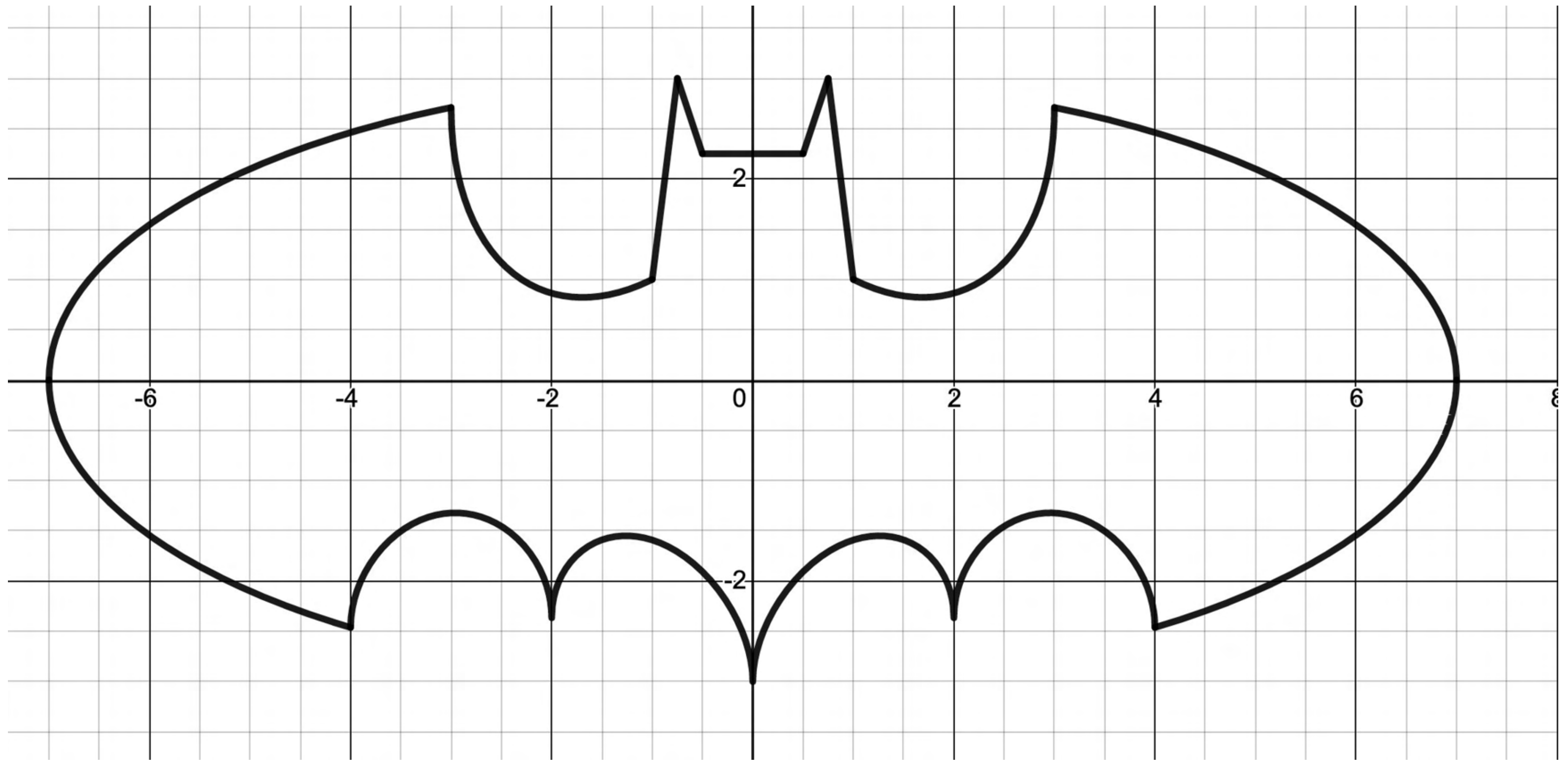


$x^2 + y^2 + z^2 = 1$

Implicit Functions



$x^2 + y^2 + z^2 = 1$



$\left\{|x| > 3 : 3\sqrt{-\left(\frac{x}{7}\right)^2 + 1}\right\}$

$\left\{|x| > 4 : -3\sqrt{-\left(\frac{x}{7}\right)^2 + 1}\right\}$

$\left|\frac{x}{2}\right| - \frac{3\sqrt{33} - 7}{112}x^2 + \sqrt{1 - \left(\mathrm{abs}\left(|x| - 2\right) - 1\right)}$

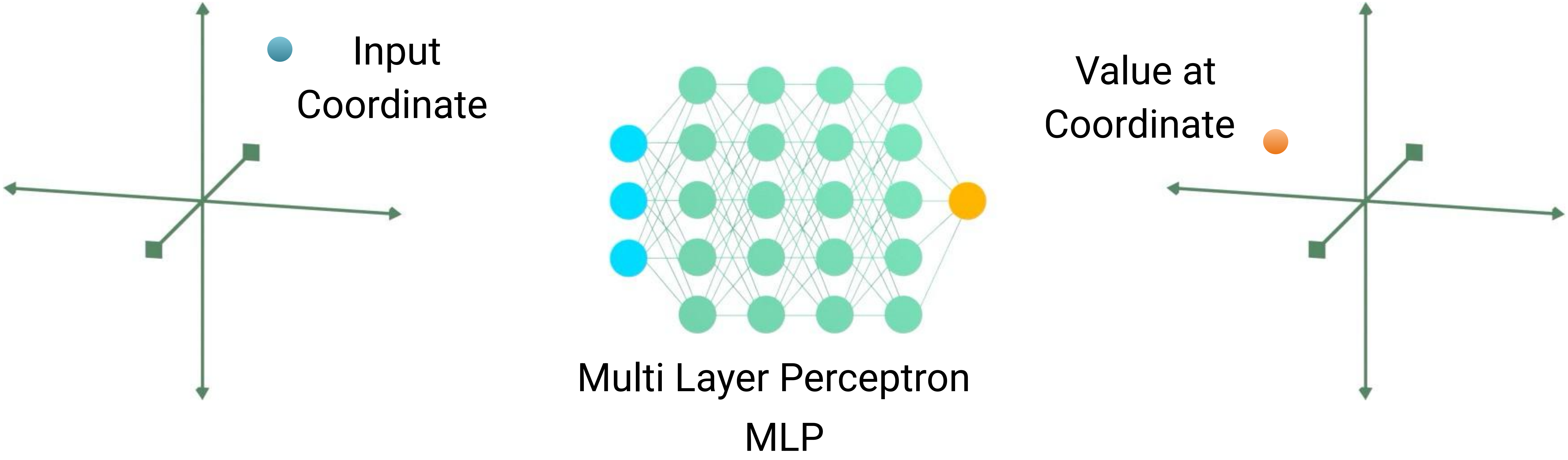
$\left\{.75 < |x| < 1 : 9 - 8|x|\right\}$

$\left\{.5 < |x| < .75 : 3|x| + .75\right\}$

$\left\{|x| < .5 : 2.25\right\}$

$\left\{|x| > 1 : 1.5 - .5|x| - \frac{6\sqrt{10}}{14}\left(\sqrt{3 - x^2 + 2|x|}\right.\right.$

Coordinate Based Neural Network

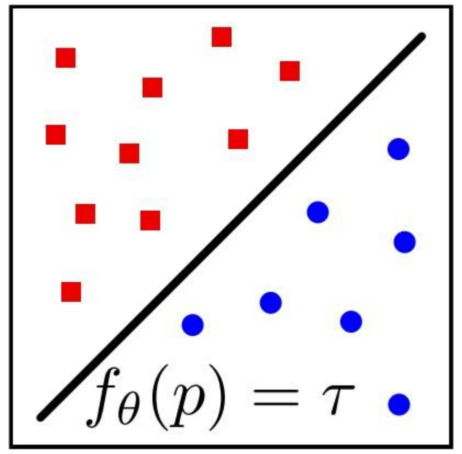


Neural networks as a continuous shape representation

Occupancy Networks

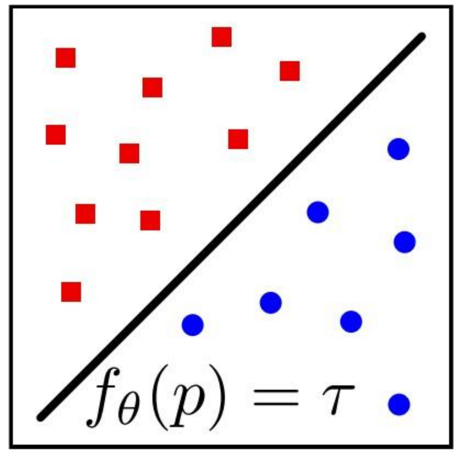
(Mescheder et al. 2019)

$(x, y, z) \longrightarrow$ occupancy

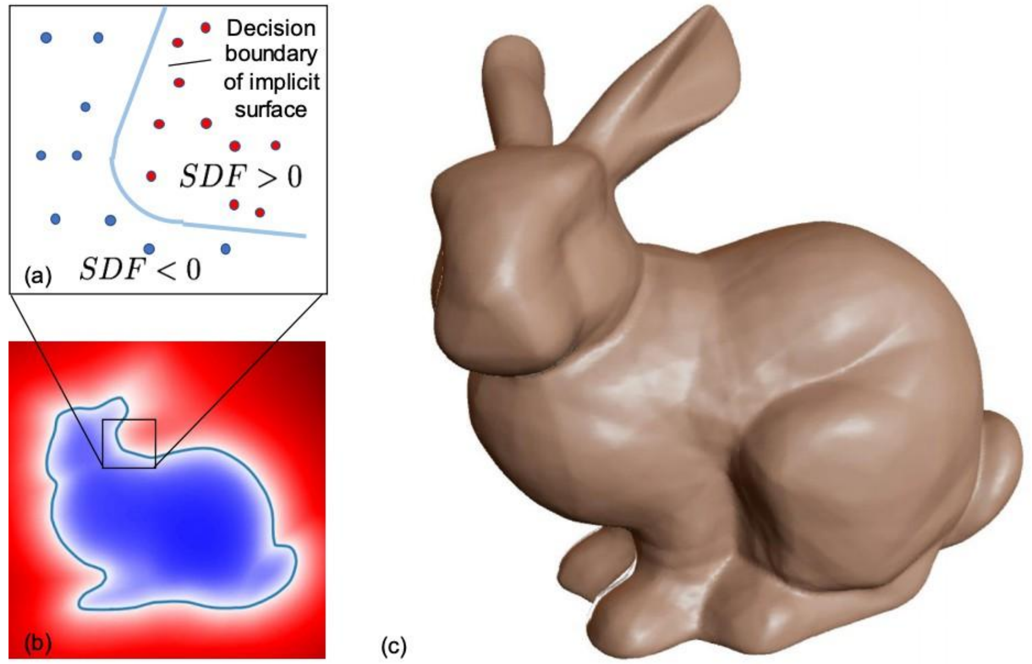


Neural networks as a continuous shape representation

Occupancy Networks
(Mescheder et al. 2019)
$(x, y, z) \longrightarrow$ occupancy

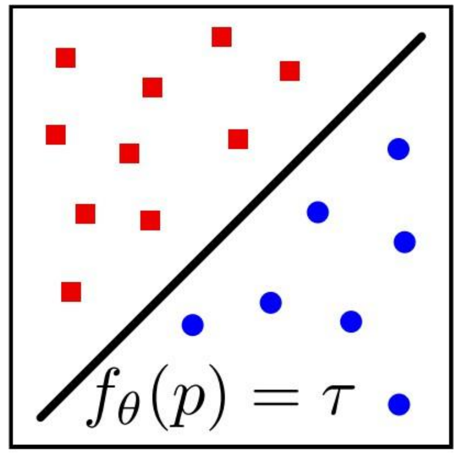


DeepSDF
(Park et al. 2019)
$(x, y, z) \longrightarrow$ distance

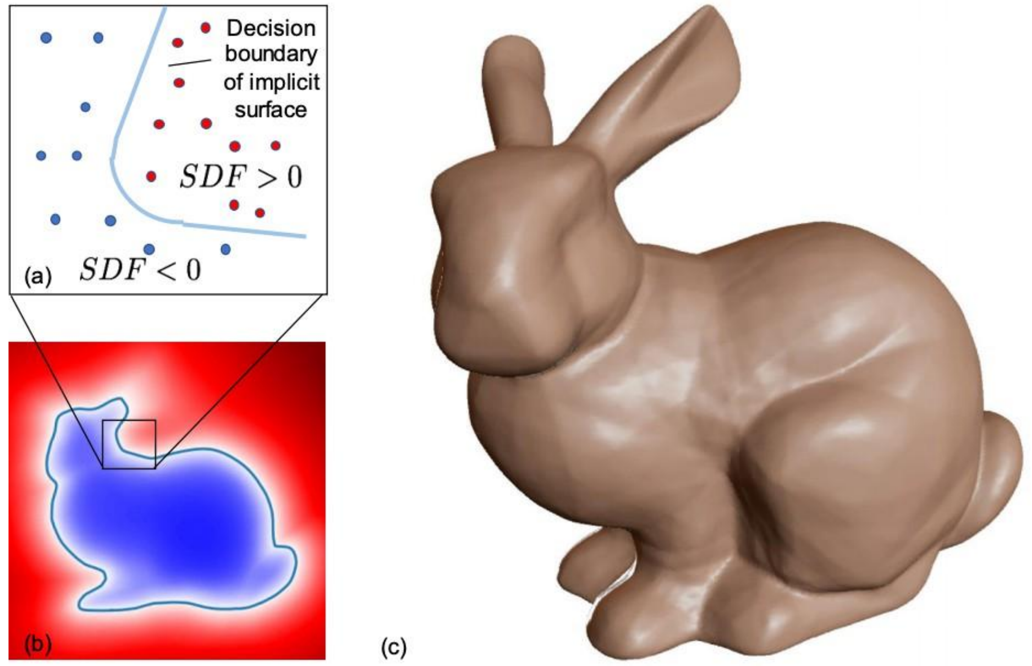


Neural networks as a continuous shape representation

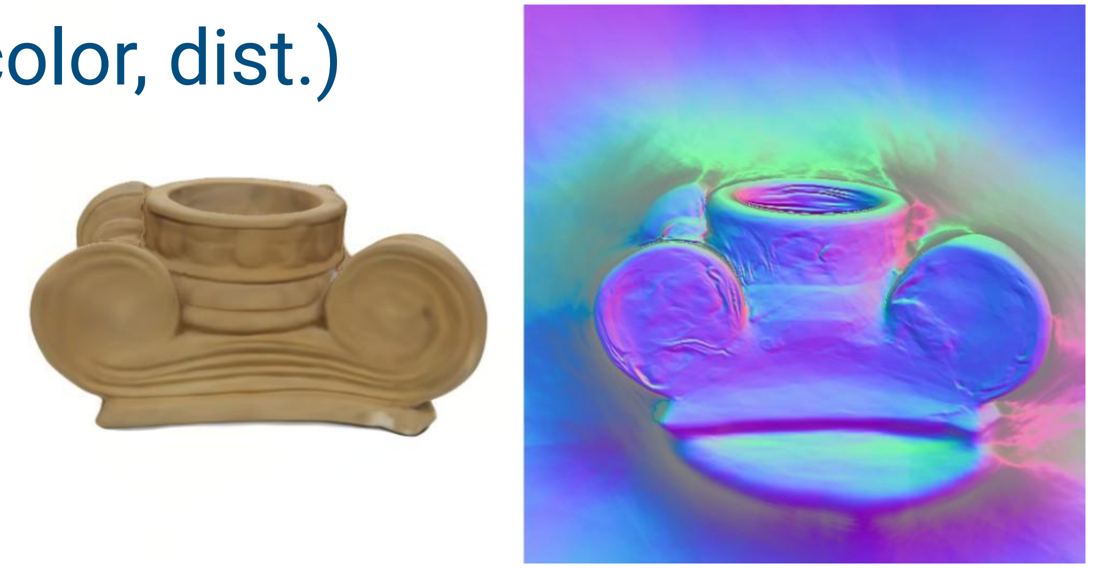
Occupancy Networks
(Mescheder et al. 2019)
$(x, y, z) \rightarrow$ occupancy



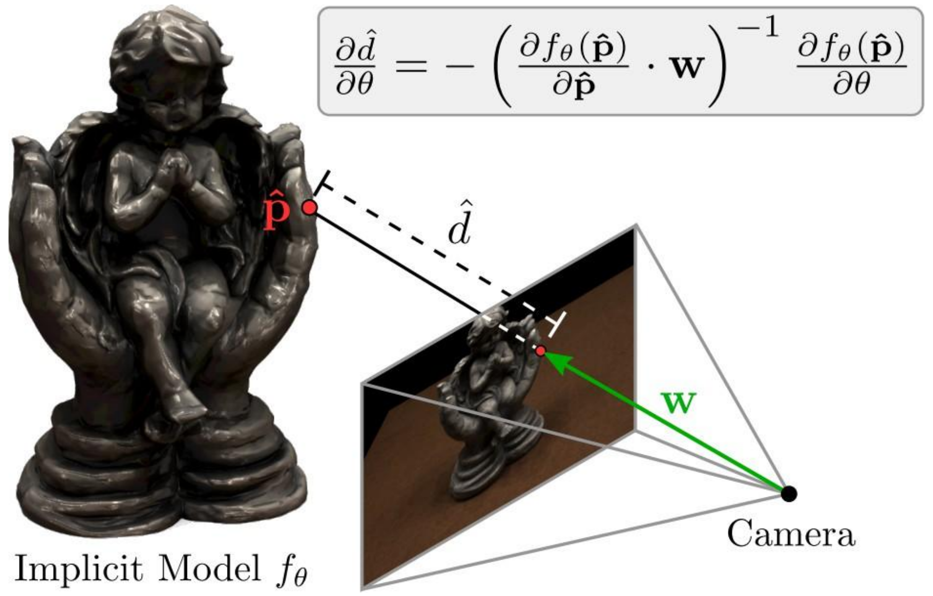
DeepSDF
(Park et al. 2019)
$(x, y, z) \rightarrow$ distance



Scene Representation Networks
(Sitzmann et al. 2019)
$(x, y, z) \rightarrow$ latent vec. (color, dist.)



Differentiable Volumetric Rendering
(Niemeyer et al. 2020)
$(x, y, z) \rightarrow$ color, occ.



NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis

Matthew Tancik*[1] Pratul P. Srinivasan*[1,3] Ben Mildenhall*[1,3]

Jonathan T. Barron[3] Ravi Ramamoorthi[2] Ren Ng[1]

1 UC San Diego 2 Google 3

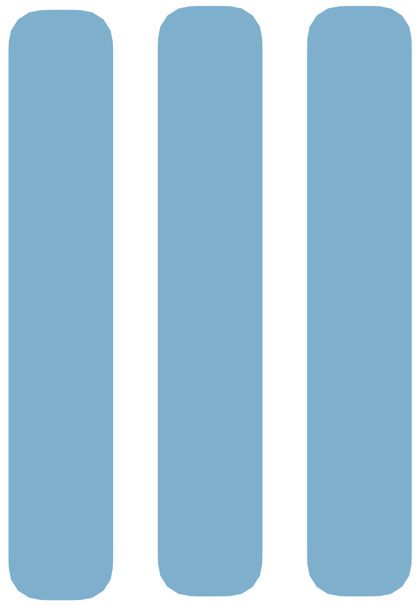
* Denotes Equal Contribution

NeRF (neural radiance fields):
Neural networks as a *volume* representation, using volume rendering to do view synthesis.

$$(x, y, z, \theta, \phi) \longrightarrow \text{color, opacity}$$

Representing a scene as a continuous 5D function

$$\underbrace{(x, y, z,}_{\text{Spatial location}} \underbrace{\theta, \phi)}_{\text{Viewing direction}} \rightarrow F_{\Omega} \rightarrow \underbrace{(r, g, b,}_{\text{Output color}} \underbrace{\sigma)}_{\text{Output density}}$$



MLP
9 layers,
256 channels

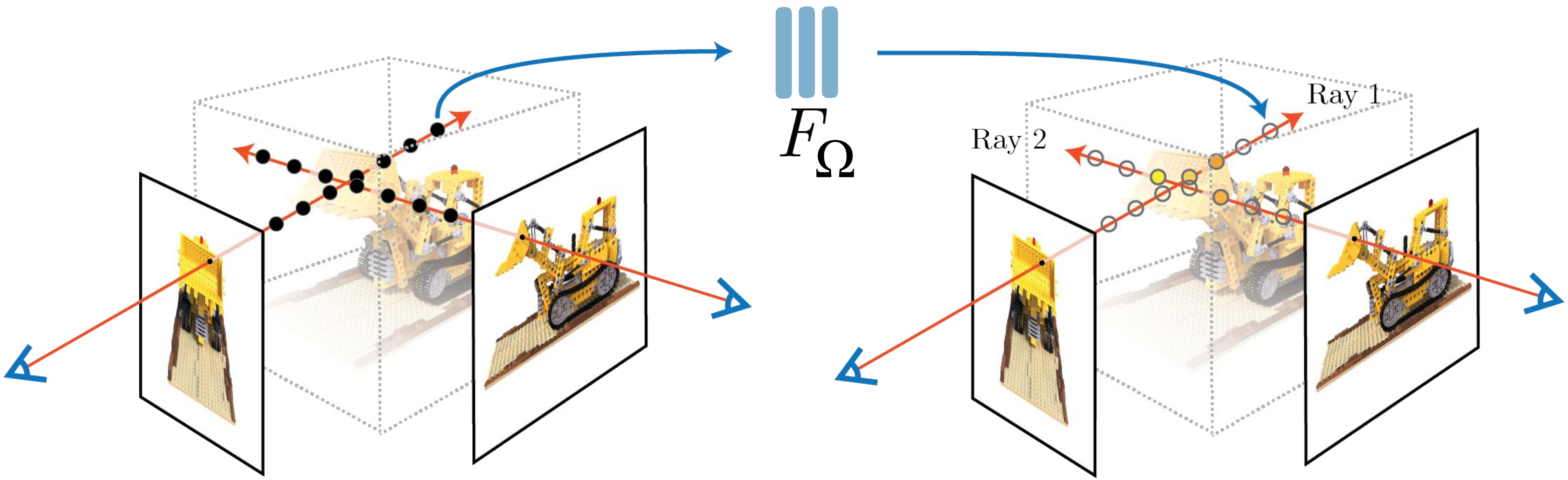
Recall “radiance” from previous lectures

Radiance

Light Traveling Along A Ray

1. Radiance is the fundamental field quantity that describes the distribution of light in an environment
    - Radiance is the quantity associated with a ray
    - Rendering is all about computing radiance
2. Radiance is invariant along a ray in a vacuum

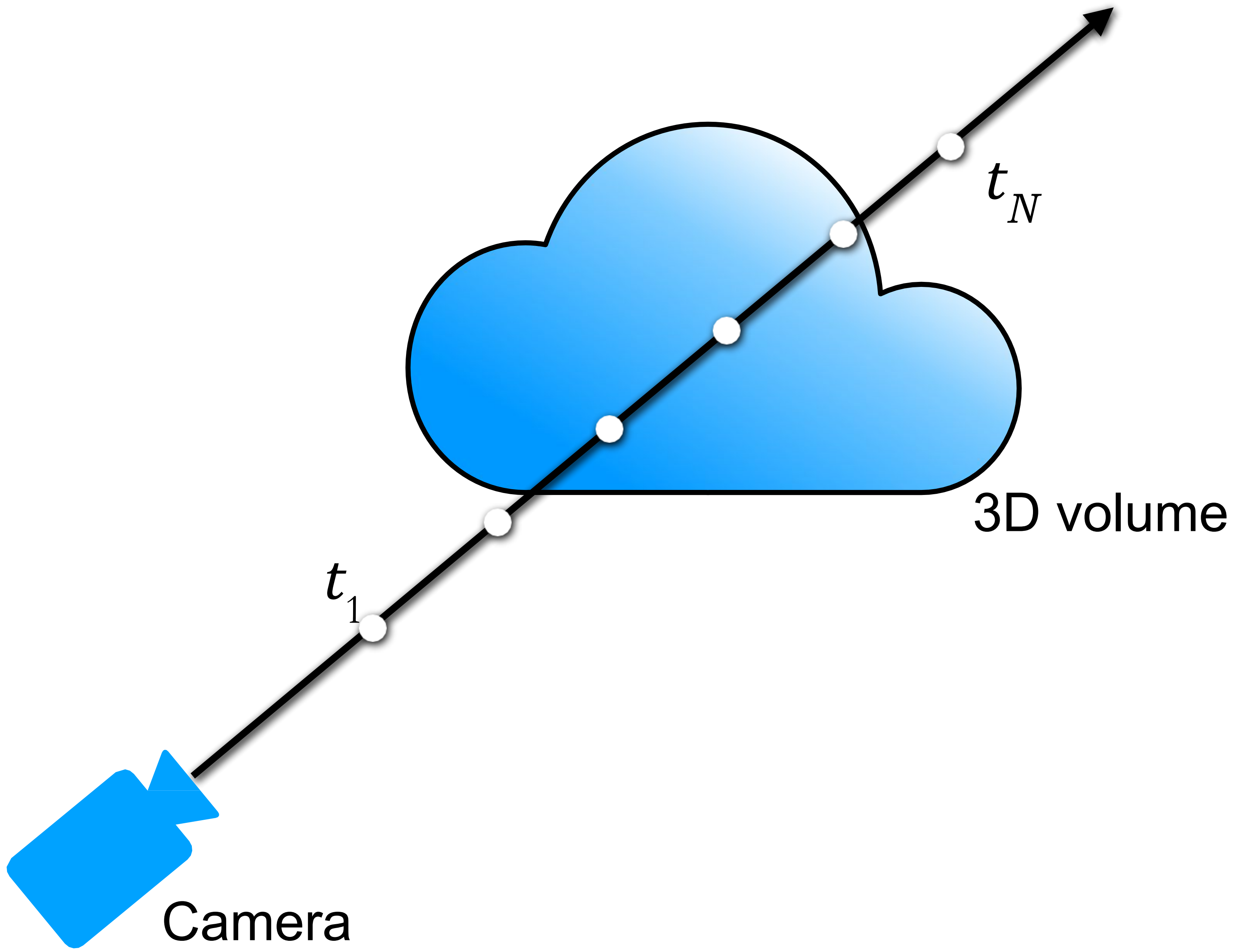
Generate views with traditional volume rendering



$$\min_{\Omega} \sum_i \|\mathrm{render}^{(i)}(F_\Omega) - I_{\mathrm{gt}}^{(i)}\|^2$$

Generate views with traditional volume rendering

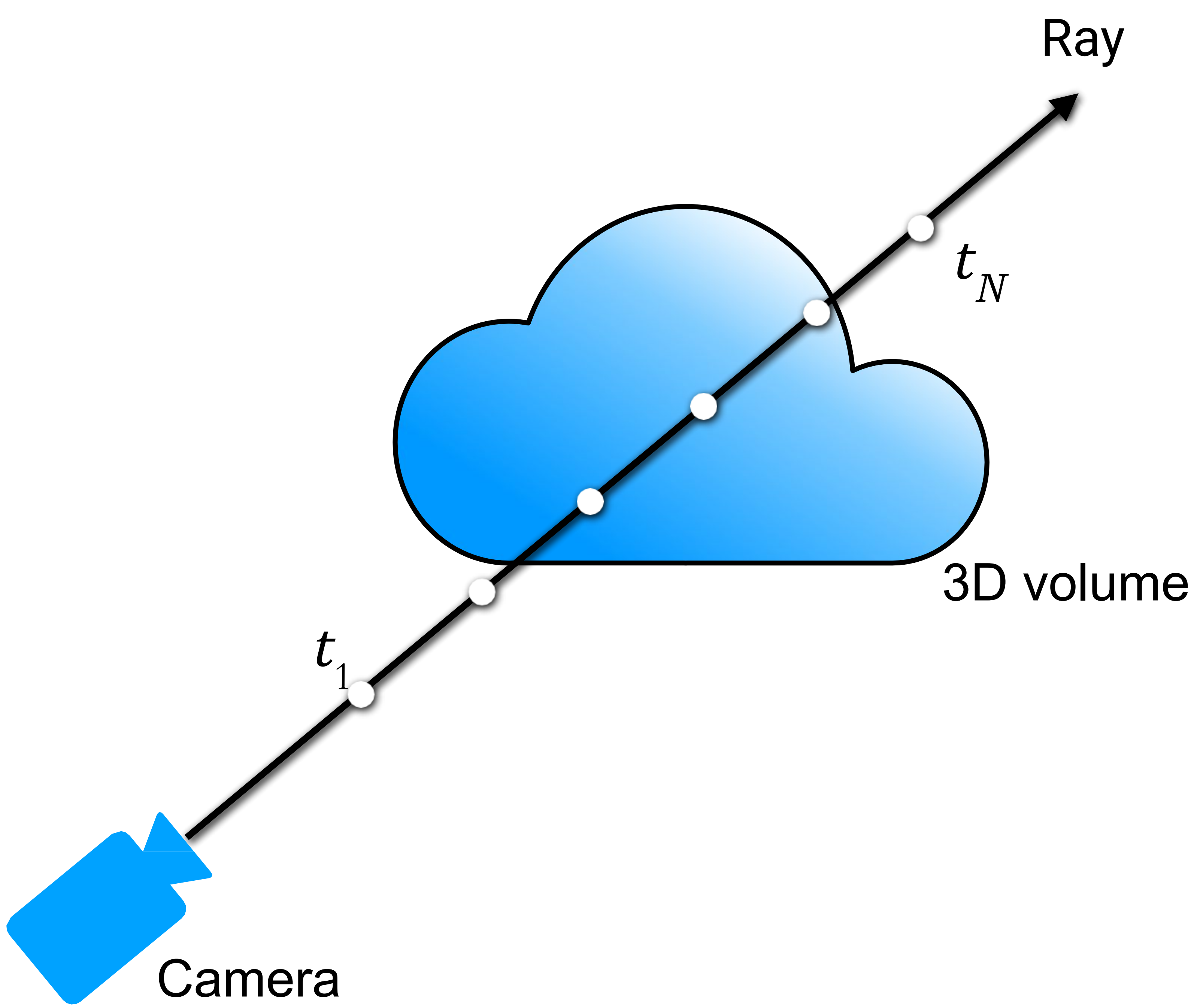
Rendering model for ray $r(t) = o + td$:



Generate views with traditional volume rendering

Rendering model for ray r(t) = o + td:

$$C \approx \sum_{i=1}^{N} \underbrace{T_i \alpha_i}_{\text{weights}} \underbrace{c_i}_{\text{colors}}$$



Generate views with traditional volume rendering

Rendering model for ray r(t) = o + td:

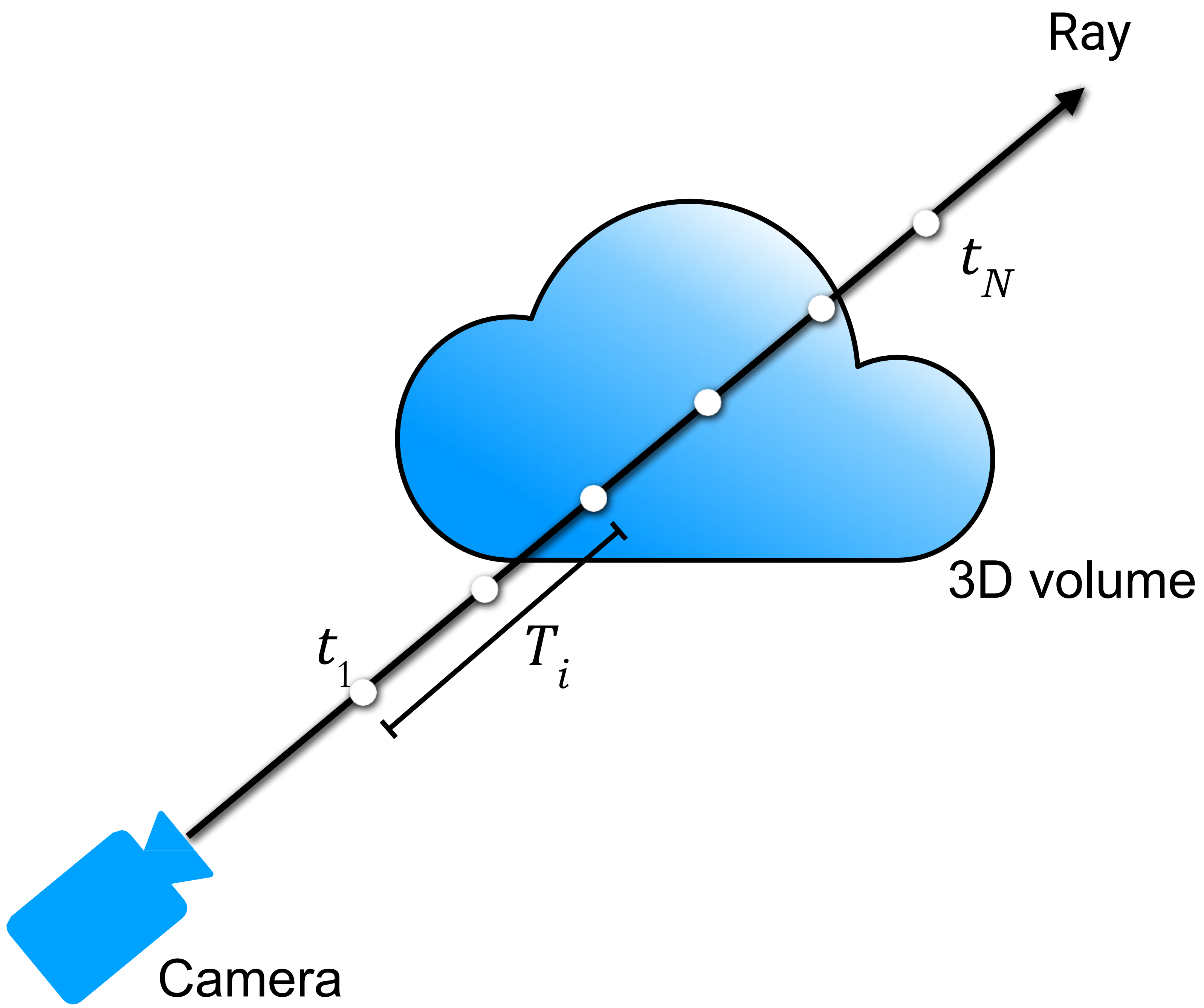
$$C \approx \sum_{i=1}^{N} T_i \alpha_i c_i$$

weights

colors

How much light is blocked earlier along ray:

$$T_i = \prod_{j=1}^{i-1} (1 - \alpha_j)$$



Generate views with traditional volume rendering

Rendering model for ray r(t) = o + td:

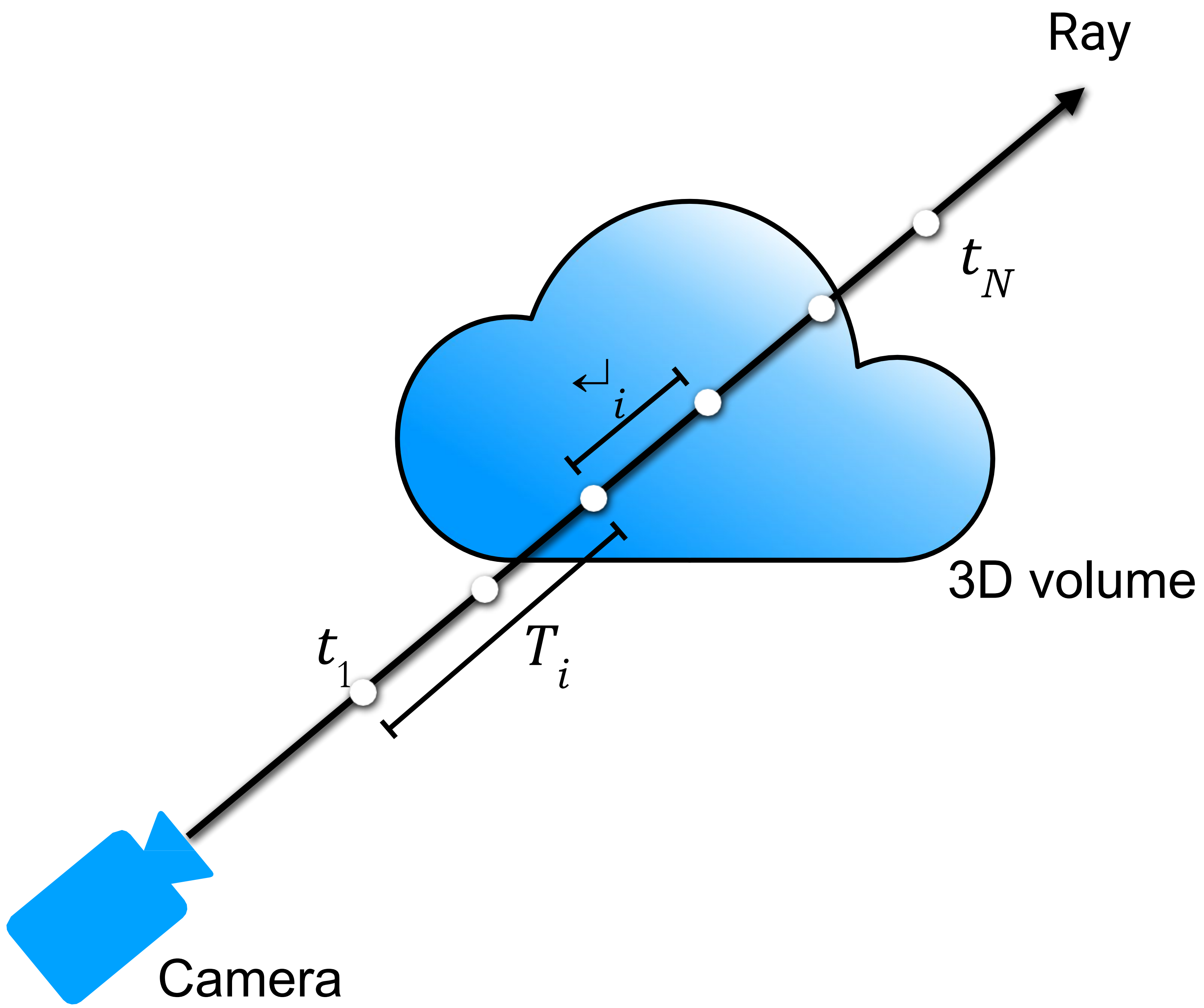
$$C \approx \sum_{i=1}^{N} \underbrace{T_i\alpha_i}_{\text{weights}} \underbrace{c_i}_{\text{colors}}$$

How much light is blocked earlier along ray:

$$T_i = \prod_{j=1}^{i-1} (1 - \alpha_j)$$

How much light is contributed by ray segment *i*:

$$\alpha_i = 1 - e^{-\sigma_i \delta t_i}$$



Effective resolution is tied to distance between samples

Rendering model for ray r(t) = o + td:

$$C \approx \sum_{i=1}^{N} T_i \alpha_i c_i$$

weights

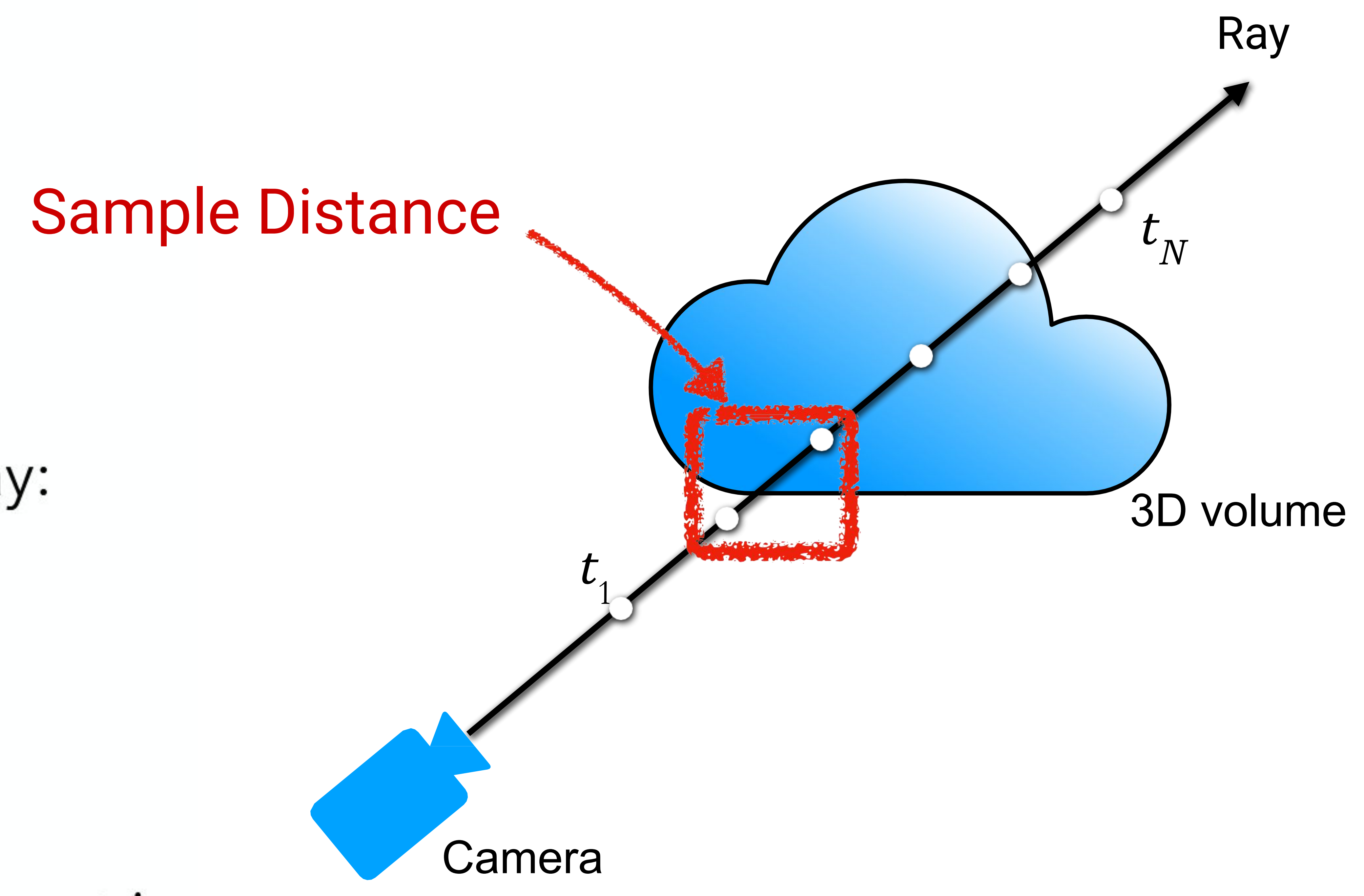
colors

How much light is blocked earlier along ray:

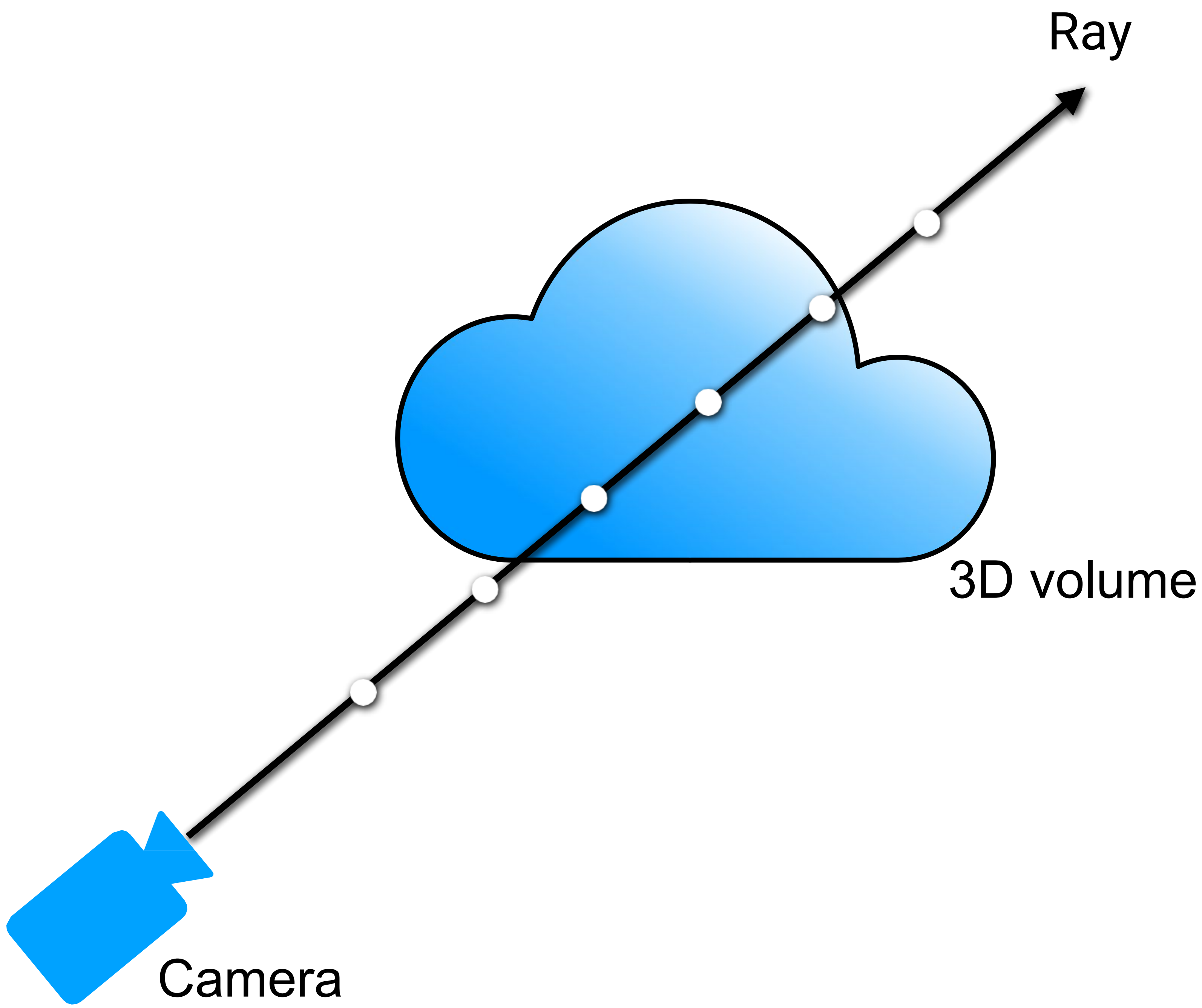
$$T_i = \prod_{j=1}^{i-1} (1 - \alpha_j)$$

How much light is contributed by ray segment *i*:

$$\alpha_i = 1 - e^{-\sigma_i \delta t_i}$$



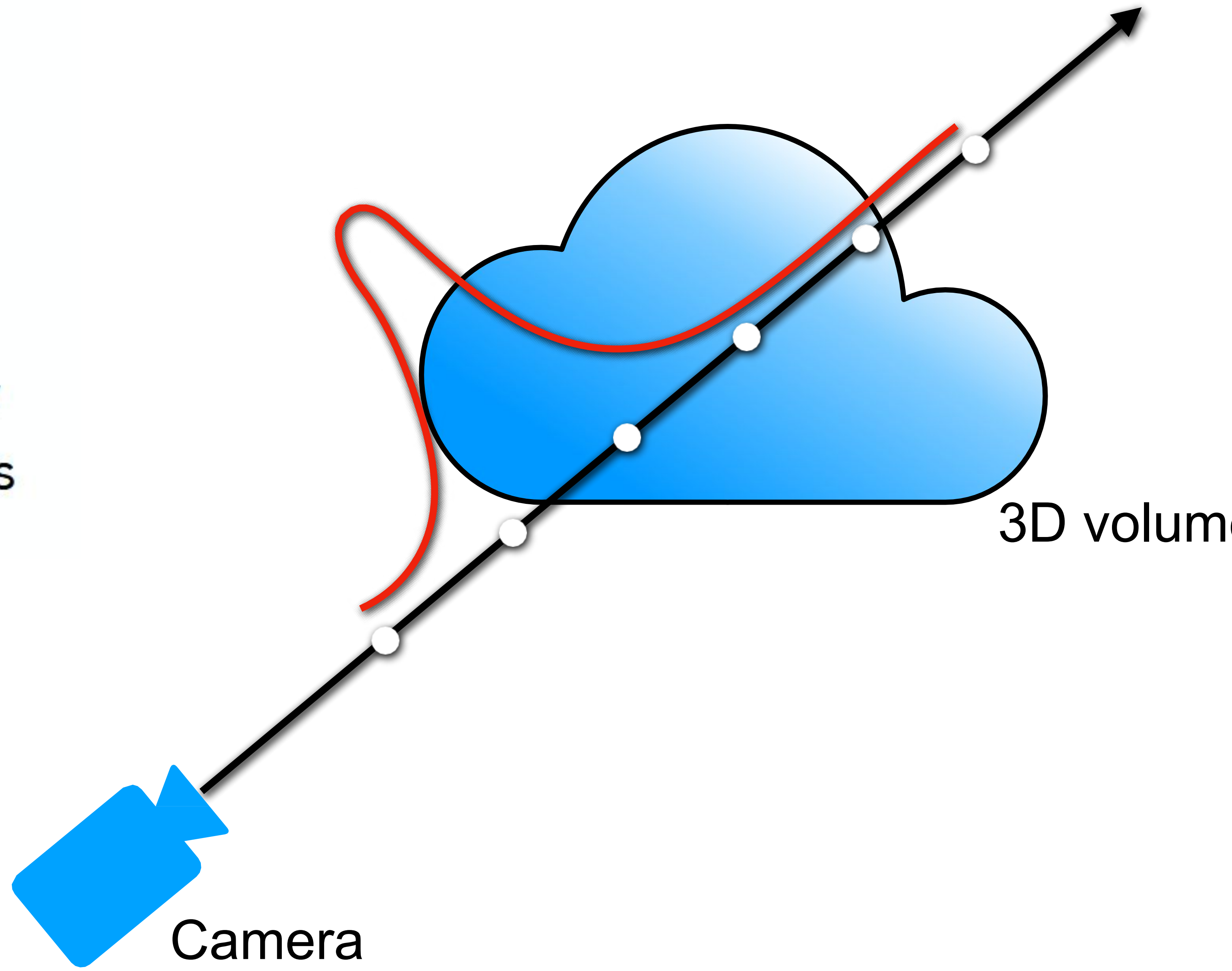
Can we allocate samples more efficiently? Two pass rendering



Two pass rendering: coarse

$$C \approx \sum_{i=1}^{N} \boxed{T_i \alpha_i} c_i$$

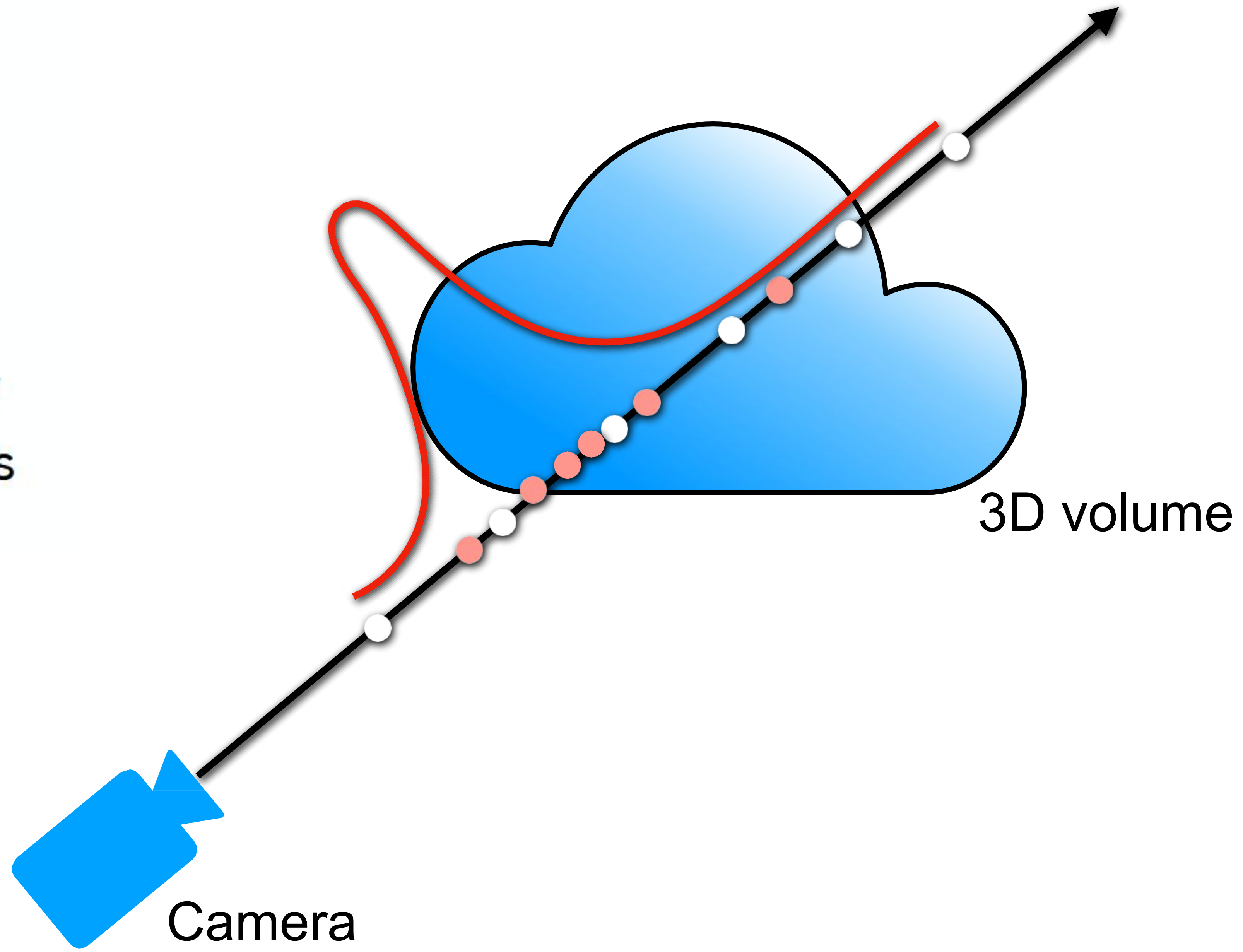
treat weights as probability distribution for new samples



Two pass rendering: fine

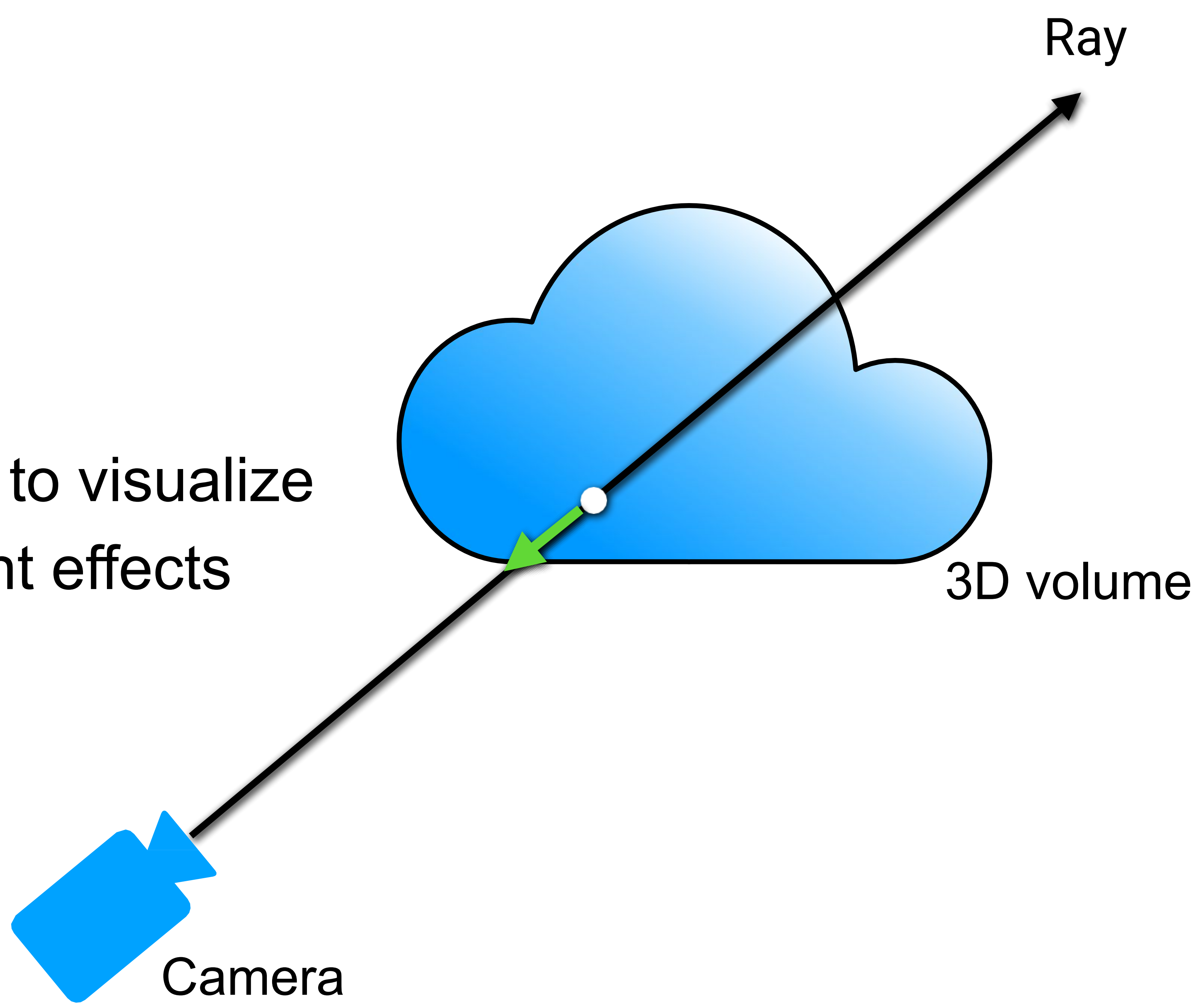
$$C \approx \sum_{i=1}^{N} \boxed{T_i \alpha_i} c_i$$

treat weights as probability distribution for new samples



Viewing directions as input

Manipulate (θ, ϕ) to visualize view-dependent effects



Volume rendering is differentiable

Rendering model for ray r(t) = o + td:

$$\boxed{C \approx \sum_{i=1}^{N} T_i \alpha_i c_i}$$

colors

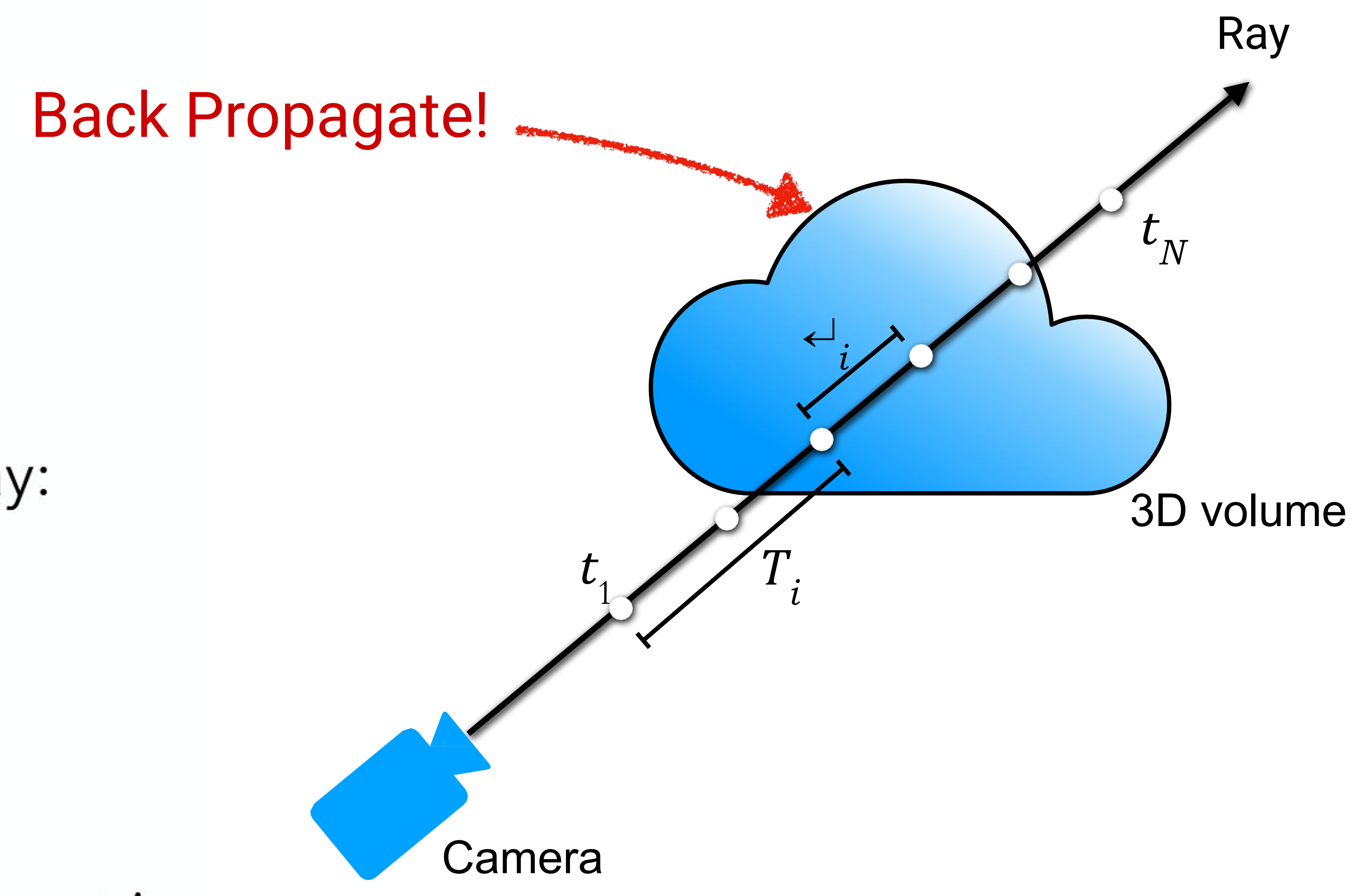
weights

How much light is blocked earlier along ray:

$$T_i = \prod_{j=1}^{i-1} (1 - \alpha_j)$$

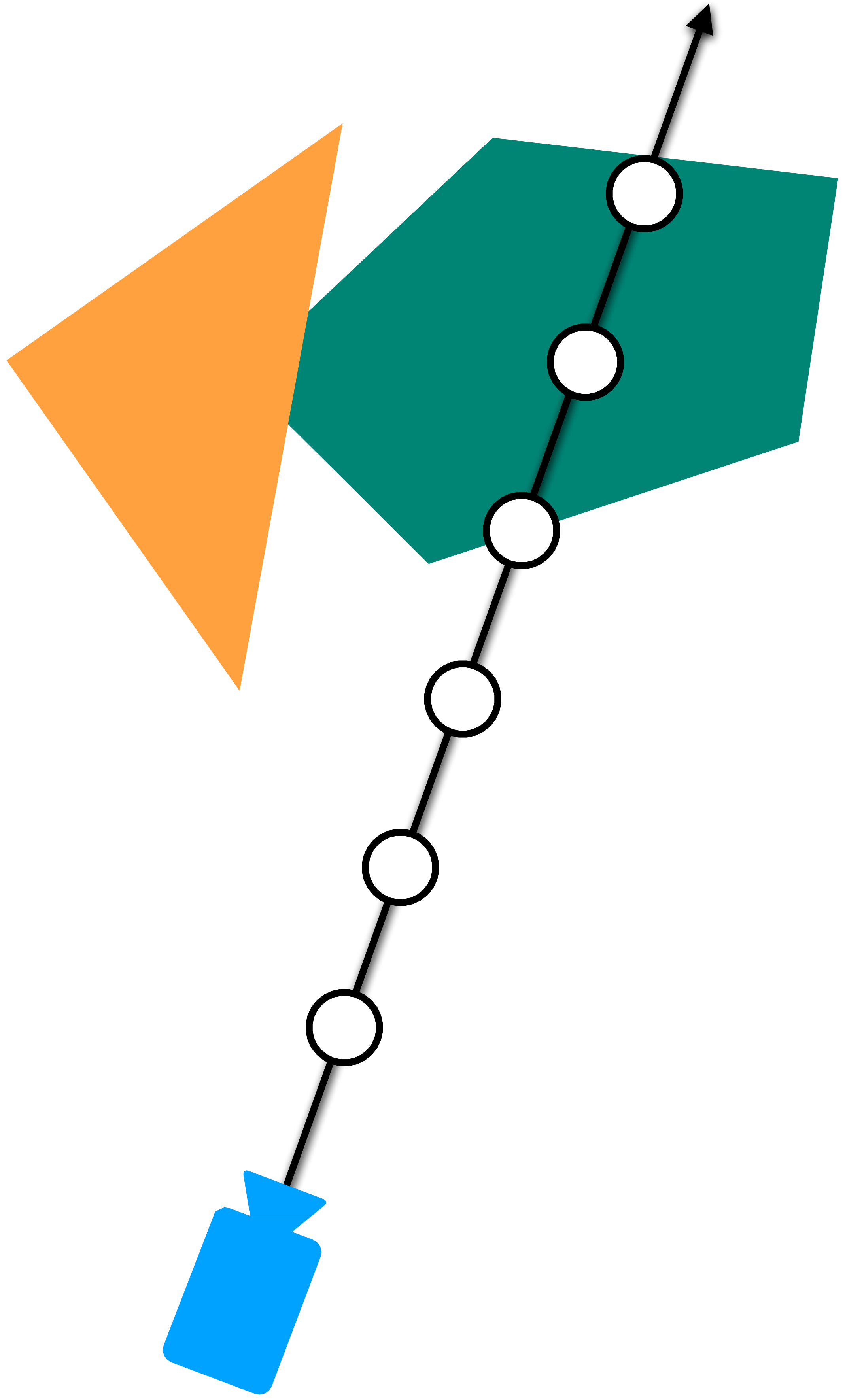
How much light is contributed by ray segment *i*:

$$\alpha_i = 1 - e^{-\sigma_i \delta t_i}$$

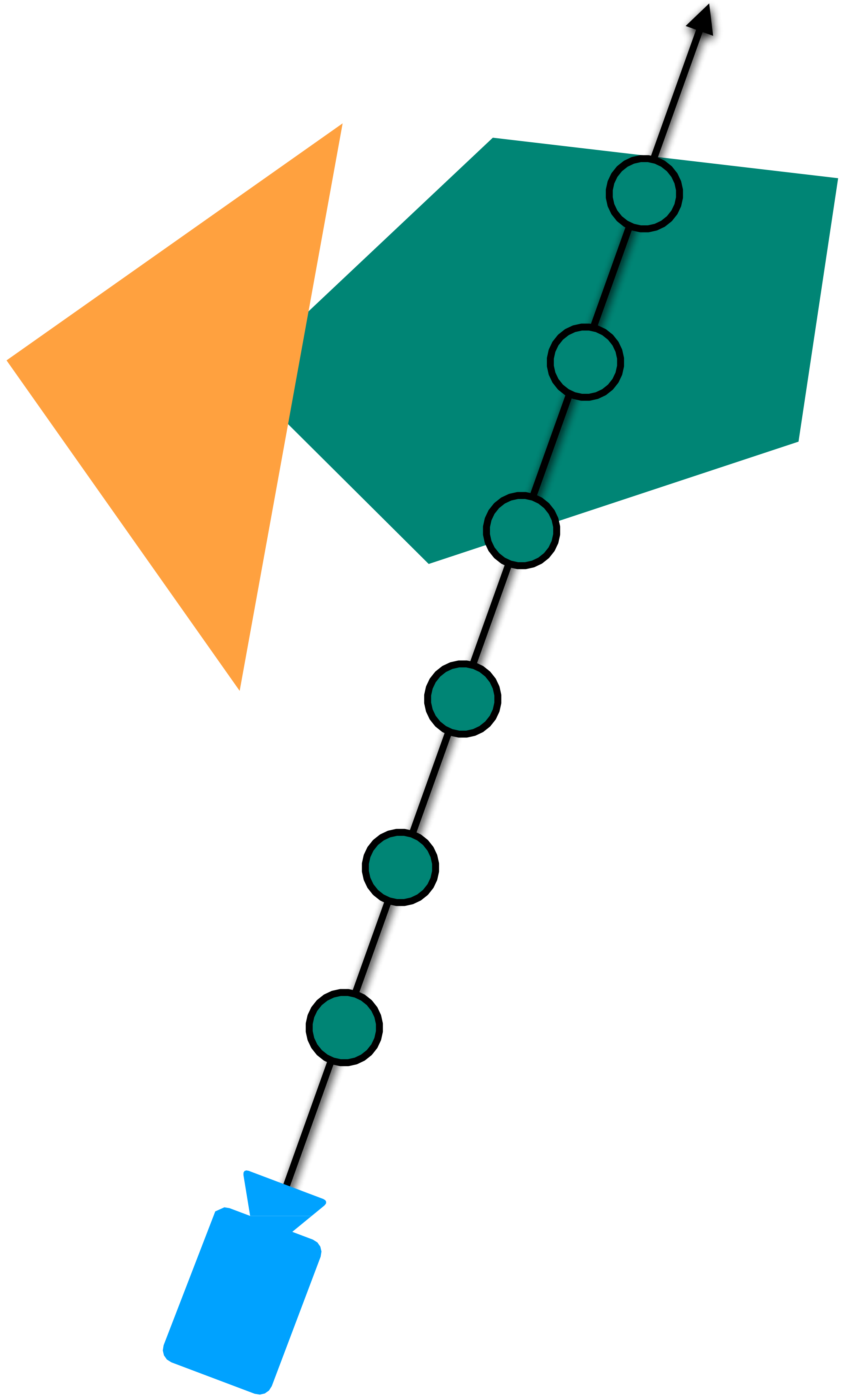


Training network to reproduce all input views of the scene

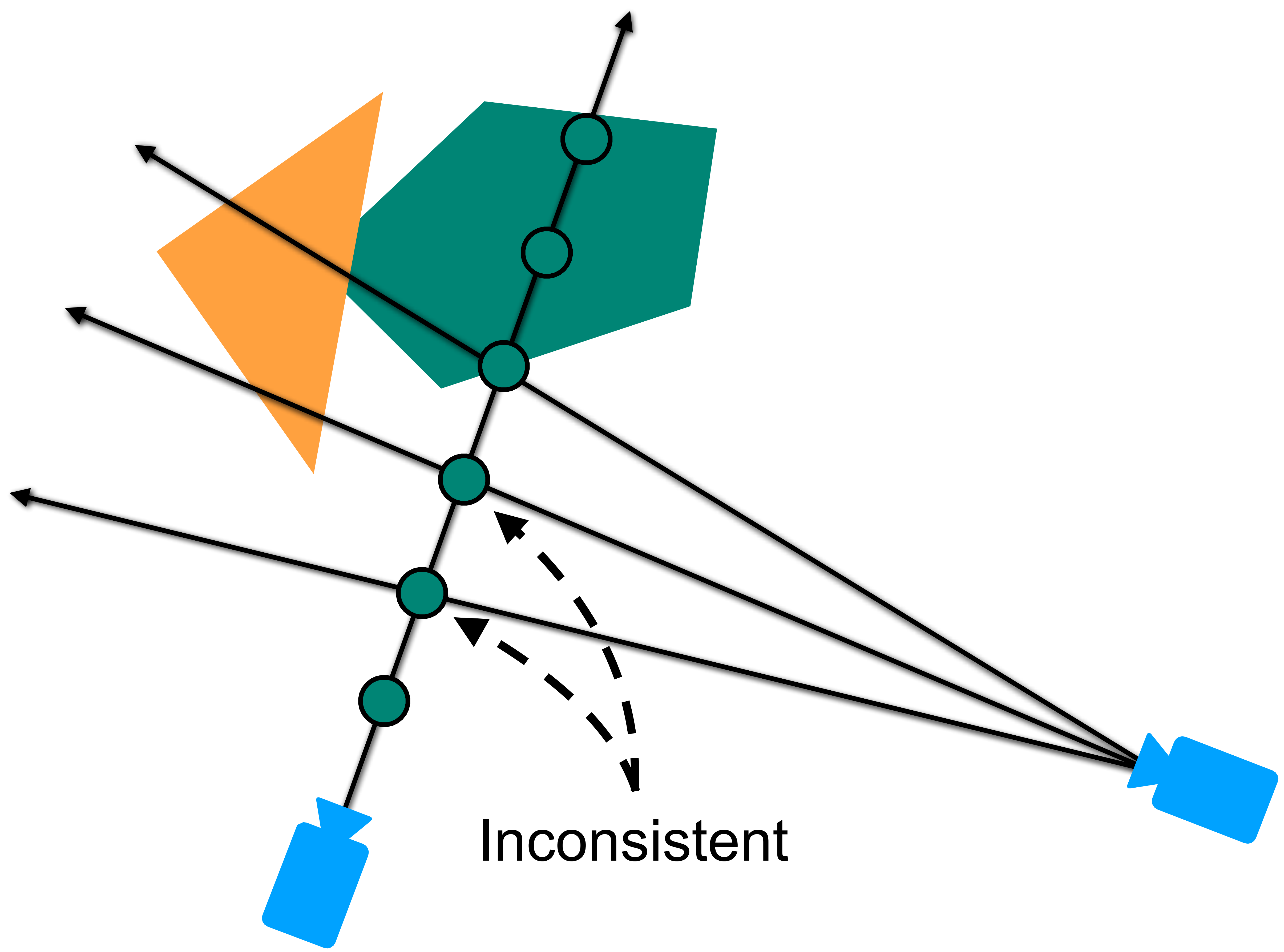
Multiview Consistency as Supervision



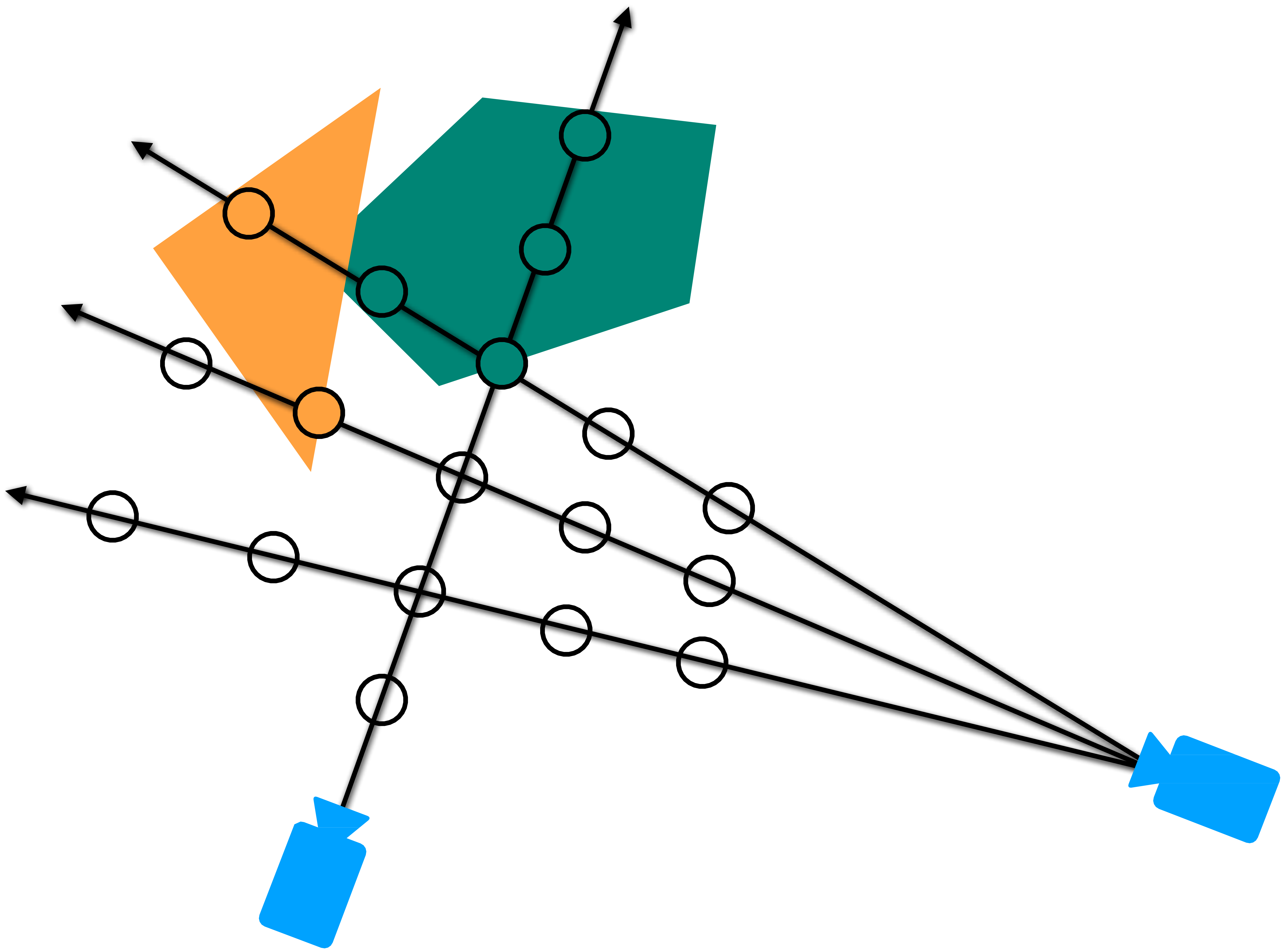
Multiview Consistency as Supervision



Multiview Consistency as Supervision



Multiview Consistency as Supervision



Results


NeRF encodes convincing view-dependent effects using directional dependence

NeRF encodes convincing view-dependent effects using directional dependence

NeRF encodes detailed scene geometry with occlusion effects

NeRF encodes detailed scene geometry with occlusion effects

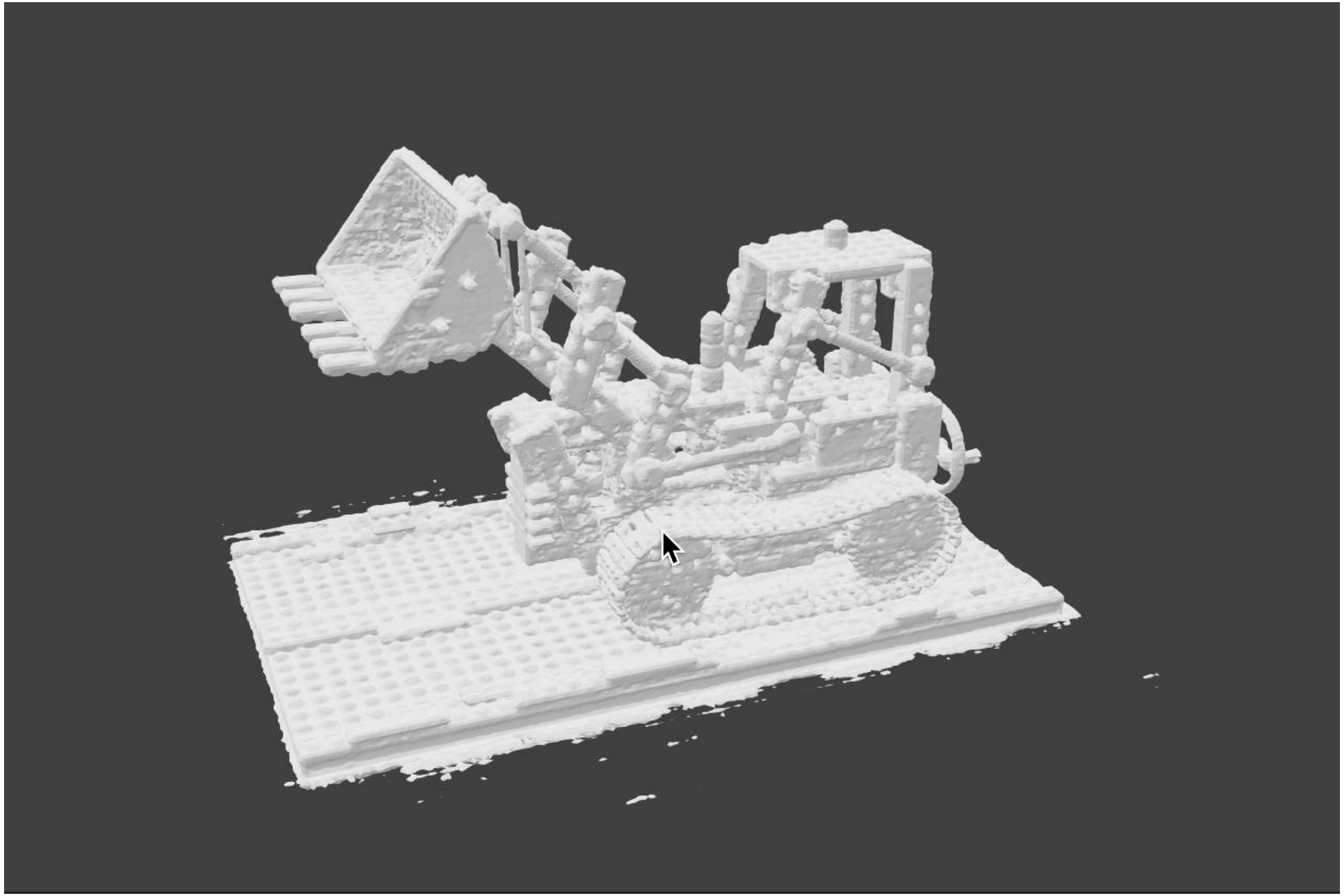
NeRF encodes detailed scene geometry with occlusion effects

NeRF encodes detailed scene geometry with occlusion effects

A great example! (not from the NeRF paper)

Notice the occlusion!

NeRF encodes detailed scene geometry



Jenin

Naive implementation produces blurry results

NeRF (Naive)

Naive implementation produces blurry results

NeRF (Naive)

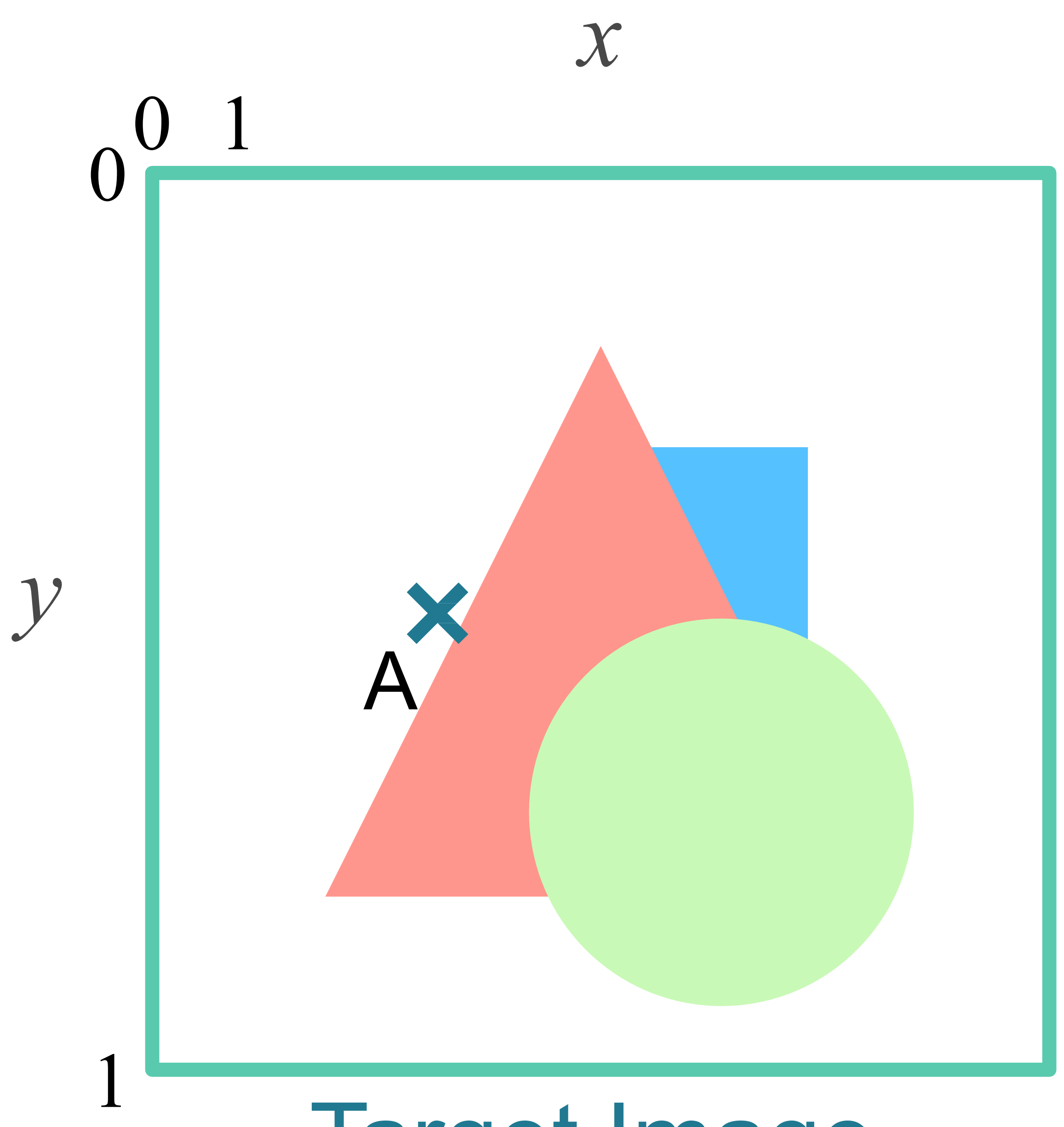
NeRF (with positional encoding)

PositionalEncodings

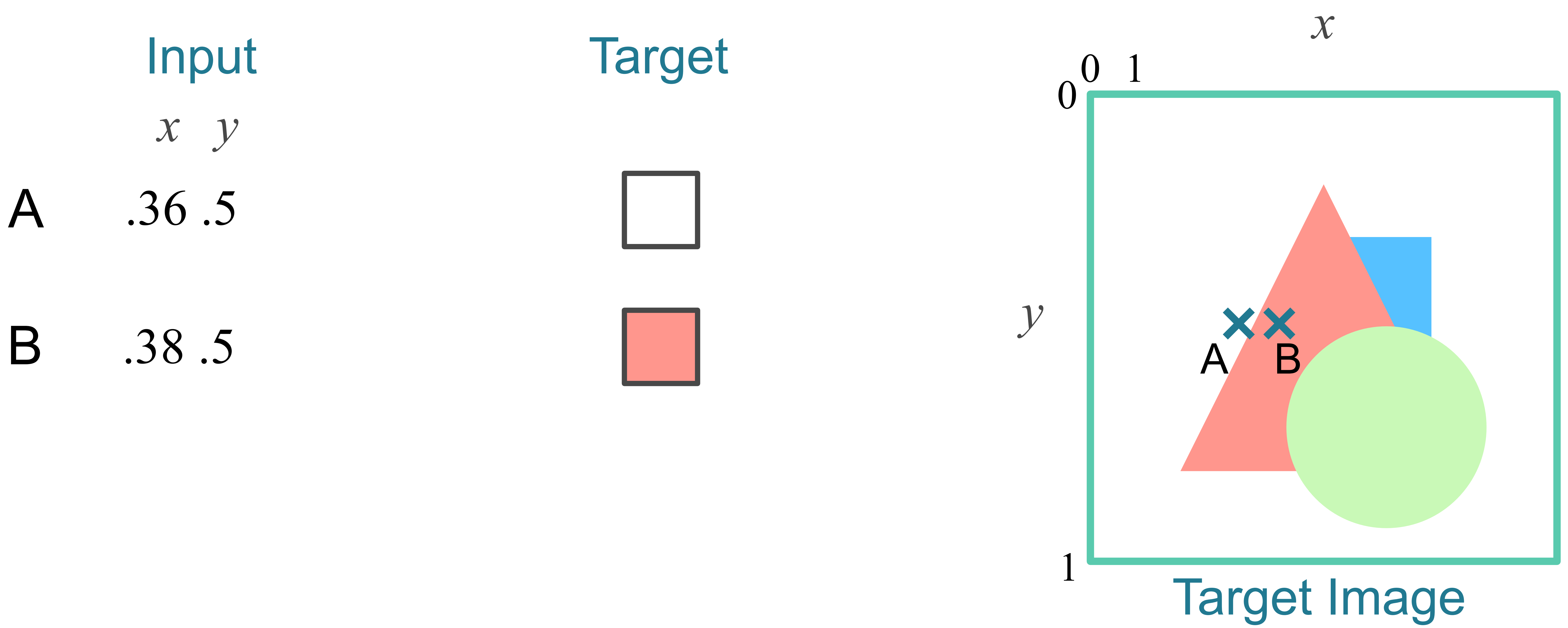
How to get neural networks to represent higher frequency functions?

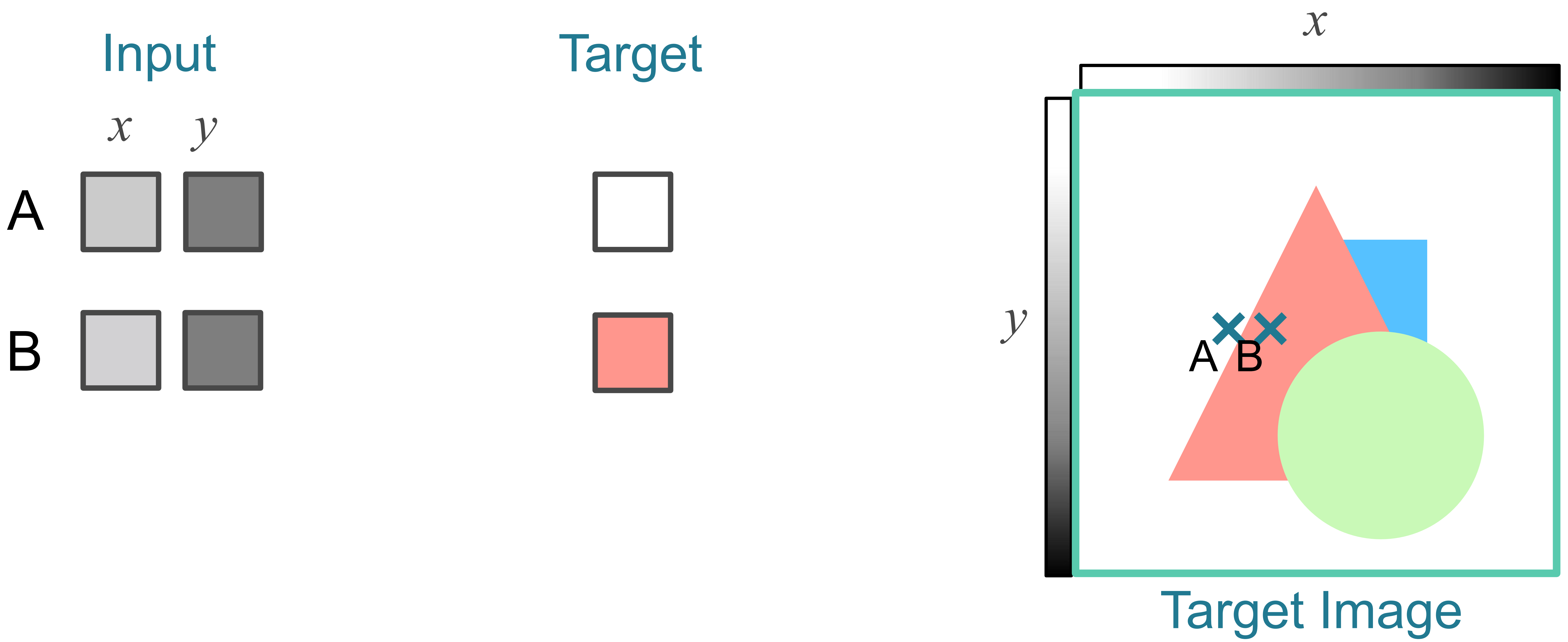
Target Image

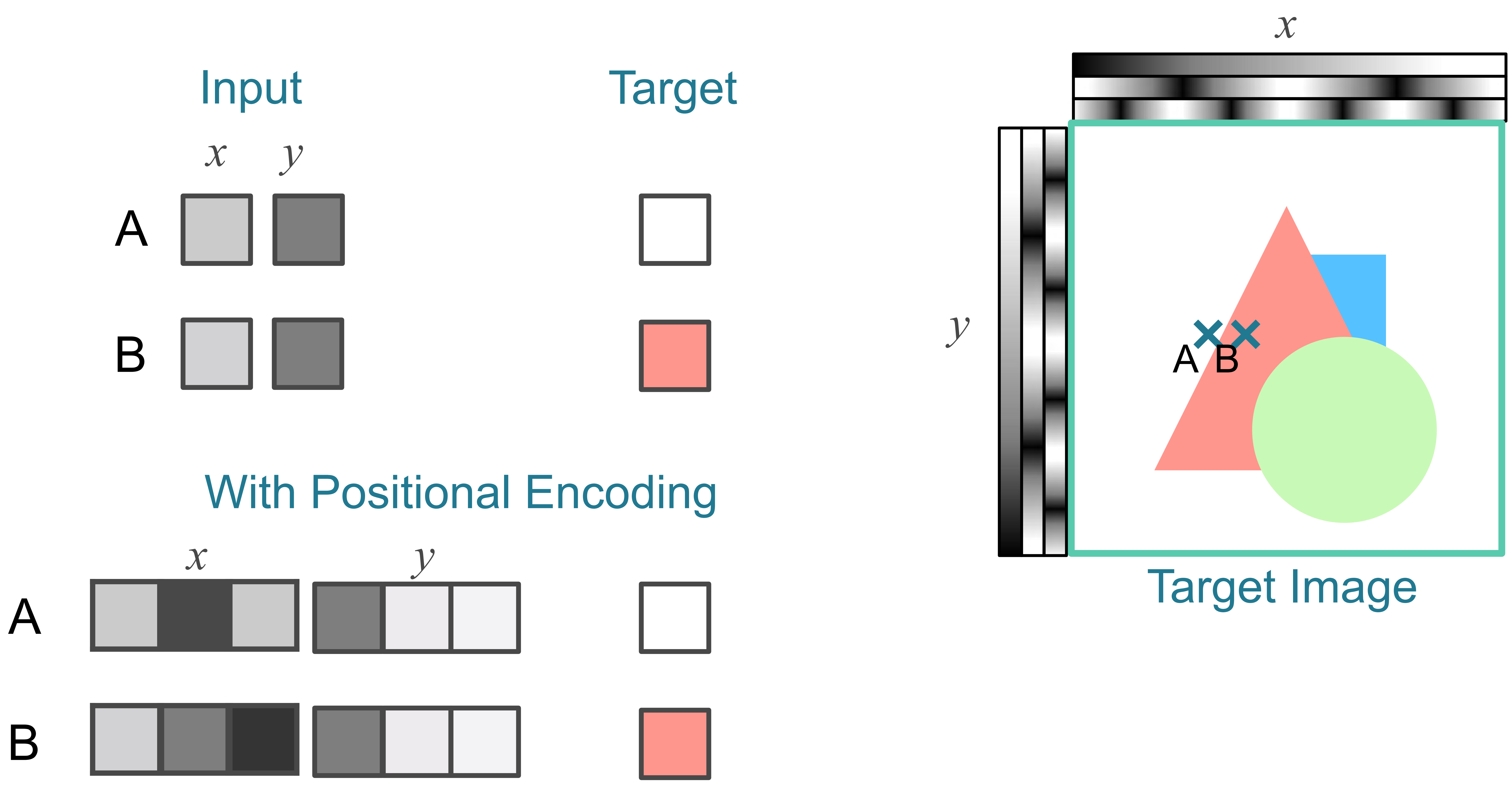
	Input		Target
	x	y	
A	.36	.5	



Target Image

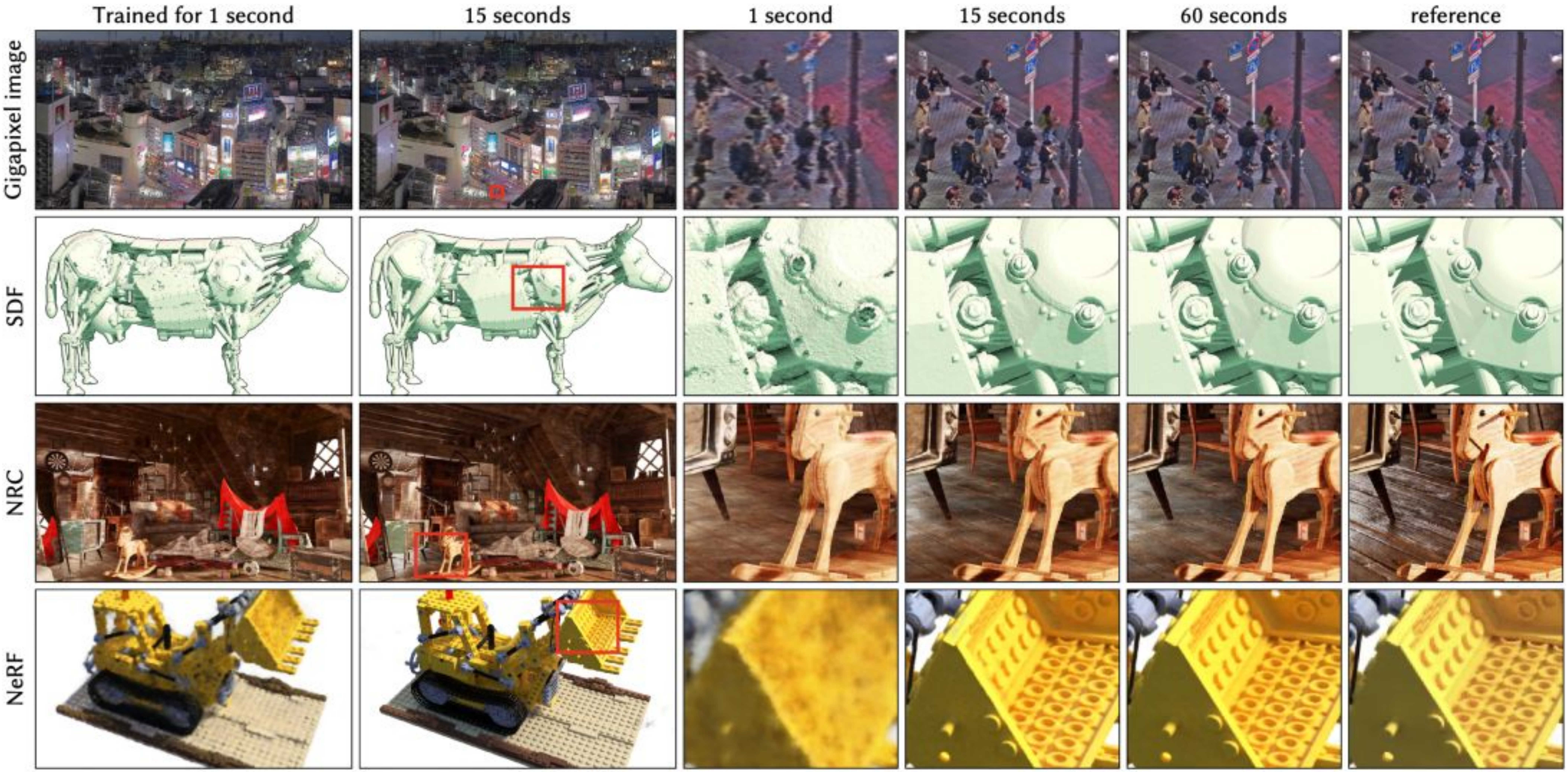

Input
x y
A .36 .5
B .38 .5
Target
x
0
1
0
y
A
B
1
Target Image


Input
x
y
A
B
Target
x
y
A
B
Target Image



Input
Target
x
y
A
B
With Positional Encoding
x
y
A
B
x
y
A
B
Target Image

Better positional encodings decreased training time

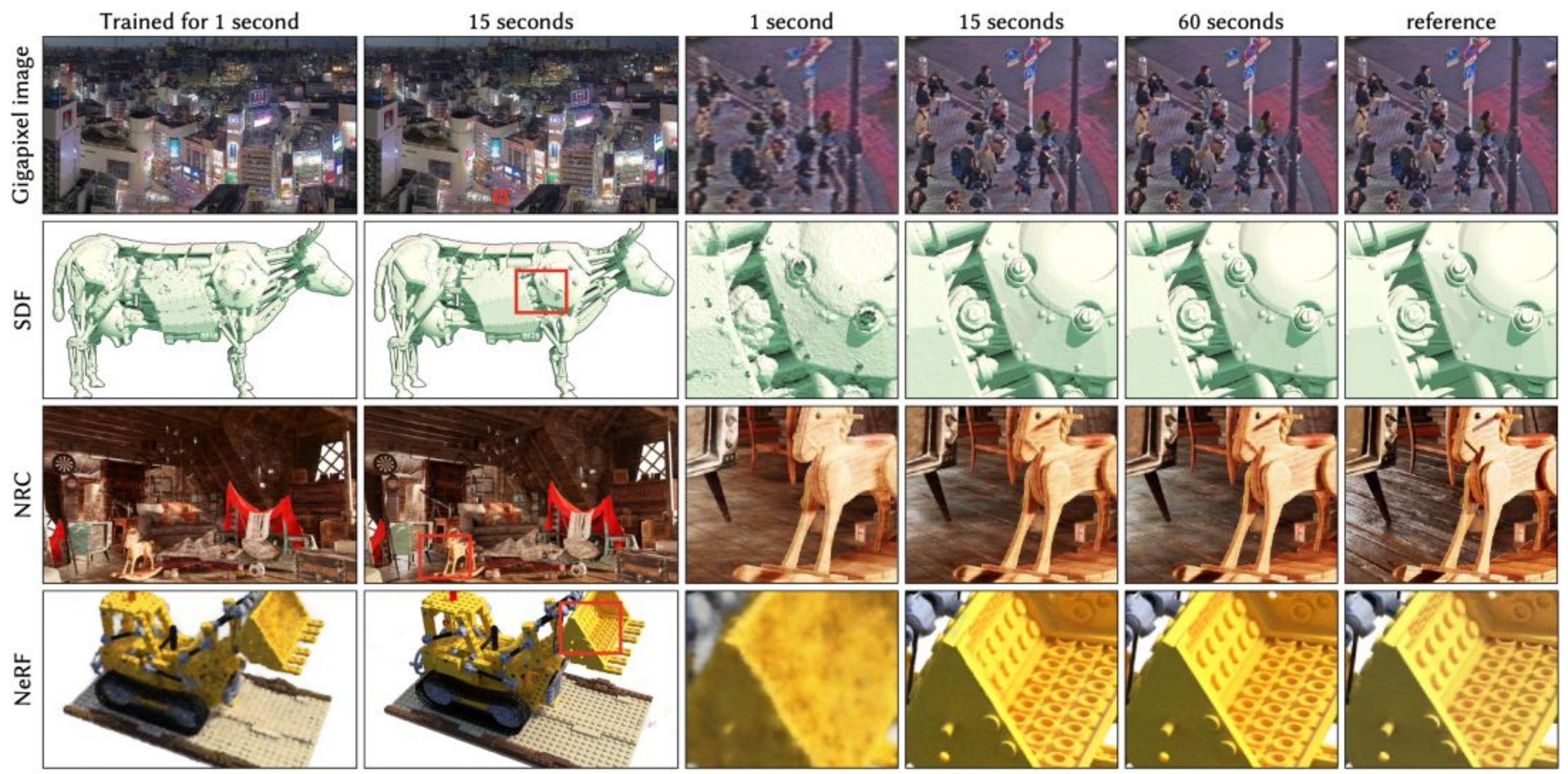


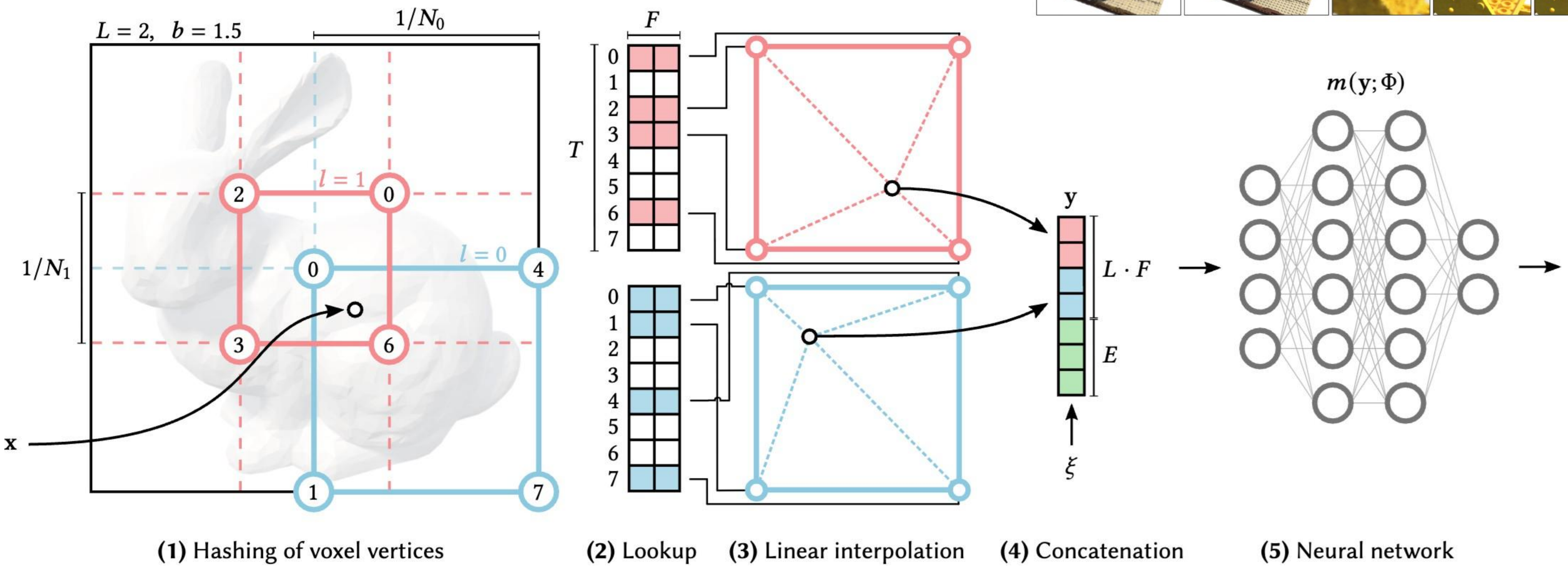
Better positional encodings decreased training time

Instant Neural Graphics Primitives with a Multiresolution Hash Encoding

THOMAS MÜLLER, NVIDIA, Switzerland
ALEX EVANS, NVIDIA, United Kingdom
CHRISTOPH SCHIED, NVIDIA, USA
ALEXANDER KELLER, NVIDIA, Germany

https://nvlabs.github.io/instant-ngp





Scene Contraction

How can we represent unbounded spaces?

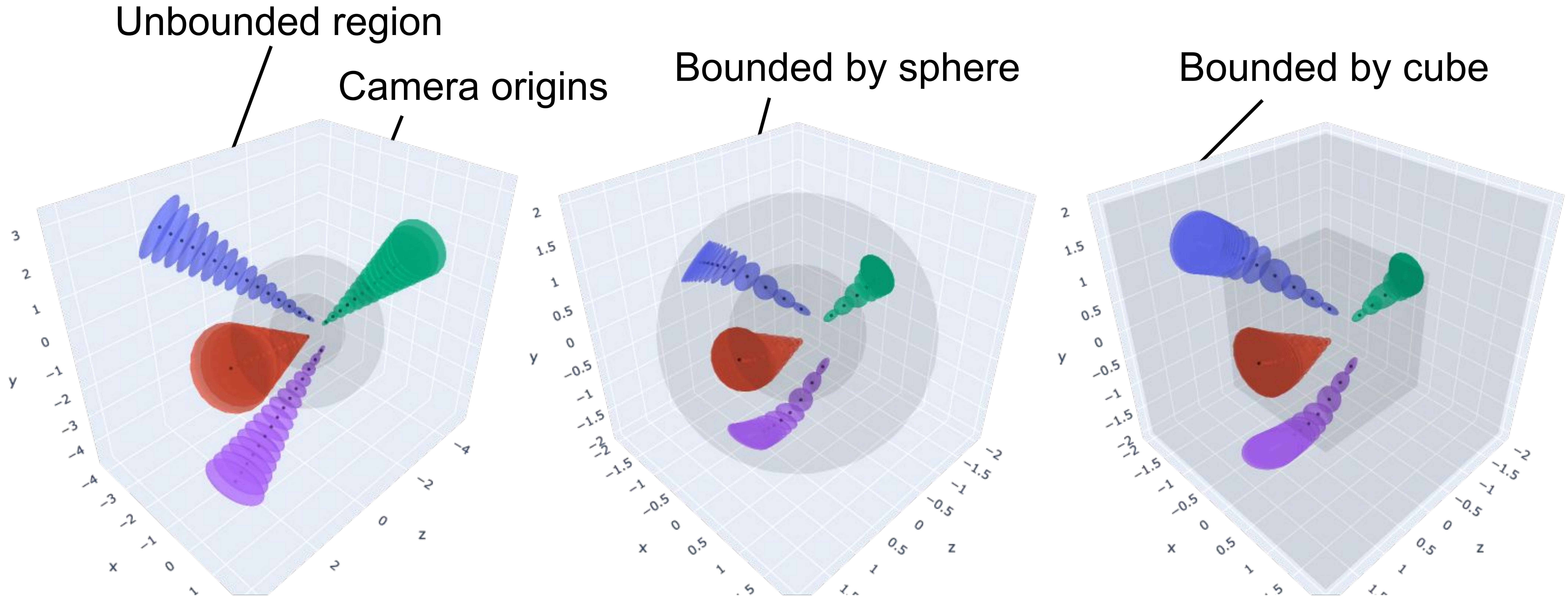
Mip-NeRF 360

Mip-NeRF 360: Unbounded Anti-Aliased Neural Radiance Fields

Jonathan T. Barron[1] Ben Mildenhall[1] Dor Verbin[1,2]
Pratul P. Srinivasan[1] Peter Hedman[1]
[1]Google Research [2]Harvard University

Major idea: use a contracted and bounded region as input to an MLP or hash grid

https://jonbarron.info/mipnerf360/



No Scene Contraction $\qquad$ L_2 (MipNeRF-360) Scene Contraction $\qquad$ L_∞ Scene Contraction

Major idea: use a contracted and bounded region as input to an MLP or hash grid

https://docs.nerf.studio/nerfology/model_components/visualize_spatial_distortions.html

Appearance Embeddings

How can we handle varying camera exposure or lighting changes?

NeRF in the Wild

NeRF in the Wild: Neural Radiance Fields for Unconstrained Photo Collections

Ricardo Martin-Brualla, Noha Radwan, Mehdi S. M. Sajjadi, Jonathan T. Barron, Alexey Dosovitskiy, Daniel Duckworth

Brandenburg Gate in Berlin

NeRF in the Wild

NeRF in the Wild: Neural Radiance Fields for Unconstrained Photo Collections

Ricardo Martin-Brualla, Noha Radwan, Mehdi S. M. Sajjadi, Jonathan T. Barron, Alexey Dosovitskiy, Daniel Duckworth

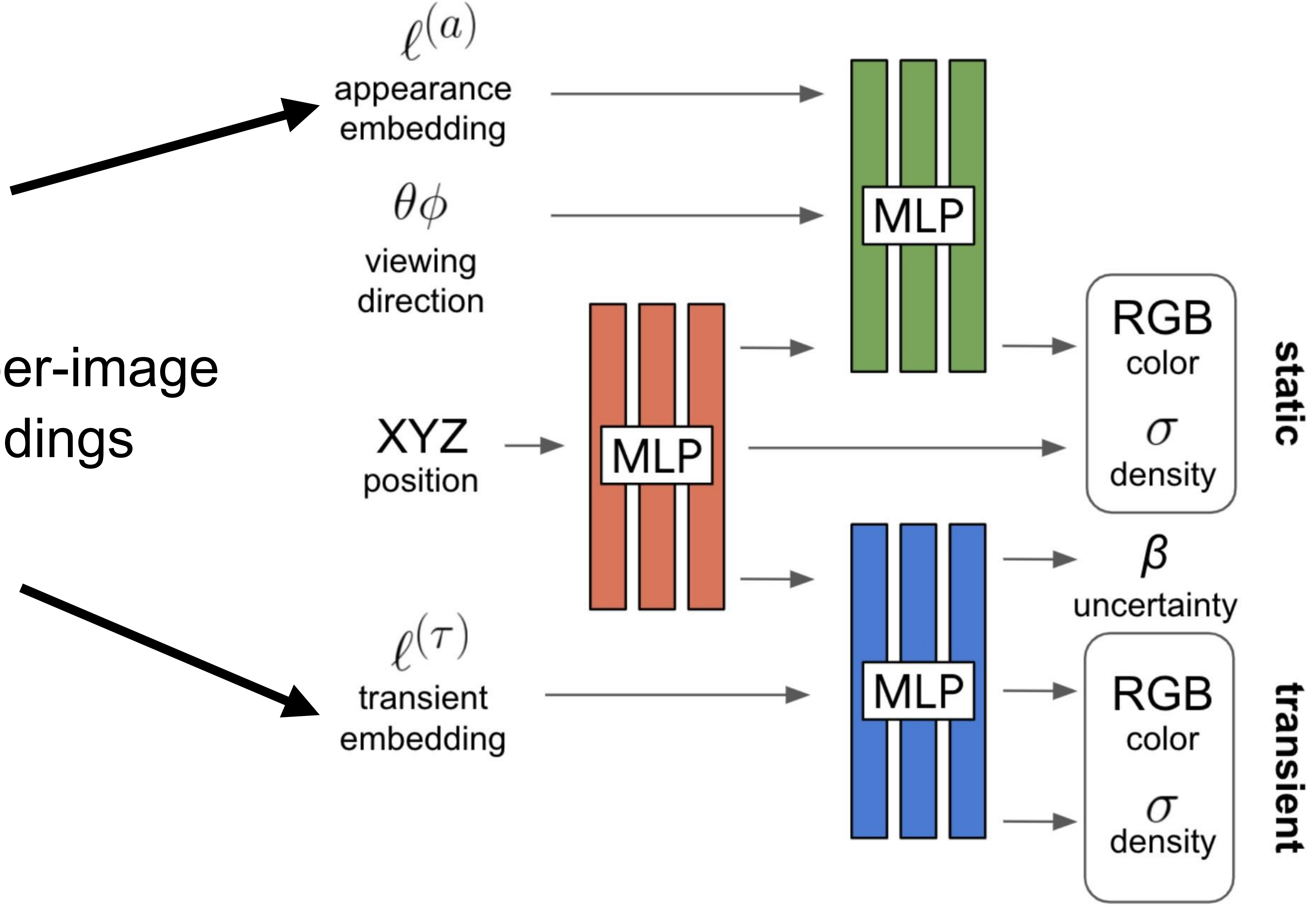
Brandenburg Gate in Berlin

NeRF in the Wild

NeRF in the Wild: Neural Radiance Fields for Unconstrained Photo Collections

Ricardo Martin-Brualla, Noha Radwan, Mehdi S. M. Sajjadi, Jonathan T. Barron, Alexey Dosovitskiy, Daniel Duckworth

Learned per-image embeddings



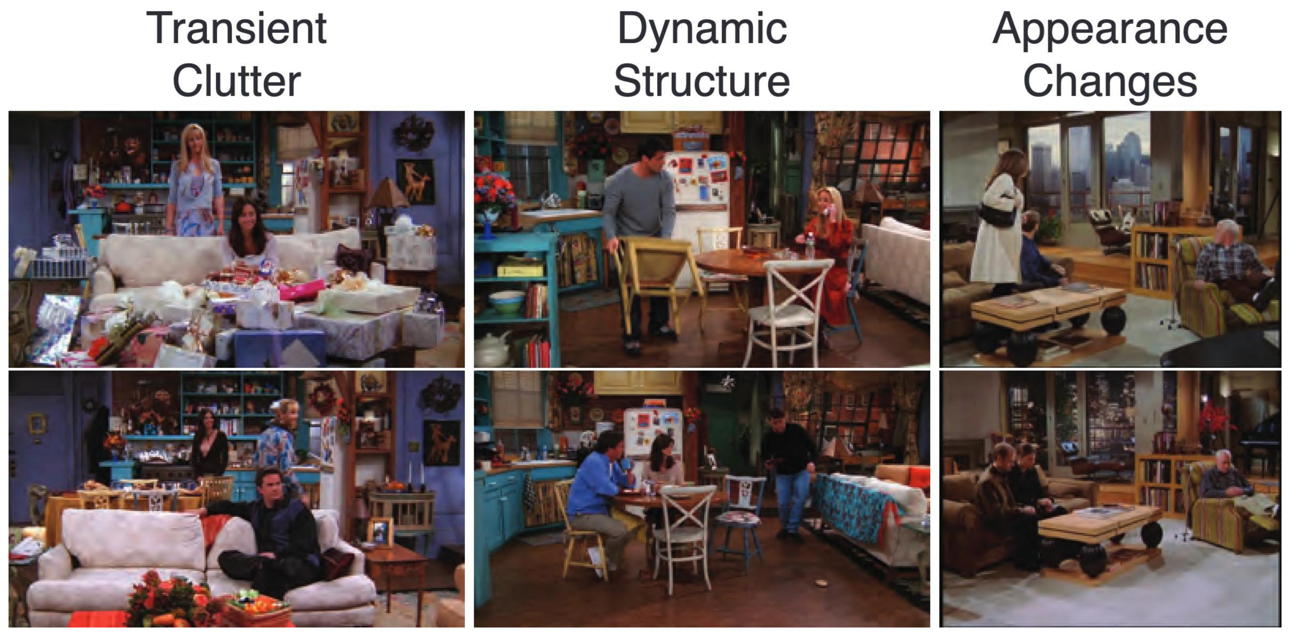
Appearance Embeddings

NeRF-W (Martin-Brualla* & Radwan* et al

Block-NeRF (Tancik et al)

Splatfacto-W (Xu et al)

3D Sitcoms



Nerfacto (Nerfstudio)

Thank You!

3D Reconstruction

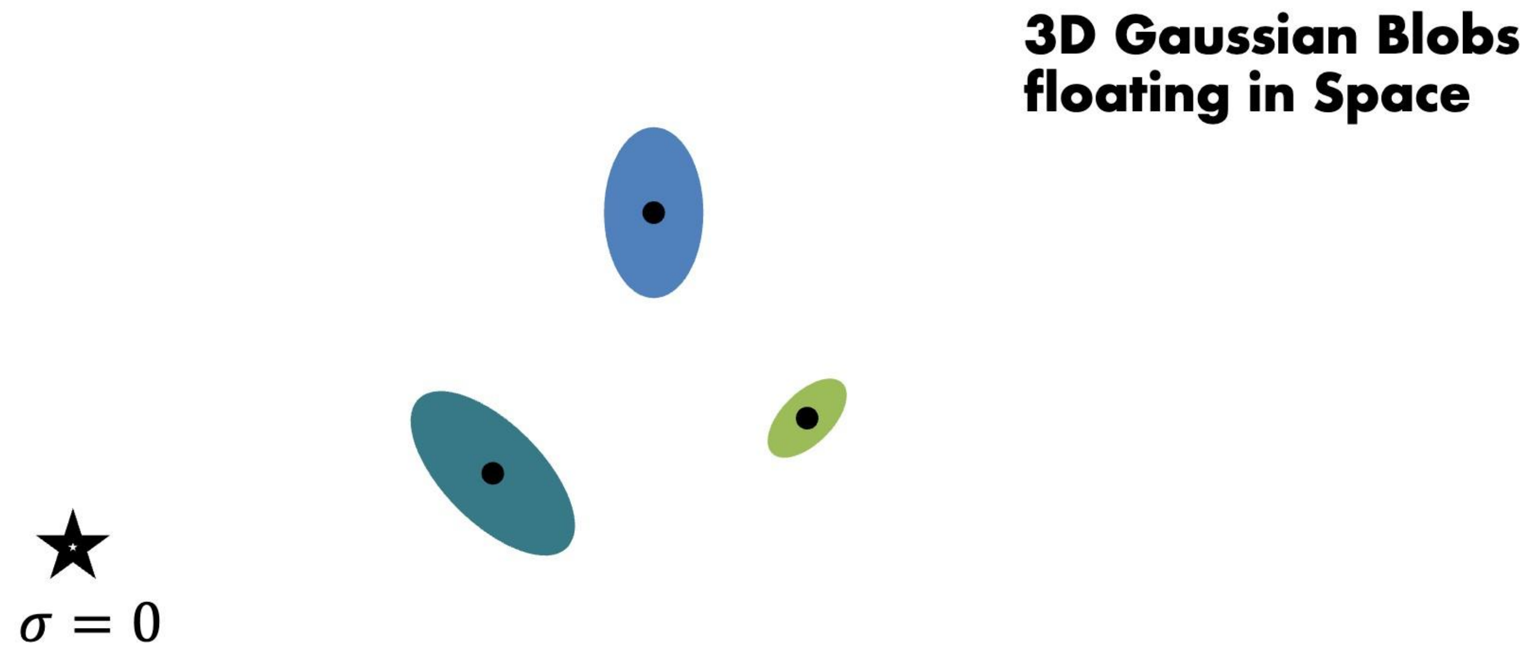
Novel-View Synthesis

NeRF

$\boxed{\text{3D Gaussian Splatting}}$

3D Gaussian Splatting (3DGS)

Key Idea: Parameterize Radiance Field *sparsely, only where density is nonzero*



Resource: lecture notes from Stanford (adapted from MIT) here

3D Gaussian Splatting for Real-Time Radiance Field Rendering

BERNHARD KERBL*, Inria, Université Côte d'Azur, France
GEORGIOS KOPANAS*, Inria, Université Côte d'Azur, France
THOMAS LEIMKÜHLER, Max-Planck-Institut für Informatik, Germany
GEORGE DRETTAKIS, Inria, Université Côte d'Azur, France

Rasterize shape primitives instead of sampling a field

3D Gaussian Splatting (3DGS)

gsplat (https://docs.gsplat.studio)

2D toy example on an image

Parameters to optimize:

- **Color**
- **Opacity**
- **Position**
- **Scale(s)**
- **Rotation**

Optimization tricks:

Initialization Culling & pruning Splitting & densify Coarse to fine

3D Gaussian Splatting (3DGS)

r/GaussianSplatting Search in r/GaussianSplatting

Going into iconic movie scenes using gaussian splats

Here are the interactive gaussian splats for each scene:

LOTR: https://lumalabs.ai/capture/176ED9AA-514F-4A45-9343-D4C708C86570

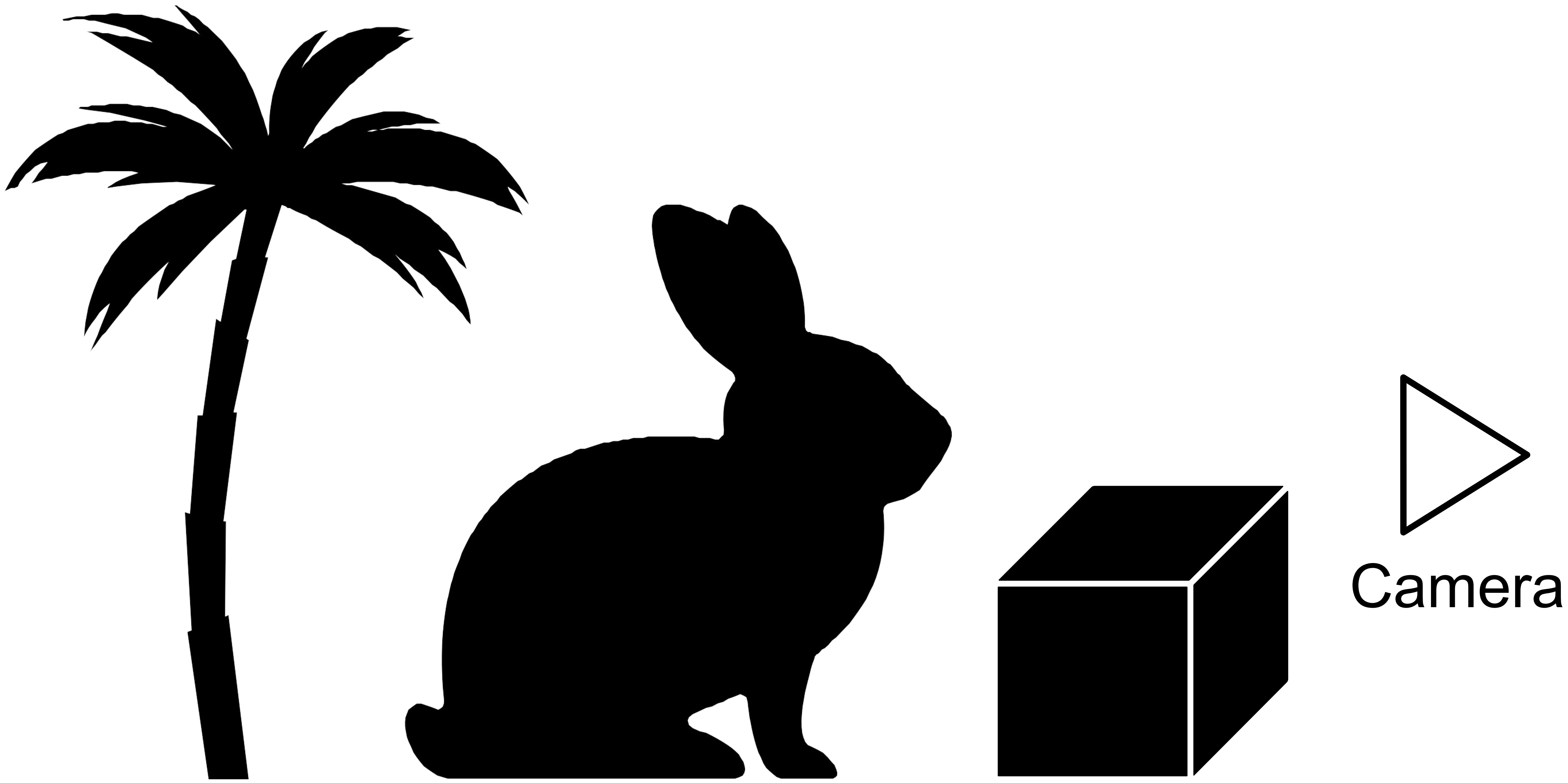
Matrix: https://lumalabs.ai/capture/F358C359-42BE-44B6-BA81-D58C7F75E19D

Citizen Kane: https://lumalabs.ai/capture/4ED192E4-44C9-4550-BC80-2CB130753F5D

Wizard of Oz: https://lumalabs.ai/capture/3D8B463B-62FF-43AF-AD42-B1E47C1213D5

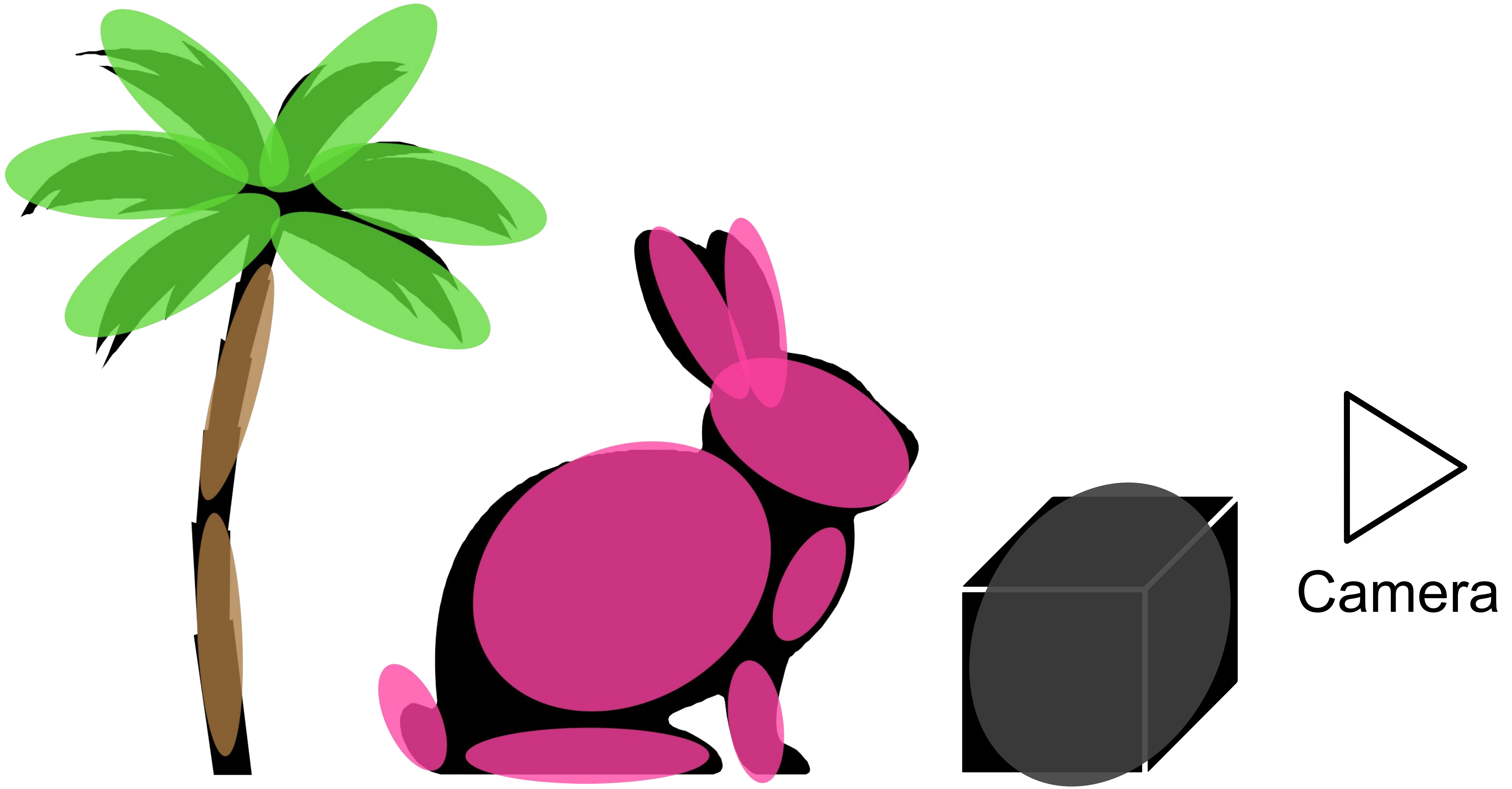
Terminator 2: https://lumalabs.ai/capture/220C2F41-E512-455C-B3EE-47CDD4398743

3D Gaussian Splatting (3DGS)



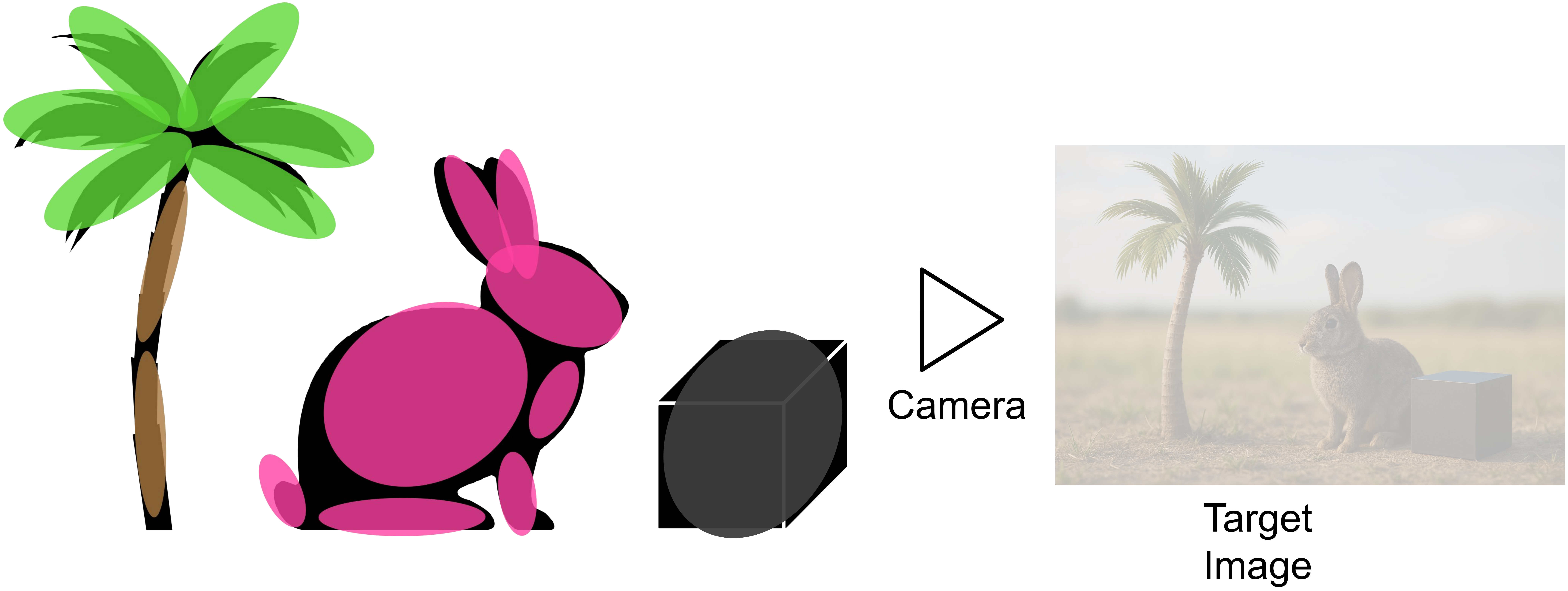
Target
Image

3D Gaussian Splatting (3DGS)

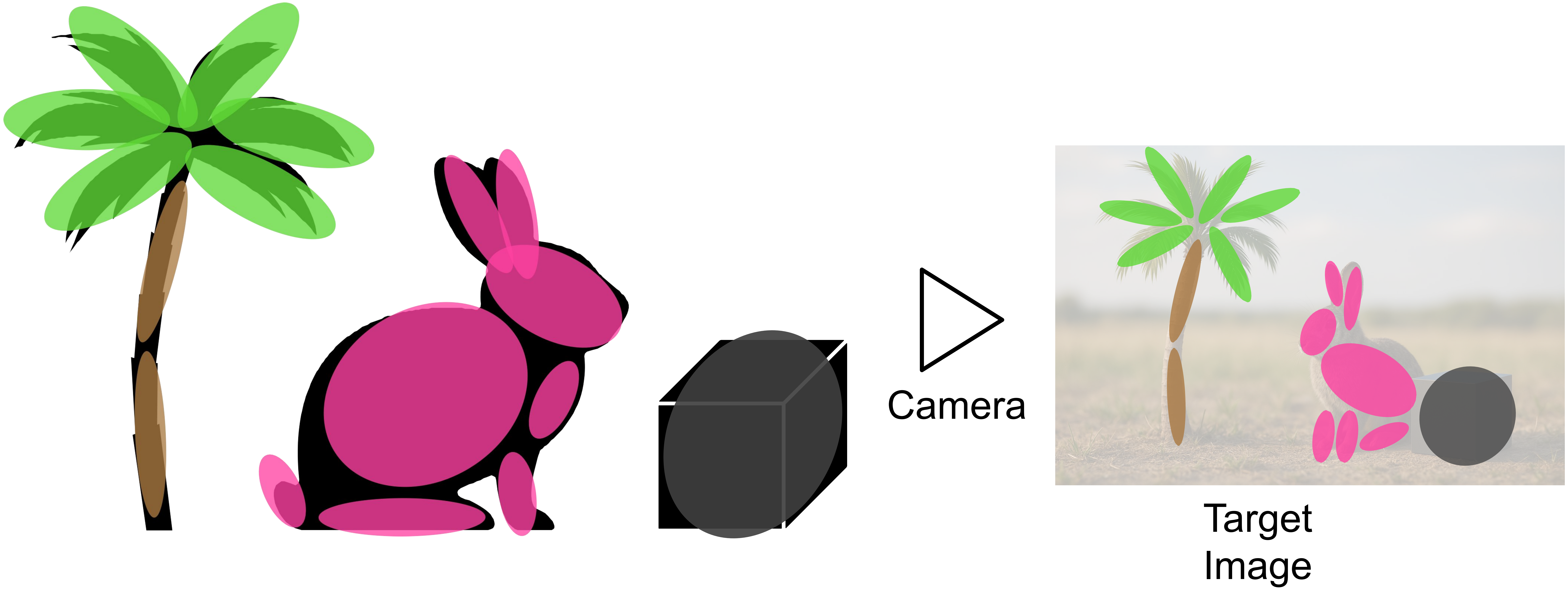


Target Image

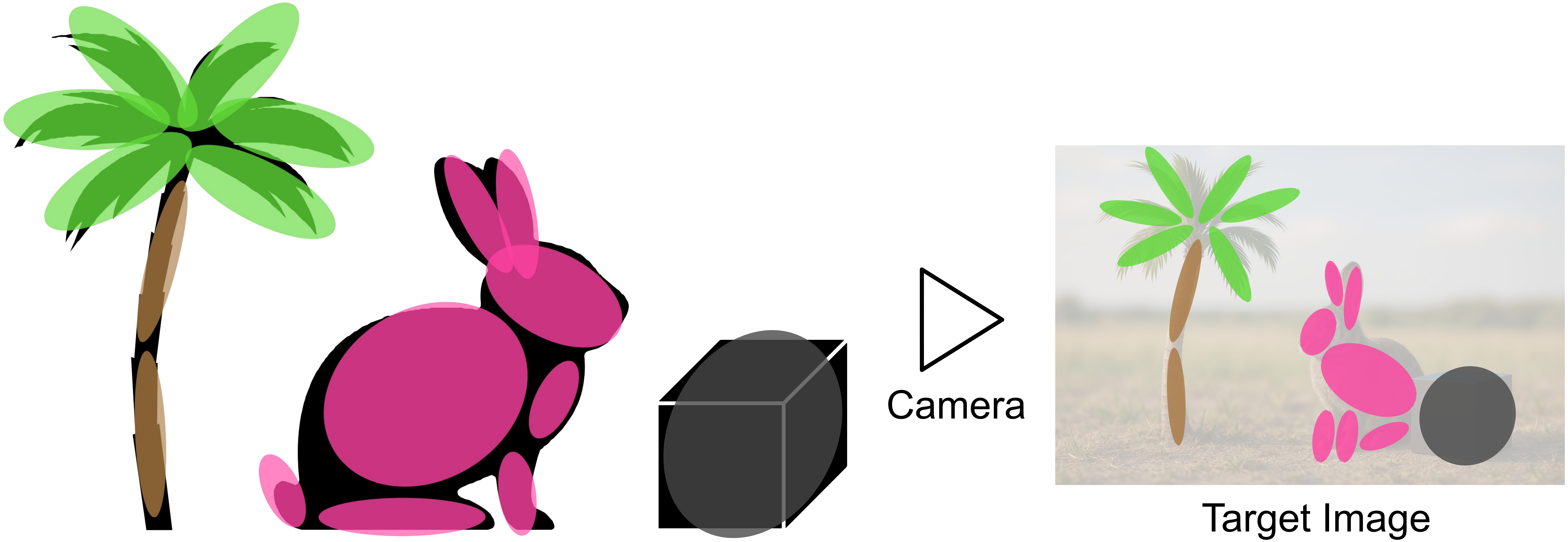
3D Gaussian Splatting (3DGS)



3D Gaussian Splatting (3DGS)

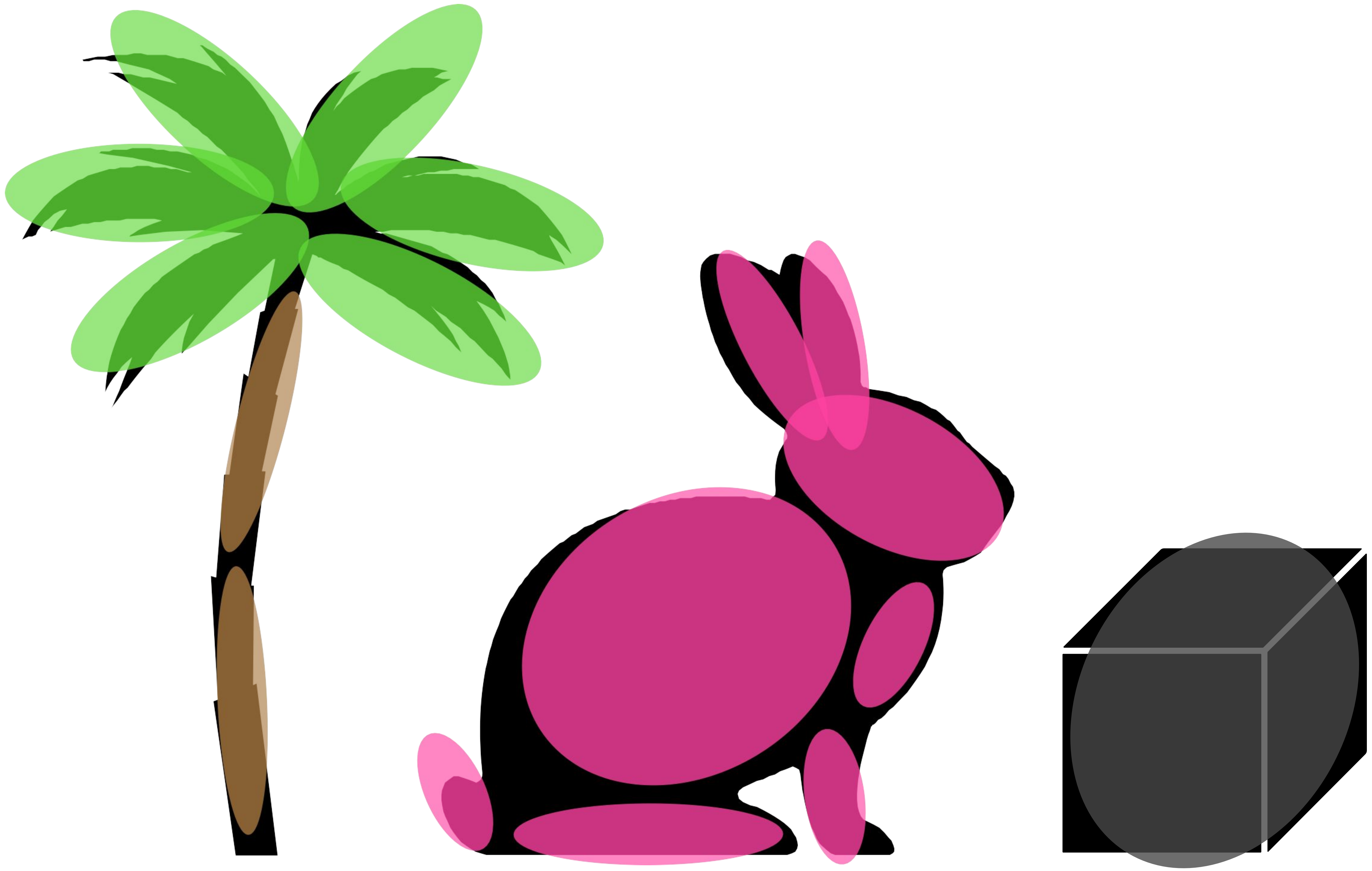


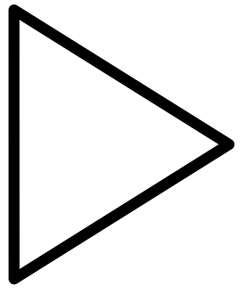
3D Gaussian Splatting (3DGS)



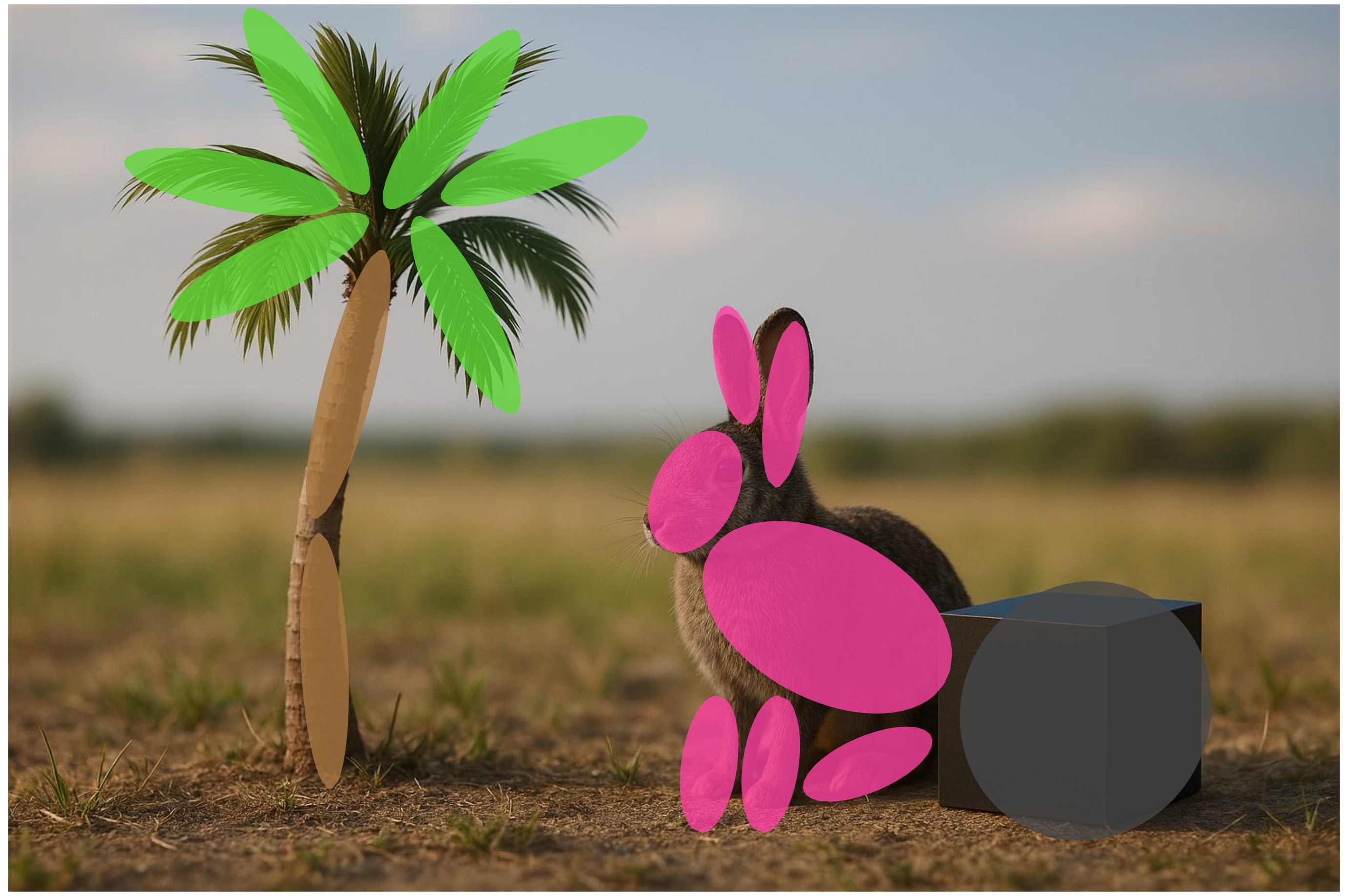
Optimize Gaussians to match the target image

3D Gaussian Splatting (3DGS)





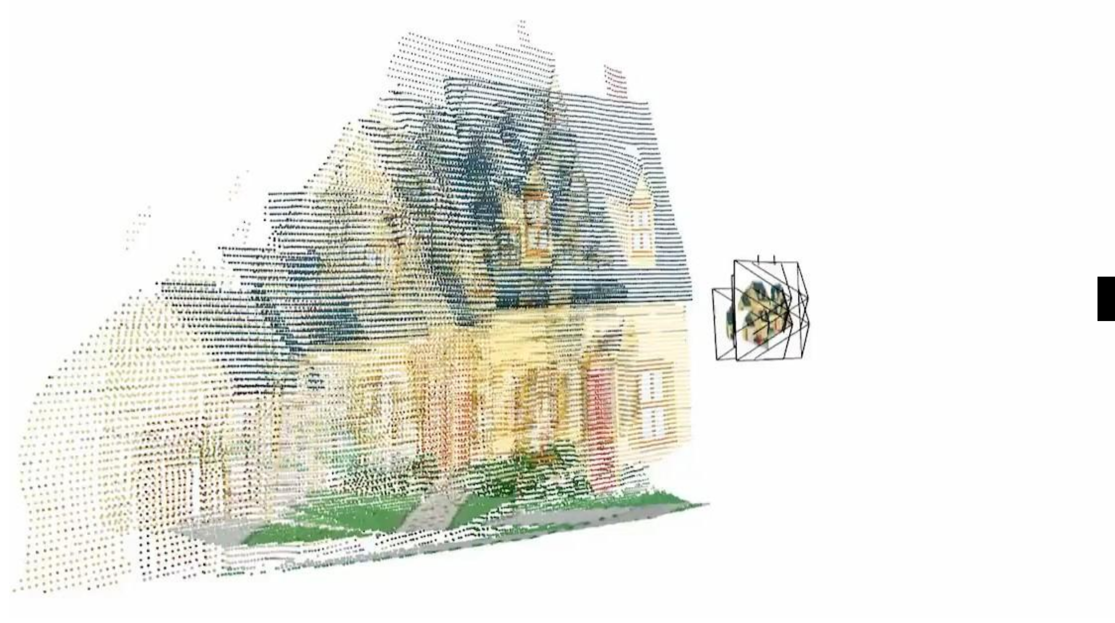
Camera



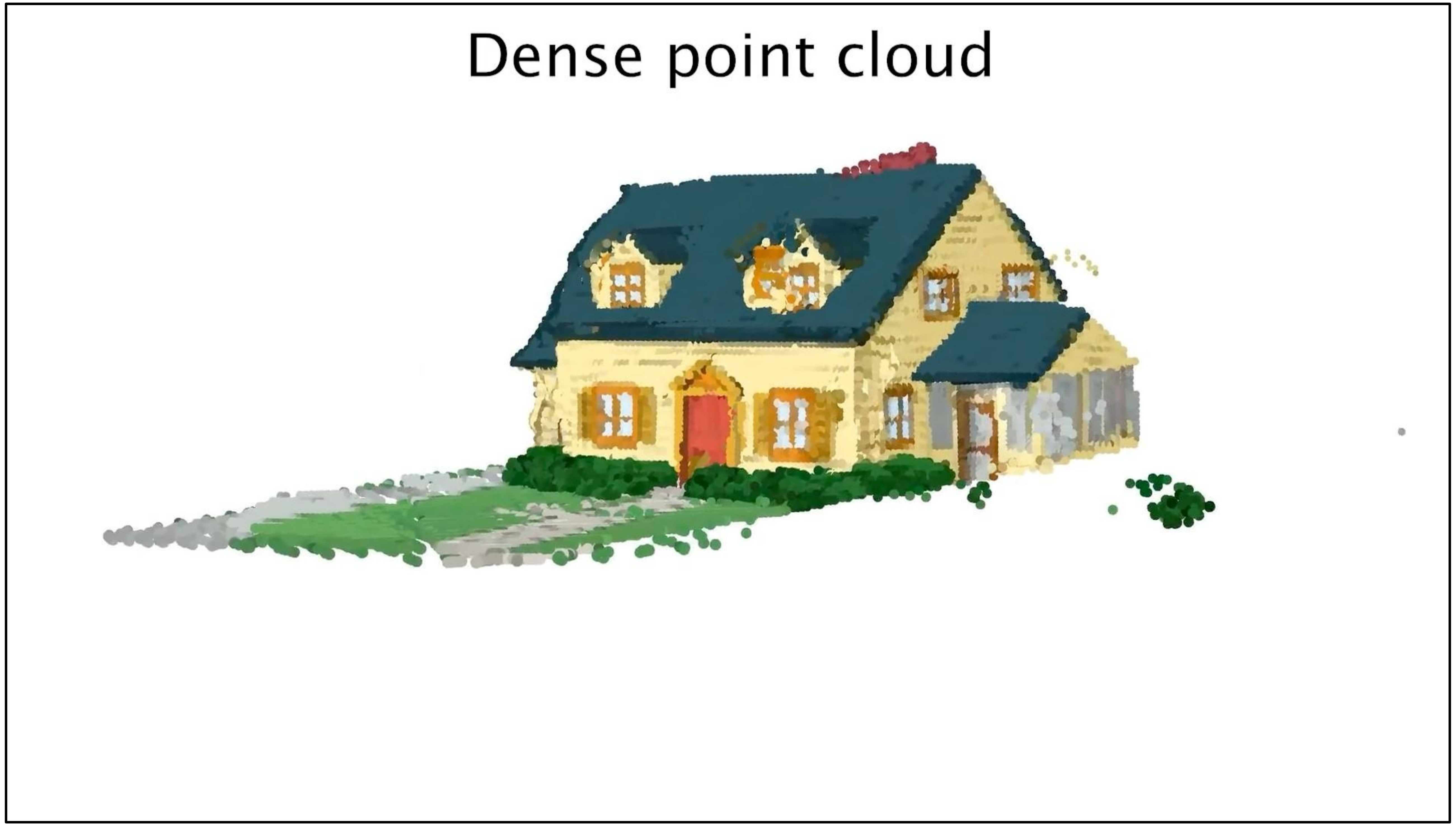
Target Image

3D Gaussian Splatting (3DGS)

Toon3D Gaussian Splatting For Better Visualizations



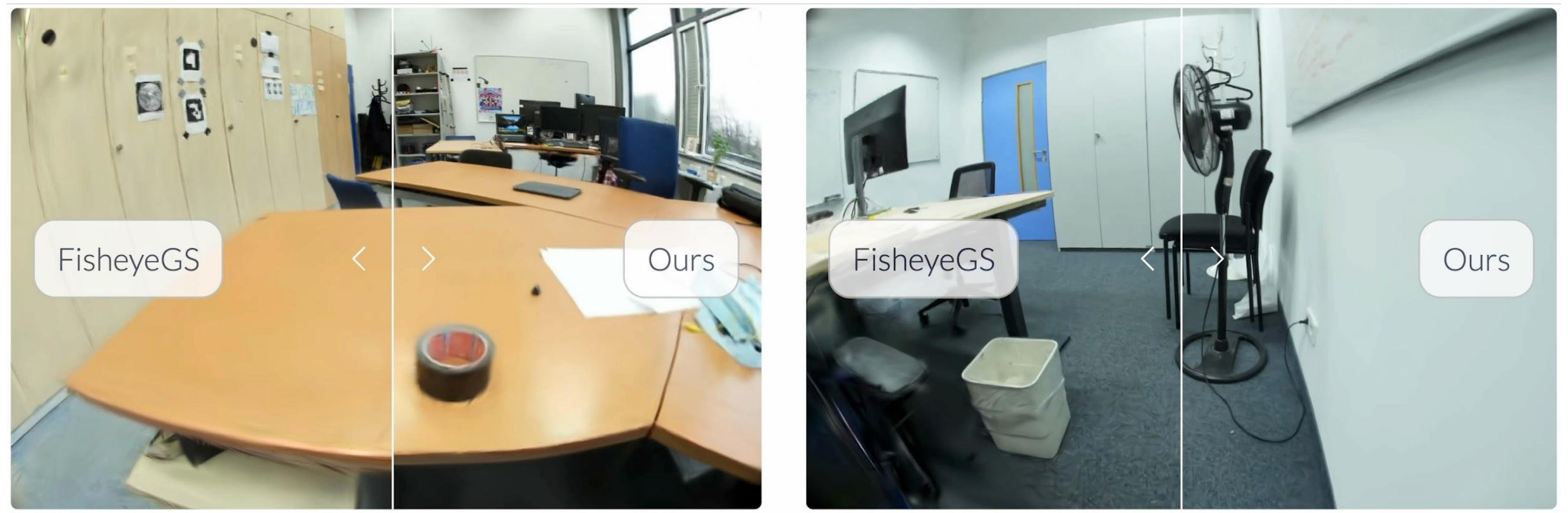
Optimize camera and align points

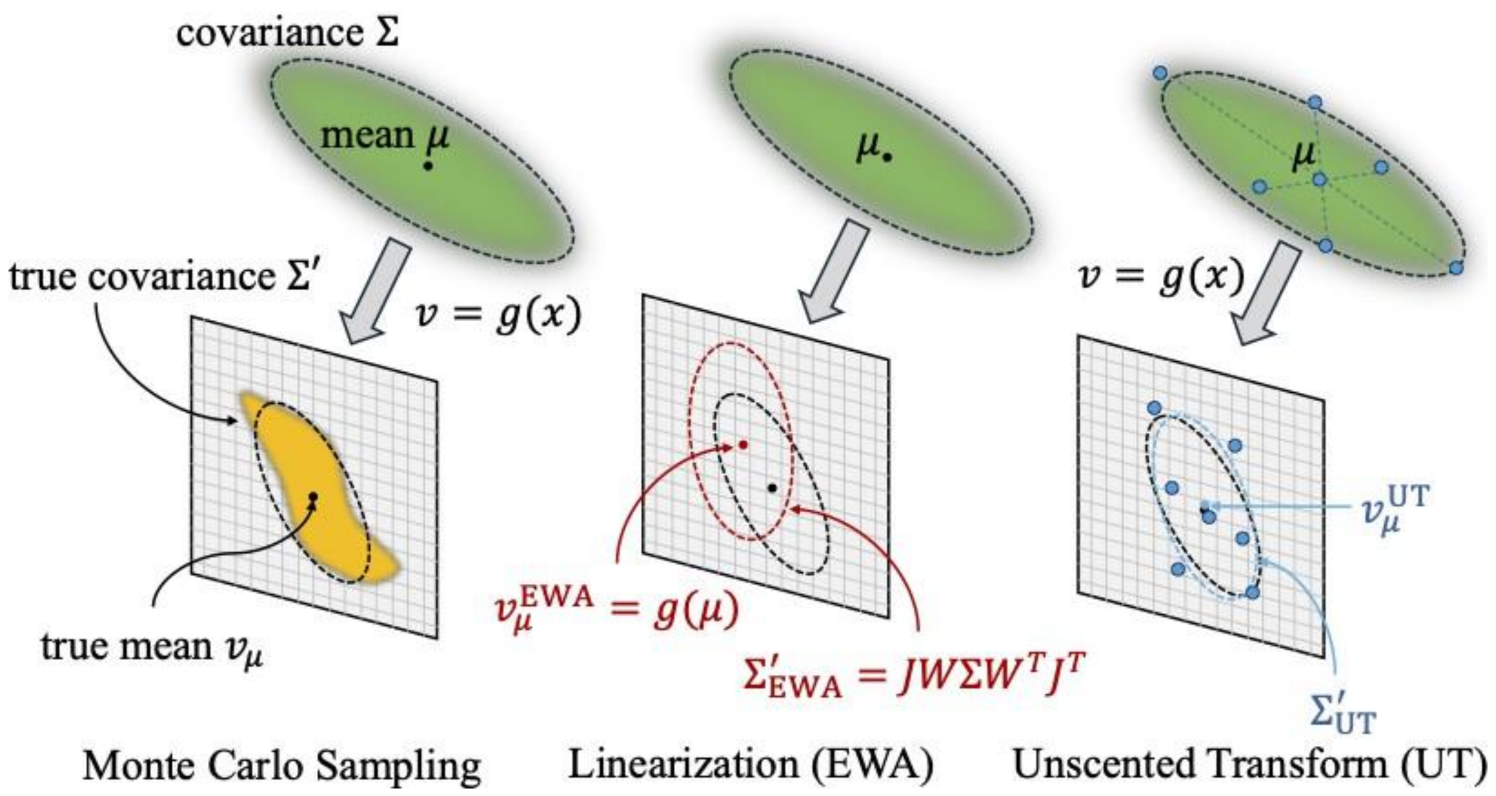


Follow-ups to Gaussian Splatting

3DGUT: Enabling Distorted Cameras and Secondary Rays in Gaussian Splatting

Qi Wu[1,*], Janick Martinez Esturo[1,*], Ashkan Mirzaei[1,2], Nicolas Moenne-Loccoz[1], Zan Gojcic[1]
[1]NVIDIA, [2]University of Toronto



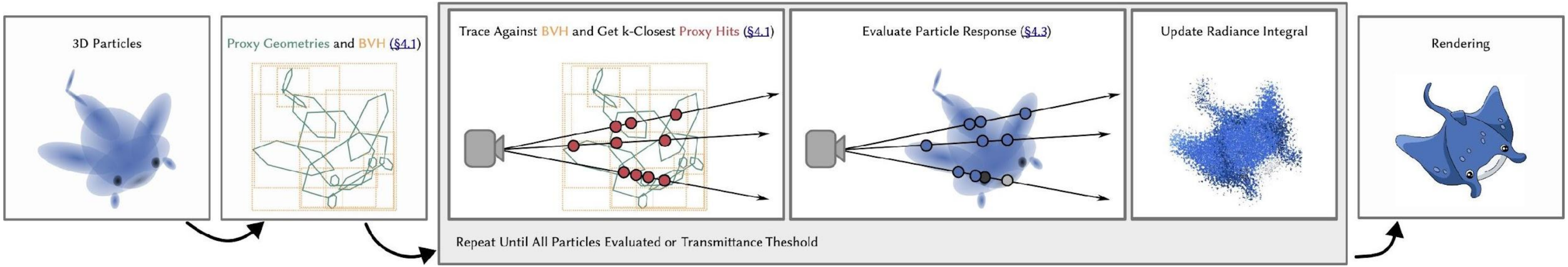


3D Gaussian Ray Tracing: Fast Tracing of Particle Scenes

NICOLAS MOENNE-LOCCOZ*, NVIDIA, Canada
ASHKAN MIRZAEI*, NVIDIA, Canada and University of Toronto, Canada
OR PEREL, NVIDIA, Israel
RICCARDO DE LUTIO, NVIDIA, USA
JANICK MARTINEZ ESTURO, NVIDIA, Germany
GAVRIEL STATE, NVIDIA, Canada
SANJA FIDLER, NVIDIA, Canada, University of Toronto, Canada, and Vector Institute, Canada
NICHOLAS SHARP†, NVIDIA, USA
ZAN GOJCIC†, NVIDIA, Switzerland

3D Gaussian Ray Tracing: Fast Tracing of Particle Scenes

NICOLAS MOENNE-LOCCOZ*, NVIDIA, Canada
ASHKAN MIRZAEI*, NVIDIA, Canada and University of Toronto, Canada
OR PEREL, NVIDIA, Israel
RICCARDO DE LUTIO, NVIDIA, USA
JANICK MARTINEZ ESTURO, NVIDIA, Germany
GAVRIEL STATE, NVIDIA, Canada
SANJA FIDLER, NVIDIA, Canada, University of Toronto, Canada, and Vector Institute, Canada
NICHOLAS SHARP†, NVIDIA, USA
ZAN GOJCIC†, NVIDIA, Switzerland



3D Reconstruction

Novel-View Synthesis

NeRF

$\boxed{\text{3D Gaussian Splatting}}$

A Modular Framework for NeRF Development

Matthew Tancik*, Ethan Weber*, Evonne Ng*, Ruilong Li, Brent Yi, Justin Kerr, Terrance Wang, Alexander Kristoffersen, Jake Austin, Kamyar Salahi, Abhik Ahuja, David McAllister, Angjoo Kanazawa

+130 additional Github collaborators

BAIR
BERKELEY ARTIFICIAL INTELLIGENCE RESEARCH

main | 1 branch | 0 tags

Go to file | Add file | Code

andybak Merge pull request #125 from xmeng525/main ... e5c8cfc 2 weeks ago 376 commits

.github	Update pull_request_template.md	2 years ago
citations	Update cvpr2034 paper NeAT	2 weeks ago
.DS_Store	Add R2L (ECCV 2022) and MobileR2L (Arxiv) for faster inference	3 months ago
.gitignore	Ignore IDE files	4 months ago
LICENSE	Initial commit	3 years ago
NeRF-and-Beyond.bib	Add R2L (ECCV 2022) and MobileR2L (Arxiv) for faster inference	3 months ago
README.md	Update cvpr2034 paper NeAT	2 weeks ago
how-to-PR.md	Create how-to-PR.md	2 years ago

README.md

Awesome Neural Radiance Fields awesome

A curated list of awesome neural radiance fields papers, inspired by awesome-computer-vision.

How to submit a pull request?

Want to help maintain the list?

Table of Contents

About

A curated list of awesome neural radiance fields papers

nerf

Readme

MIT license

4.5k stars

194 watching

462 forks

Report repository

Releases

No releases published

Packages

No packages published

Contributors 57

Nerfstudio Design Goals

Easy to:

Use

Develop

Learn

An End-to-End Framework

Mobile

Polycam
Record3D
KIRI Engine

Desktop

COLMAP
Metashape
RealityCapture

Input

→

Samplers

Uniform
Occupancy
PDF
Proposal

Fields

Fused MLP
Voxel Grid

Encoders

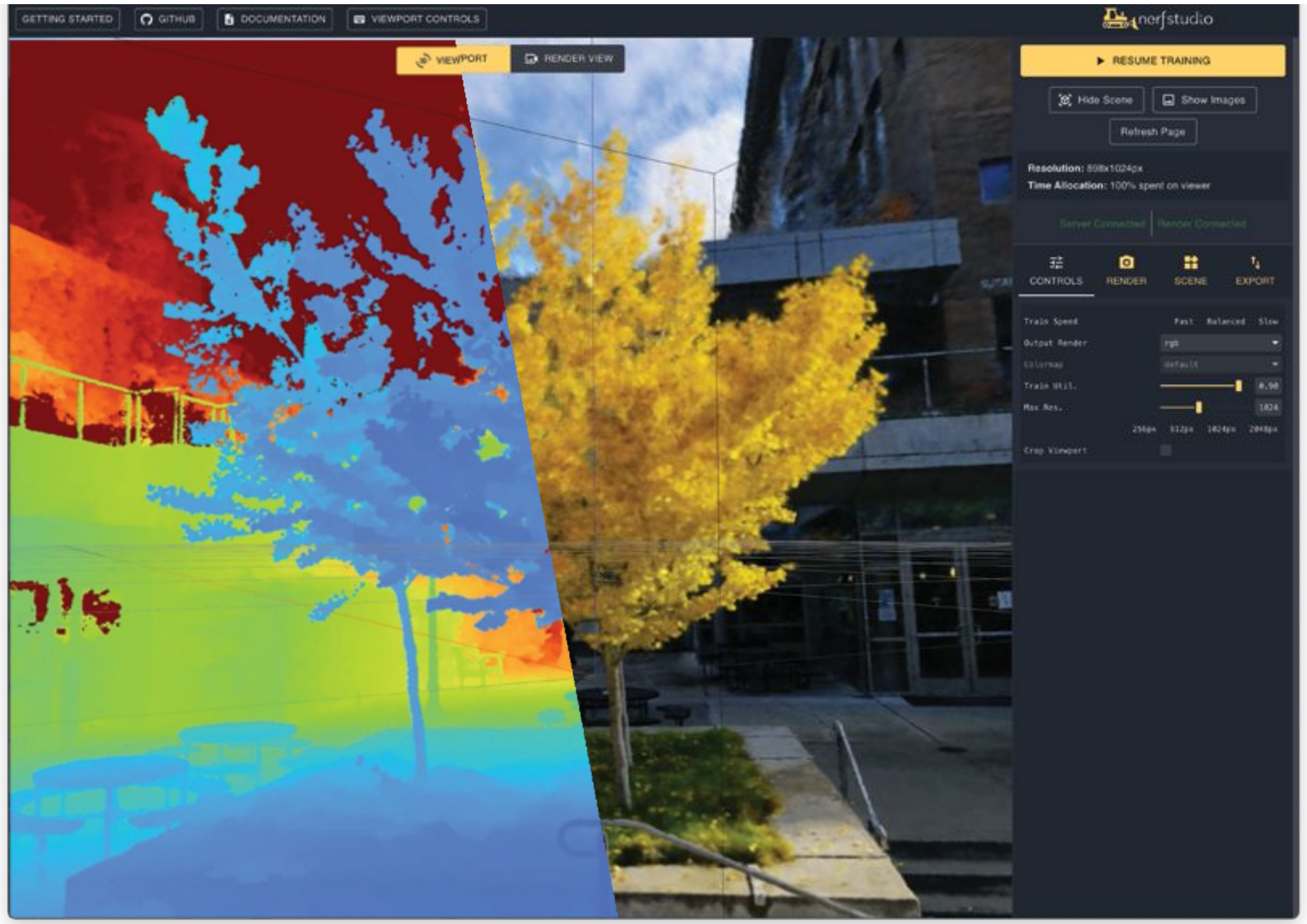
Positional Encoding
Fourier Features
Hash Encoding
Spherical Harmonics
Matrix Decomposition

Renderers

RGB
RGB-SH
Depth
Accumulation
Normals

Modular Components

→



Real-time web viewer

→

Video

Point Cloud

Mesh

Export

Data Pipelines

Onboarding Pipelines

- COLMAP
- Polycam
- Record3D
- MetaShape
- RealityCapture
- Kiri Engine

Easy to Develop

Sampling

Fields &
Encoders

Volumetric
Rendering

Pythonic and Modular

Easy to Develop

Sampling

- Uniform
- Occupancy
- PDF
- Proposal
- Spacing Fn

Fields & Encoders

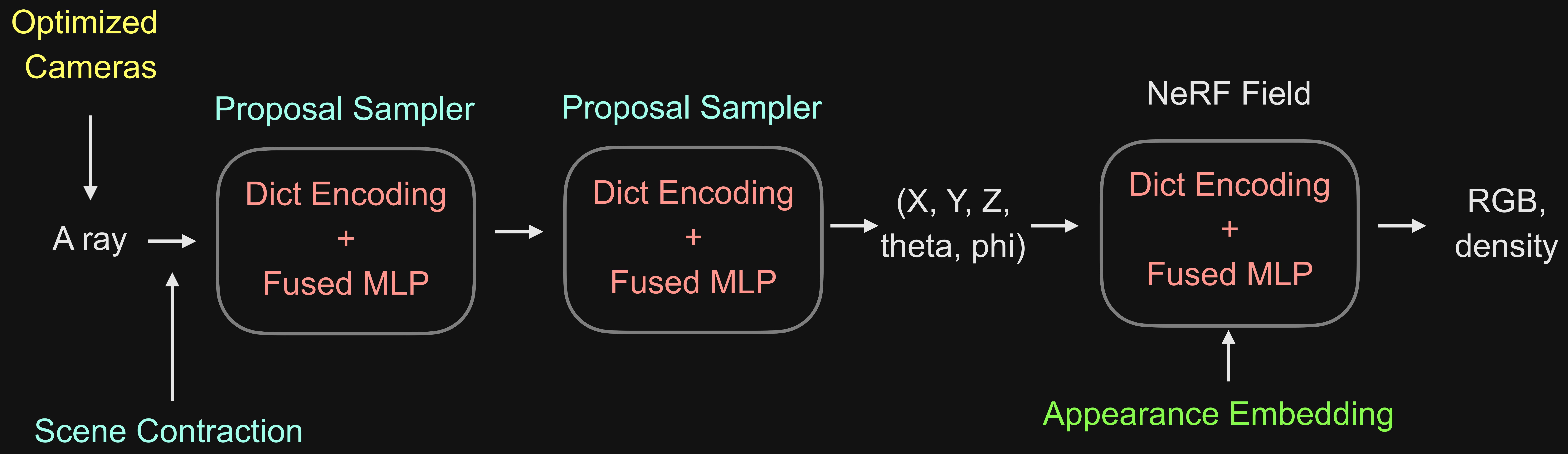
- Positional Encoding
- Fourier Features
- Hash Encoding
- Spherical Harmonics
- Matrix Decomposition
- Fused MLP
- Voxel Grid

Volumetric Rendering

- RGB
- RGB-SH
- Depth
- Accumulation
- Normals

Pythonic and Modular

Striking the balance between performance & easy development



Current Model

mip-NeRF 360, NeRF-W,
NeRF--/BaRF, InstantNGP

Nerfacto Variants

(Bigger models work better)

Model	Description	Memory	Speed
nerfacto	Default Model	~6GB	Fast
nerfacto-big	Larger higher quality	~12GB	Slow
Nerfacto-huge	Even larger and higher quality	~24Gb	Slower


Nerfacto


Nerfacto-huge


Nerfacto


Nerfacto-huge