ADVANCES IN COMPUTER VISION

MODULE 1: Geometry

DIFFERENTIABLE RENDERING: DATA STRUCTURES & SIGNAL REPRESENTATIONS



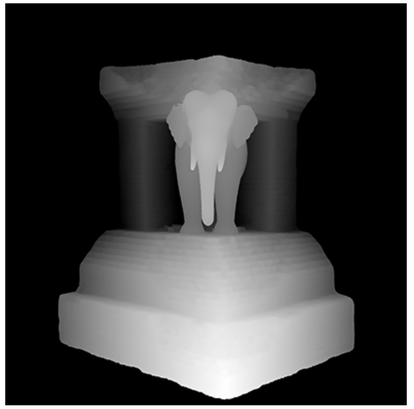
Prof. Vincent Sitzmann

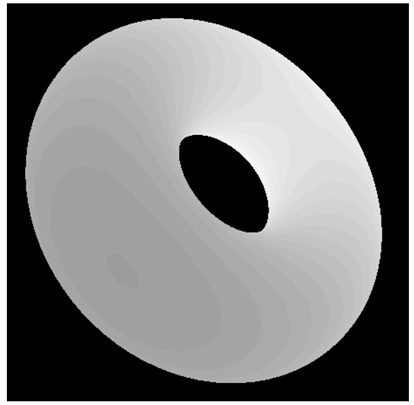
Some Slides adapted from…

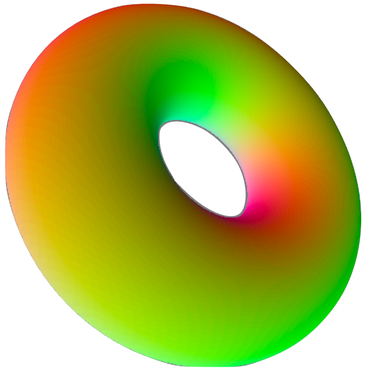
CMU 16-889: Learning for 3D Vision

Prof. Shubham Tulsiani

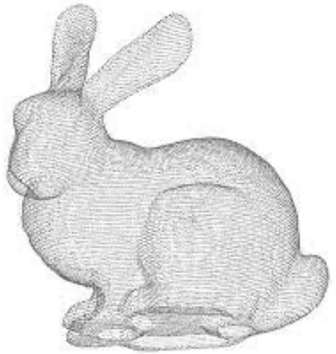
2.5D Representations

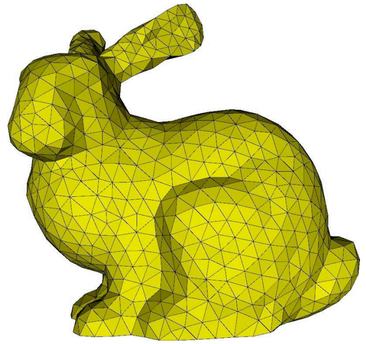




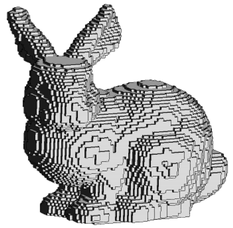


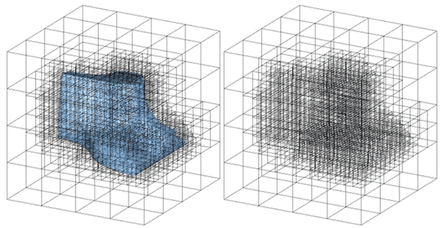
Surface Representations

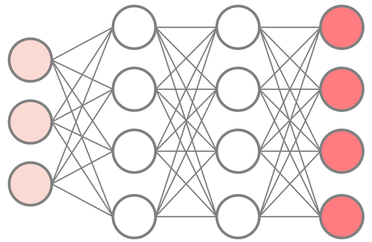


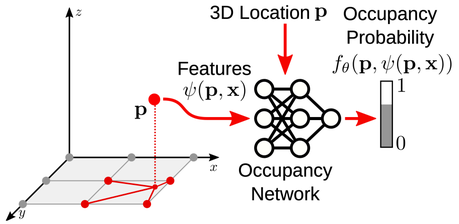


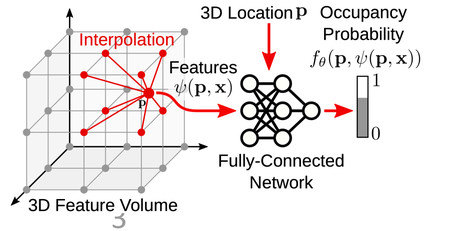
Field Parameterizations



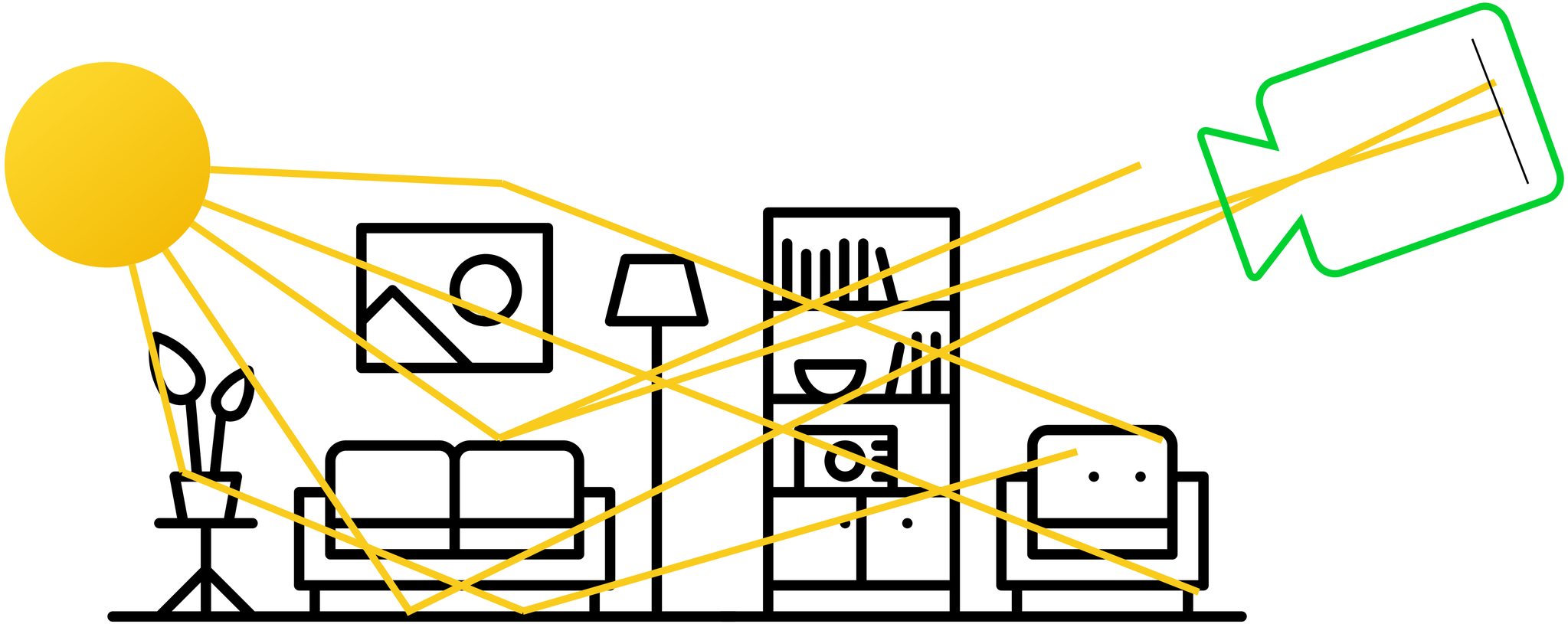




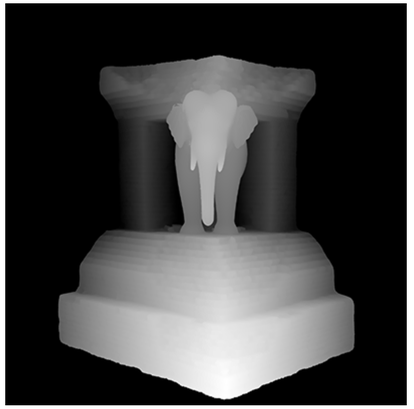


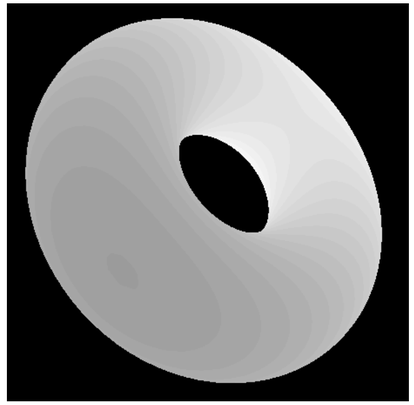


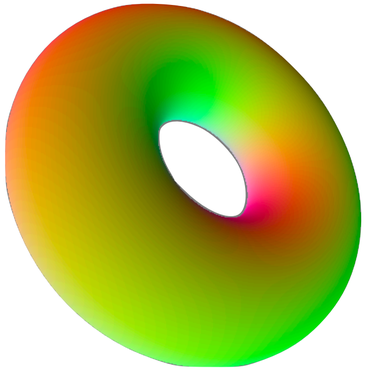
What does the camera see?



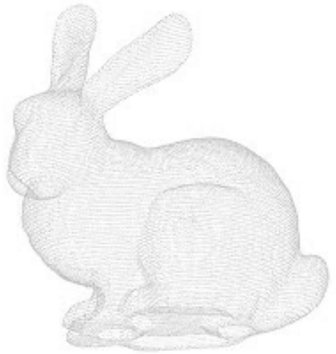
2.5D Representations

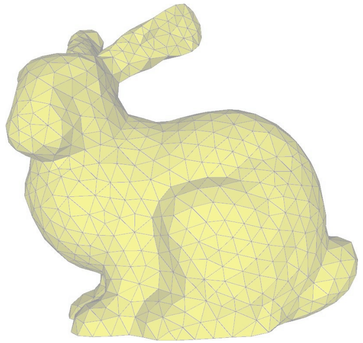




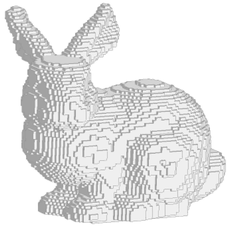


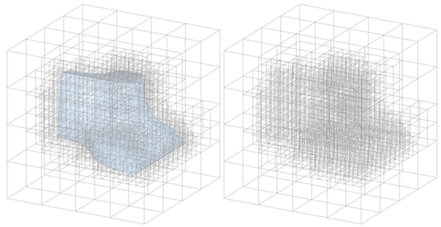
Surface Representations

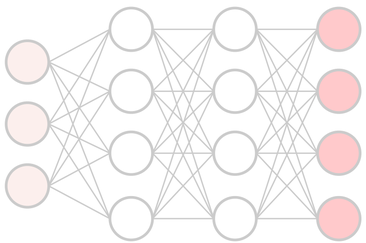


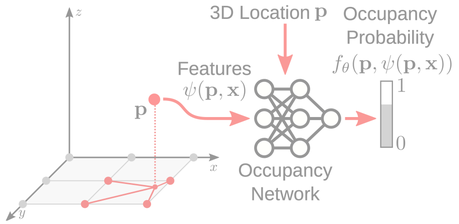


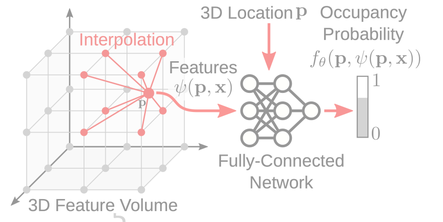
Field Parameterizations











Depth Maps







Slide credit: CMU 16-889: Learning for 3D Vision, Prof. Shubham Tulsiani

Depth Maps







Common modality, captured with Stereo, LIDAR, ….

Slide credit: CMU 16-889: Learning for 3D Vision, Prof. Shubham Tulsiani

Depth Maps



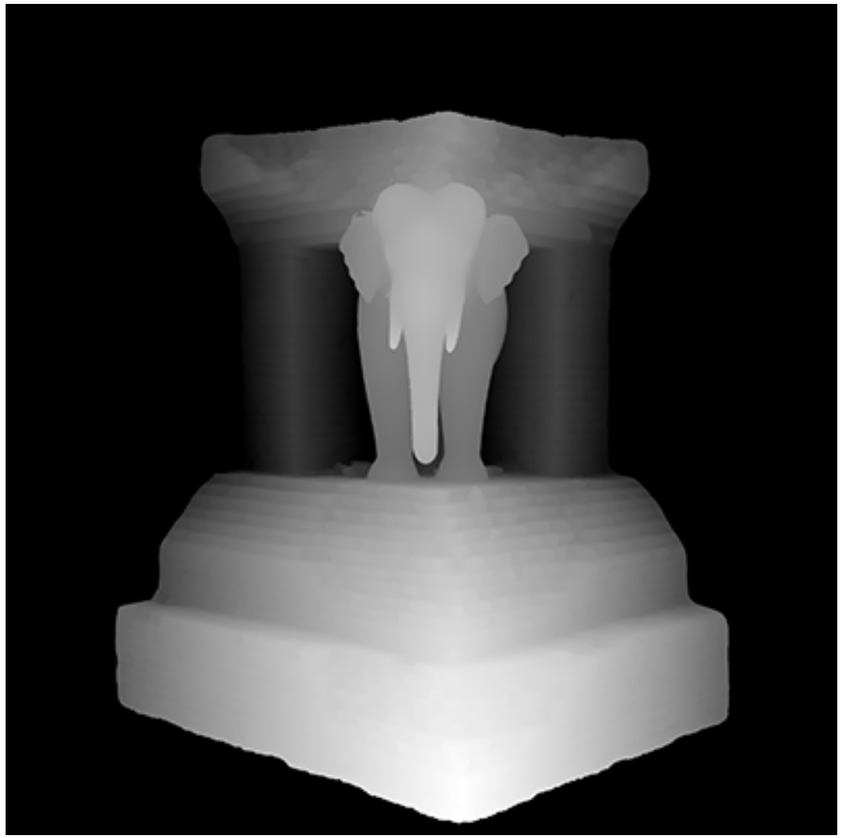
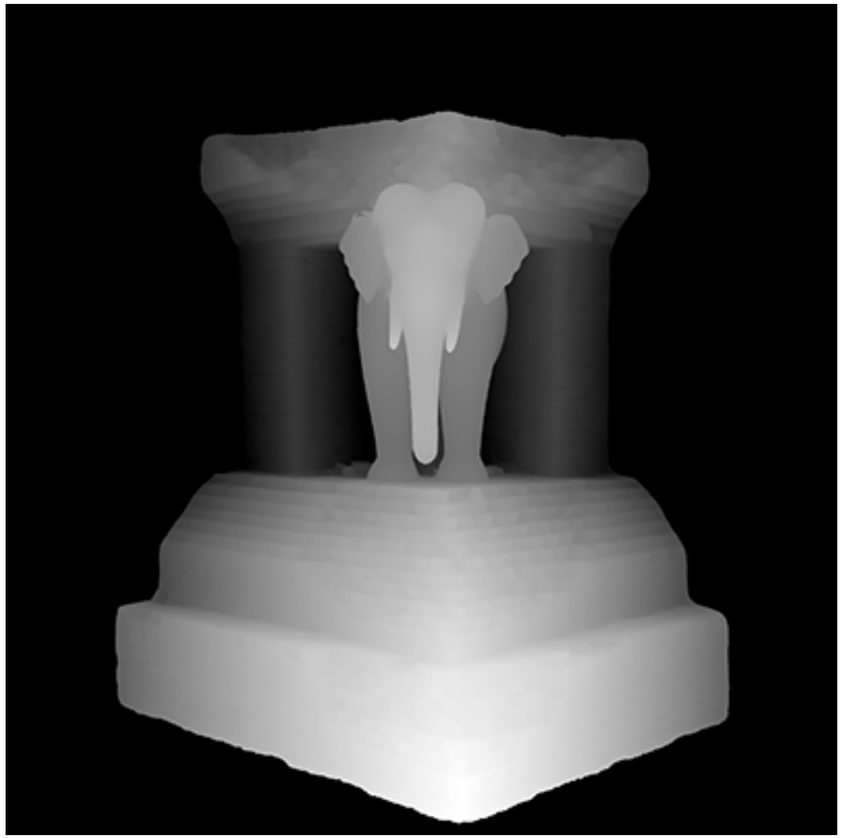


Image credits: Paul Bourke, Slide credit: CMU 16-889: Learning for 3D Vision, Prof. Shubham Tulsiani

Depth Maps



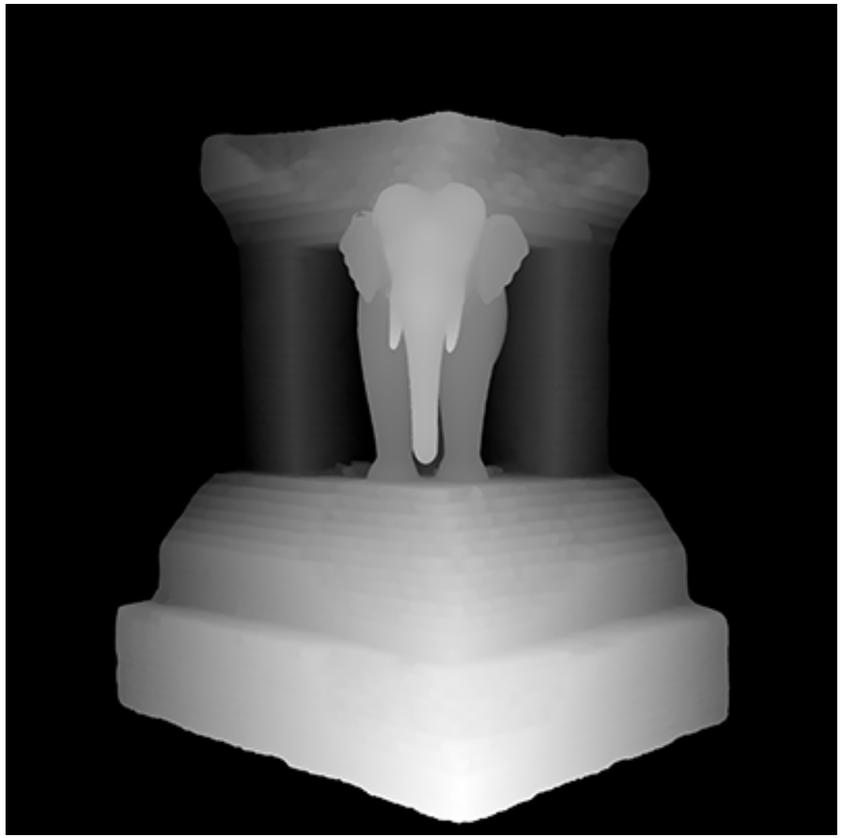


An **image** that represents how far each pixel **p** is

Image credits: Paul Bourke, Slide credit: CMU 16-889: Learning for 3D Vision, Prof. Shubham Tulsiani

Depth Maps





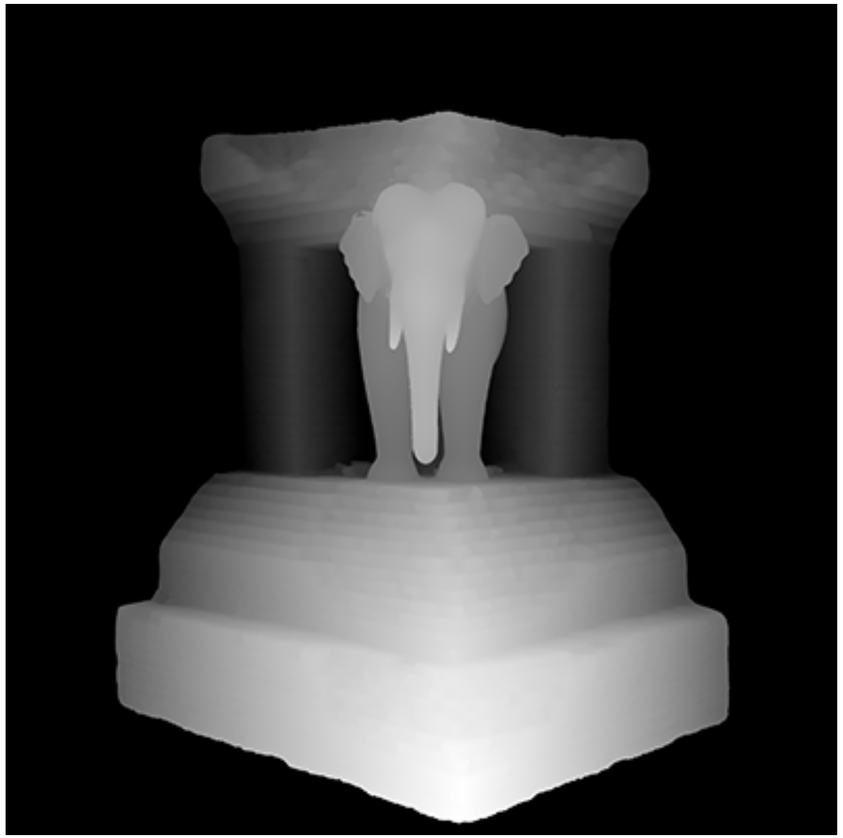
An **image** that represents how far each pixel **p** is

$$D[\mathbf{p}] \in \mathbb{R}^+$$

Image credits: Paul Bourke, Slide credit: CMU 16-889: Learning for 3D Vision, Prof. Shubham Tulsiani

Depth Maps

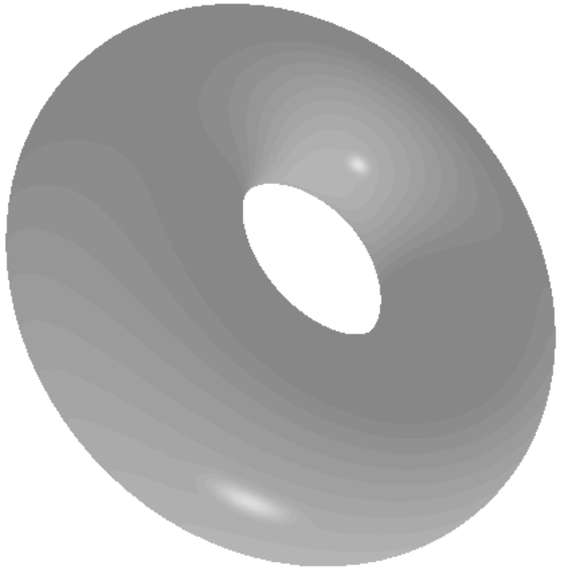


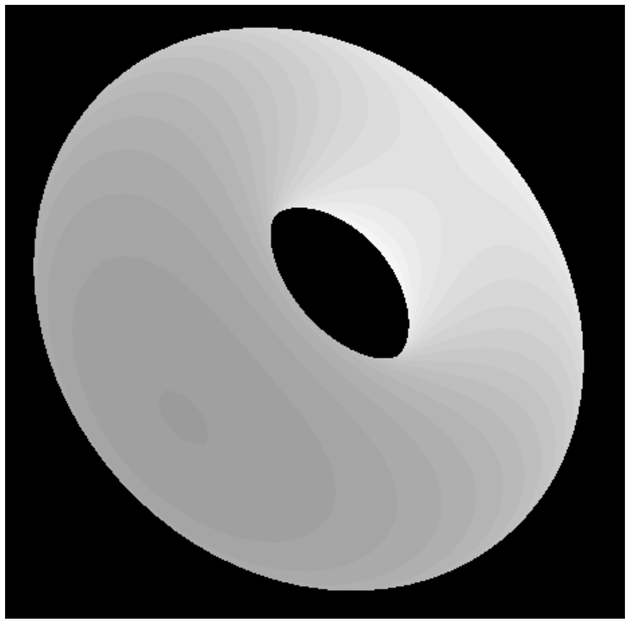


The ‘standard’ image processing tools can be used

Image credits: Paul Bourke, Slide credit: CMU 16-889: Learning for 3D Vision, Prof. Shubham Tulsiani

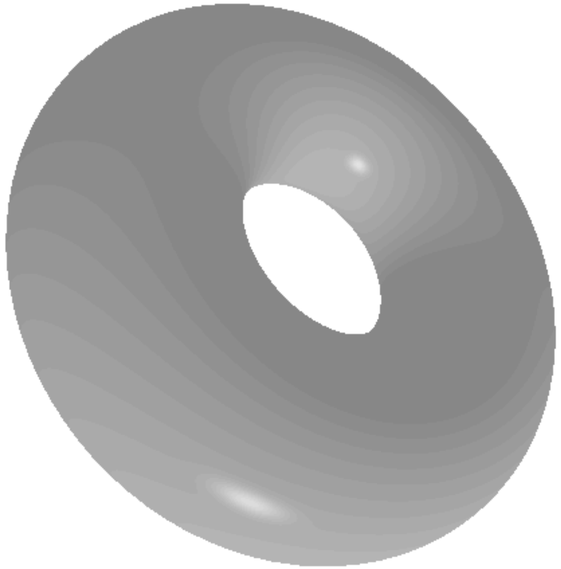
Depth Maps and Normal Maps

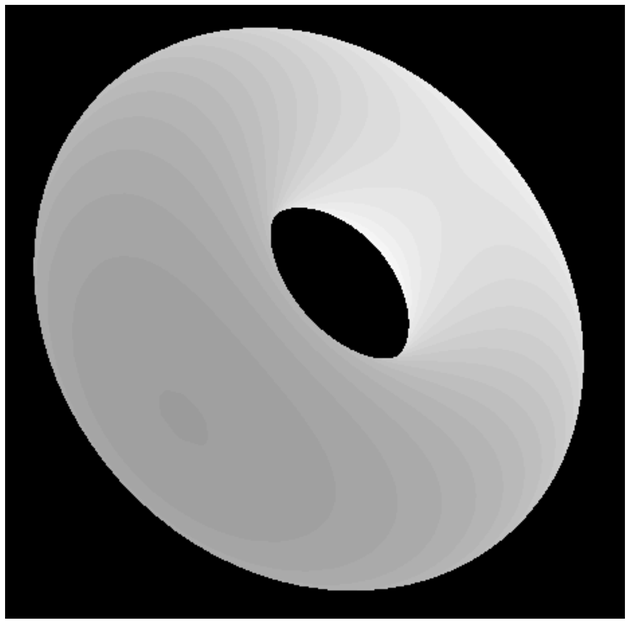


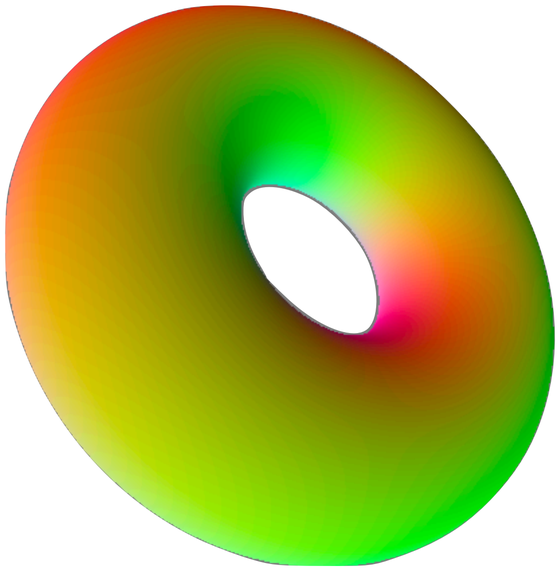


Slide credit: CMU 16-889: Learning for 3D Vision, Prof. Shubham Tulsiani

Depth Maps and Normal Maps



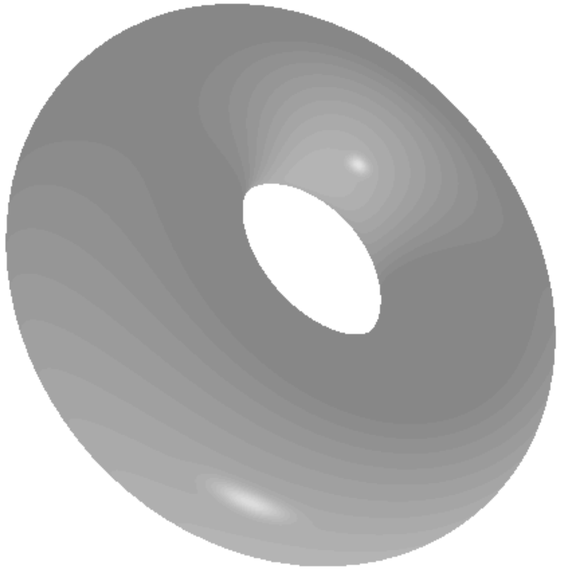


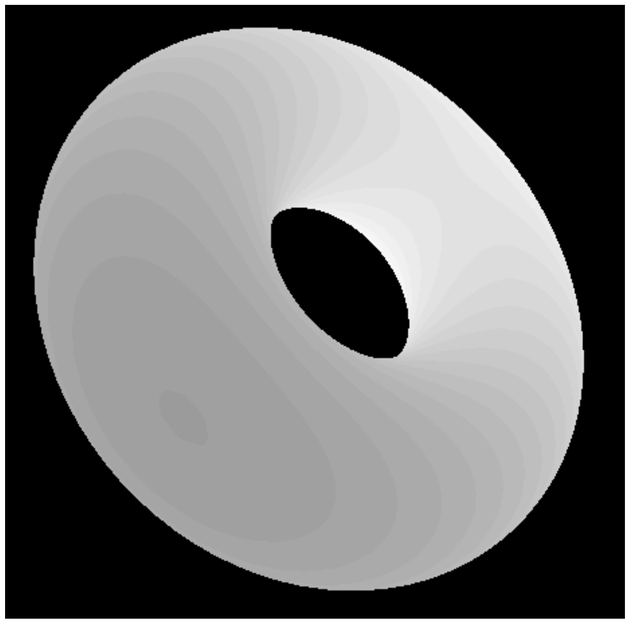


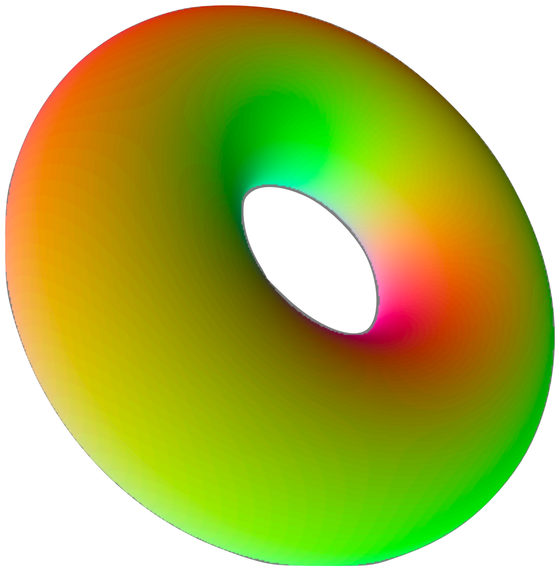
Can extend to capture other surface properties e.g. surface normals

Slide credit: CMU 16-889: Learning for 3D Vision, Prof. Shubham Tulsiani

Depth Maps and Normal Maps





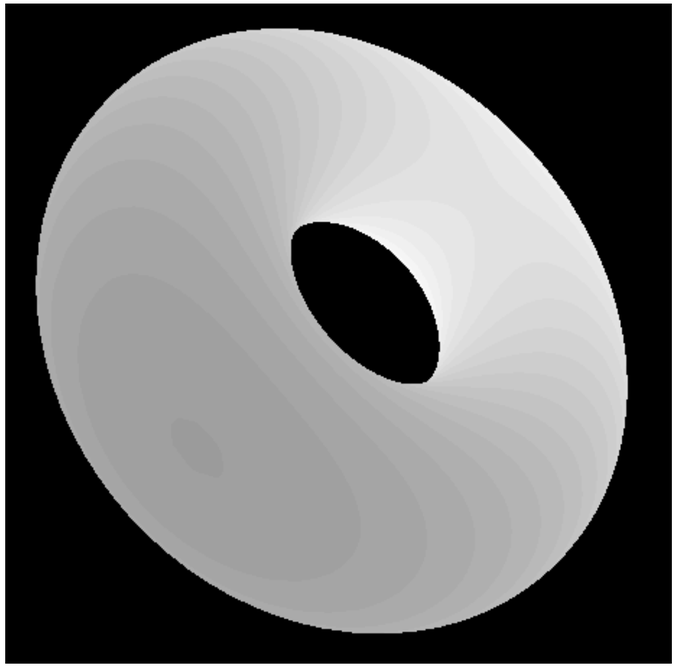


$$N[\mathbf{p}] \in \mathbb{S}^2$$

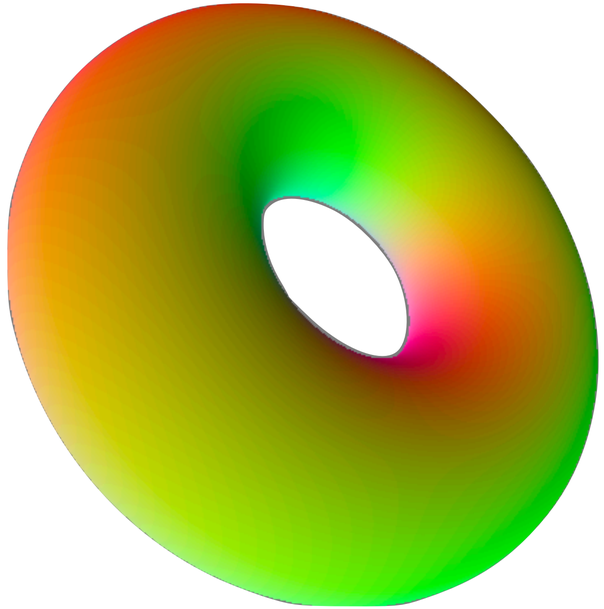
Can extend to capture other surface properties e.g. surface normals

Slide credit: CMU 16-889: Learning for 3D Vision, Prof. Shubham Tulsiani

Normal map is spatial derivative of depth map



∇_x



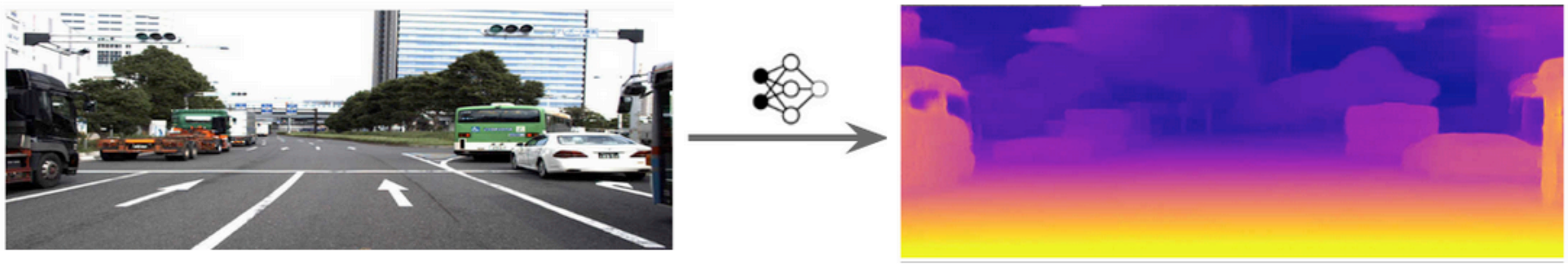
$$D : \mathbb{R}^2 \rightarrow \mathbb{R}^+$$

$$D(\mathbf{x}_{pix}) = d$$

$$N : \mathbb{R}^2 \rightarrow \mathbb{R}^3$$

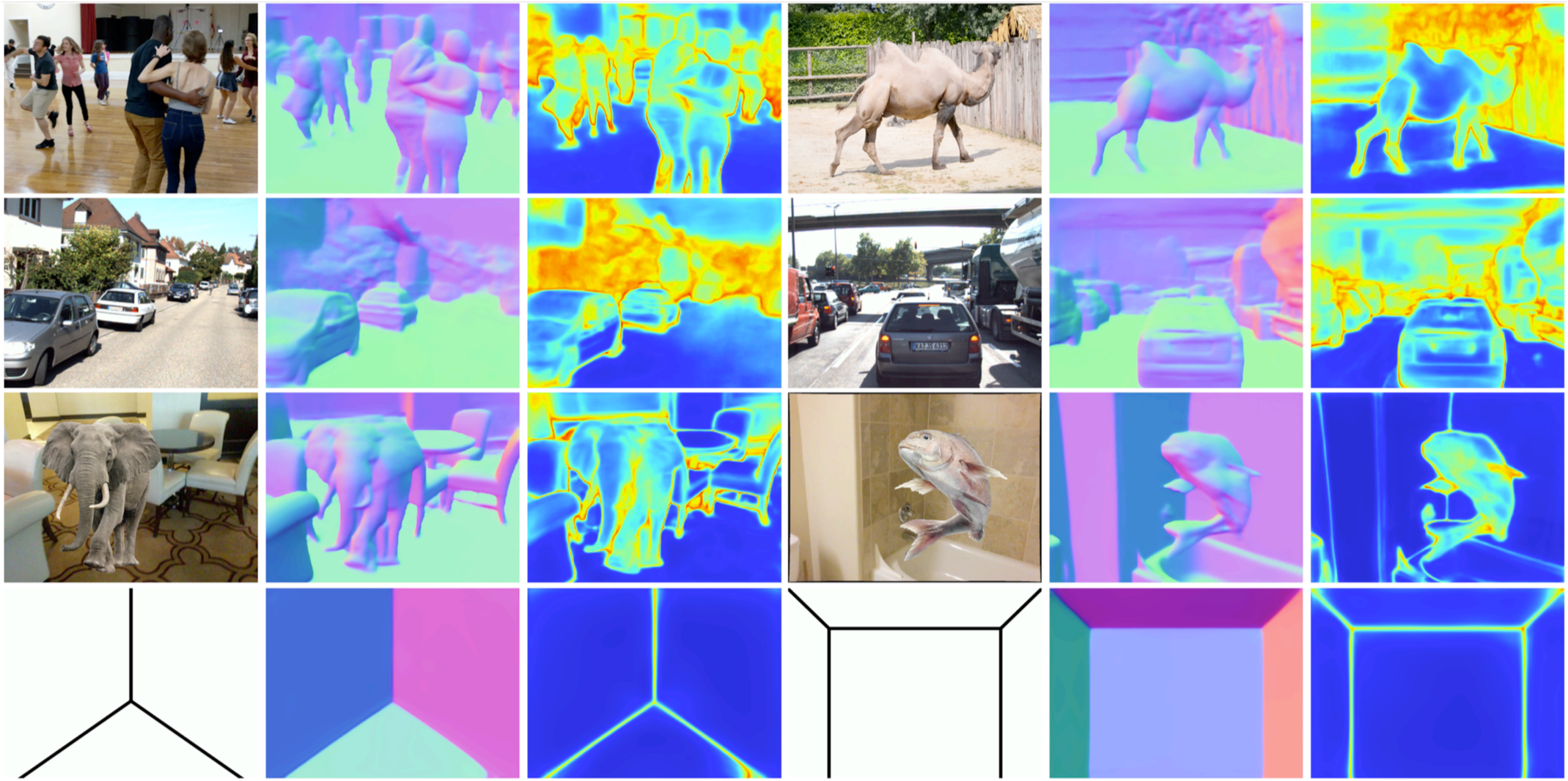
$$N(\mathbf{x}_{pix}) = \nabla_{\mathbf{x}} D(\mathbf{x}_{pix}) = \mathbf{n}$$

Supervised **Monocular** Depth Estimation



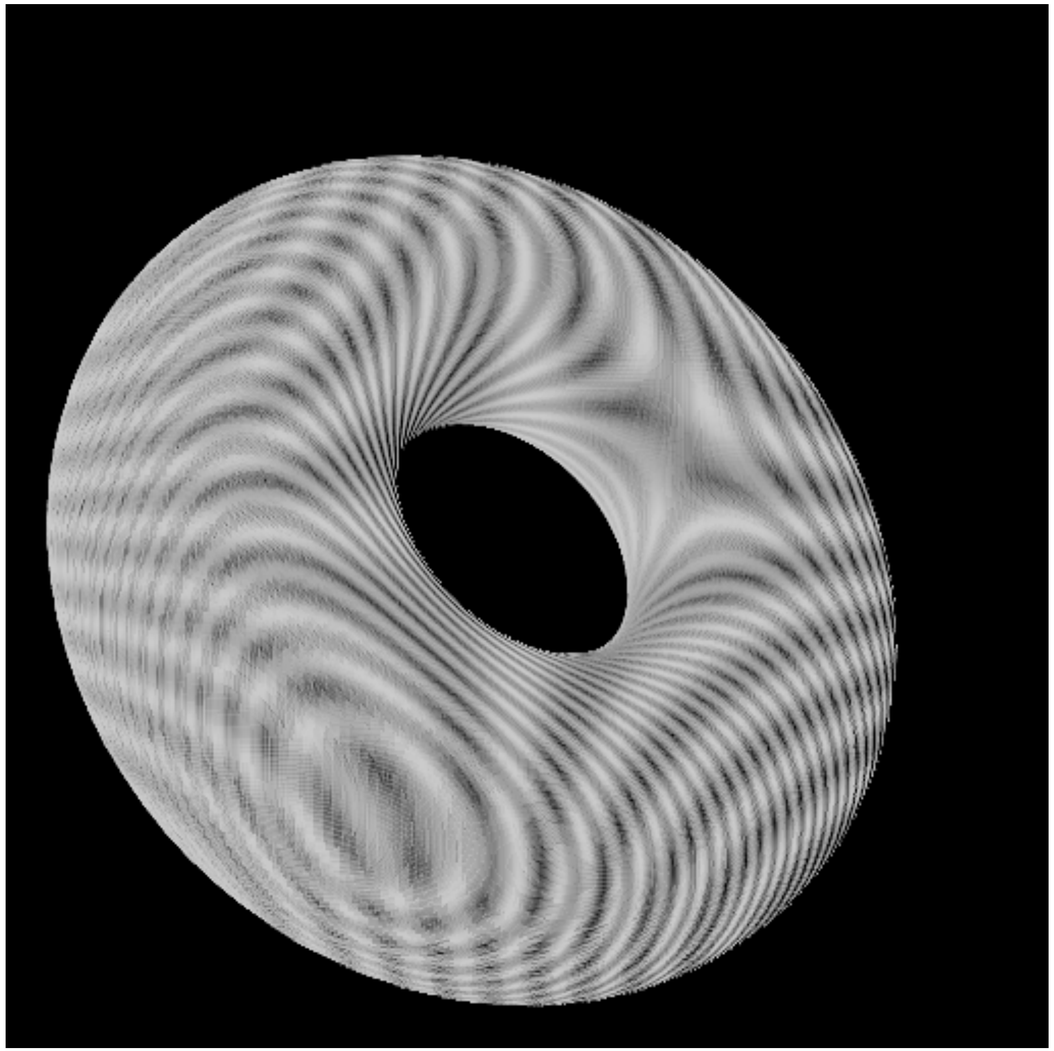
https://medium.com/toyotaresearch/self-supervised-learning-in-depth-part-1-of-2-74825baaaa04

It’s easier to predict normals from pixels than depth



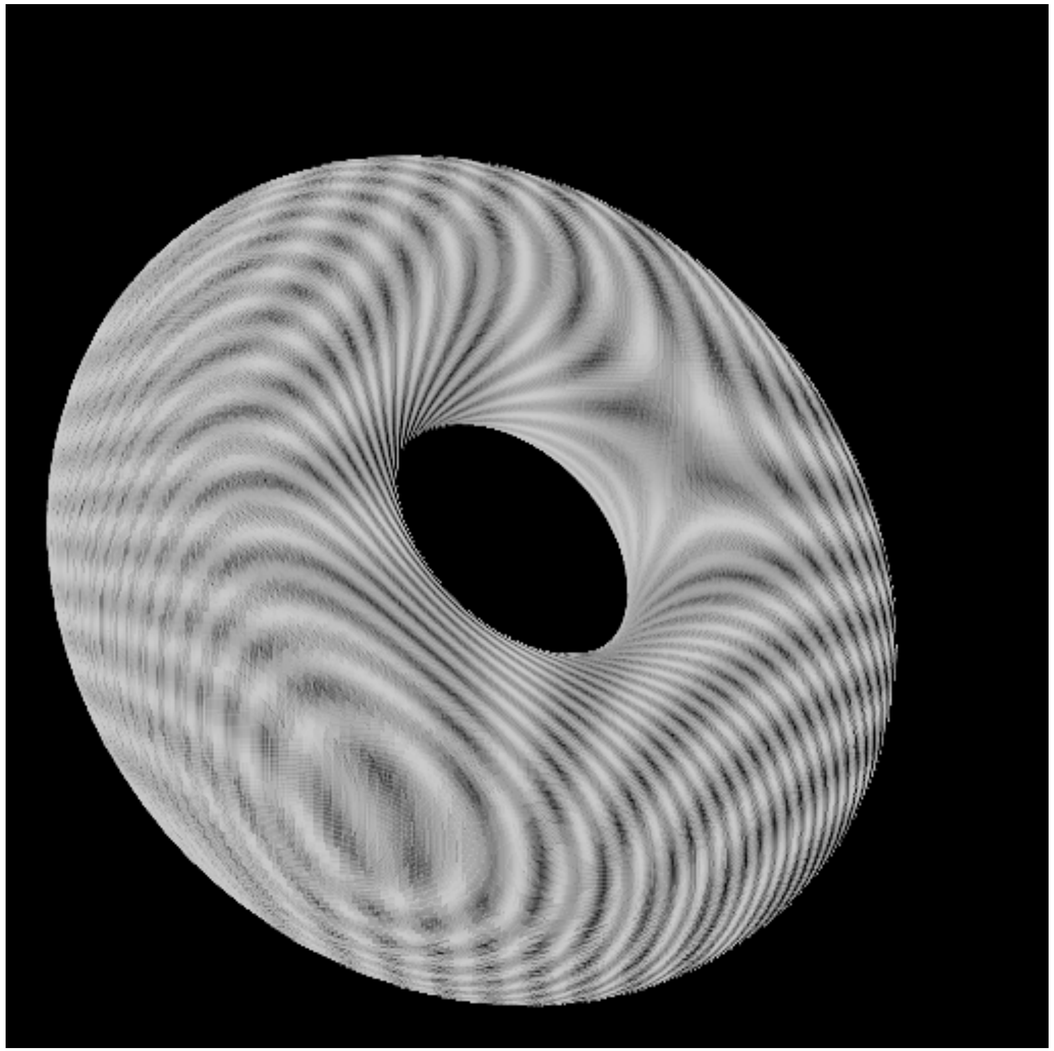
Estimating and Exploiting the Aleatoric Uncertainty in Surface Normal Estimation, Bae et al., ICCV 2021

Depth Maps — Representing the Visible



Slide credit: CMU 16-889: Learning for 3D Vision, Prof. Shubham Tulsiani

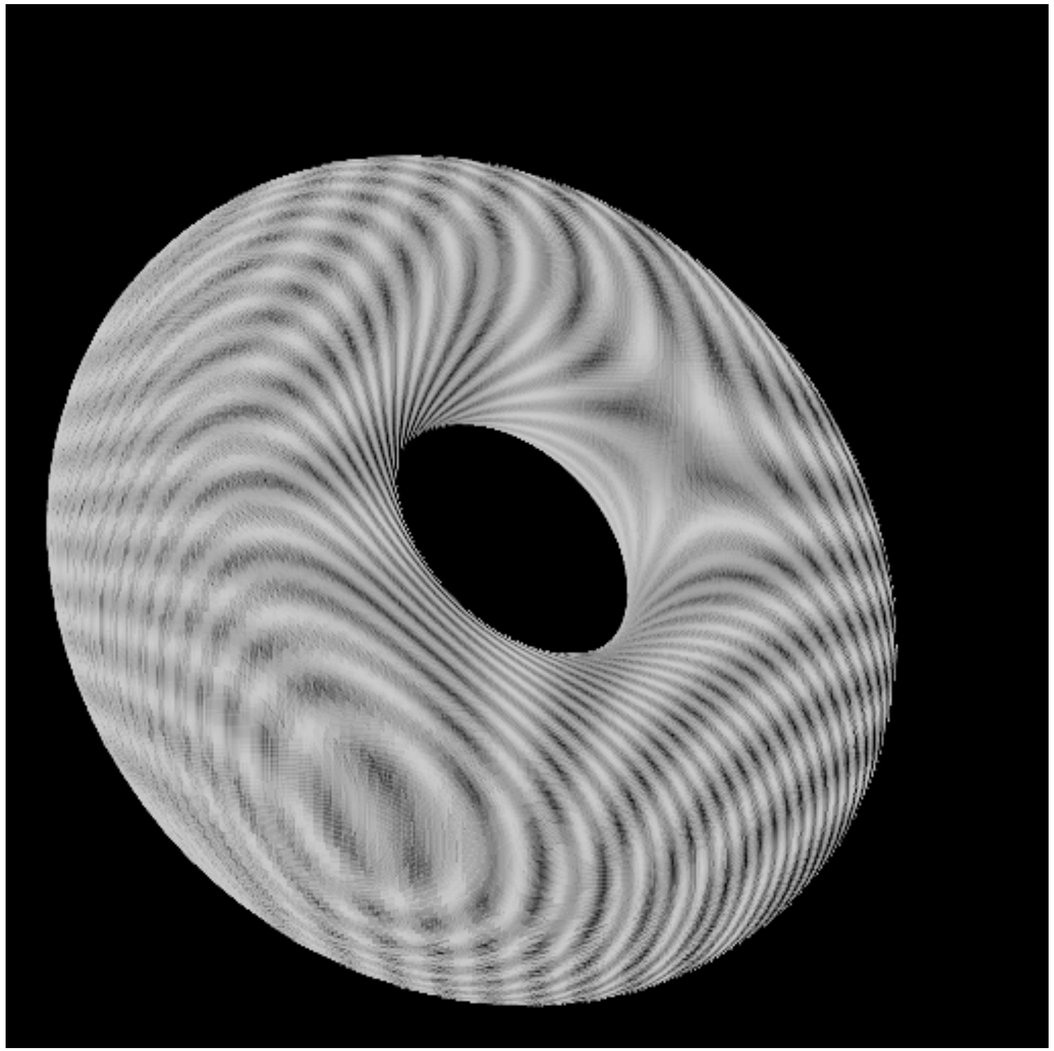
Depth Maps — Representing the Visible



Does not capture the ‘full’ 3D structure

Slide credit: CMU 16-889: Learning for 3D Vision, Prof. Shubham Tulsiani

Depth Maps — Representing the Visible

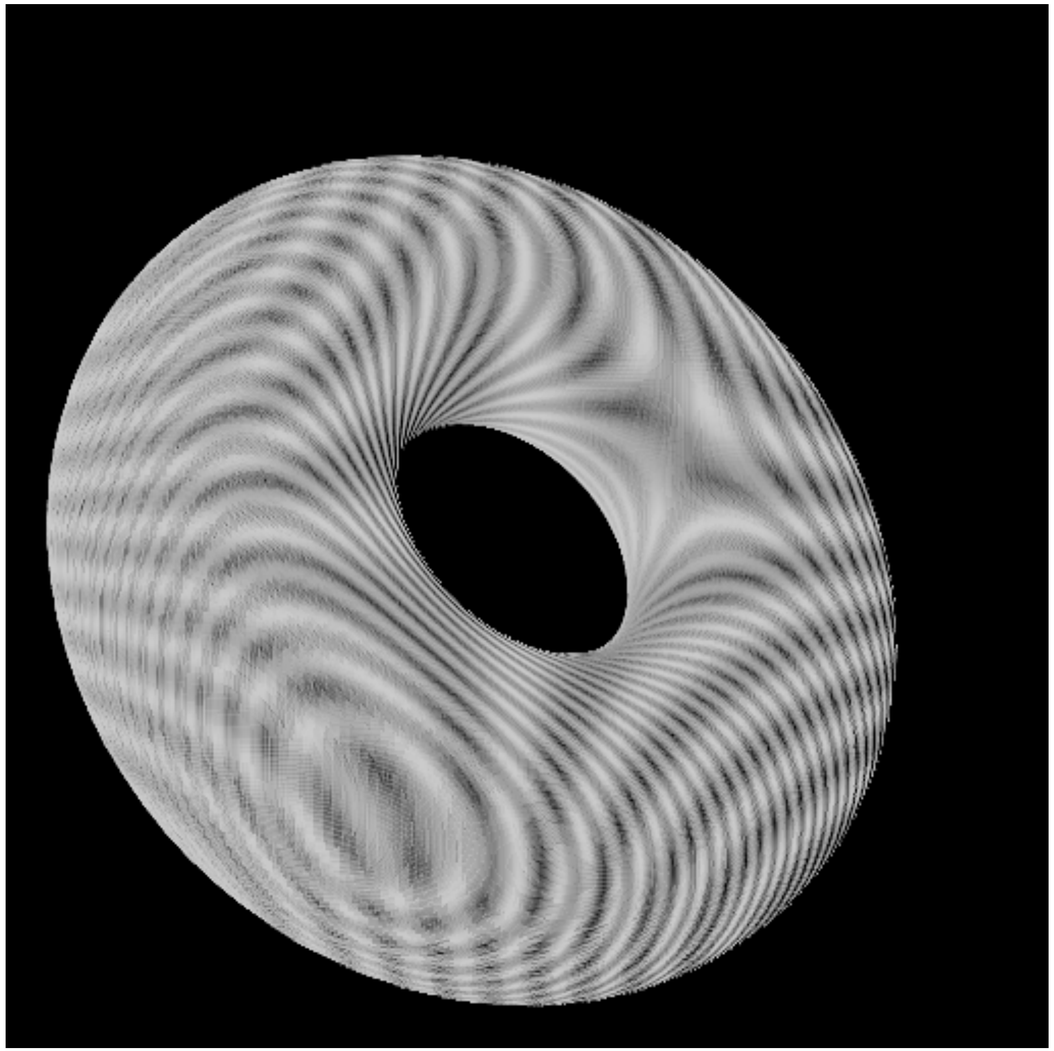


‘2.5D’ representation — properties associated to image pixels

Does not capture the ‘full’ 3D structure

Slide credit: CMU 16-889: Learning for 3D Vision, Prof. Shubham Tulsiani

Depth Maps — Representing the Visible



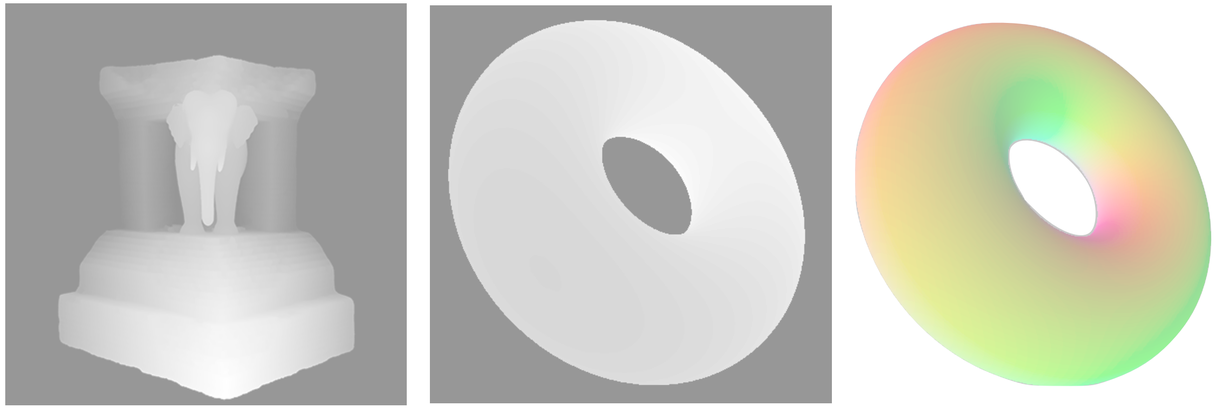
‘2.5D’ representation — properties associated to image pixels

Does not capture the ‘full’ 3D structure

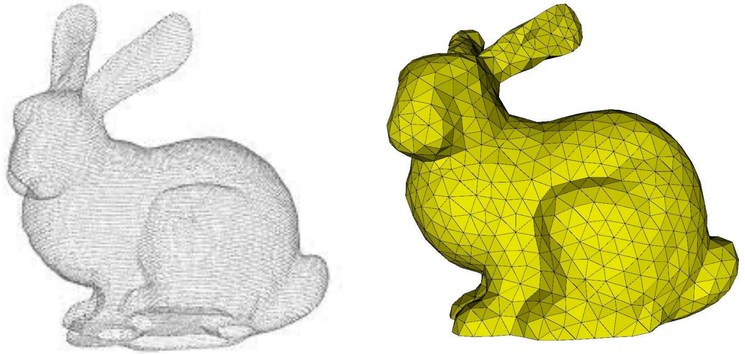
can we relax this ‘image-dependence’ constraint?

Slide credit: CMU 16-889: Learning for 3D Vision, Prof. Shubham Tulsiani

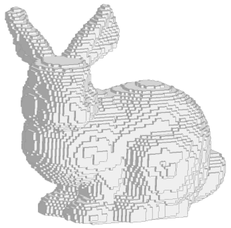
2.5D Representations

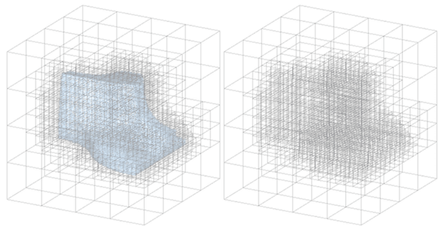


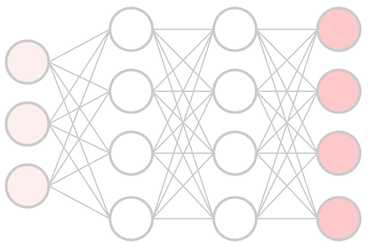
Surface Representations

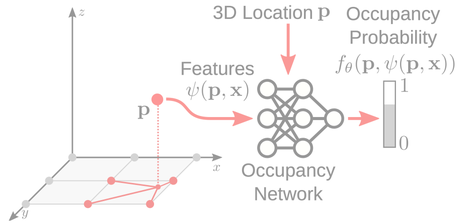


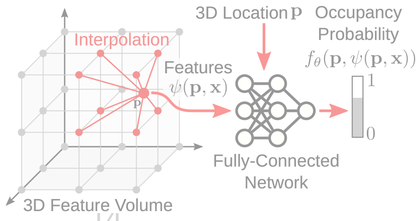
Field Parameterizations



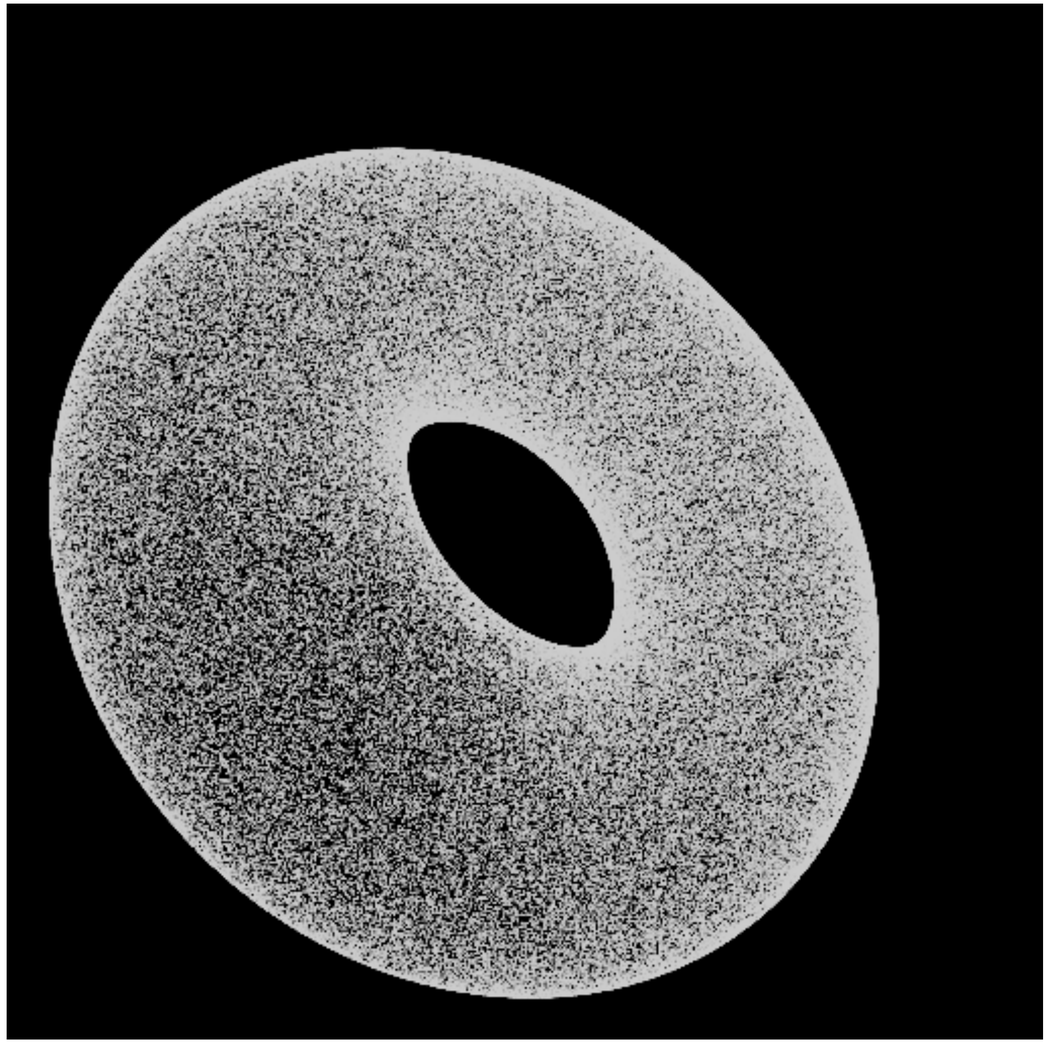






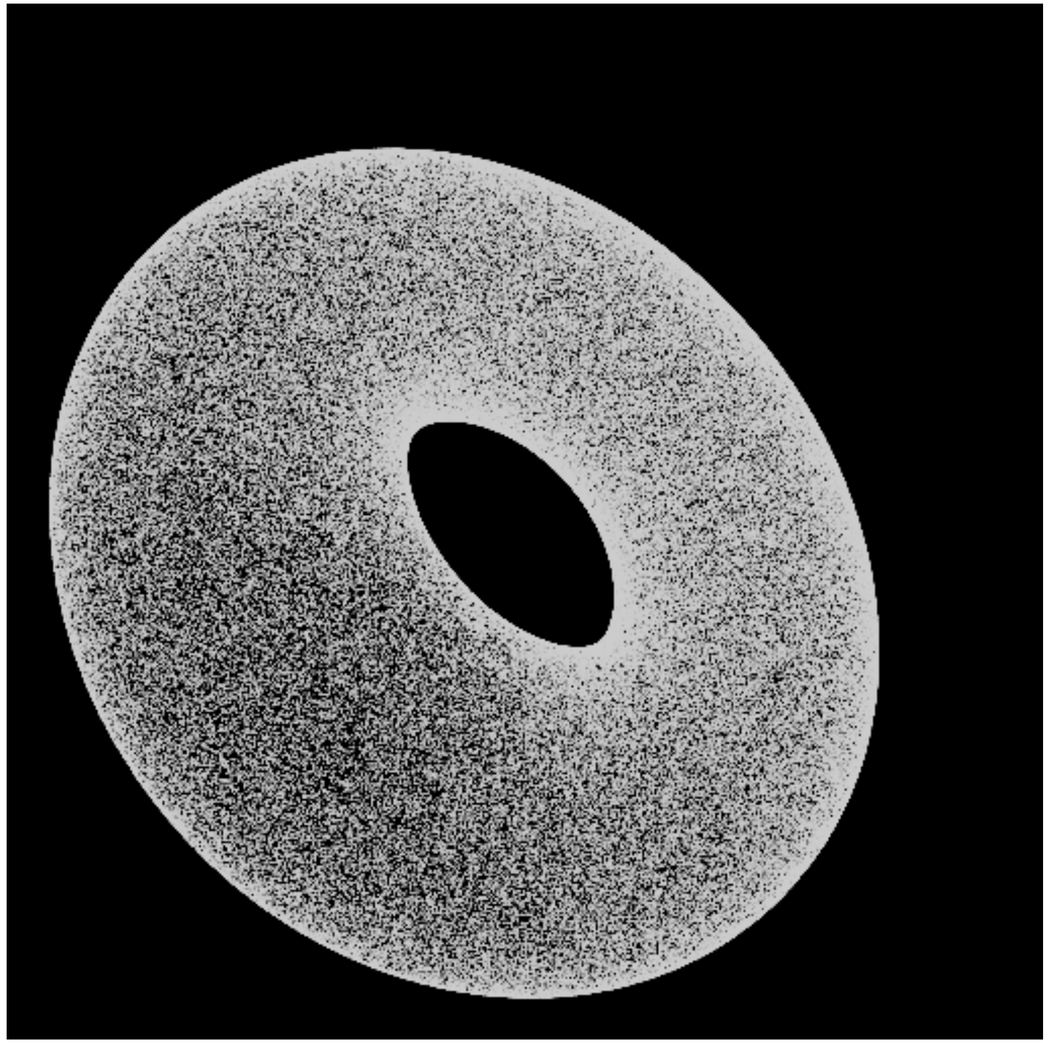


Point Clouds



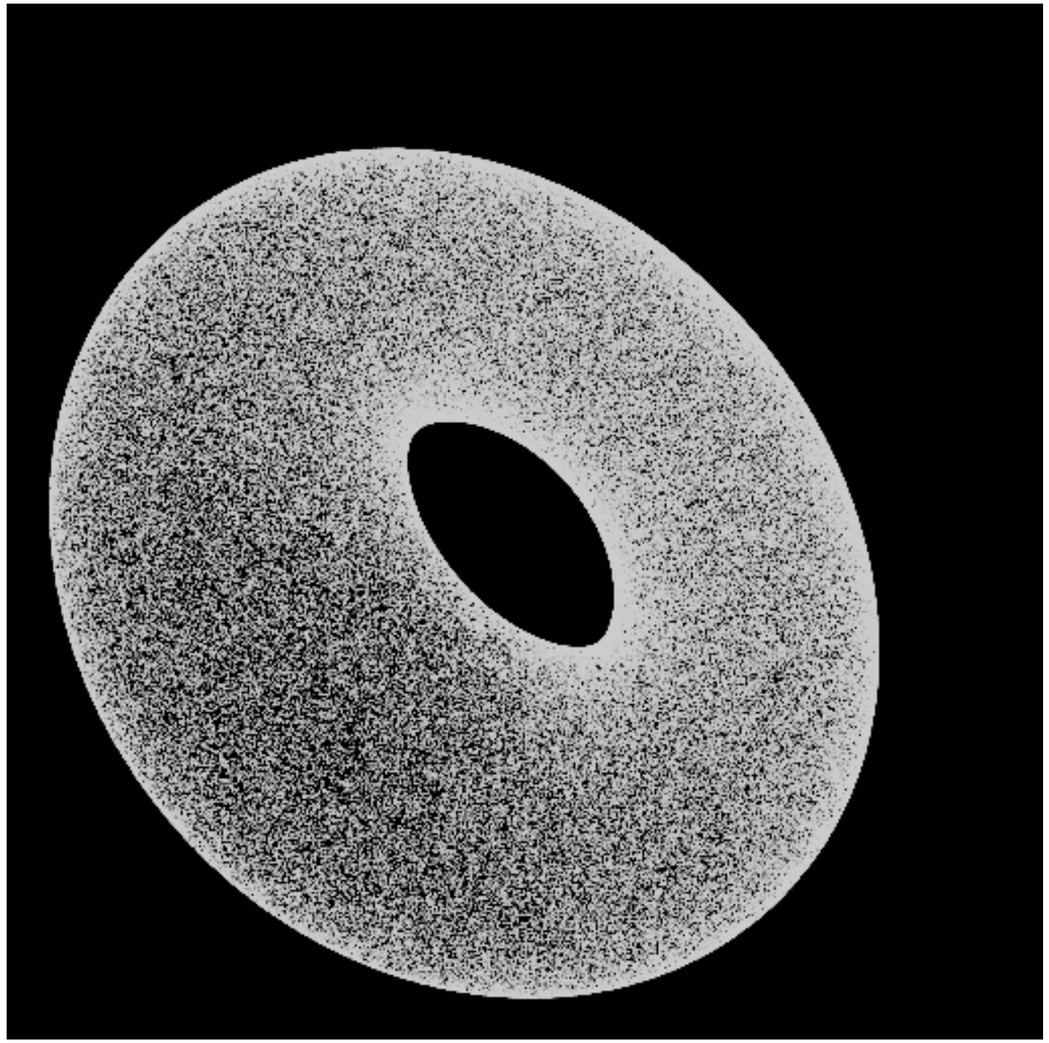
Slide credit: CMU 16-889: Learning for 3D Vision, Prof. Shubham Tulsiani

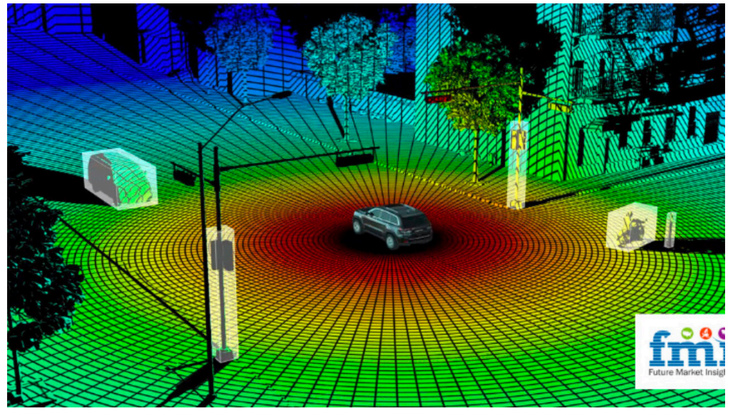
Point Clouds



Slide credit: CMU 16-889: Learning for 3D Vision, Prof. Shubham Tulsiani

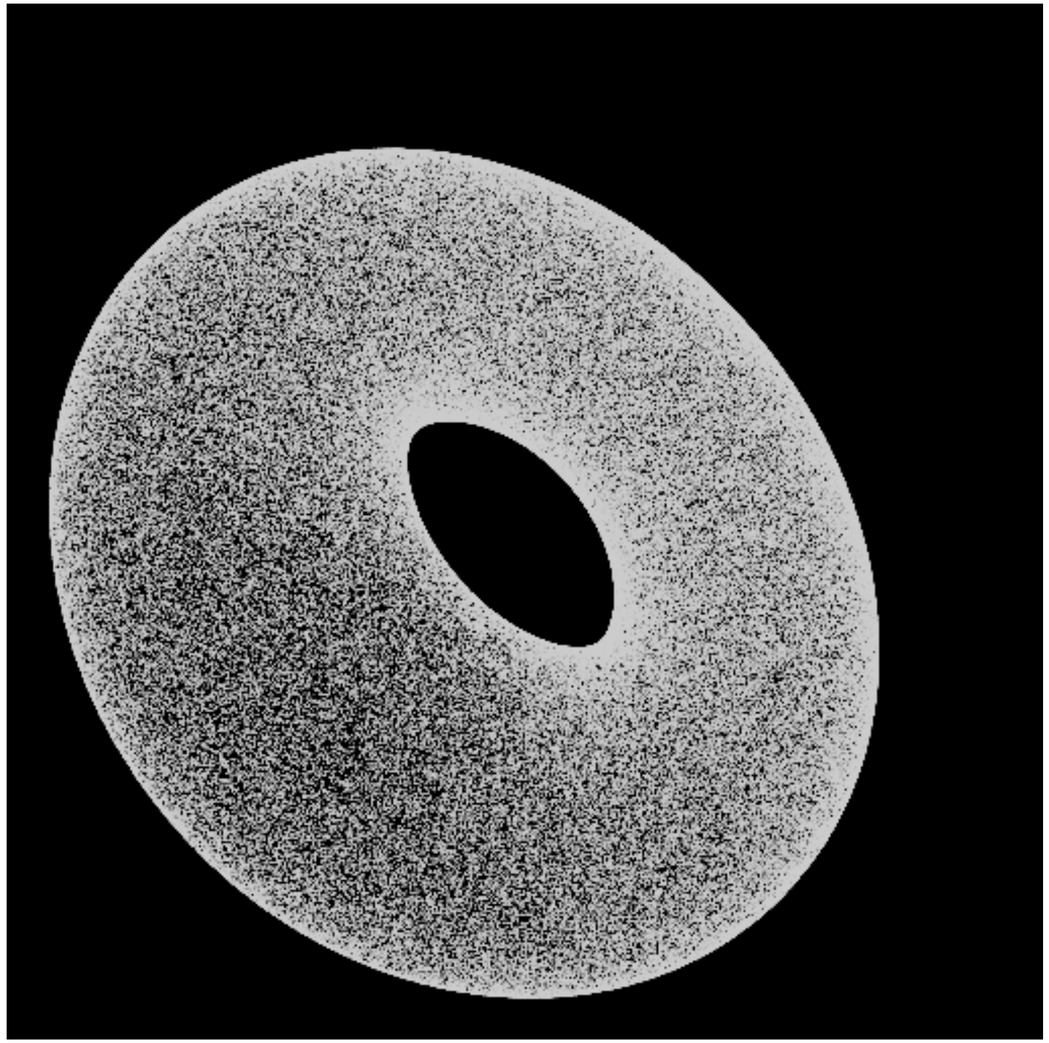
Point Clouds

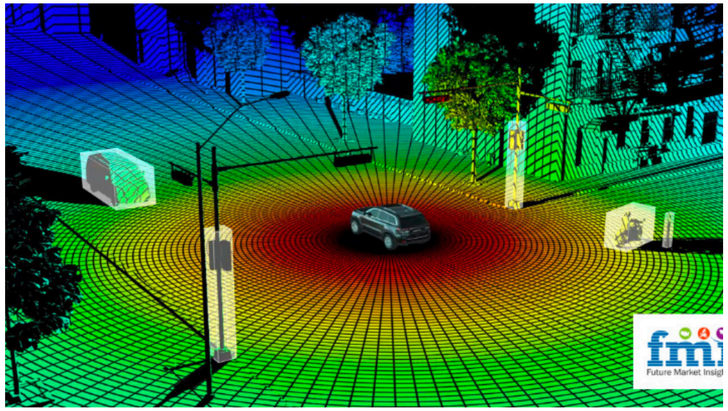




Slide credit: CMU 16-889: Learning for 3D Vision, Prof. Shubham Tulsiani

Point Clouds

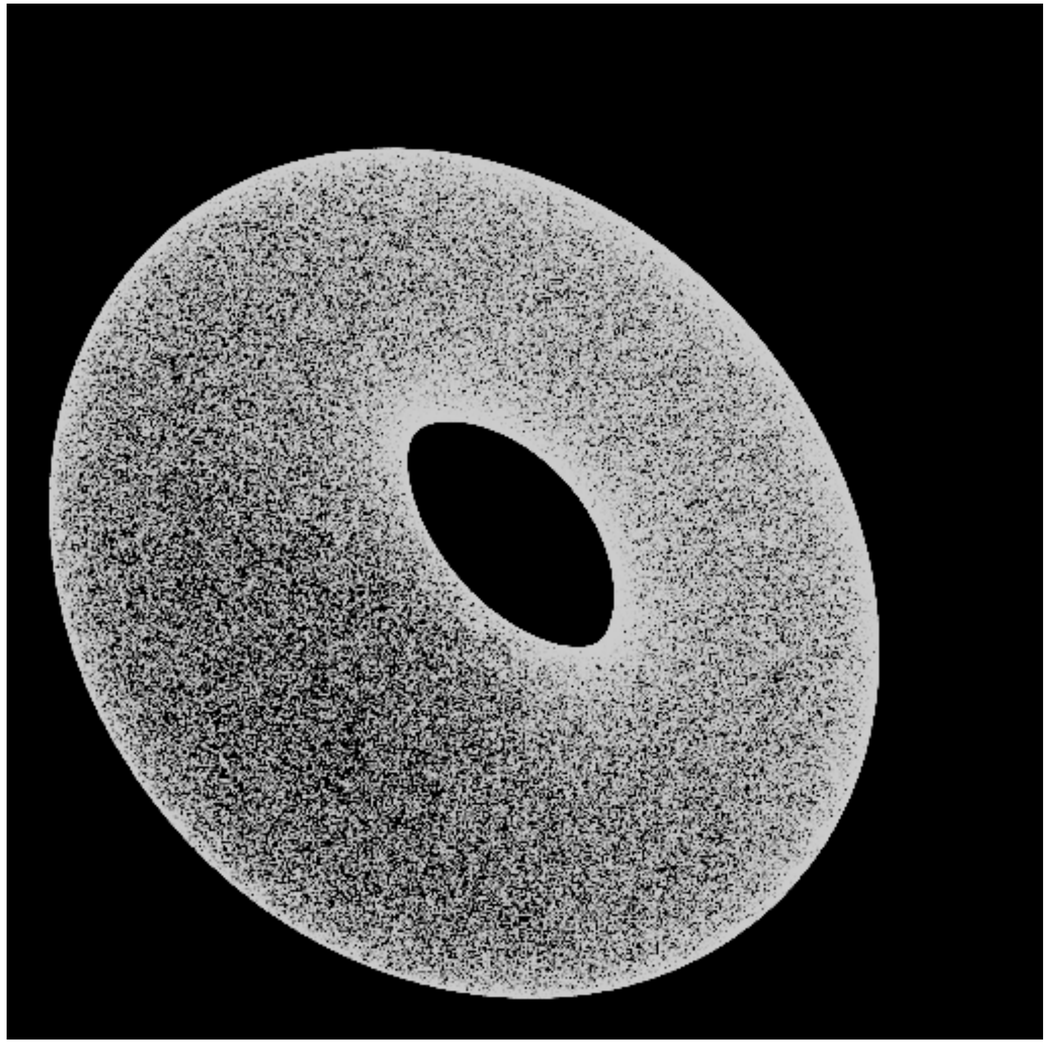


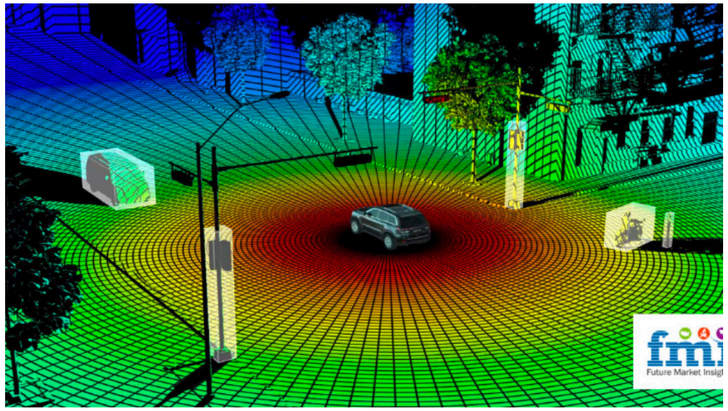




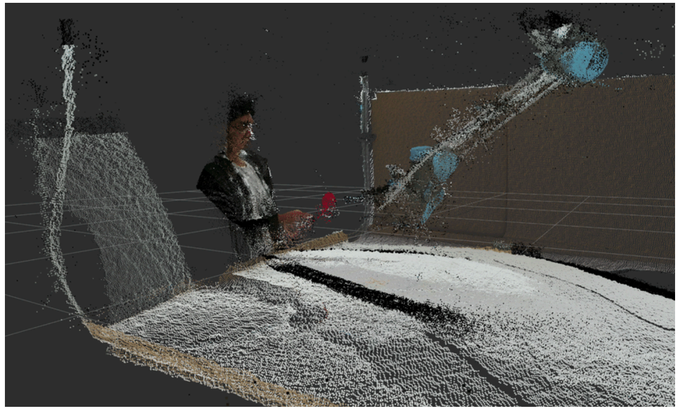
Slide credit: CMU 16-889: Learning for 3D Vision, Prof. Shubham Tulsiani

Point Clouds



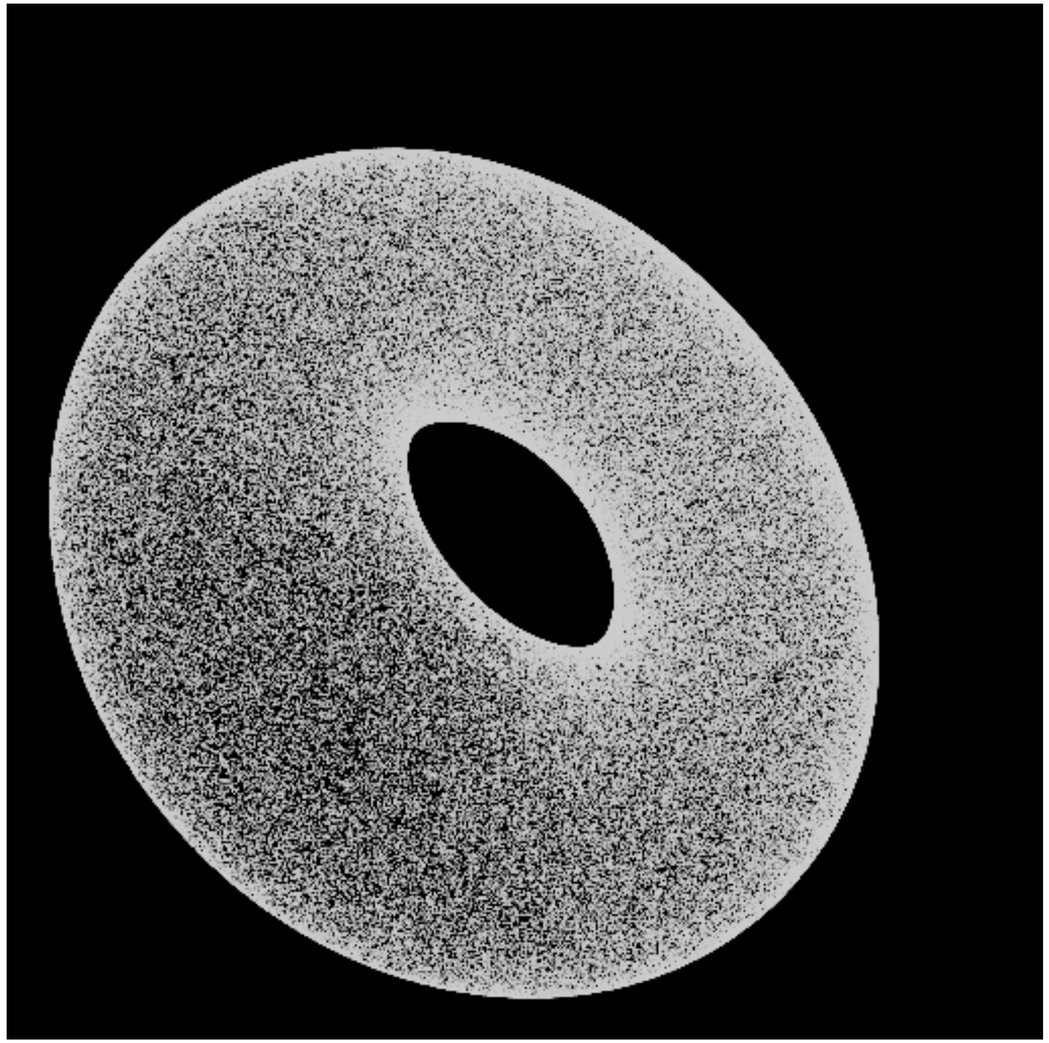


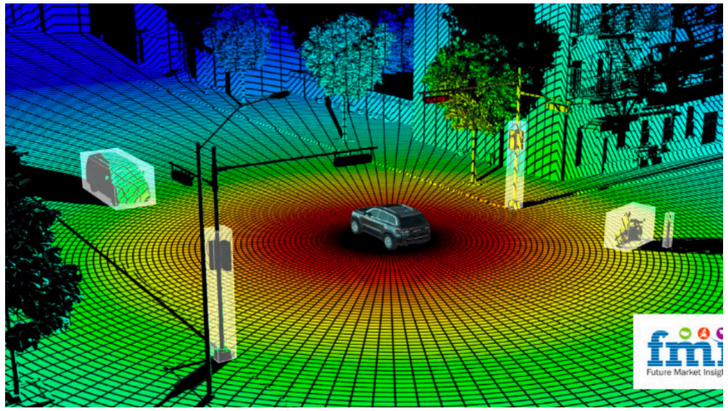




Slide credit: CMU 16-889: Learning for 3D Vision, Prof. Shubham Tulsiani

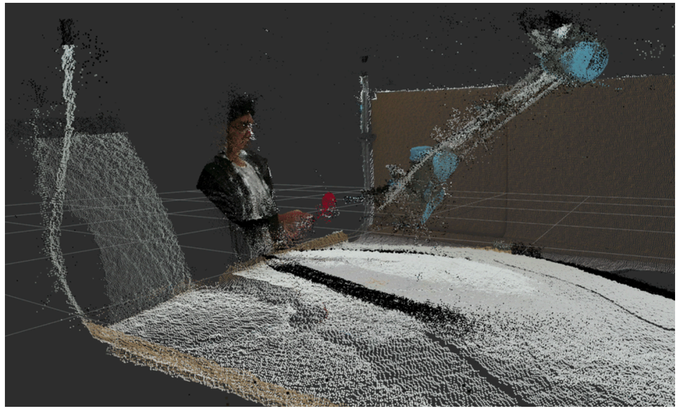
Point Clouds





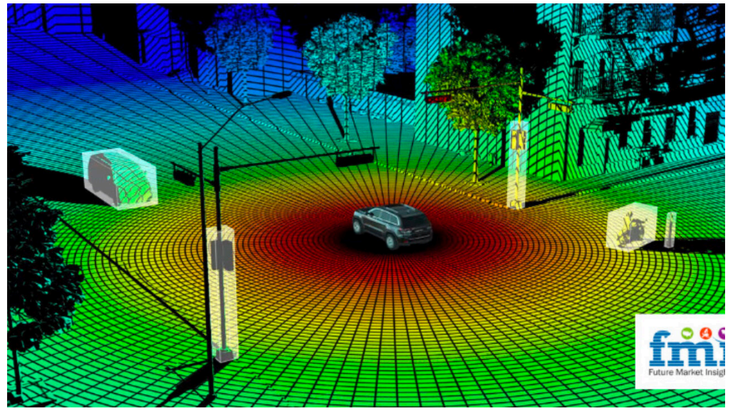






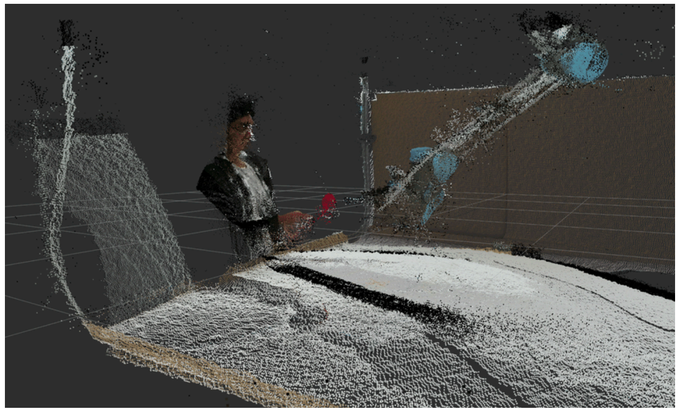
Slide credit: CMU 16-889: Learning for 3D Vision, Prof. Shubham Tulsiani

Point Clouds





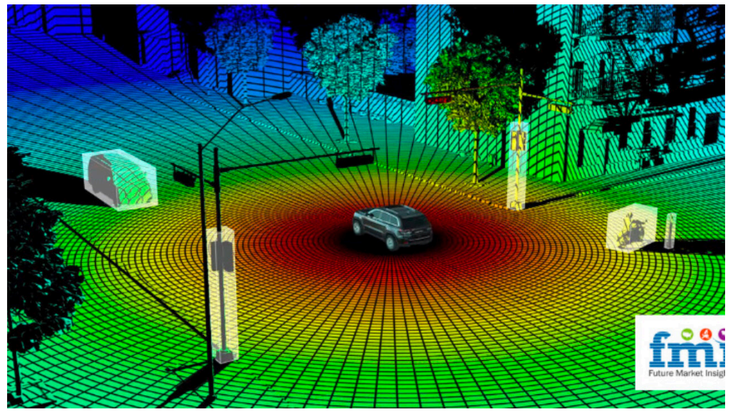




Slide credit: CMU 16-889: Learning for 3D Vision, Prof. Shubham Tulsiani

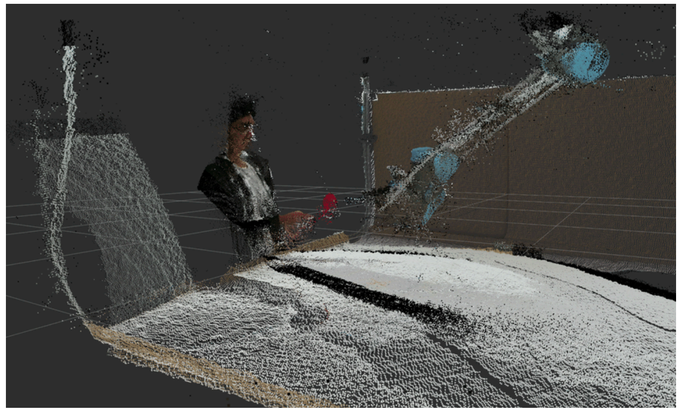
Point Clouds

Often acquired in an image-like manner from sensors but can be useful to ‘forget’ this







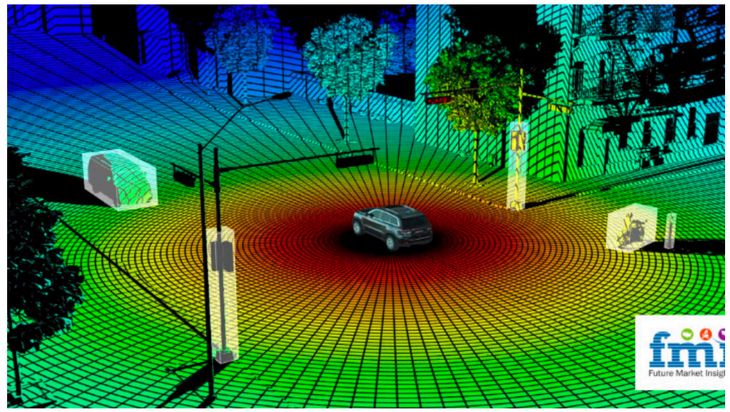


Slide credit: CMU 16-889: Learning for 3D Vision, Prof. Shubham Tulsiani

Point Clouds

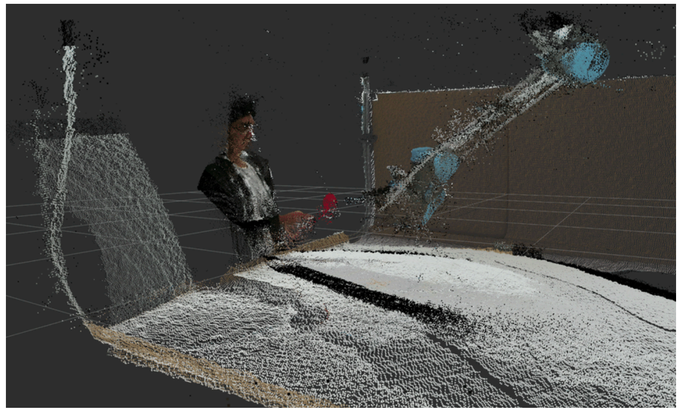
Often acquired in an image-like manner from sensors but can be useful to ‘forget’ this

e.g. point clouds from multiple views can make a single point cloud









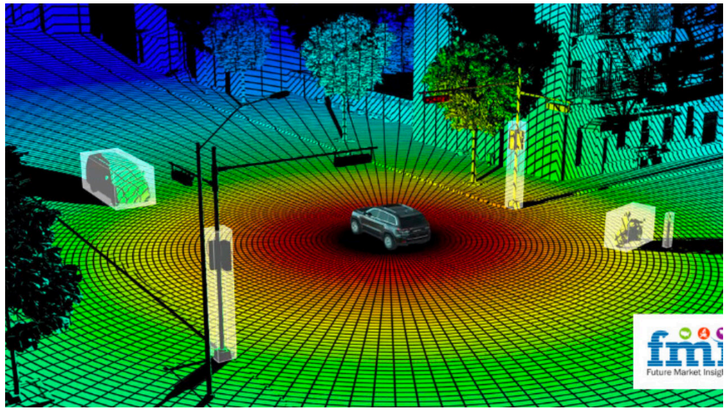
Slide credit: CMU 16-889: Learning for 3D Vision, Prof. Shubham Tulsiani

Point Clouds

Often acquired in an image-like manner from sensors but can be useful to ‘forget’ this

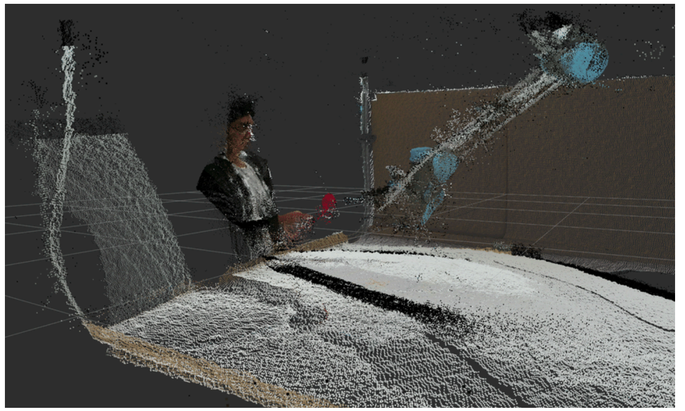
e.g. point clouds from multiple views can make a single point cloud

can help unify approaches independent of sensing modality



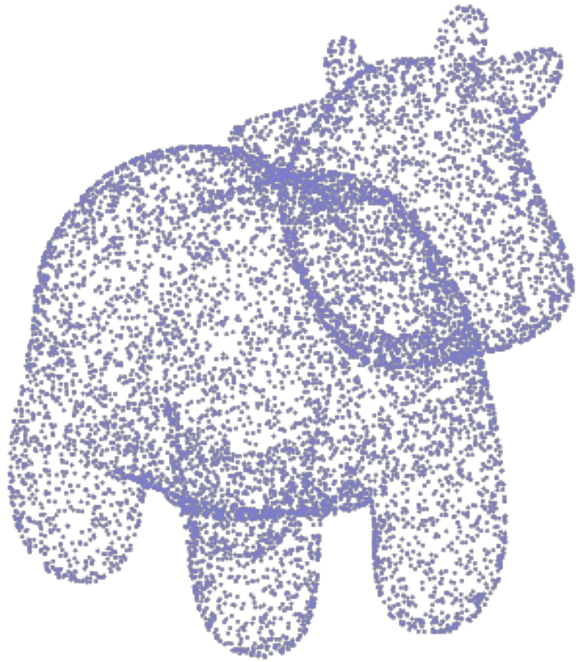






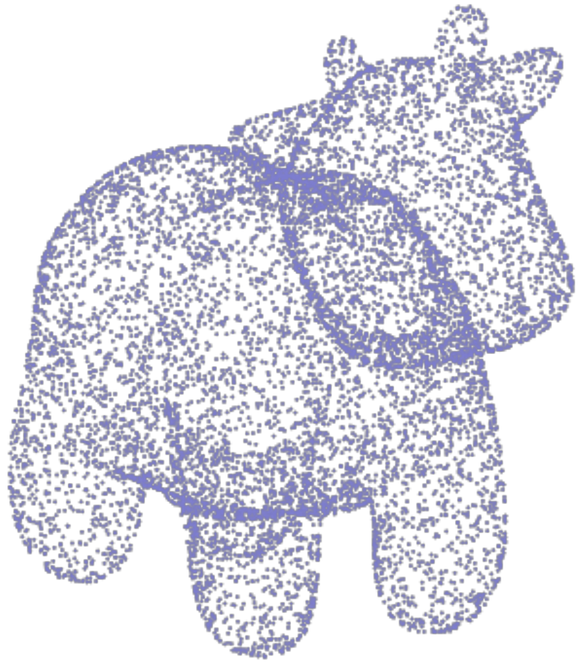
Slide credit: CMU 16-889: Learning for 3D Vision, Prof. Shubham Tulsiani

Point Clouds



Slide credit: CMU 16-889: Learning for 3D Vision, Prof. Shubham Tulsiani

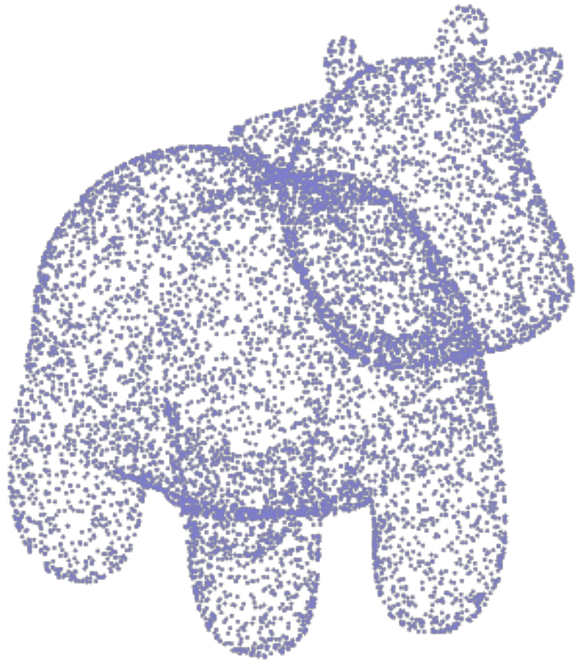
Point Clouds



$$\{\mathbf{p}_1, \mathbf{p}_2, \cdots, \mathbf{p}_N\}$$

Slide credit: CMU 16-889: Learning for 3D Vision, Prof. Shubham Tulsiani

Point Clouds

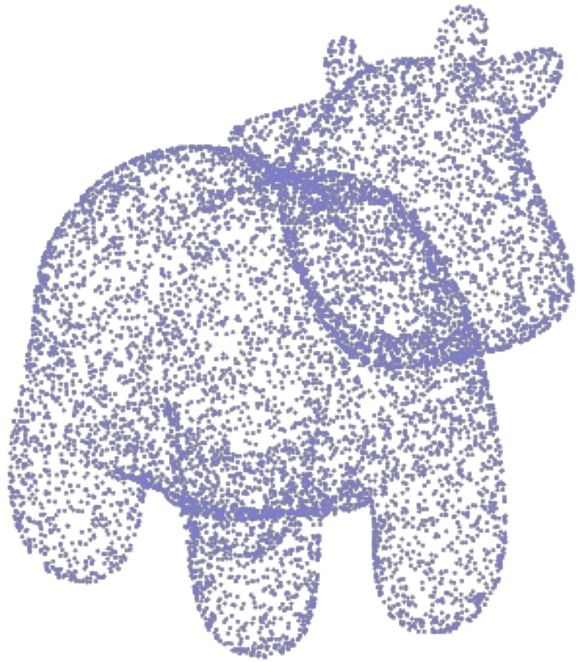


Permutation σ

$$\{\mathbf{p}_1, \mathbf{p}_2, \cdots, \mathbf{p}_N\}$$

Slide credit: CMU 16-889: Learning for 3D Vision, Prof. Shubham Tulsiani

Point Clouds



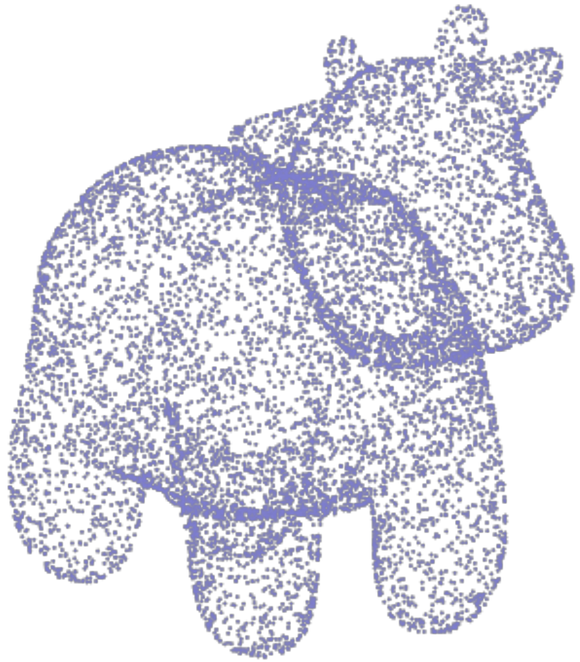
Permutation σ

$$\{\mathbf{p}_1, \mathbf{p}_2, \cdots, \mathbf{p}_N\}$$

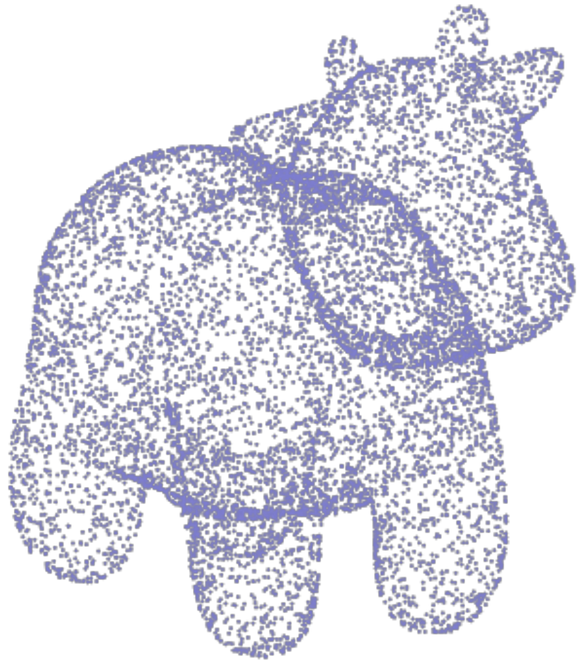
$$\{\mathbf{p}_{\sigma(1)}, \mathbf{p}_{\sigma(2)}, \cdots, \mathbf{p}_{\sigma(N)}\}$$

Slide credit: CMU 16-889: Learning for 3D Vision, Prof. Shubham Tulsiani

Point Clouds



Permutation σ

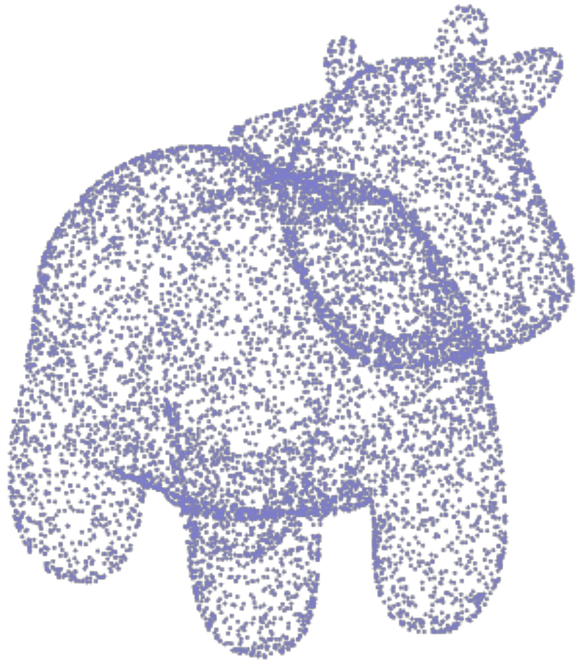


$$\{\mathbf{p}_1, \mathbf{p}_2, \cdots, \mathbf{p}_N\}$$

$$\{\mathbf{p}_{\sigma(1)}, \mathbf{p}_{\sigma(2)}, \cdots, \mathbf{p}_{\sigma(N)}\}$$

Slide credit: CMU 16-889: Learning for 3D Vision, Prof. Shubham Tulsiani

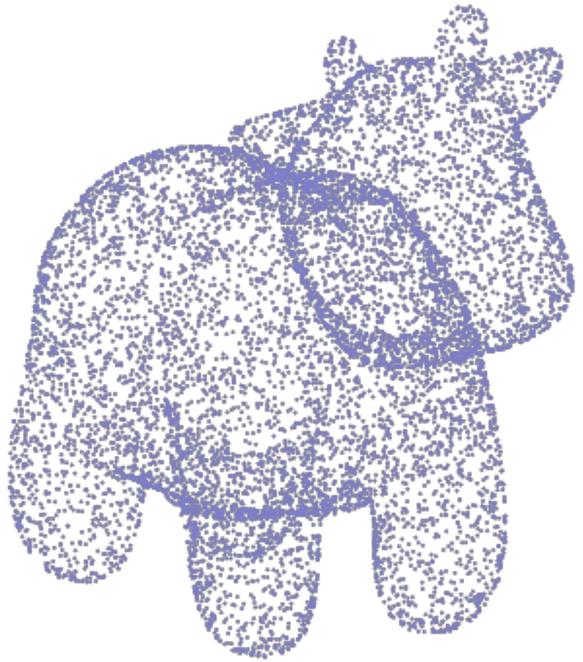
Point Clouds



$$\{\mathbf{p}_1, \mathbf{p}_2, \cdots, \mathbf{p}_N\}$$

Slide credit: CMU 16-889: Learning for 3D Vision, Prof. Shubham Tulsiani

Point Clouds

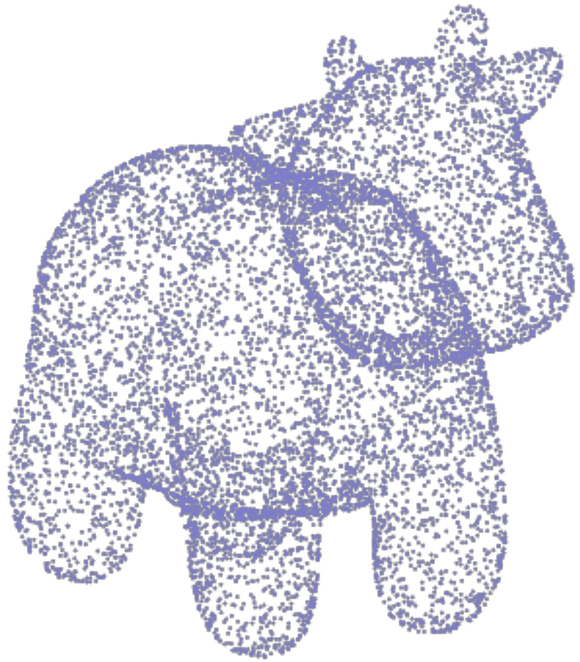


$$\{\mathbf{p}_1, \mathbf{p}_2, \cdots, \mathbf{p}_N\}$$

Unordered set of points

Slide credit: CMU 16-889: Learning for 3D Vision, Prof. Shubham Tulsiani

Point Clouds



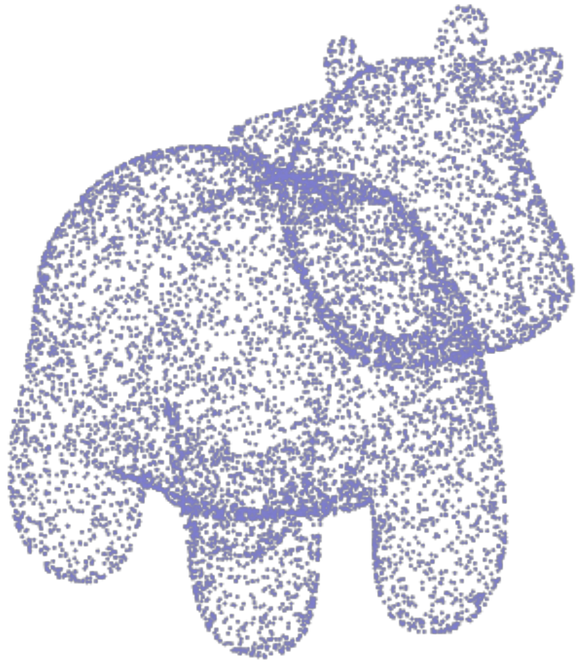
x1,y1,z1
.
.
.
.
.
.
.
.
.
.
.
.
.
.
.

$$\{\mathbf{p}_1, \mathbf{p}_2, \cdots, \mathbf{p}_N\}$$

Unordered set of points

Slide credit: CMU 16-889: Learning for 3D Vision, Prof. Shubham Tulsiani

Point Clouds



x1,y1,z1
.
.
.
.
.
.
.
.
.
.
.
.
.
.
.

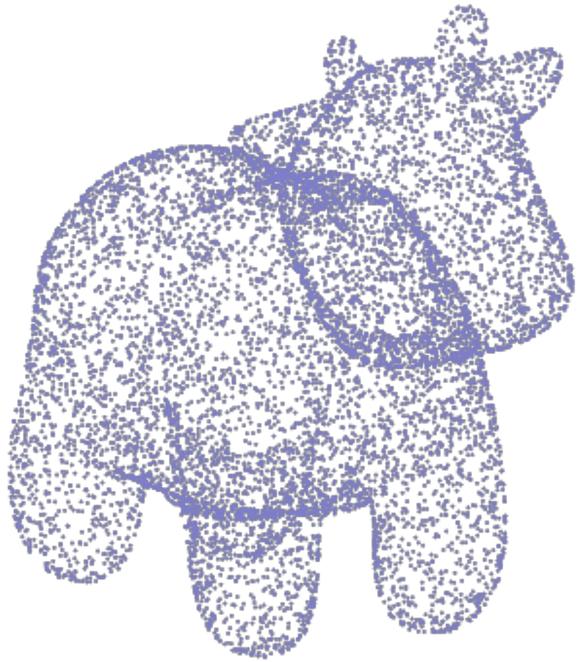
Often represented as a NX3 array, but ordering does **not** matter (unlike images)

$$\{\mathbf{p}_1, \mathbf{p}_2, \cdots, \mathbf{p}_N\}$$

Unordered set of points

Slide credit: CMU 16-889: Learning for 3D Vision, Prof. Shubham Tulsiani

Point Clouds



x1,y1,z1
.
.
.
.
.
.
.
.
.
.
.
.
.
.
.

Often represented as a NX3 array, but ordering does **not** matter (unlike images)

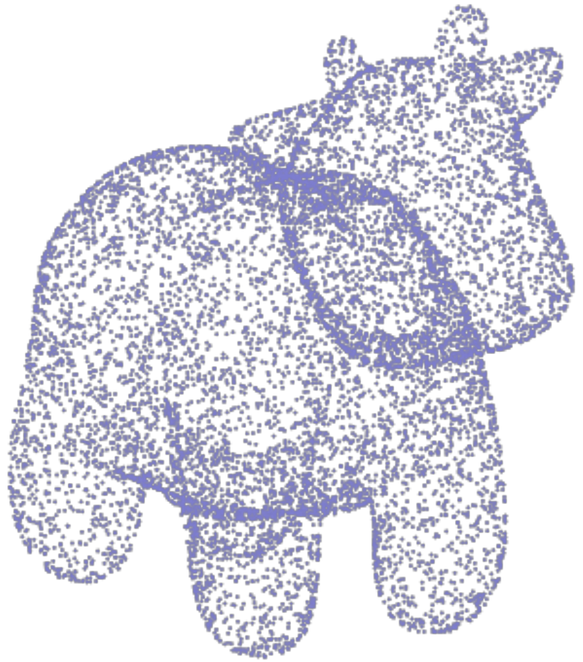
Need processing/generation methods that are permutation invariant (e.g. fully connected layer will not work)

$$\{\mathbf{p}_1, \mathbf{p}_2, \cdots, \mathbf{p}_N\}$$

Unordered set of points

Slide credit: CMU 16-889: Learning for 3D Vision, Prof. Shubham Tulsiani

Point Clouds

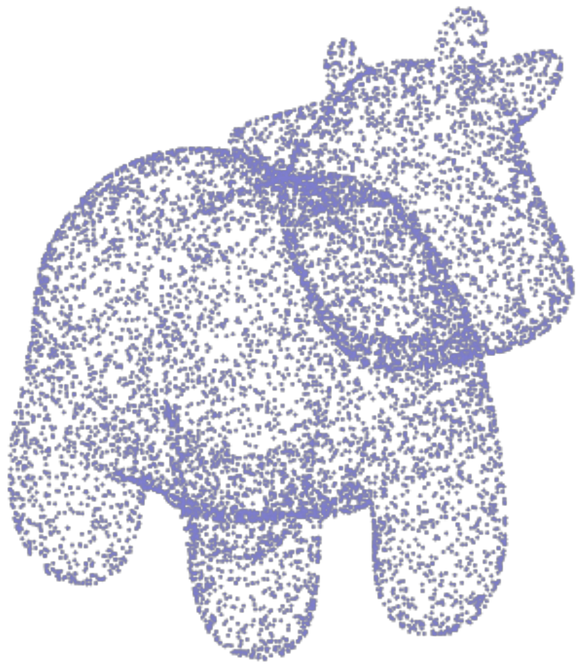


$$\{\mathbf{p}_1, \mathbf{p}_2, \cdots, \mathbf{p}_N\}$$

Unordered set of points

Slide credit: CMU 16-889: Learning for 3D Vision, Prof. Shubham Tulsiani

Point Clouds



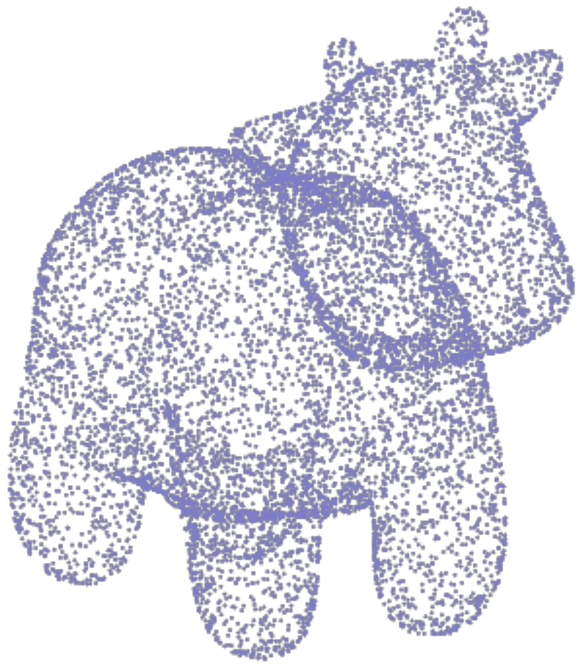
No explicit ‘connectivity’ information

$$\{\mathbf{p}_1, \mathbf{p}_2, \cdots, \mathbf{p}_N\}$$

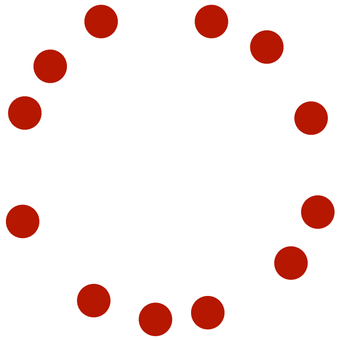
Unordered set of points

Slide credit: CMU 16-889: Learning for 3D Vision, Prof. Shubham Tulsiani

Point Clouds



No explicit ‘connectivity’ information

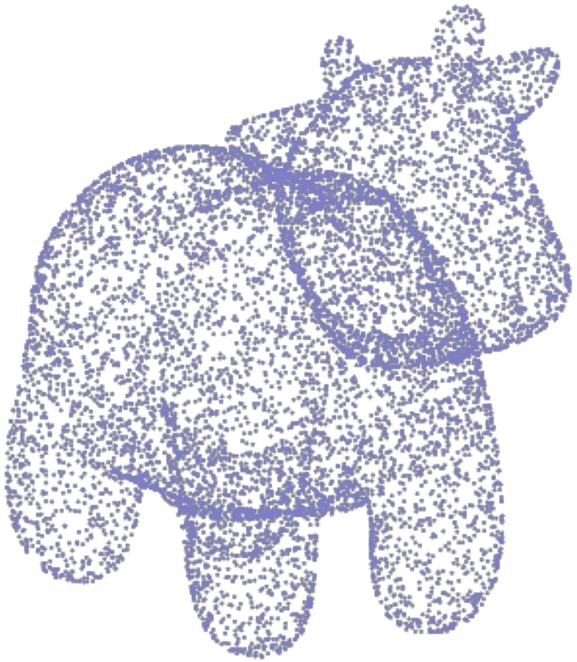


$$\{\mathbf{p}_1, \mathbf{p}_2, \cdots, \mathbf{p}_N\}$$

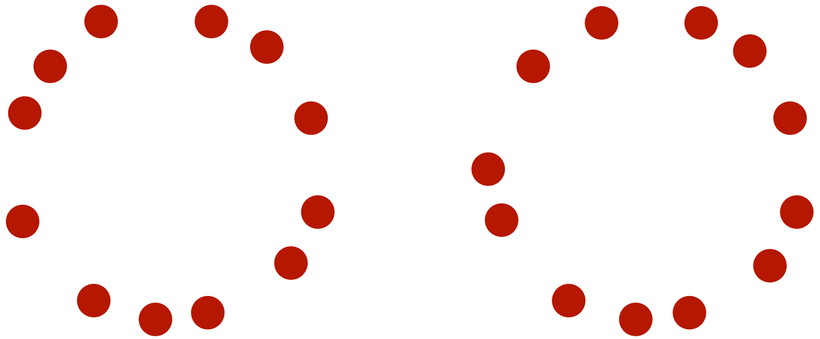
Unordered set of points

Slide credit: CMU 16-889: Learning for 3D Vision, Prof. Shubham Tulsiani

Point Clouds



No explicit ‘connectivity’ information

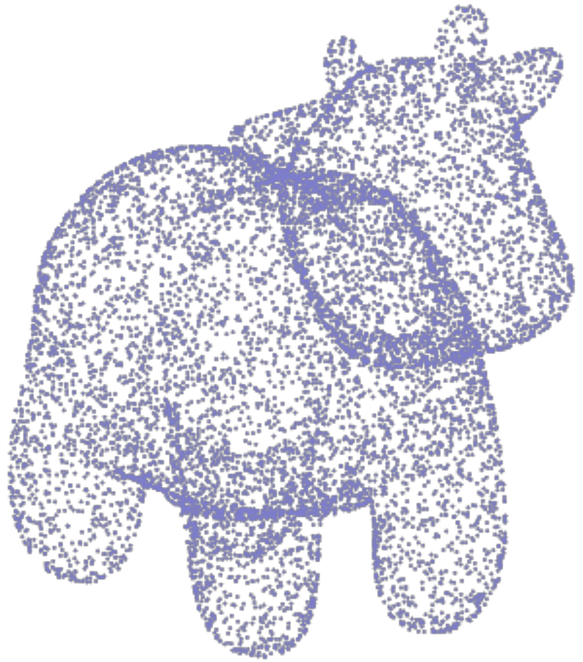


$$\{\mathbf{p}_1, \mathbf{p}_2, \cdots, \mathbf{p}_N\}$$

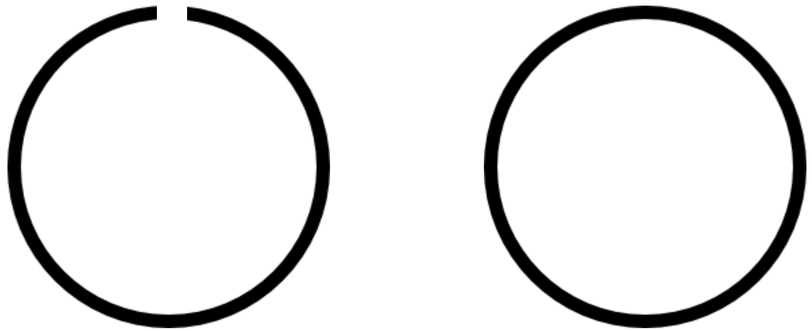
Unordered set of points

Slide credit: CMU 16-889: Learning for 3D Vision, Prof. Shubham Tulsiani

Point Clouds



No explicit ‘connectivity’ information

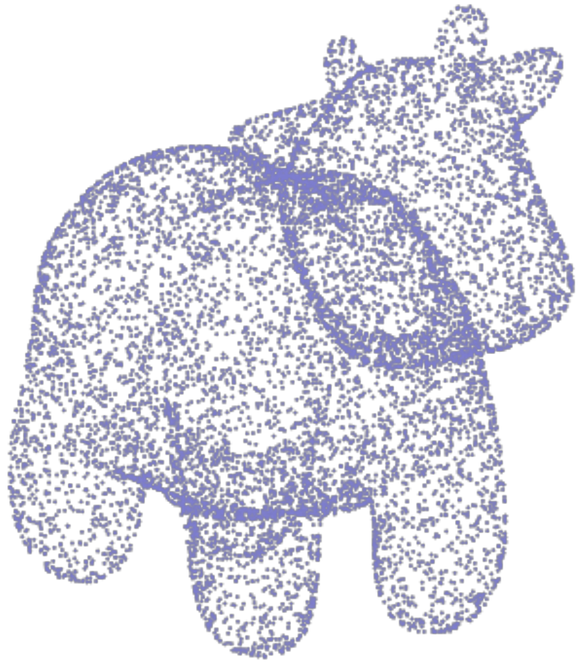


$$\{\mathbf{p}_1, \mathbf{p}_2, \cdots, \mathbf{p}_N\}$$

Unordered set of points

Slide credit: CMU 16-889: Learning for 3D Vision, Prof. Shubham Tulsiani

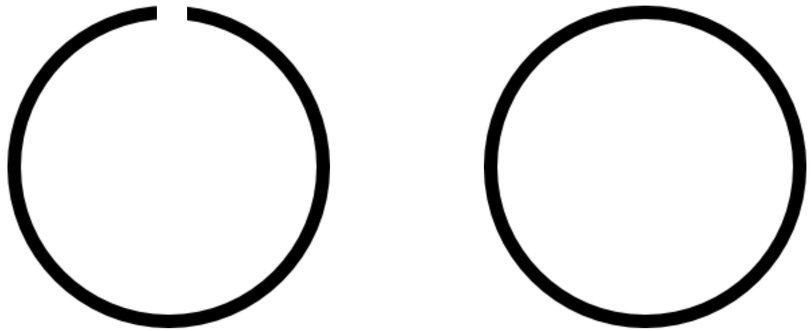
Point Clouds



$$\{\mathbf{p}_1, \mathbf{p}_2, \cdots, \mathbf{p}_N\}$$

Unordered set of points

No explicit ‘connectivity’ information



(yes, we can sample more - but more efficient to add edges)

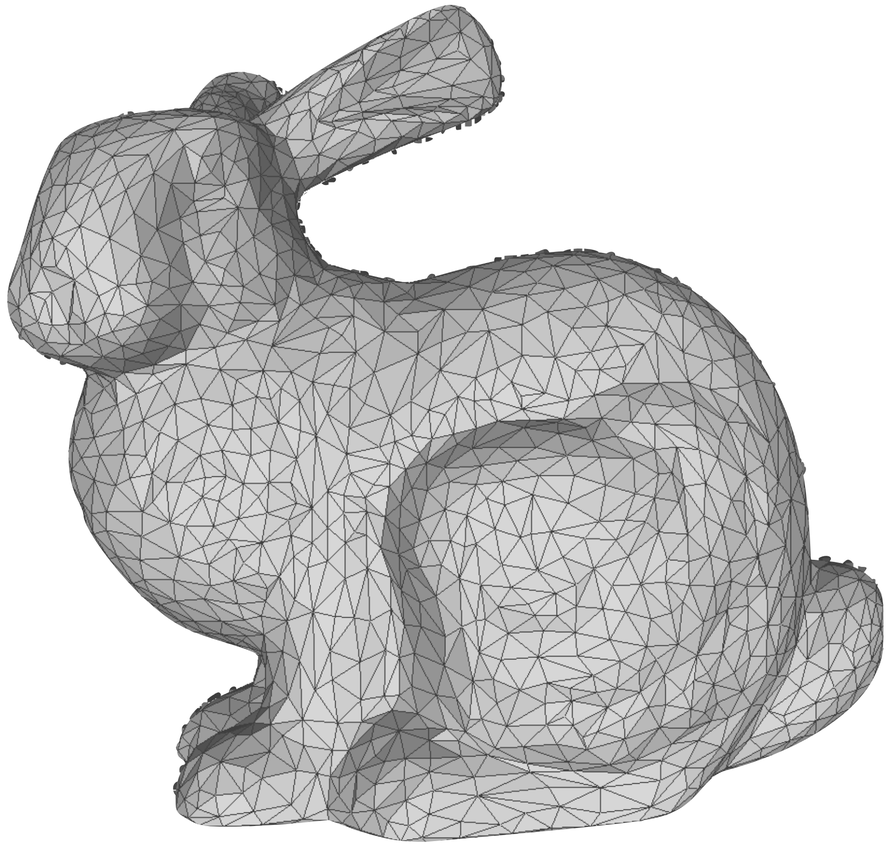
Slide credit: CMU 16-889: Learning for 3D Vision, Prof. Shubham Tulsiani

Meshes



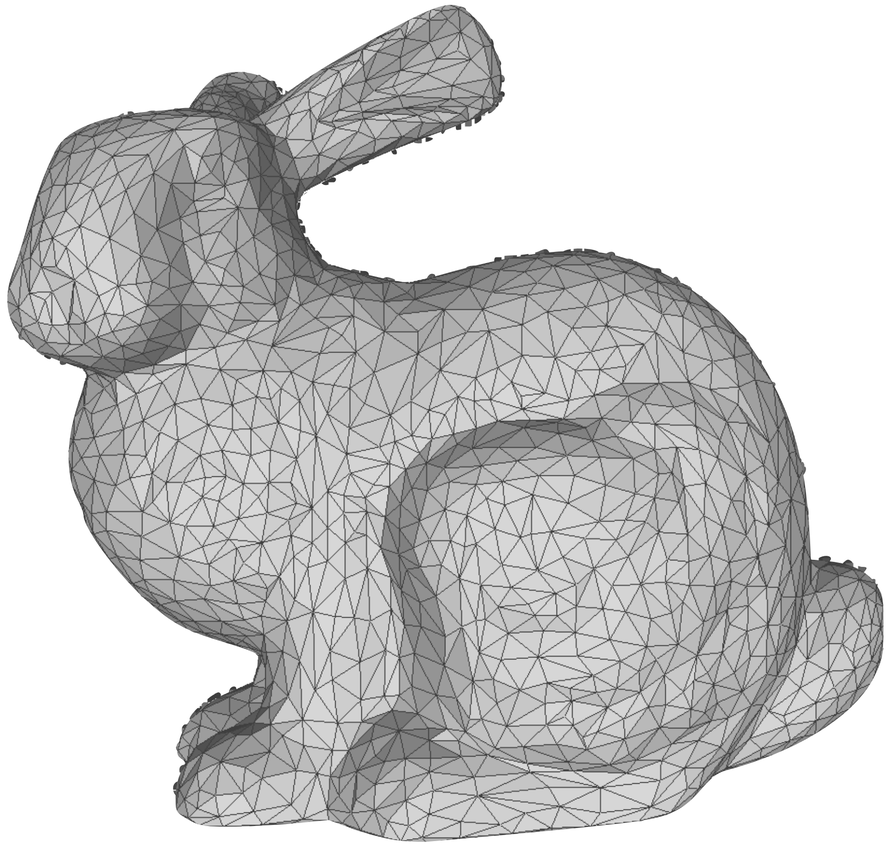
Slide credit: CMU 16-889: Learning for 3D Vision, Prof. Shubham Tulsiani

Meshes



Slide credit: CMU 16-889: Learning for 3D Vision, Prof. Shubham Tulsiani

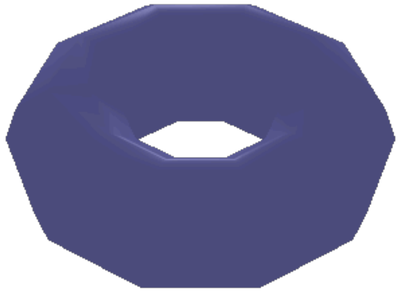
Meshes

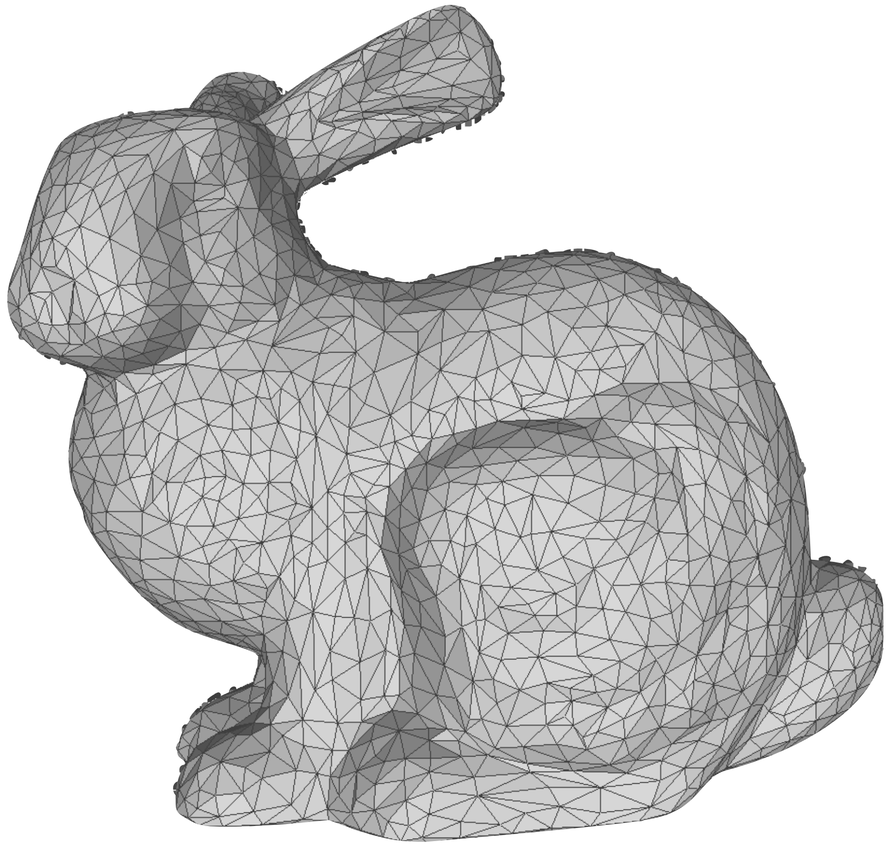


Vertices + Faces

Slide credit: CMU 16-889: Learning for 3D Vision, Prof. Shubham Tulsiani

Meshes

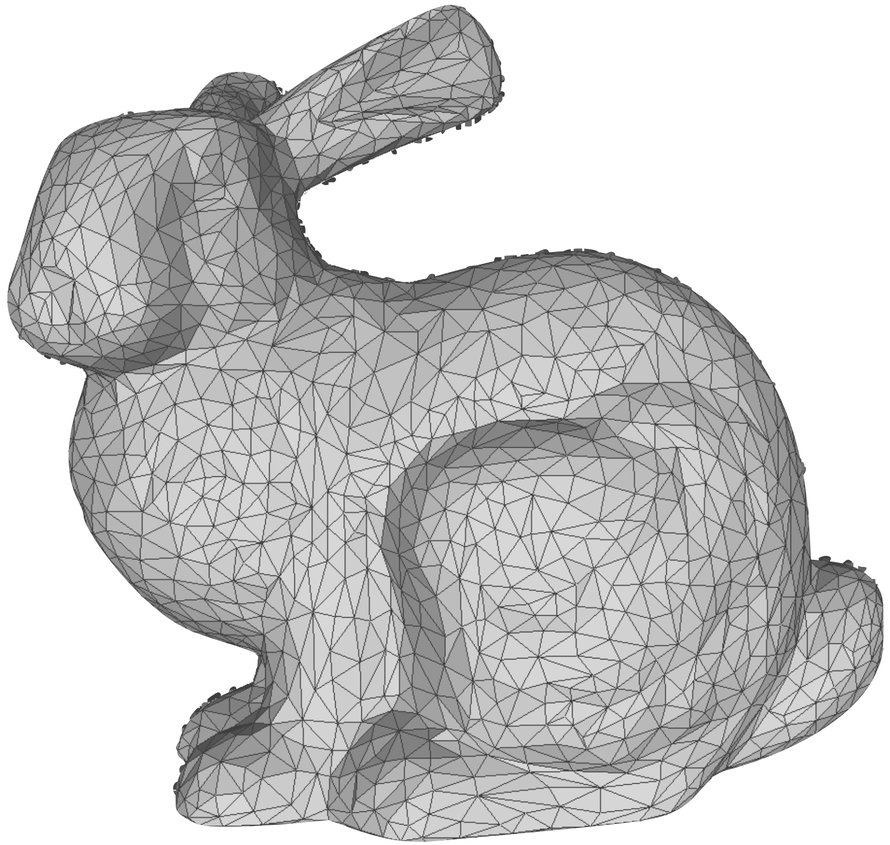




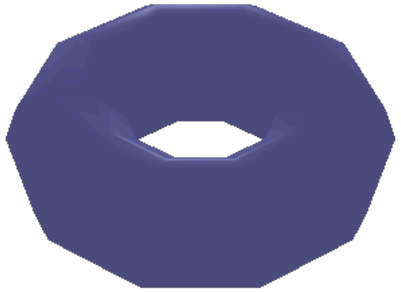
Vertices + Faces

Slide credit: CMU 16-889: Learning for 3D Vision, Prof. Shubham Tulsiani

Meshes



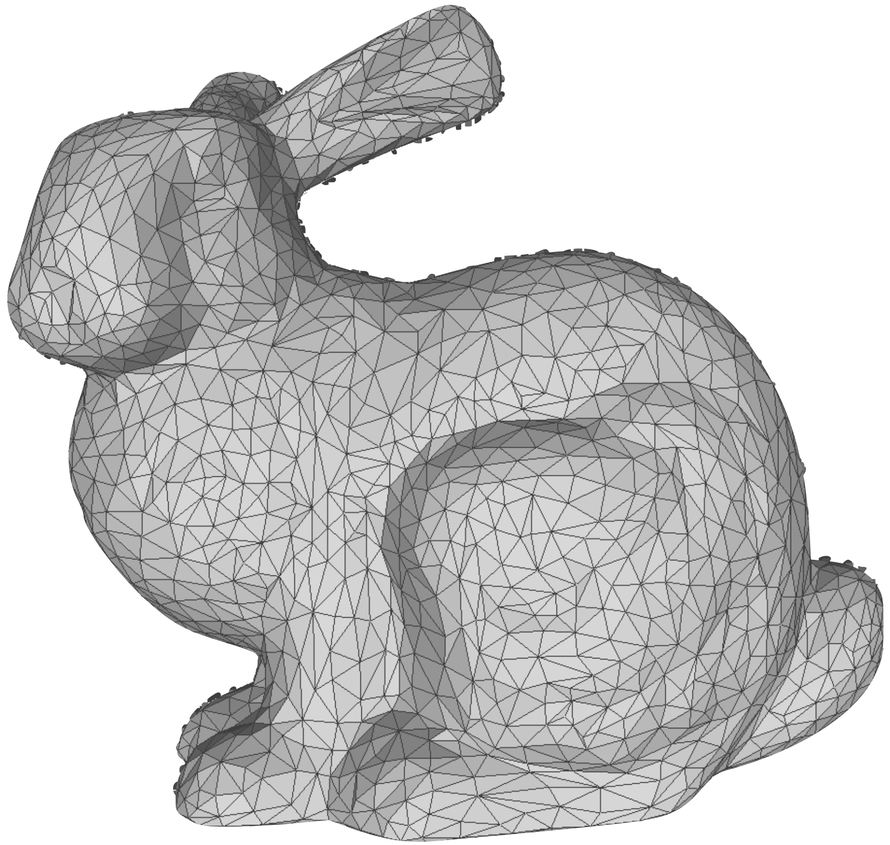
Vertices + Faces



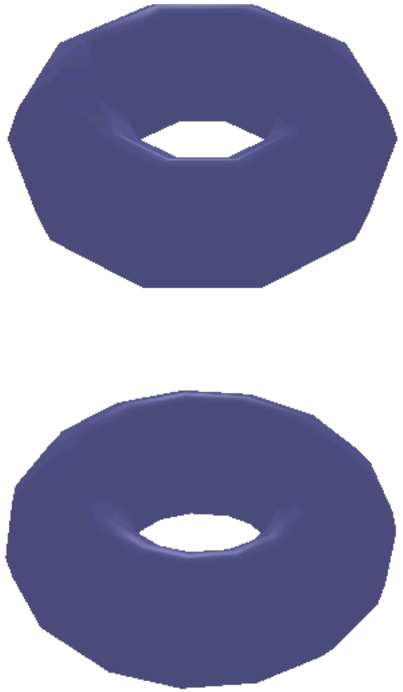
Piecewise linear approximations of underlying surface

Slide credit: CMU 16-889: Learning for 3D Vision, Prof. Shubham Tulsiani

Meshes



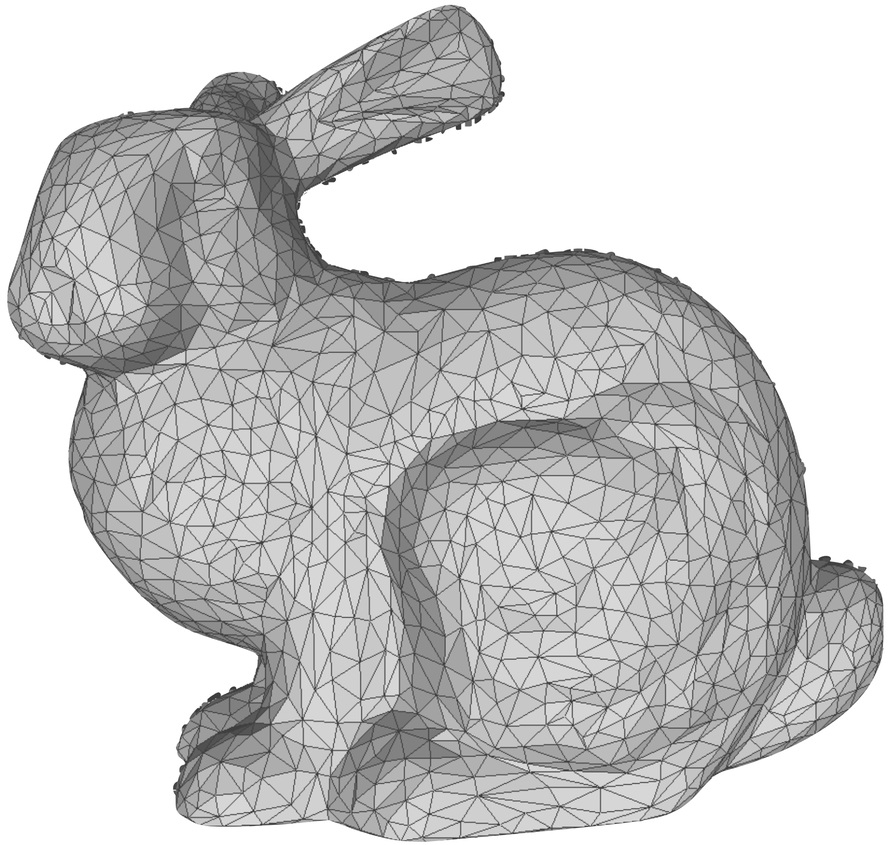
Vertices + Faces



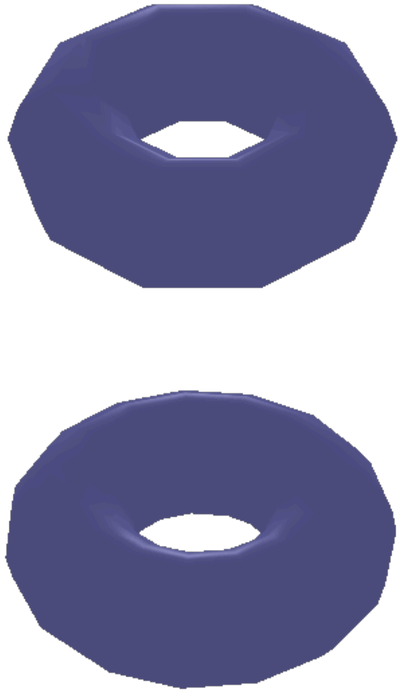
Piecewise linear approximations of underlying surface

Slide credit: CMU 16-889: Learning for 3D Vision, Prof. Shubham Tulsiani

Meshes



Vertices + Faces

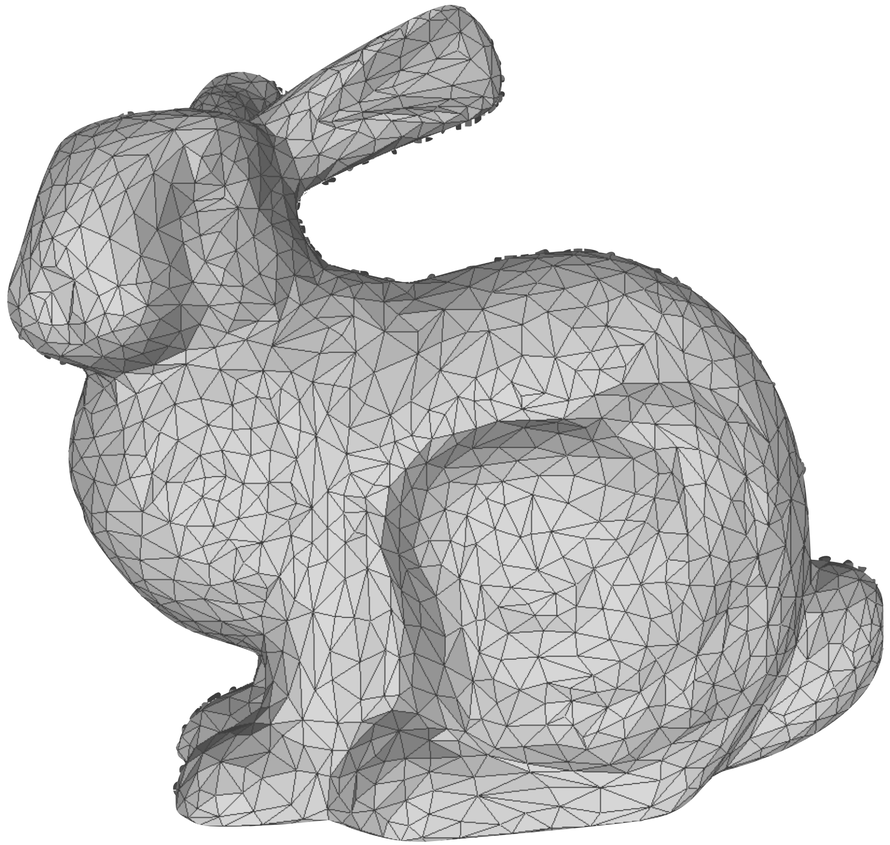


Piecewise linear approximations of underlying surface

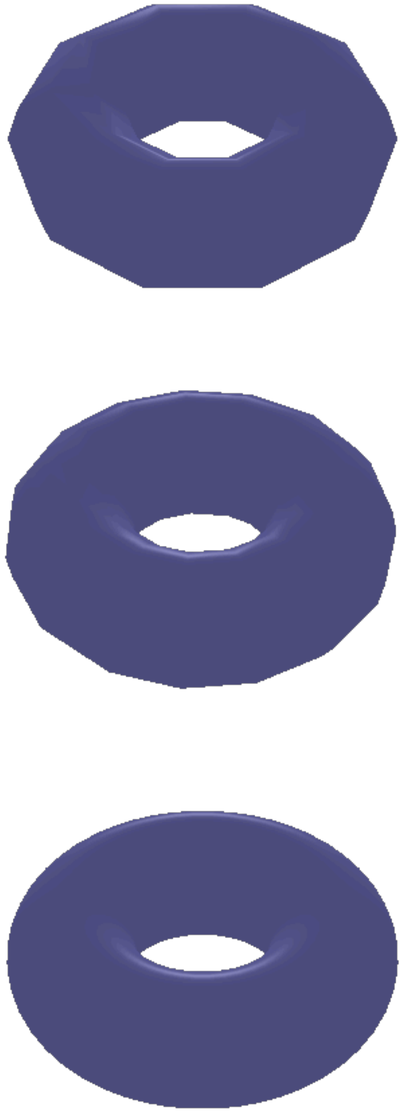
More elements allow better approximation

Slide credit: CMU 16-889: Learning for 3D Vision, Prof. Shubham Tulsiani

Meshes



Vertices + Faces

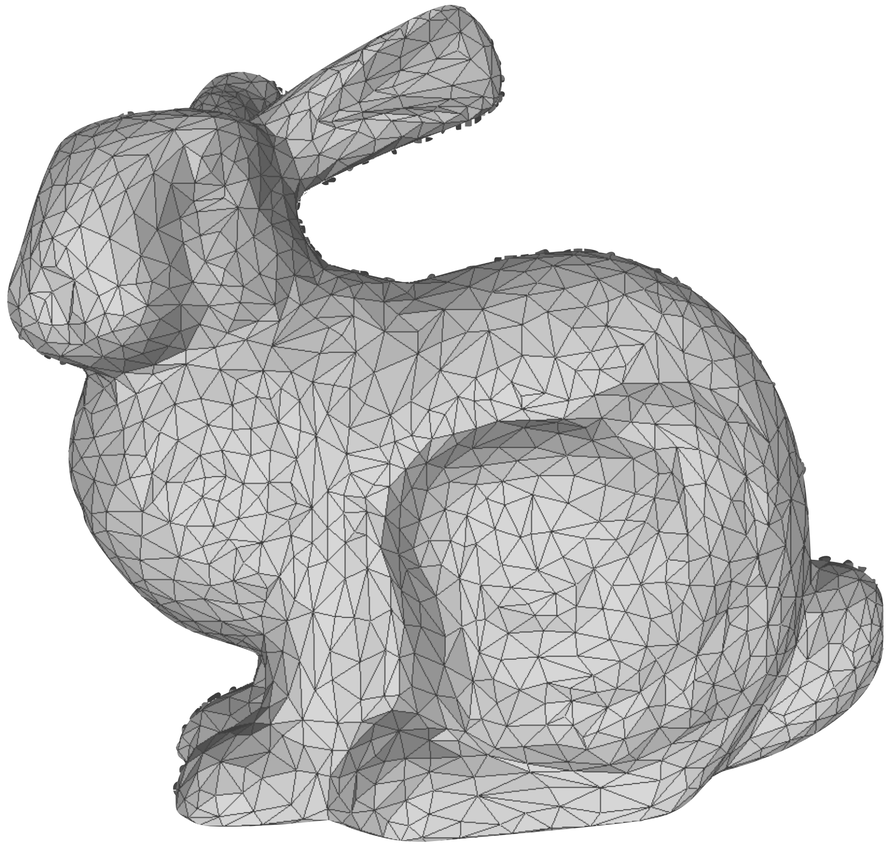


Piecewise linear approximations of underlying surface

More elements allow better approximation

Slide credit: CMU 16-889: Learning for 3D Vision, Prof. Shubham Tulsiani

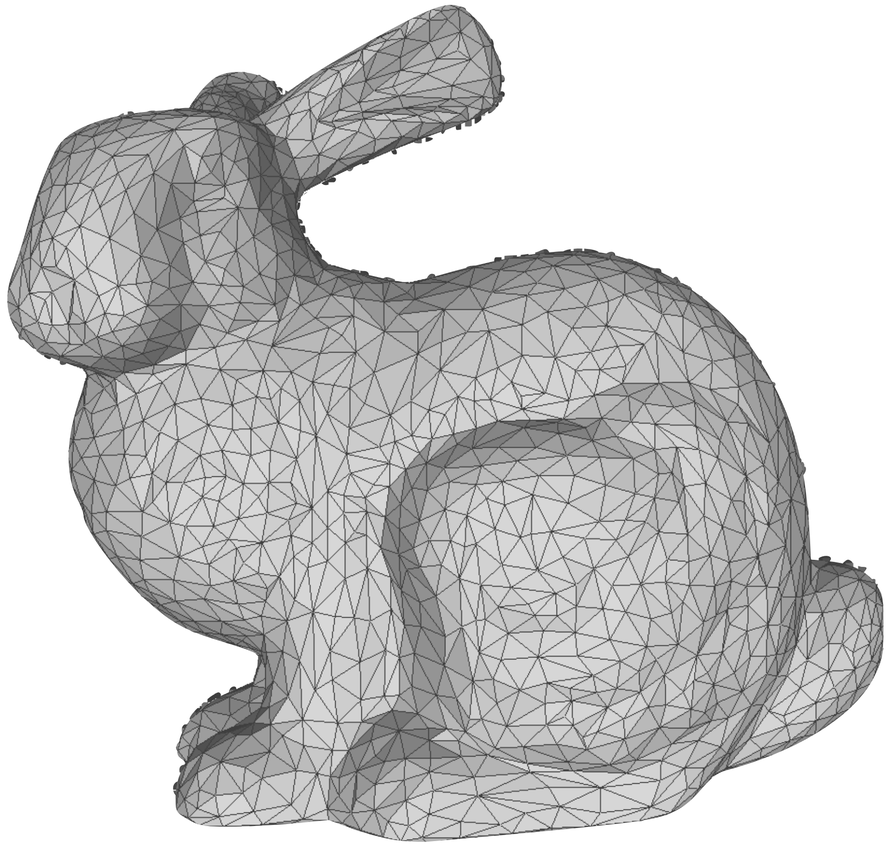
Meshes



Vertices + Faces

Slide credit: CMU 16-889: Learning for 3D Vision, Prof. Shubham Tulsiani

Meshes



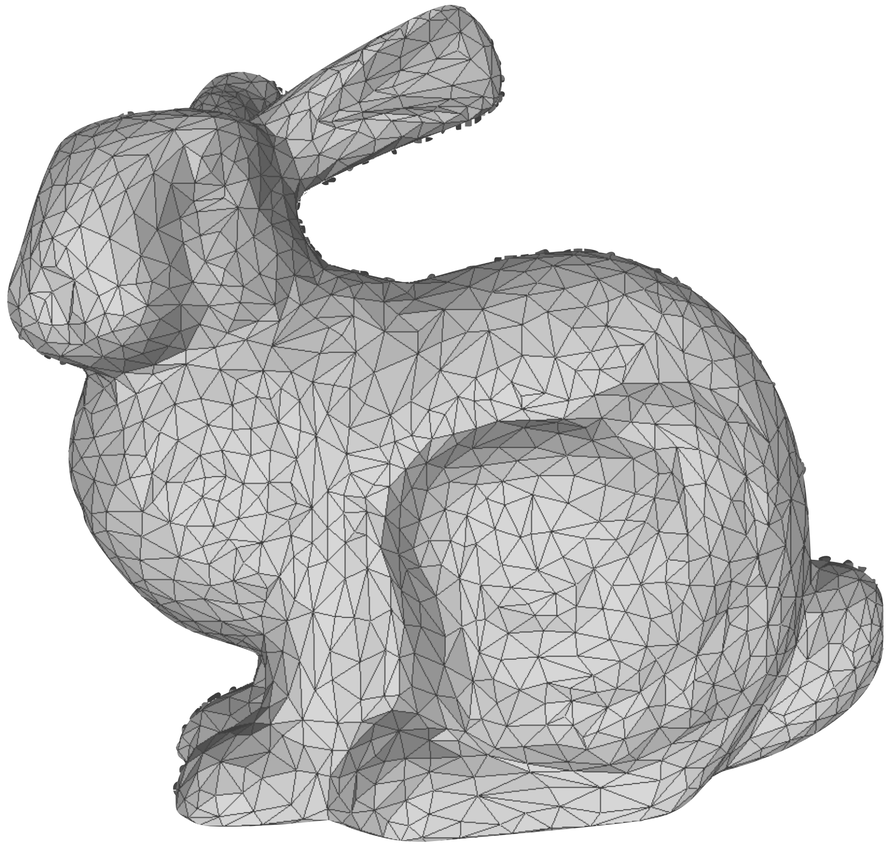
Vertices + Faces

Vertices

x1,y1,z1
.
.
.
.
.
.
.
.
.
.
.
.
.
.
.

Slide credit: CMU 16-889: Learning for 3D Vision, Prof. Shubham Tulsiani

Meshes



Vertices + Faces

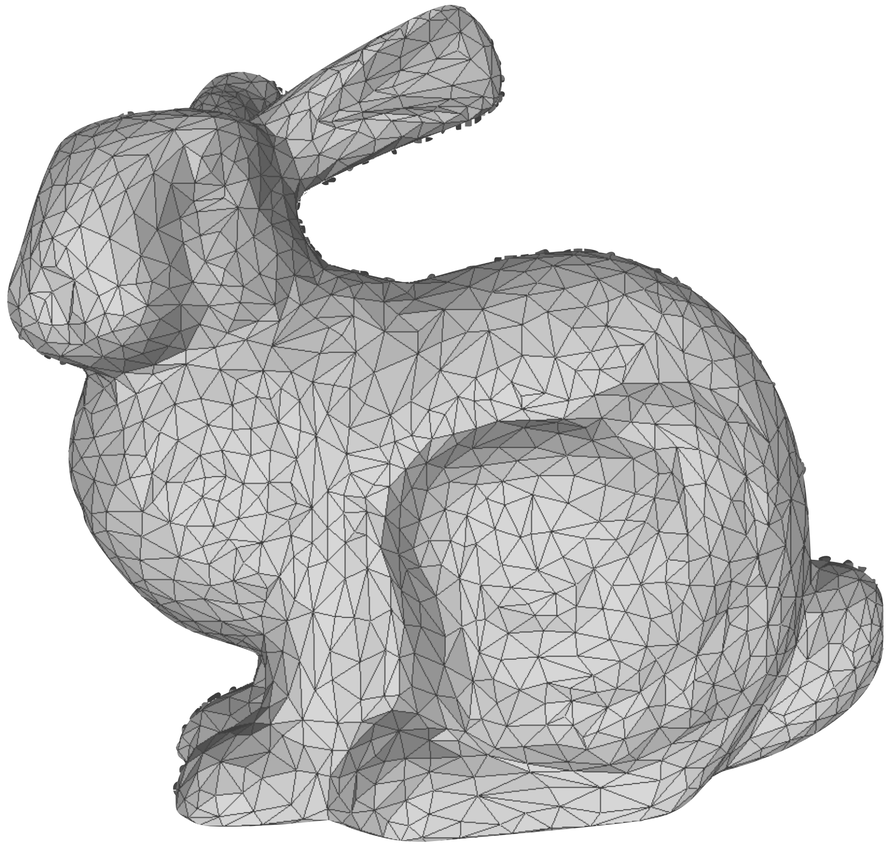
Vertices

x1,y1,z1
.
.
.
.
.
.
.
.
.
.
.
.
.
.
.

Positions of Vertices

Slide credit: CMU 16-889: Learning for 3D Vision, Prof. Shubham Tulsiani

Meshes



Vertices + Faces

Vertices

x1,y1,z1
.
.
.
.
.
.
.
.
.
.
.
.
.
.
.

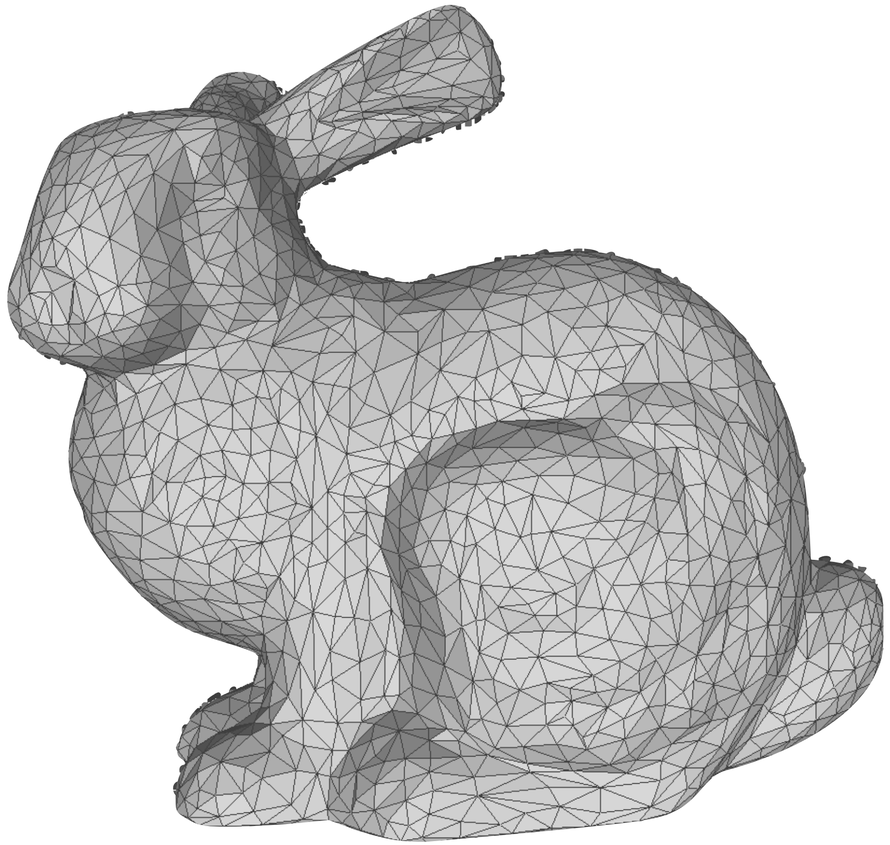
Positions of Vertices

Faces

i,j,k
.
.
.
a,b,c,d,e
.
.
.
.
.
.
.
.
.
.
.
.
.
.
.
.
.
.
.

Slide credit: CMU 16-889: Learning for 3D Vision, Prof. Shubham Tulsiani

Meshes



Vertices + Faces

Vertices

x1,y1,z1
.
.
.
.
.
.
.
.
.
.
.
.
.
.
.

Positions of Vertices

Faces

i,j,k
.
.
.
a,b,c,d,e
.
.
.
.
.
.
.
.
.
.
.
.
.
.
.
.
.
.
.
.

Connectivity
(indices of vertices
that make a ‘face’)

Slide credit: CMU 16-889: Learning for 3D Vision, Prof. Shubham Tulsiani

Meshes

Vertices + Faces

Vertices

x1,y1,z1
.
.
.
.
.
.
.
.
.
.
.
.
.
.
.

Positions of Vertices

Faces

i,j,k
.
.
.
a,b,c,d,e
.
.
.
.
.
.
.
.
.
.
.

Connectivity
(indices of vertices
that make a ‘face’)

Slide credit: CMU 16-889: Learning for 3D Vision, Prof. Shubham Tulsiani

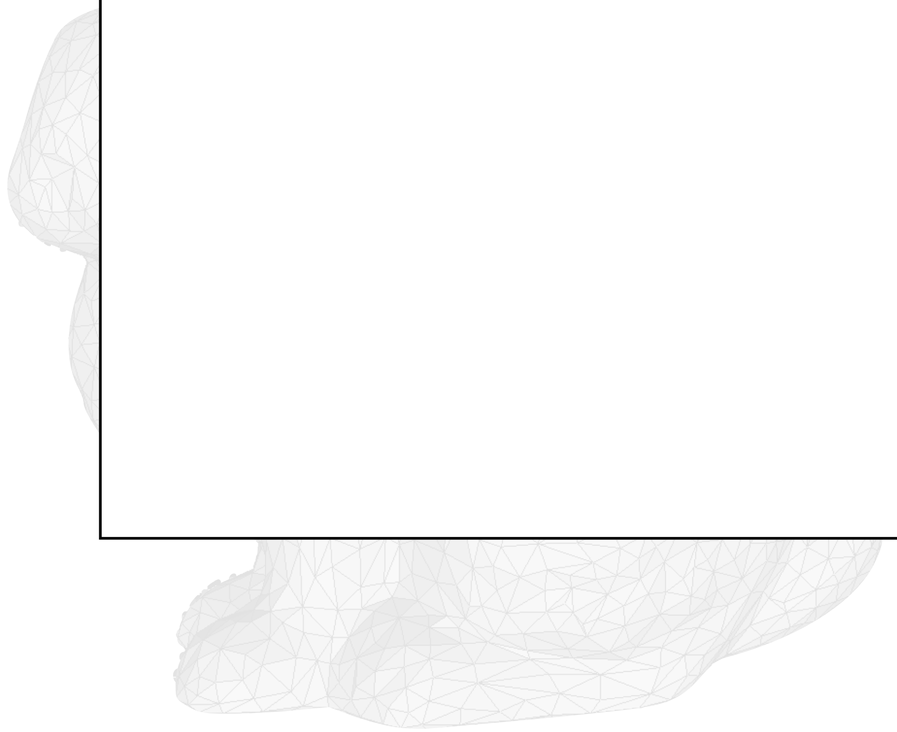
Meshes

Faces

Vertices

i,j,k

Ordering Matters in Face Indices



Vertices + Faces

Positions of Vertices

Connectivity
(indices of vertices
that make a ‘face’)

Slide credit: CMU 16-889: Learning for 3D Vision, Prof. Shubham Tulsiani

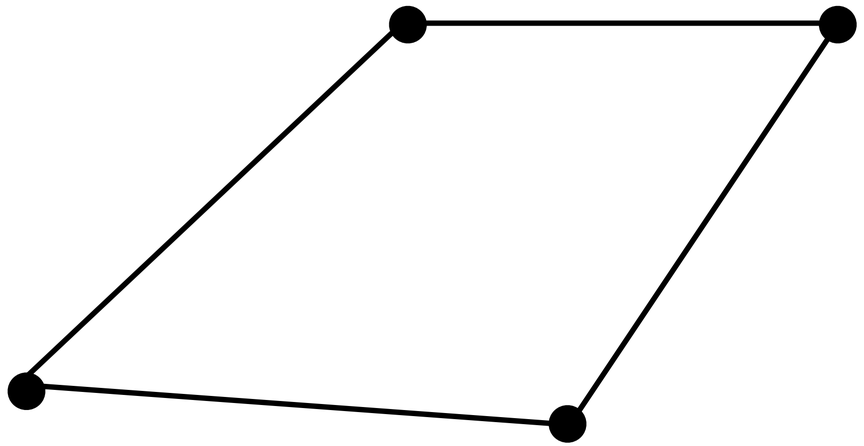
Meshes

Faces

Vertices

i,j,k

Ordering Matters in Face Indices



Positions of Vertices

Connectivity
(indices of vertices
that make a ‘face’)

Vertices + Faces

Slide credit: CMU 16-889: Learning for 3D Vision, Prof. Shubham Tulsiani

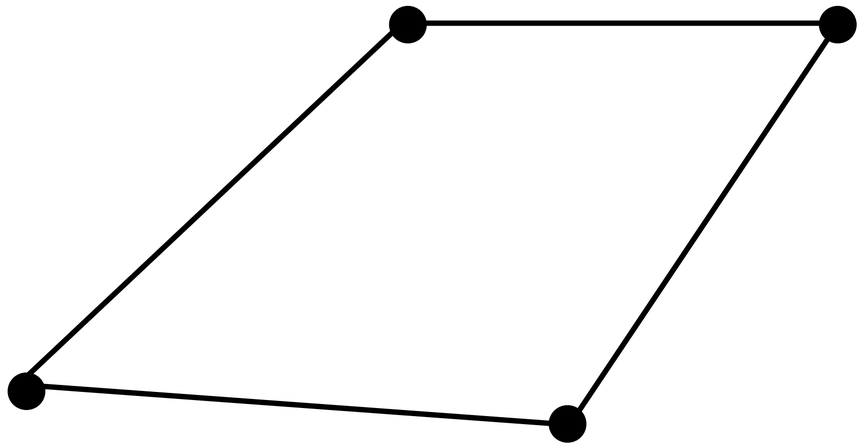
Meshes

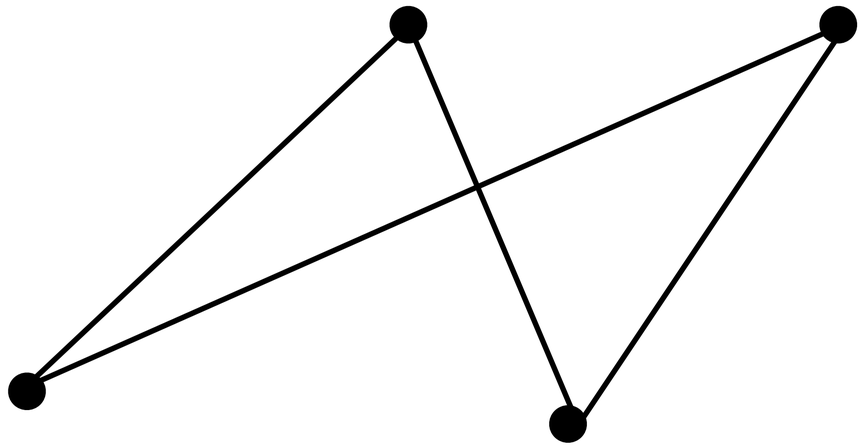
Faces

Vertices

i,j,k

Ordering Matters in Face Indices





Vertices + Faces

Positions of Vertices

Connectivity
(indices of vertices
that make a ‘face’)

Slide credit: CMU 16-889: Learning for 3D Vision, Prof. Shubham Tulsiani

Meshes

Faces

Vertices

i,j,k

Arbitrary Polygons can be non-planar

Positions of Vertices

Connectivity
(indices of vertices
that make a ‘face’)

Vertices + Faces

Slide credit: CMU 16-889: Learning for 3D Vision, Prof. Shubham Tulsiani

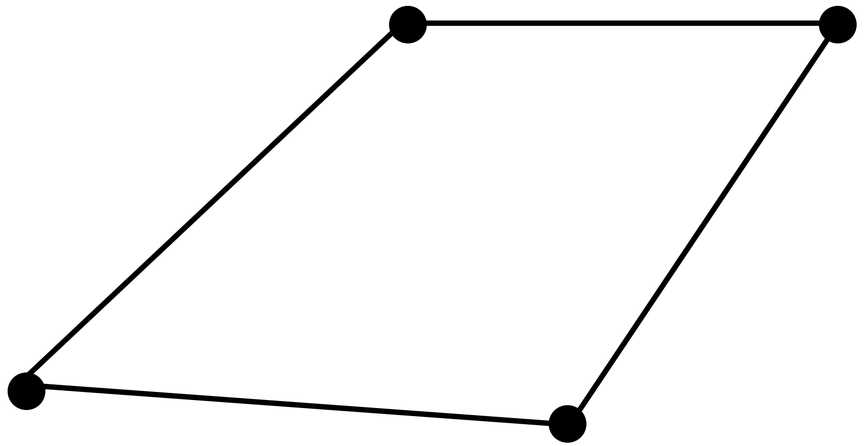
Meshes

Faces

Vertices

i,j,k
.

Arbitrary Polygons can be non-planar



Positions of Vertices

Connectivity
(indices of vertices
that make a ‘face’)

Vertices + Faces

Slide credit: CMU 16-889: Learning for 3D Vision, Prof. Shubham Tulsiani

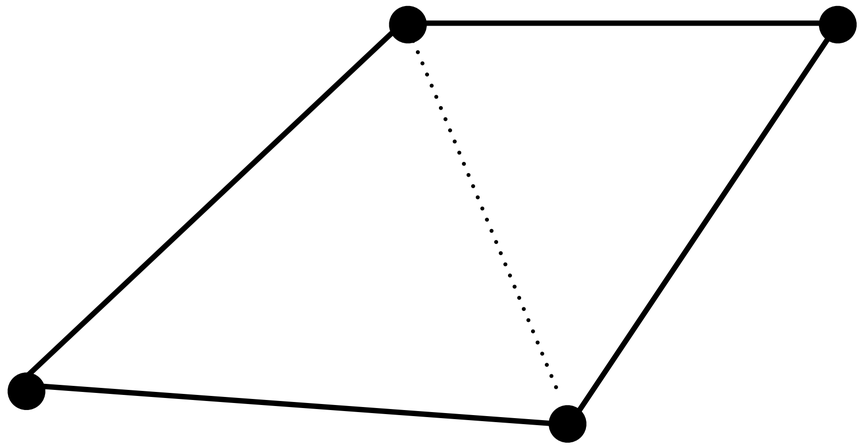
Meshes

Faces

Vertices

i,j,k

Arbitrary Polygons can be non-planar



Positions of Vertices

Connectivity
(indices of vertices
that make a ‘face’)

Vertices + Faces

Slide credit: CMU 16-889: Learning for 3D Vision, Prof. Shubham Tulsiani

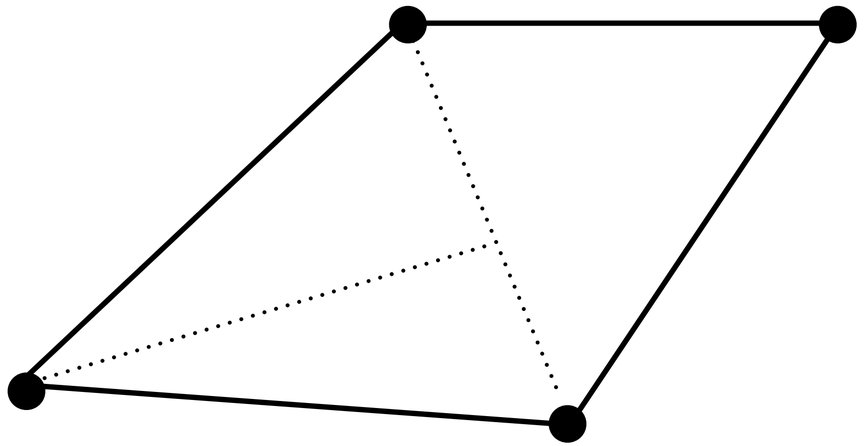
Meshes

Faces

Vertices

i,j,k

Arbitrary Polygons can be non-planar



Positions of Vertices

Connectivity
(indices of vertices
that make a ‘face’)

Vertices + Faces

Slide credit: CMU 16-889: Learning for 3D Vision, Prof. Shubham Tulsiani

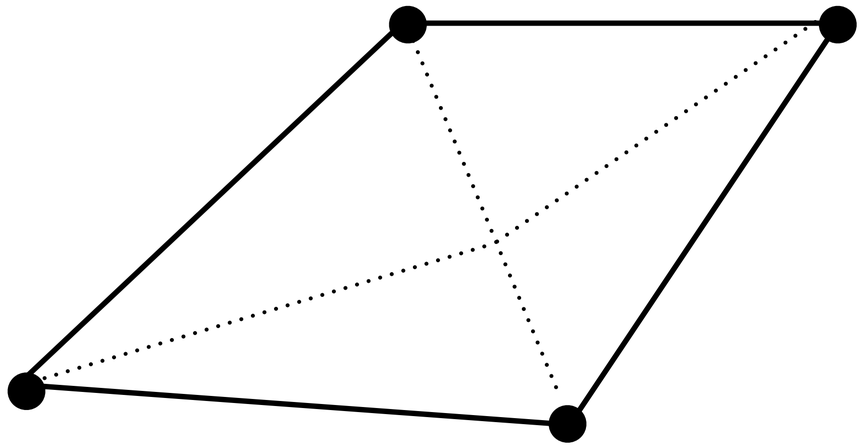
Meshes

Faces

Vertices

i,j,k

Arbitrary Polygons can be non-planar



Positions of Vertices

Connectivity
(indices of vertices
that make a ‘face’)

Vertices + Faces

Slide credit: CMU 16-889: Learning for 3D Vision, Prof. Shubham Tulsiani

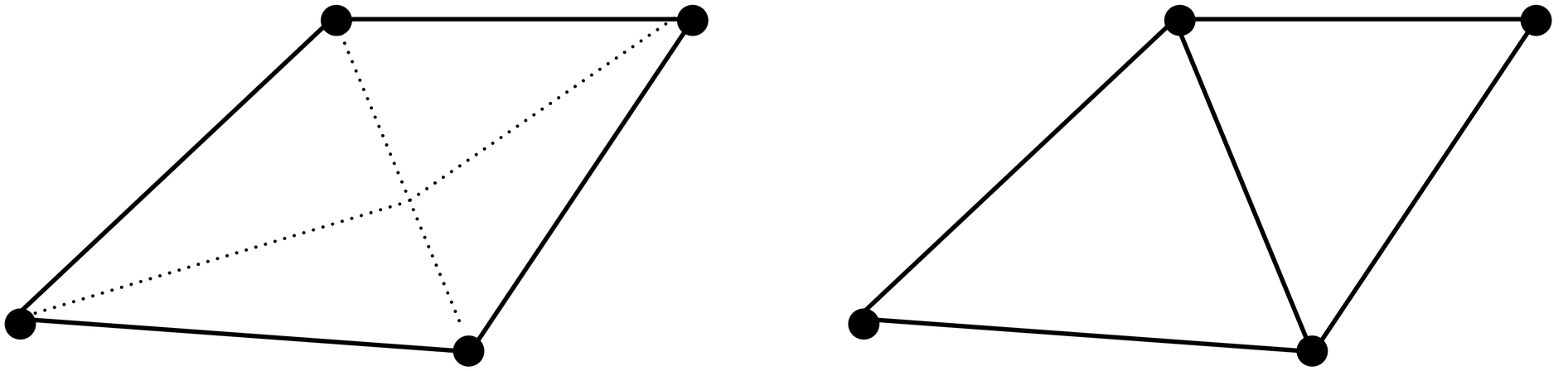
Meshes

Faces

Vertices

i,j,k

Arbitrary Polygons can be non-planar



restrict to *Triangular* Meshes

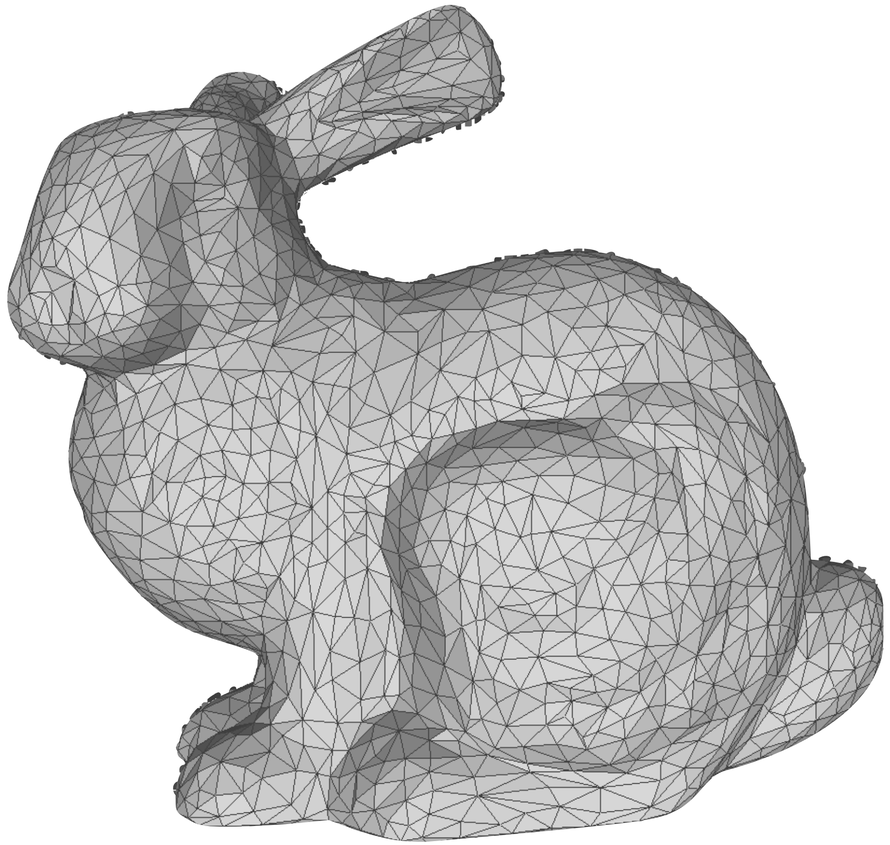
Positions of Vertices

Connectivity (indices of vertices that make a ‘face’)

Vertices + Faces

Slide credit: CMU 16-889: Learning for 3D Vision, Prof. Shubham Tulsiani

Triangular Meshes



Vertices + Faces

Vertices

x1,y1,z1
.
.
.
.
.
.
.
.
.
.
.
.
.
.

Positions of Vertices

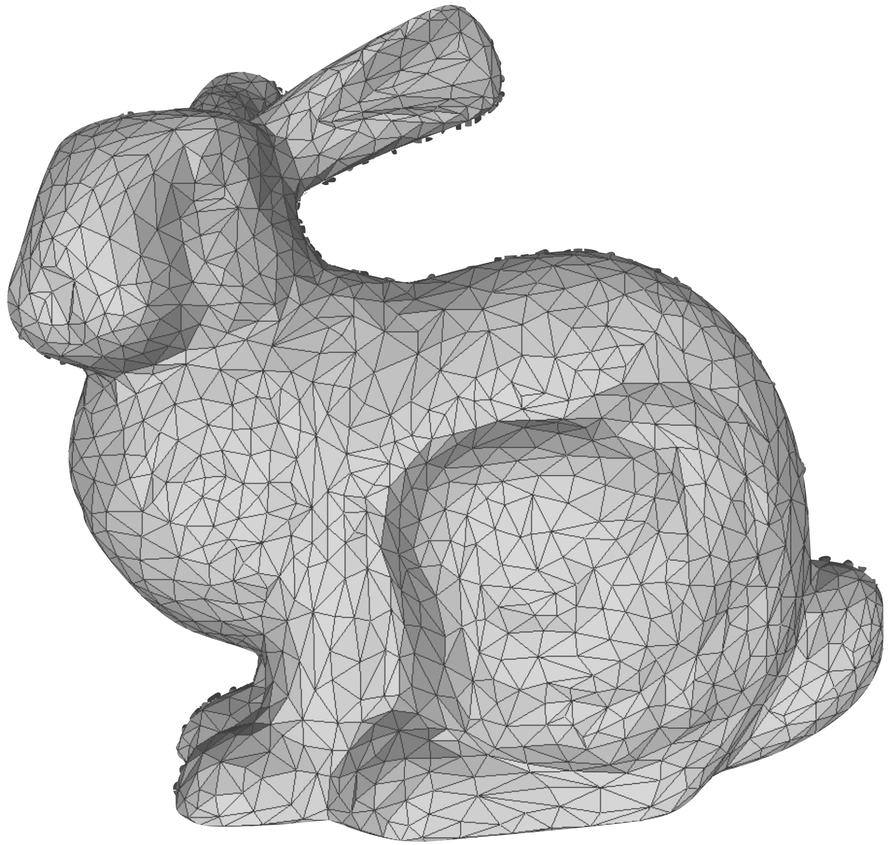
Faces

i,j,k
.
.
.
a,b,c
.
.
.
.
.
.
.
.
.
.
.
.
.
.
.
.

Connectivity (indices of **three** vertices that make a ‘face’)

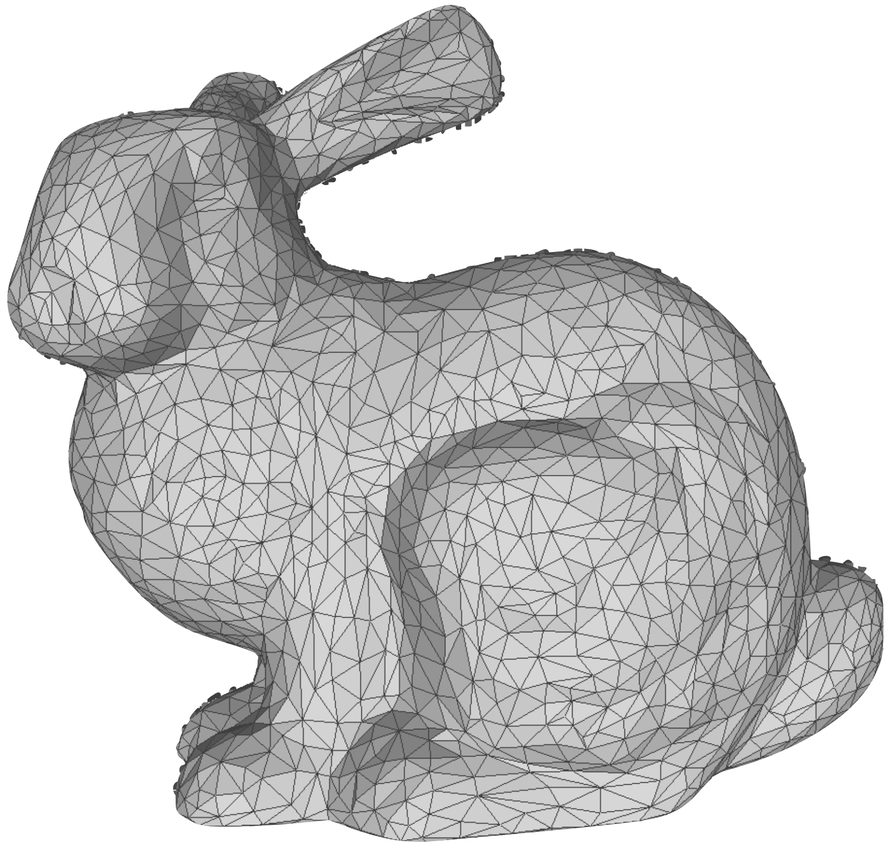
Slide credit: CMU 16-889: Learning for 3D Vision, Prof. Shubham Tulsiani

Is a point inside or outside the shape?



Vertices + Faces

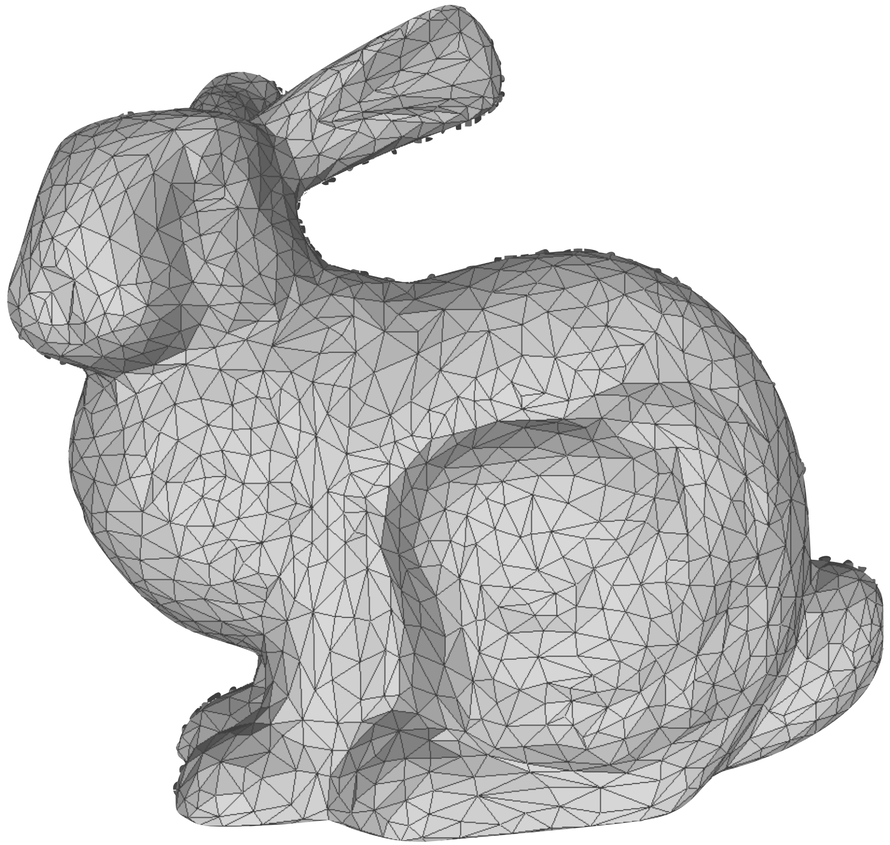
Is a point inside or outside the shape?



Vertices + Faces

Meshes can be “watertight”, i.e., if you filled the mesh with water, nothing would leak out

Is a point inside or outside the shape?

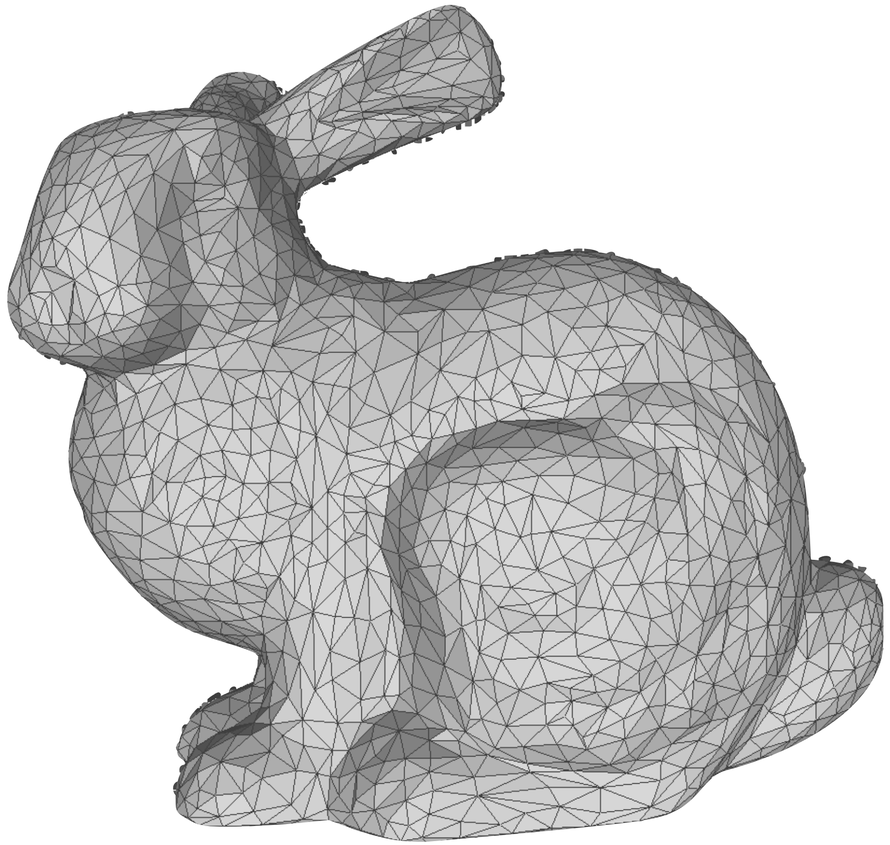


Vertices + Faces

Meshes can be “watertight”, i.e., if you filled the mesh with water, nothing would leak out

If mesh is watertight, can define “inside” and “outside”

Is a point inside or outside the shape?



Vertices + Faces

Meshes can be “watertight”, i.e., if you filled the mesh with water, nothing would leak out

If mesh is watertight, can define “inside” and “outside”

If not, needs further processing, see “Generalized Winding Numbers”

Triangular Meshes



Slide credit: CMU 16-889: Learning for 3D Vision, Prof. Shubham Tulsiani

Triangular Meshes

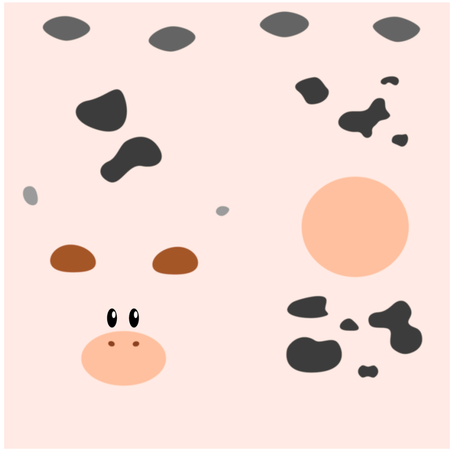


Efficient to compute ray-triangle intersections

Slide credit: CMU 16-889: Learning for 3D Vision, Prof. Shubham Tulsiani

Triangular Meshes



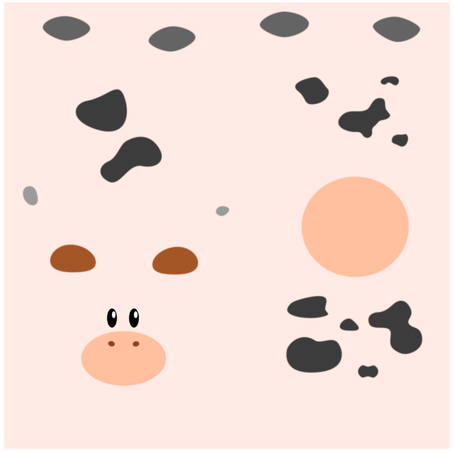




Efficient to compute ray-triangle intersections

Triangular Meshes







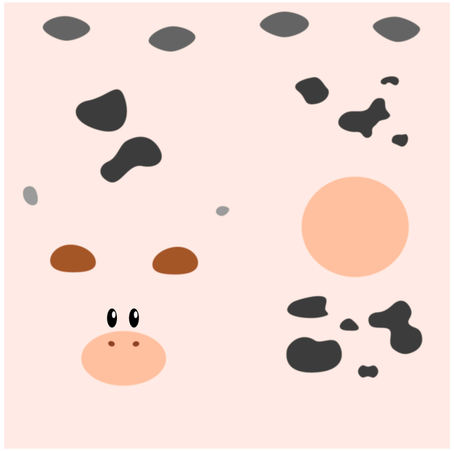


Efficient to compute ray-triangle intersections

Slide credit: CMU 16-889: Learning for 3D Vision, Prof. Shubham Tulsiani

Triangular Meshes









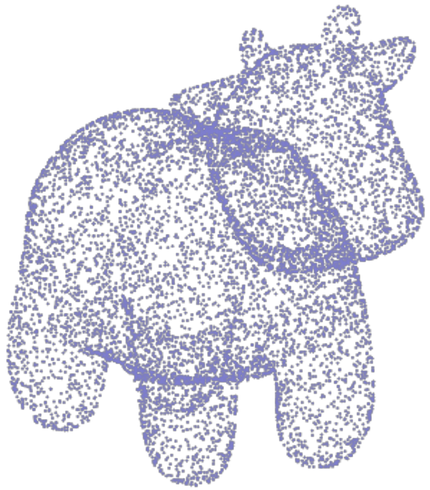
Efficient to compute ray-triangle intersections

Easy to texture and render (common representation across graphics)

Slide credit: CMU 16-889: Learning for 3D Vision, Prof. Shubham Tulsiani

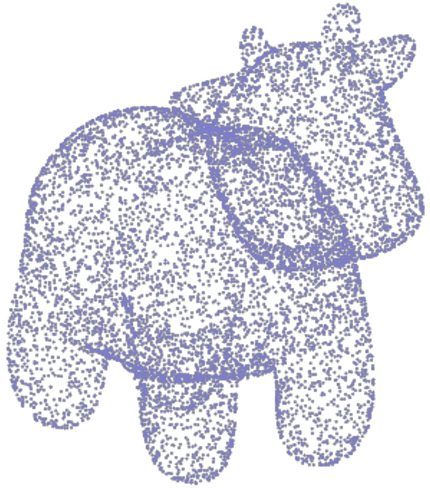
Surface Representations

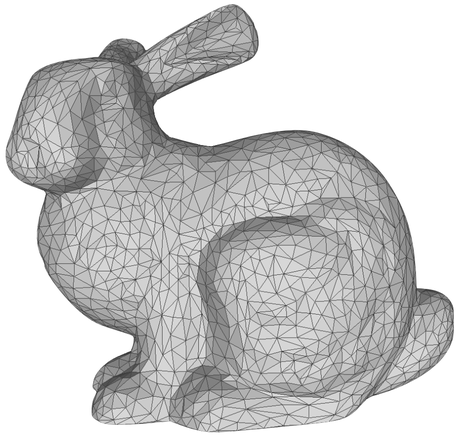
Slide credit: CMU 16-889: Learning for 3D Vision, Prof. Shubham Tulsiani



Surface Representations

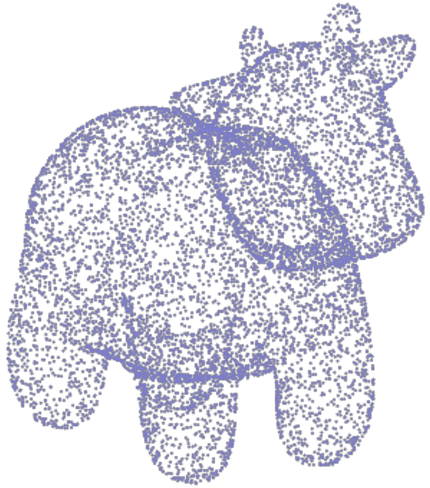
Slide credit: CMU 16-889: Learning for 3D Vision, Prof. Shubham Tulsiani

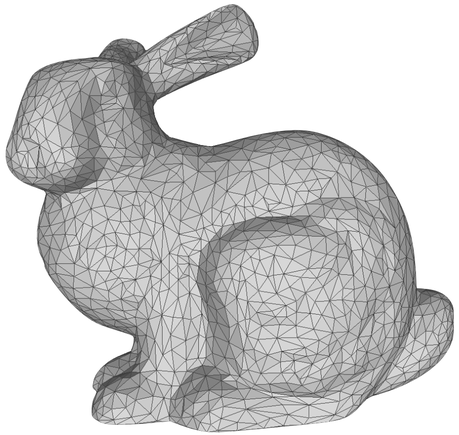


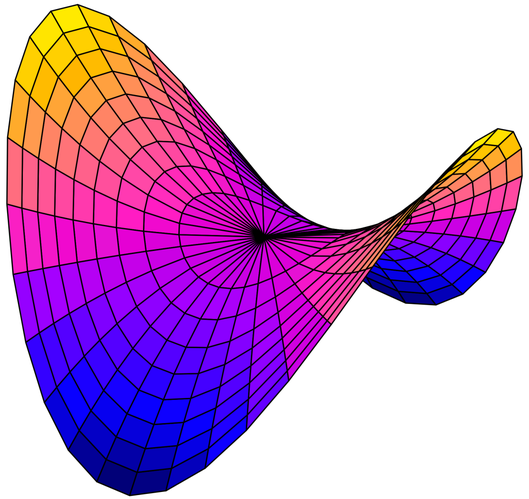


Surface Representations

Slide credit: CMU 16-889: Learning for 3D Vision, Prof. Shubham Tulsiani

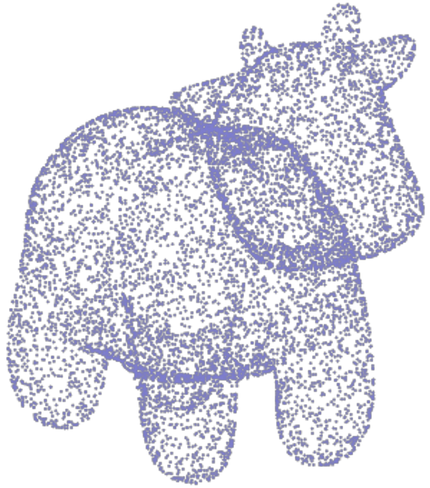


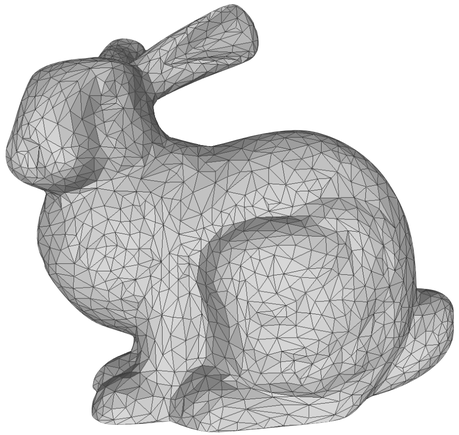


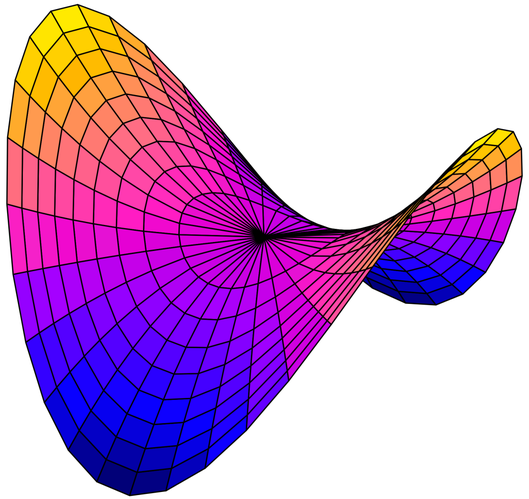


Surface Representations

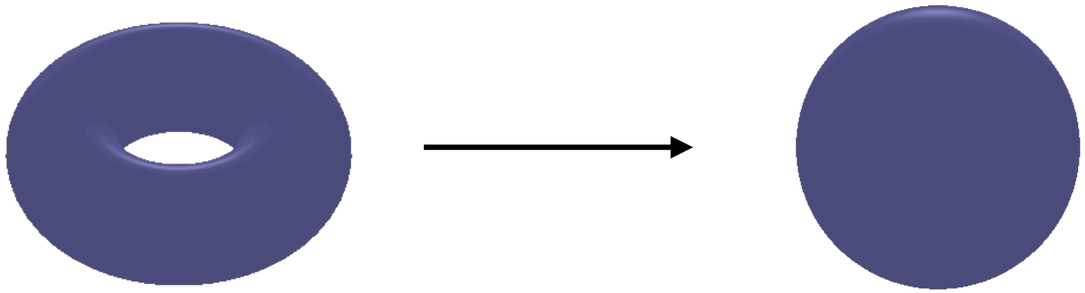
Slide credit: CMU 16-889: Learning for 3D Vision, Prof. Shubham Tulsiani



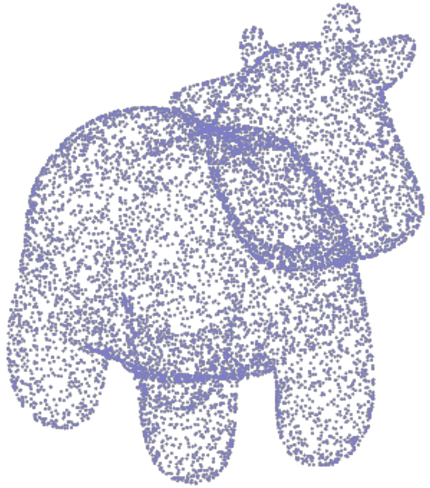


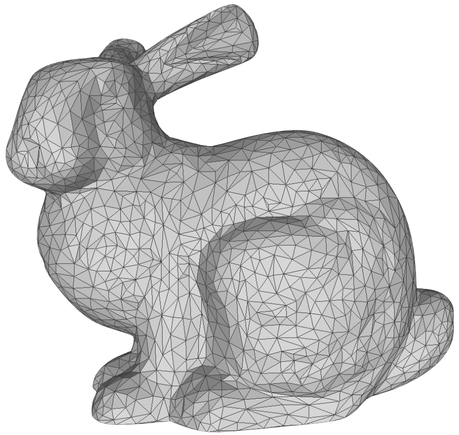


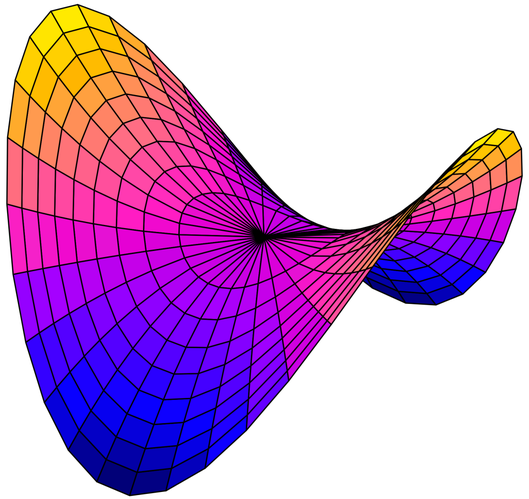
Surface Representations



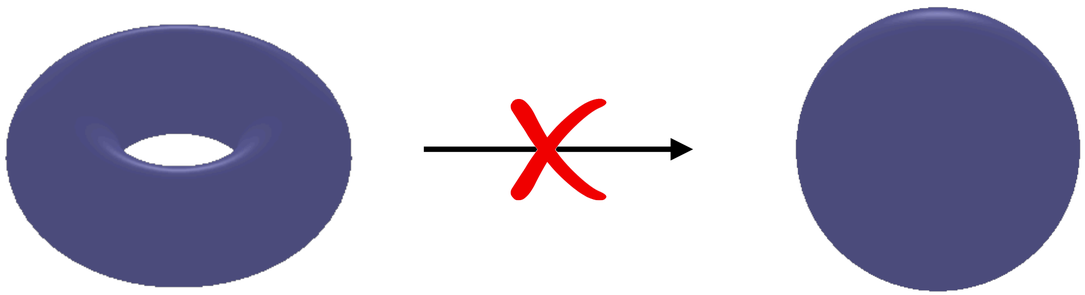
Slide credit: CMU 16-889: Learning for 3D Vision, Prof. Shubham Tulsiani





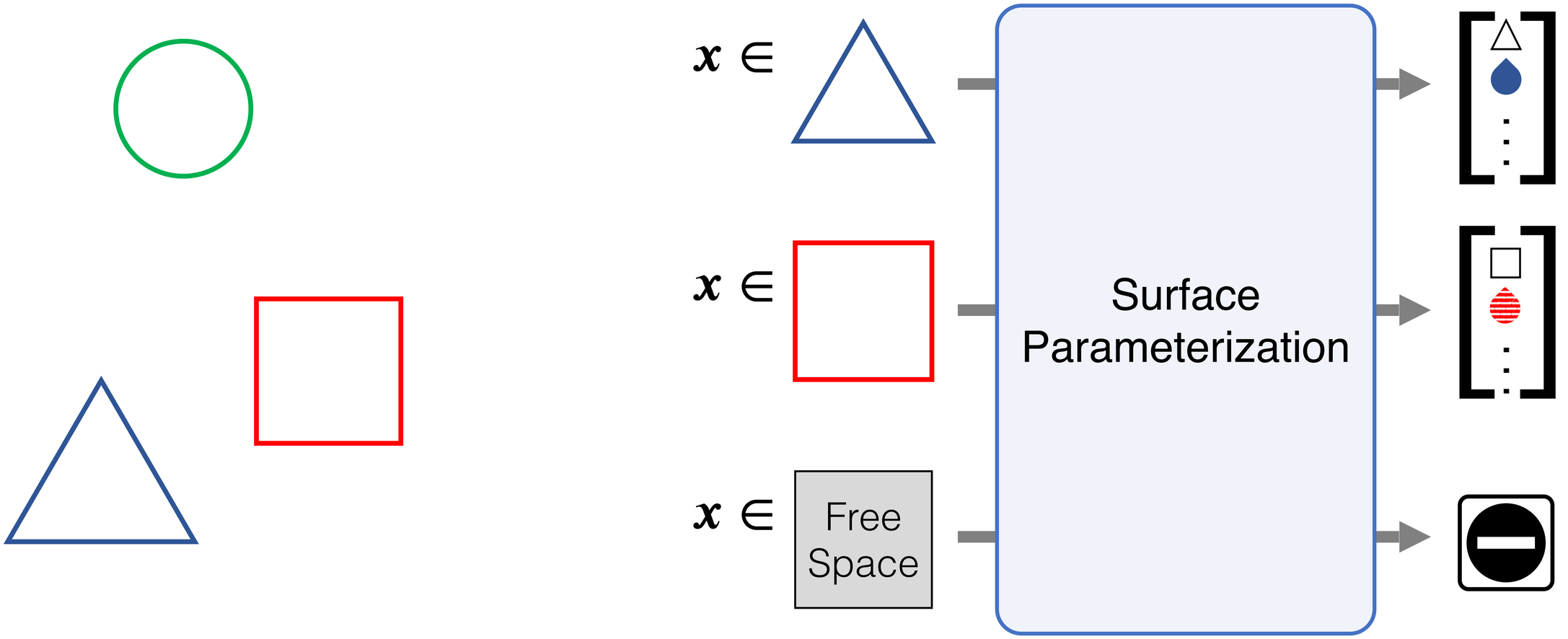


Surface Representations

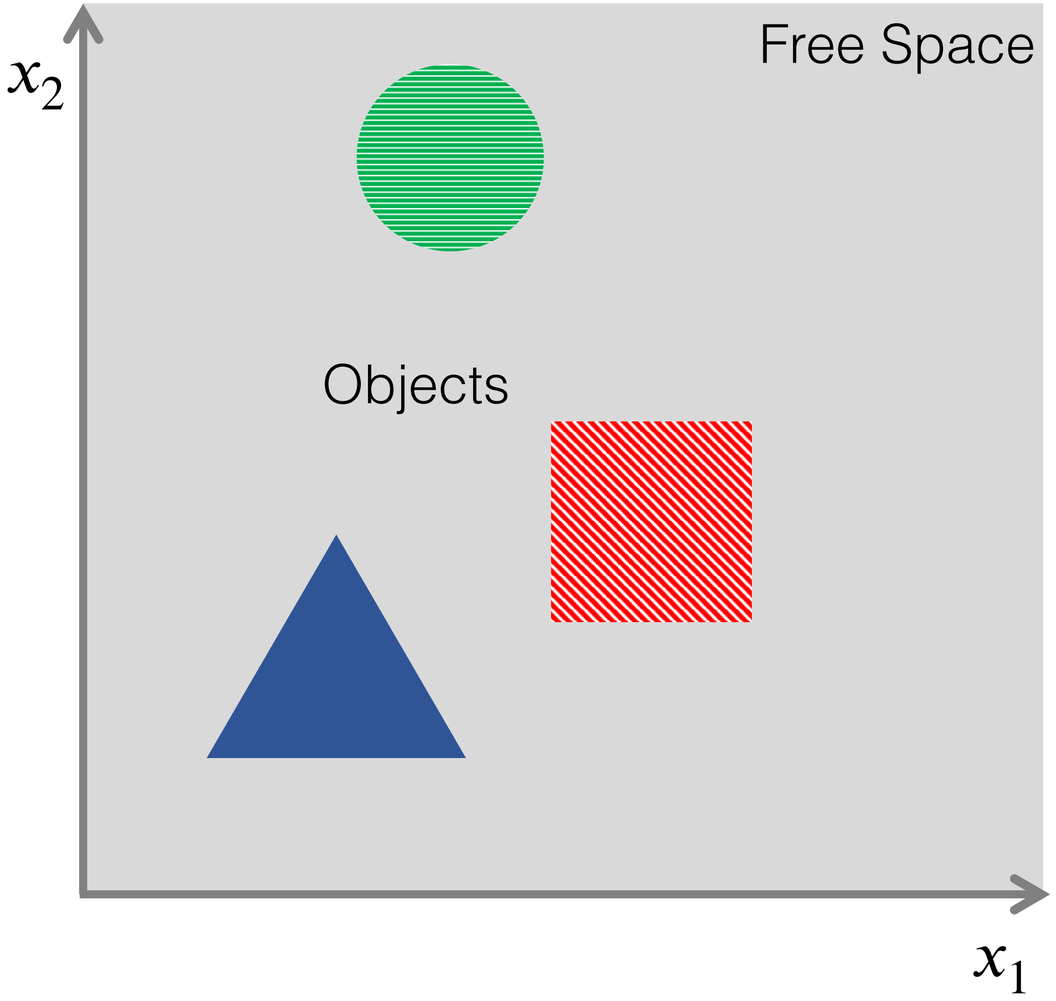


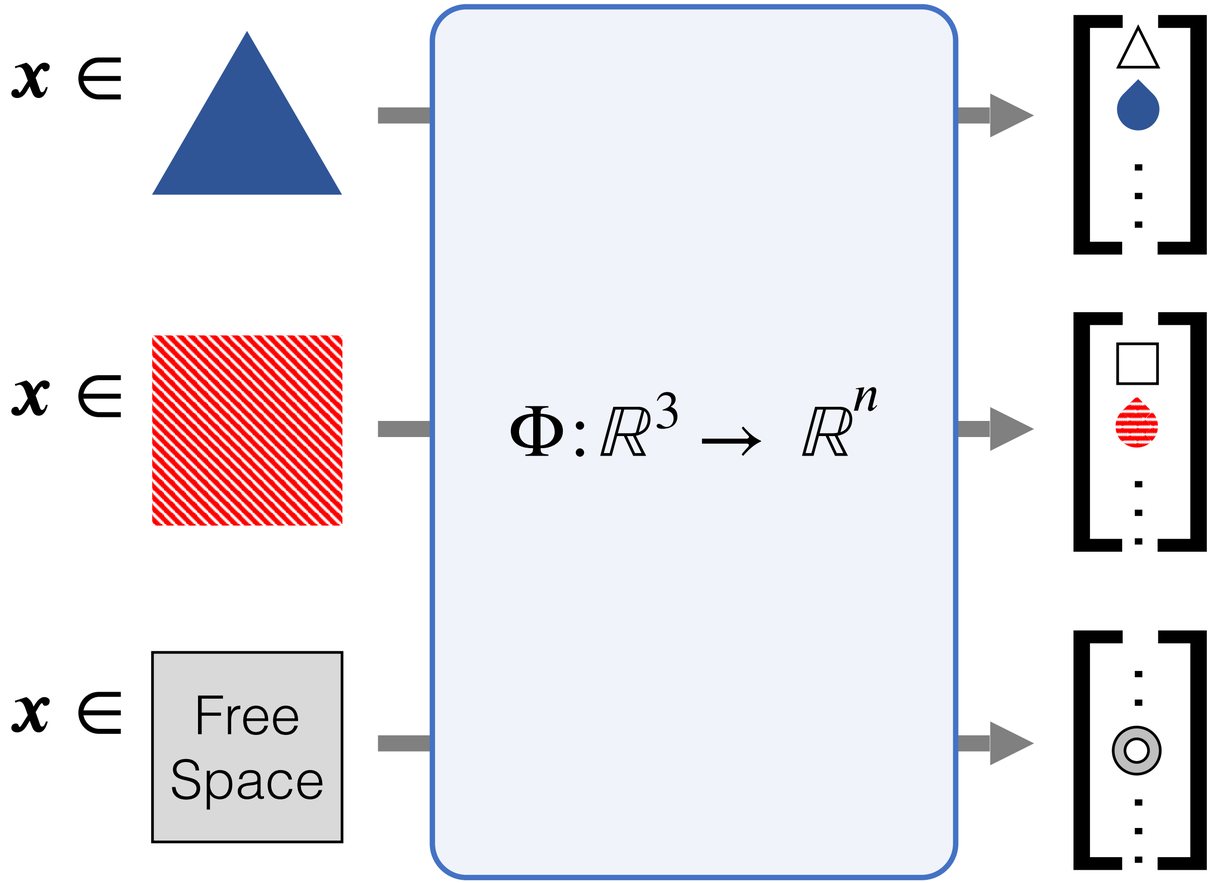
Slide credit: CMU 16-889: Learning for 3D Vision, Prof. Shubham Tulsiani

Surface representations only parameterize stuff on surfaces of things.

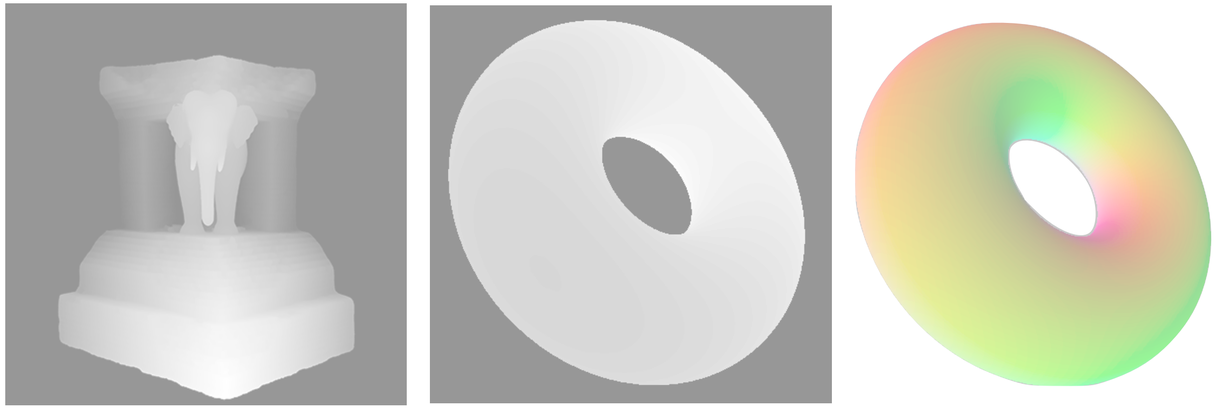


What if we wanted to parameterize stuff **everywhere?**

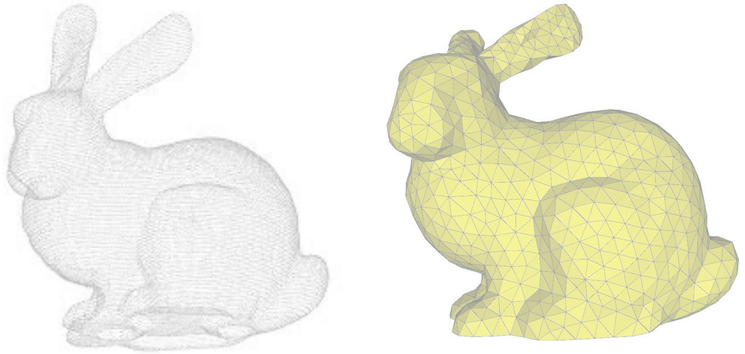




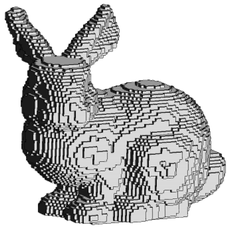
2.5D Representations

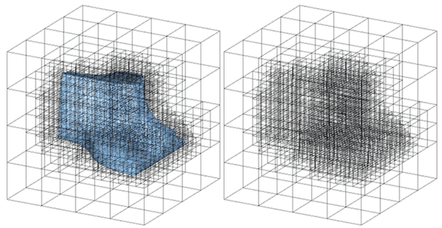


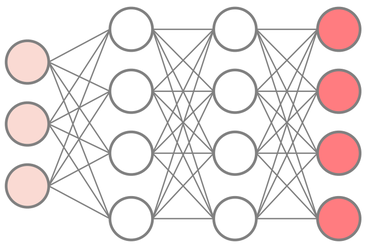
Surface Representations

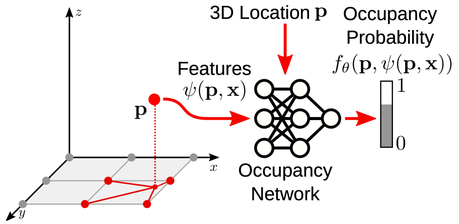


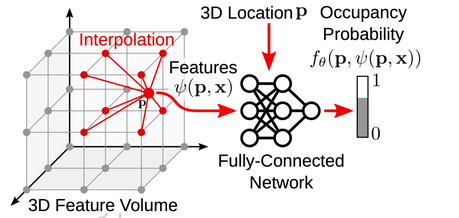
Field Parameterizations











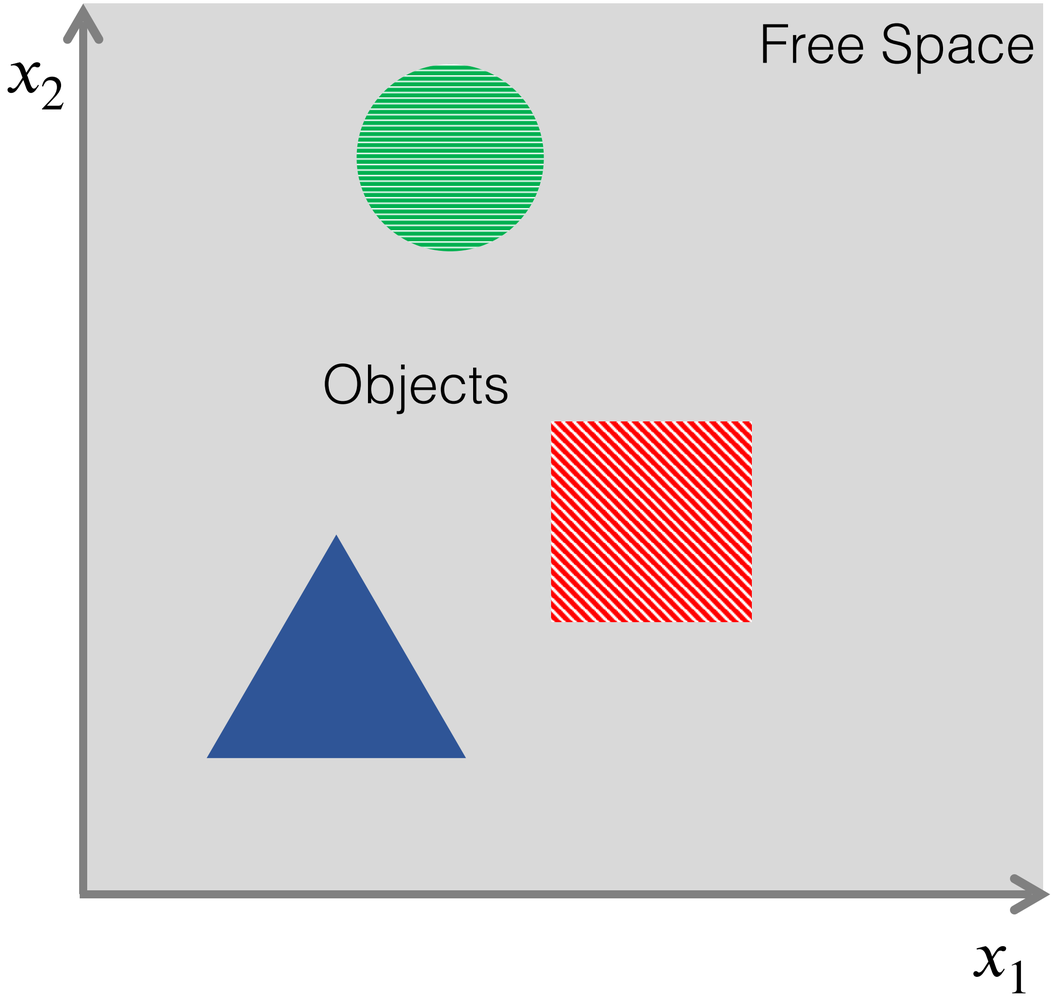
Definition: Field (physics)

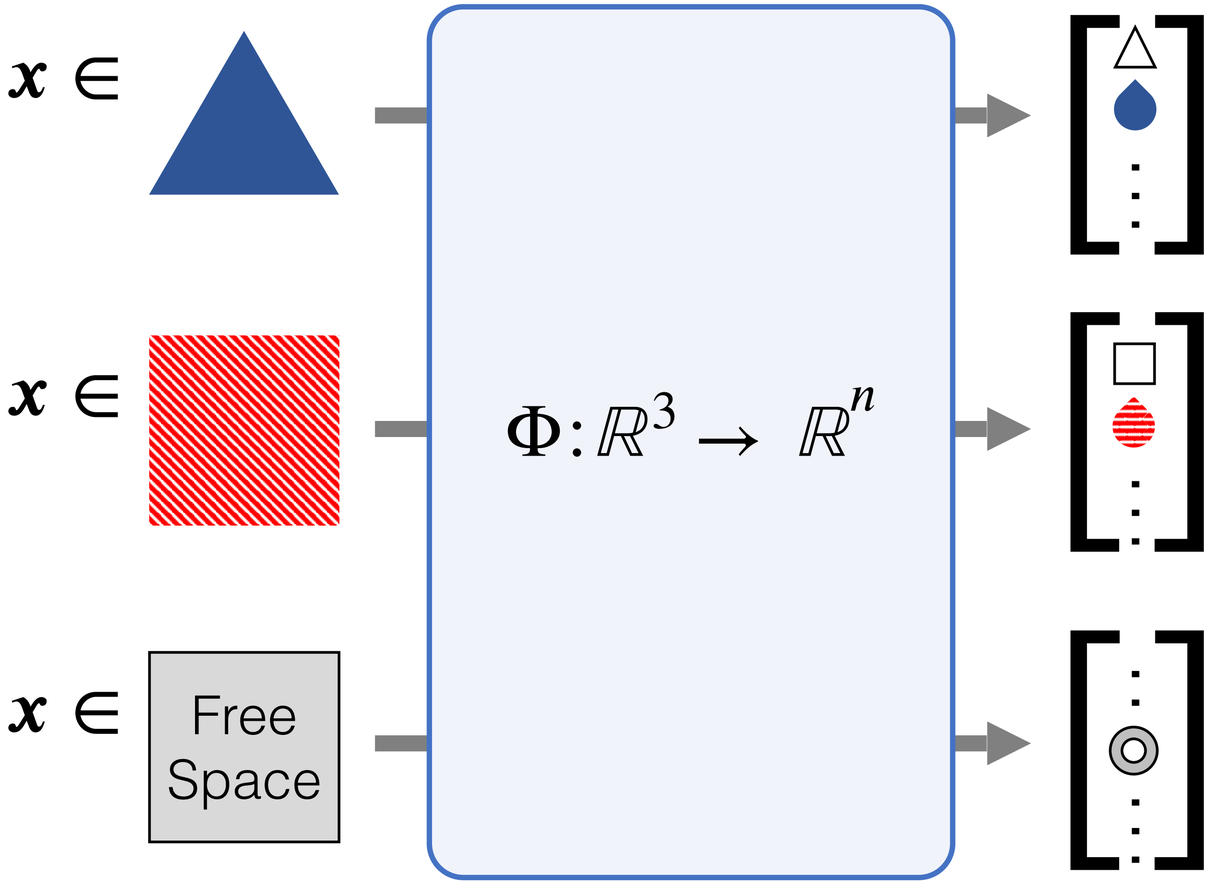
> A physical quantity represented by a scalar, vector, or tensor, that has a value for each point in space and time.

Will use term “field” for any function that takes a space, time, or space-time coordinate as input to map it to some quantity at that coordinate.

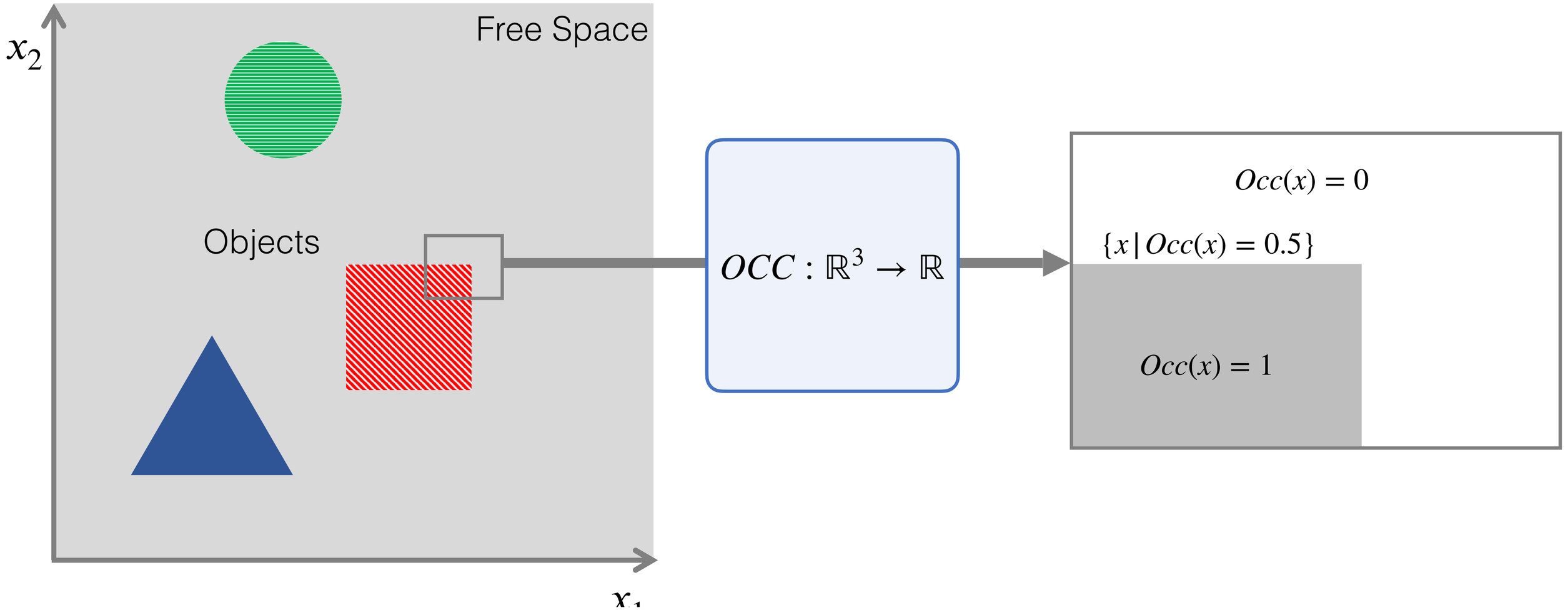
$$\Phi : \mathbb{R}^3 \rightarrow \mathbb{R}^n$$

How to parameterize surfaces?



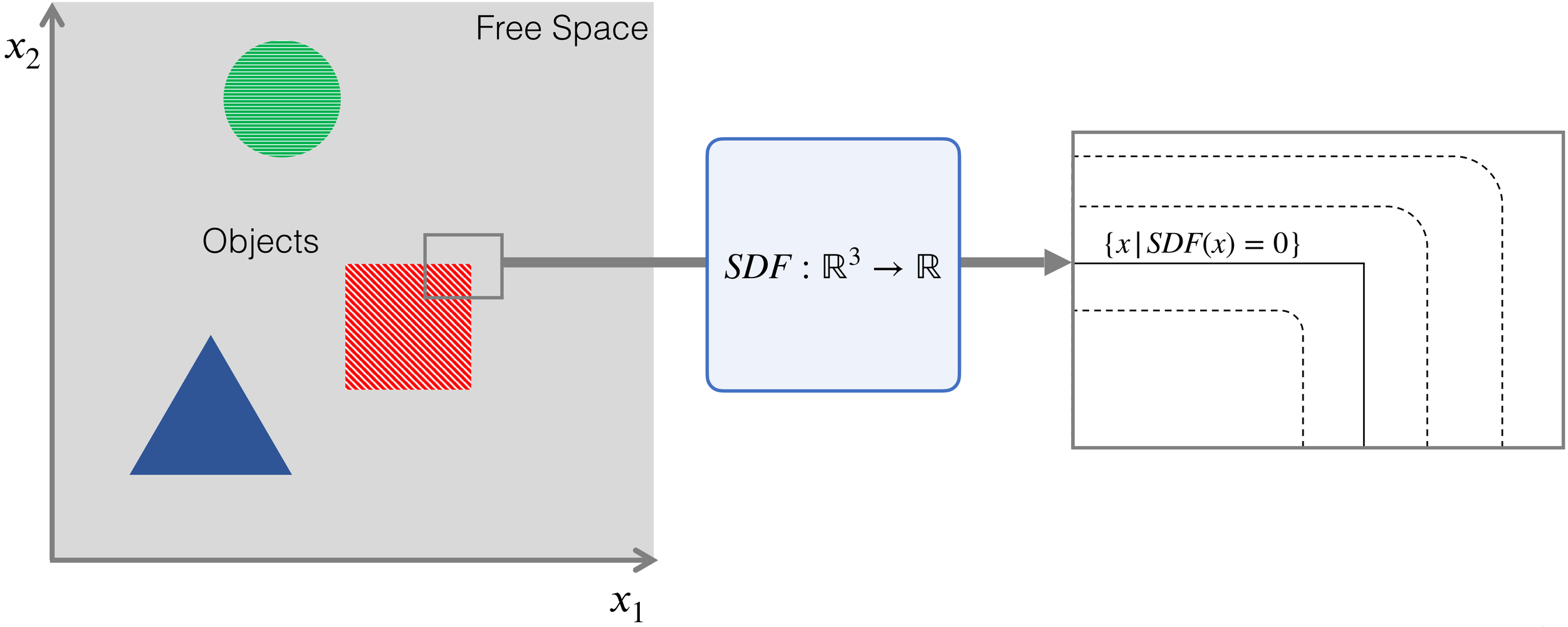


The Occupancy Field



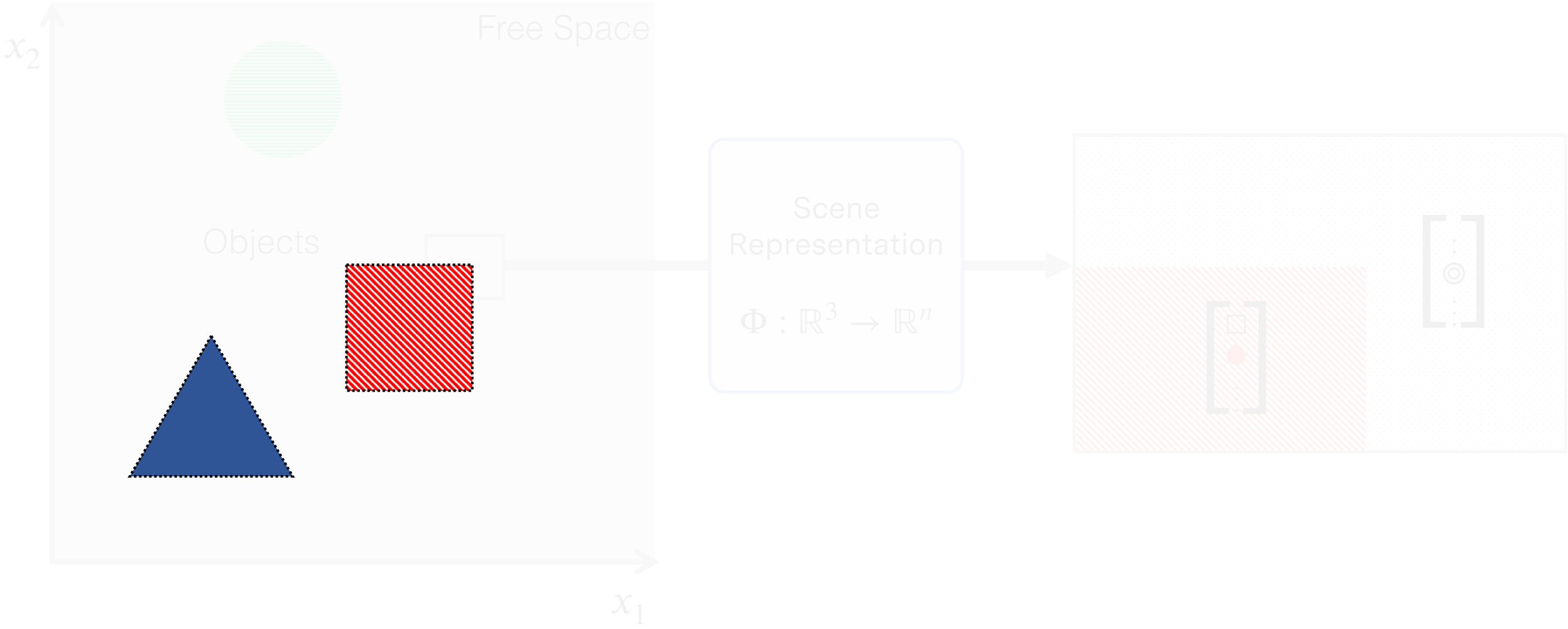
Occupancy Networks: Learning 3D Reconstruction in Function Space, Mescheder et al.
IM-Net: Learning Implicit Fields for Generative Shape Modeling, Chen et al.

The Signed Distance Field



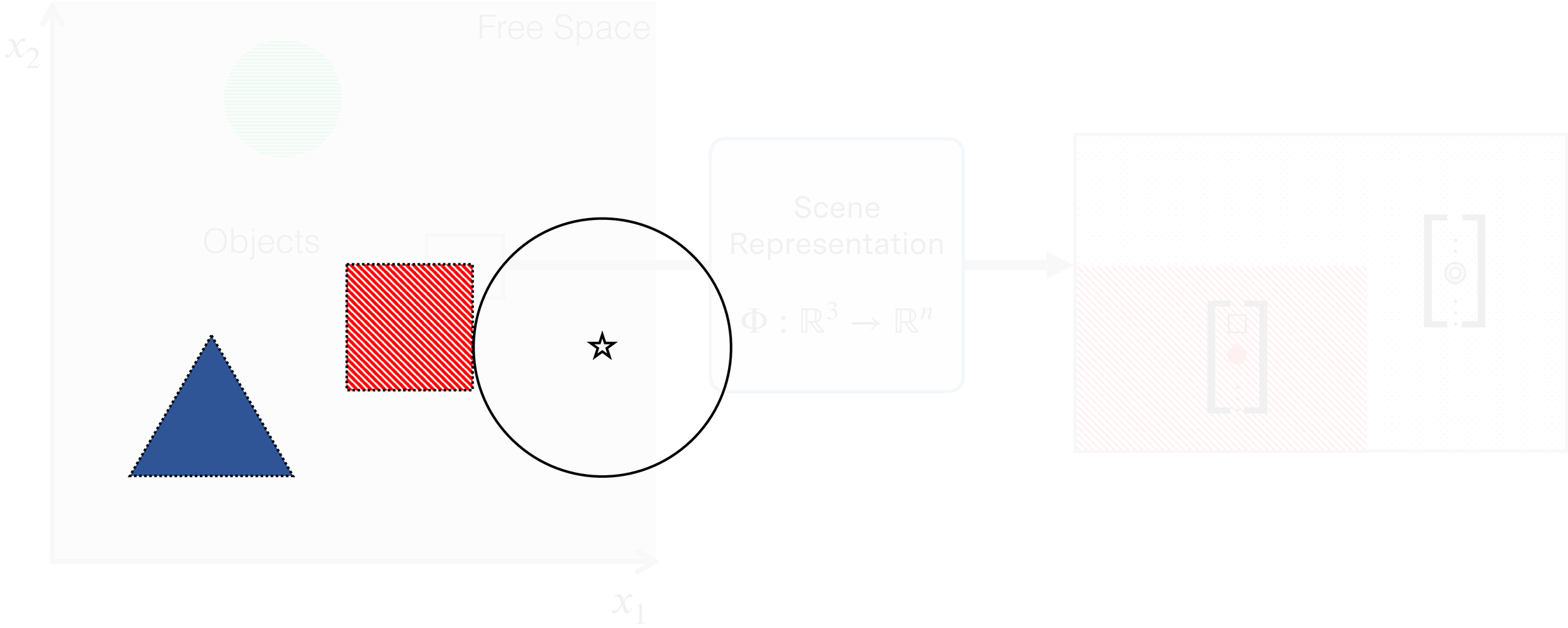
DeepSDF: Learning Continuous Signed Distance Functions for Shape Representation, Park et al. 2019

The Signed Distance Field



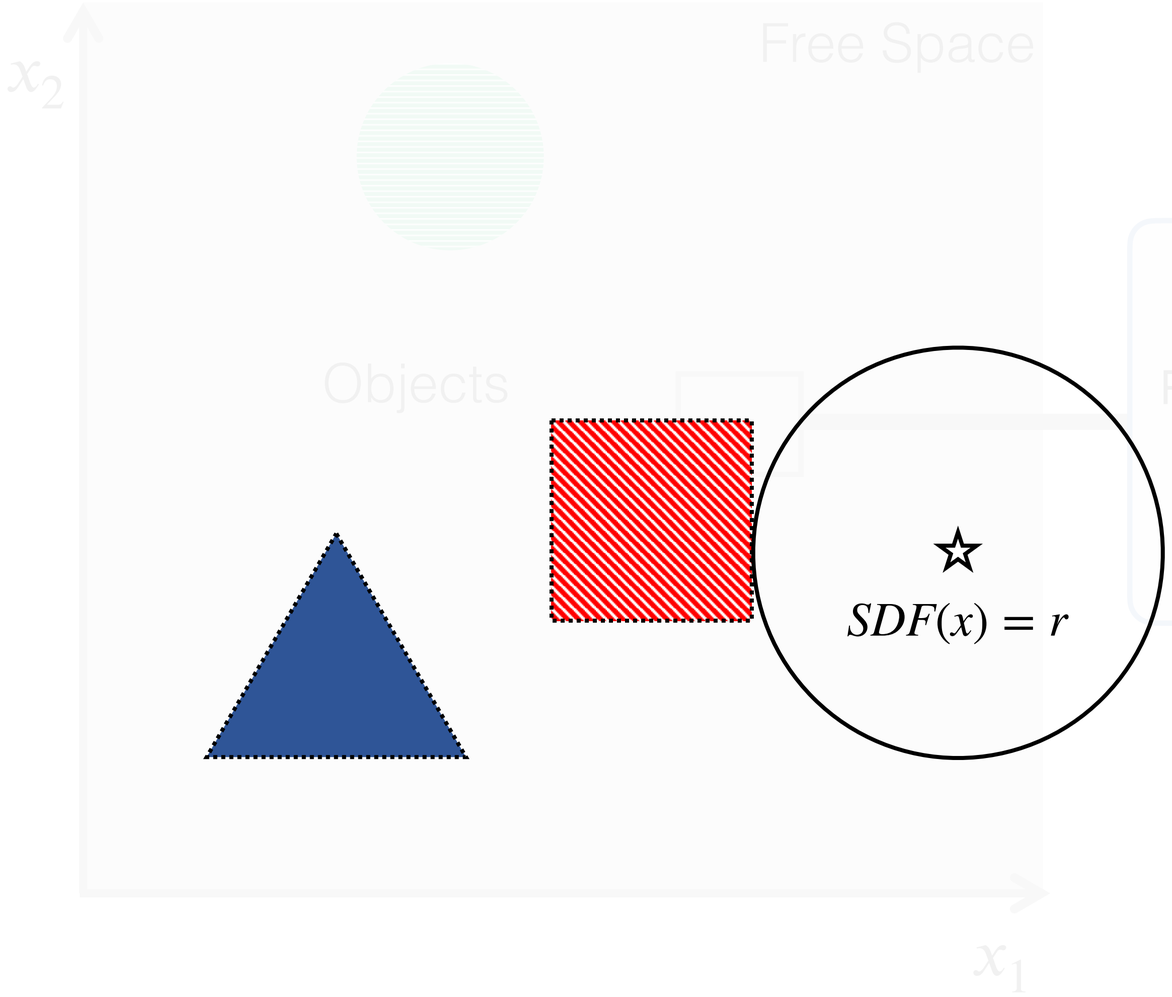
DeepSDF: Learning Continuous Signed Distance Functions for Shape Representation, Park et al. 2019

The Signed Distance Field



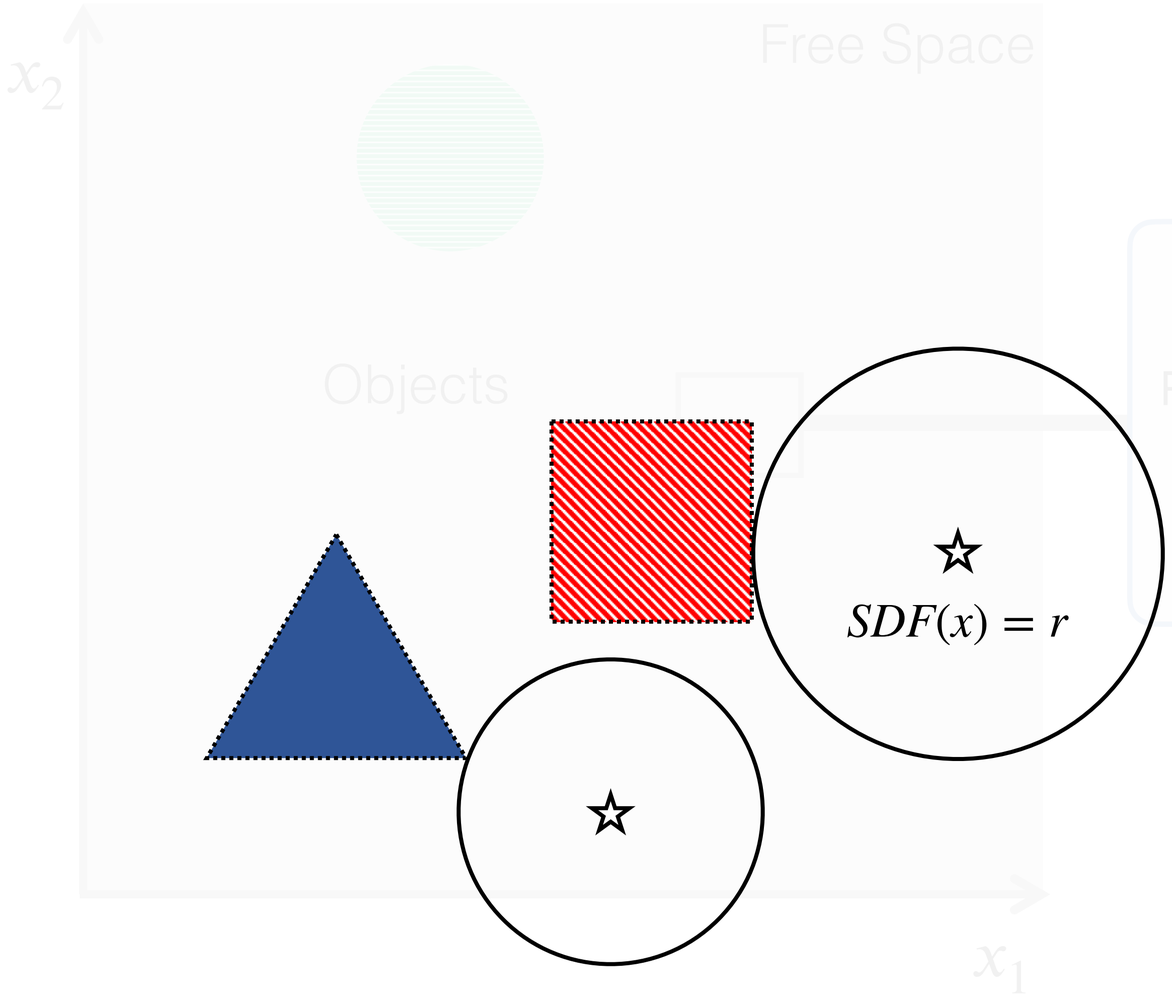
DeepSDF: Learning Continuous Signed Distance Functions for Shape Representation, Park et al. 2019

The Signed Distance Field



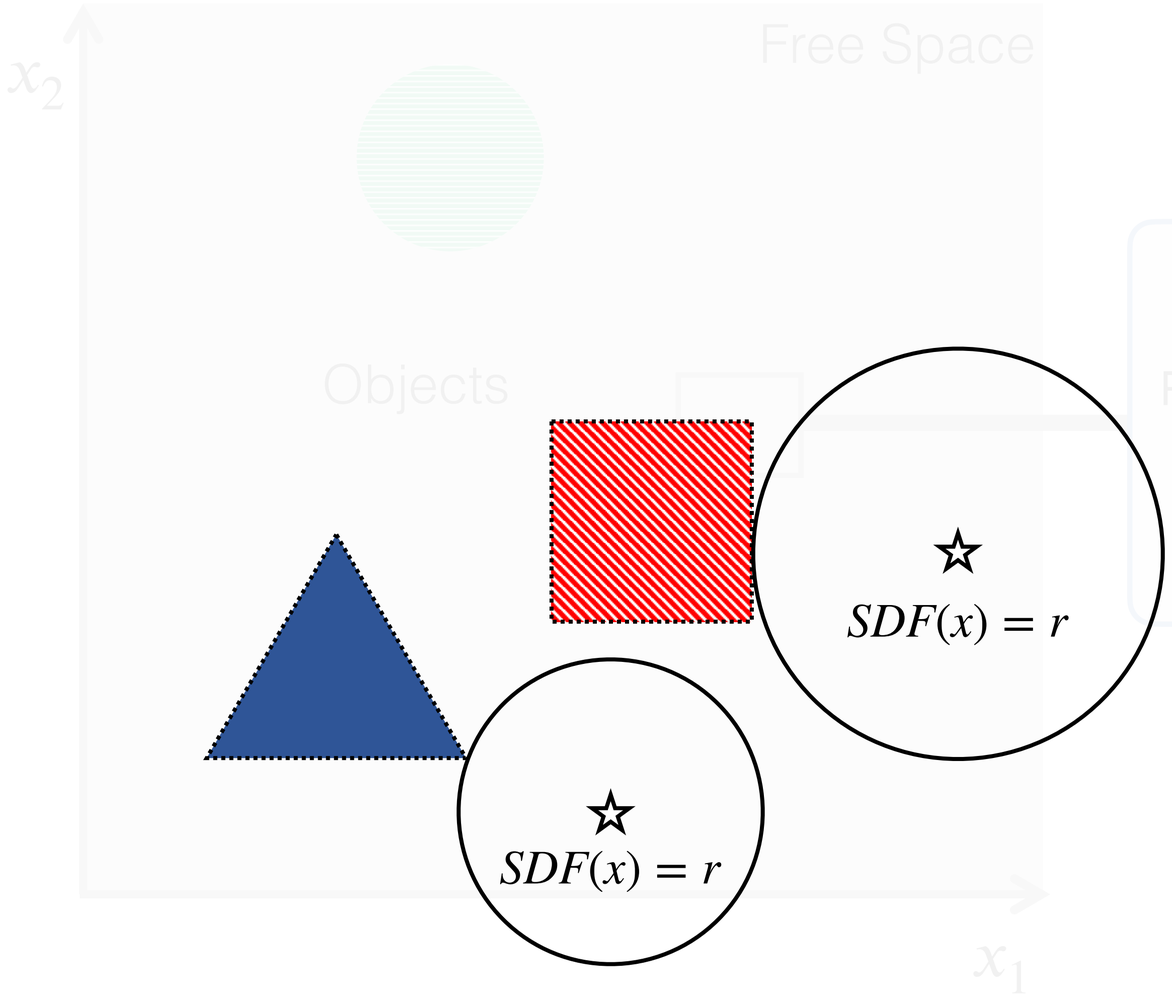
DeepSDF: Learning Continuous Signed Distance Functions for Shape Representation, Park et al. 2019

The Signed Distance Field



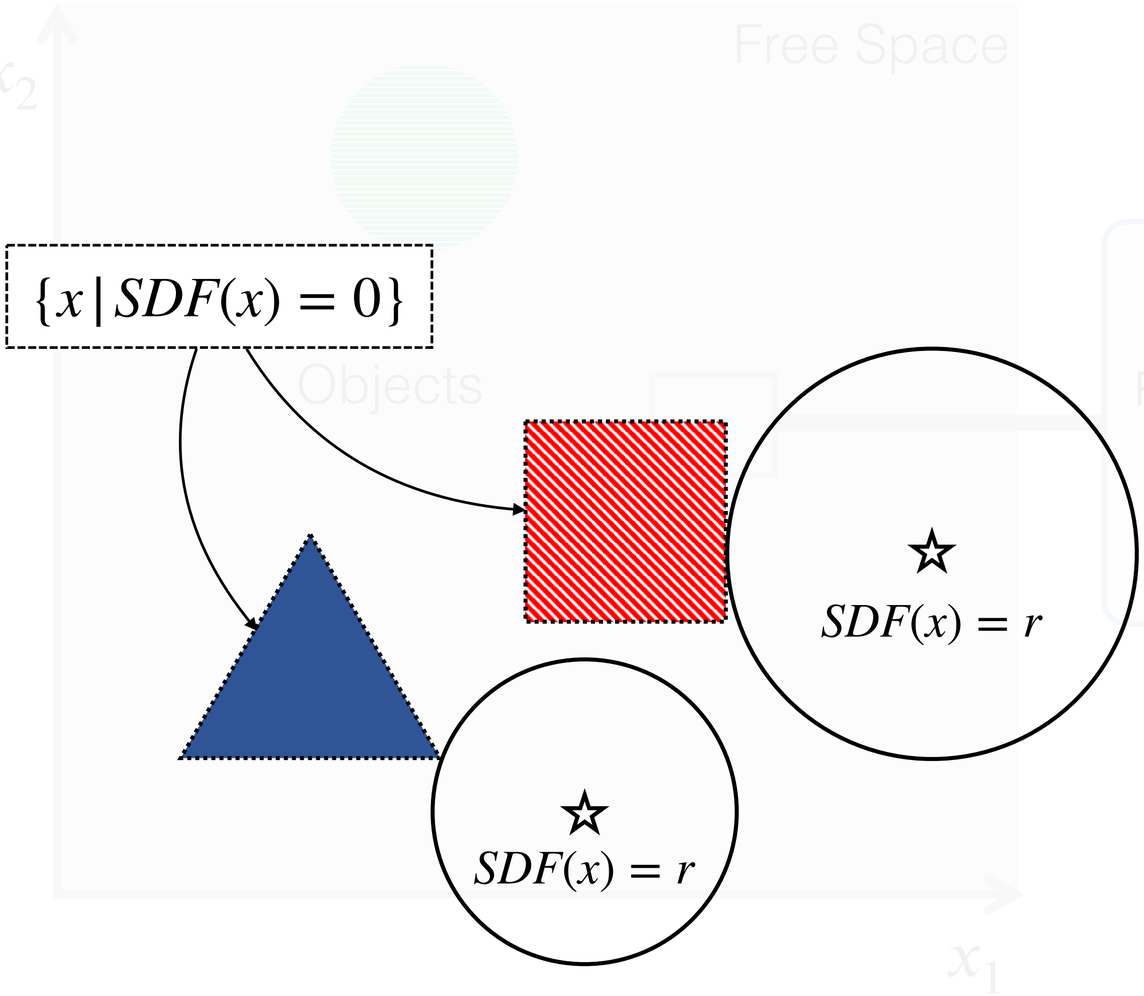
DeepSDF: Learning Continuous Signed Distance Functions for Shape Representation, Park et al. 2019

The Signed Distance Field



DeepSDF: Learning Continuous Signed Distance Functions for Shape Representation, Park et al. 2019

The Signed Distance Field



DeepSDF: Learning Continuous Signed Distance Functions for Shape Representation, Park et al. 2019

Implicit Surfaces

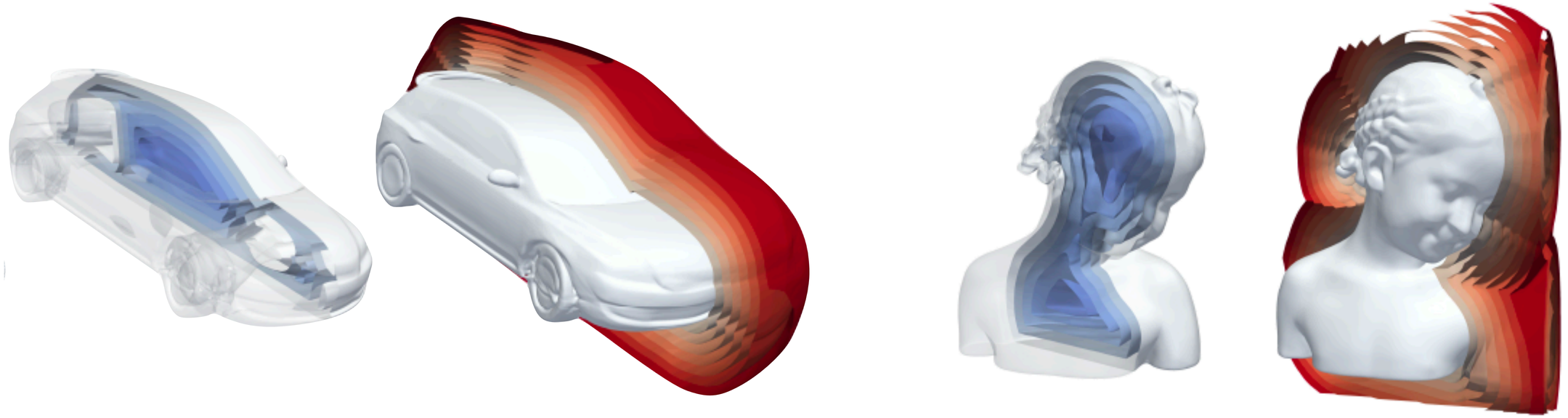
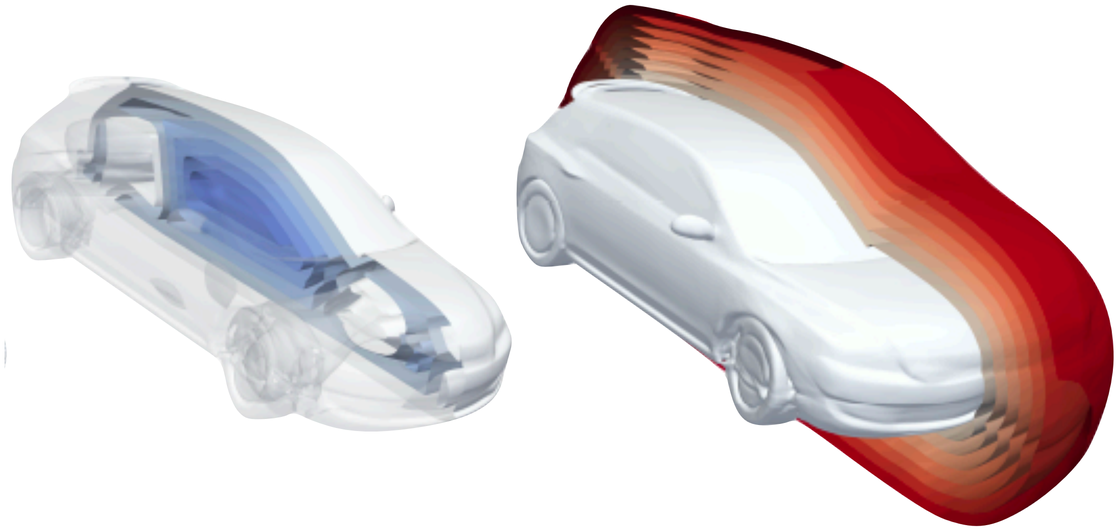


Image credits: IGR: Implicit Geometric Regularization for Learning Shapes. *Gropp et. al*

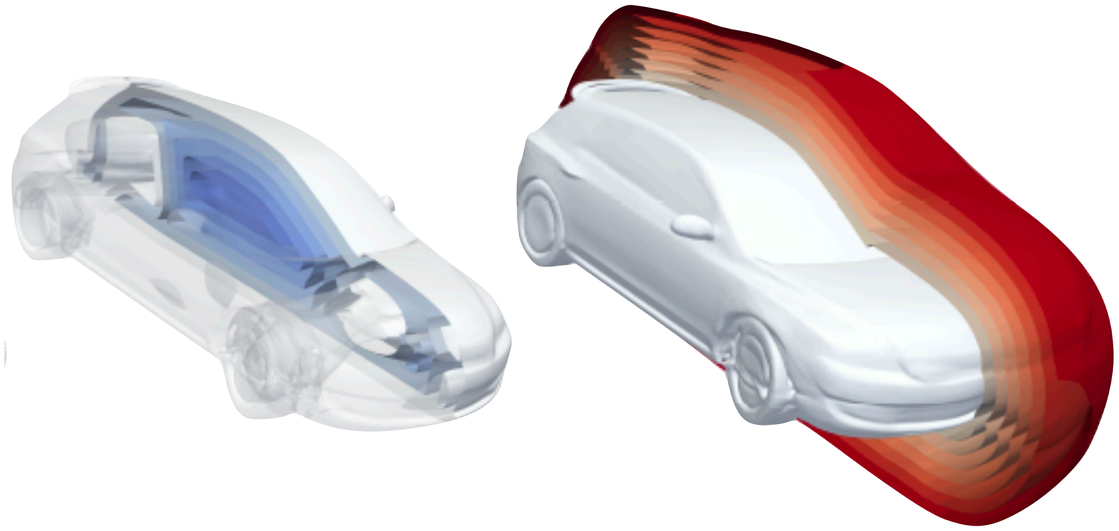
Implicit Surfaces



Level sets of Occupancy, SDF parameterize surface

Image credits: IGR: Implicit Geometric Regularization for Learning Shapes. *Gropp et. al*

Implicit Surfaces

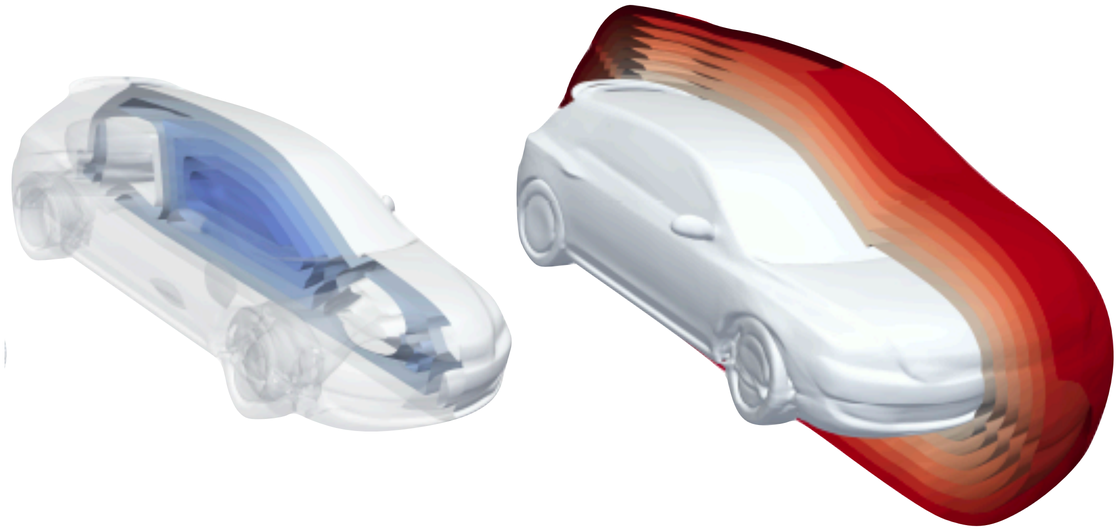


Level sets of Occupancy, SDF parameterize surface

Easy to answer questions about occupancy

Image credits: IGR: Implicit Geometric Regularization for Learning Shapes. *Gropp et. al*

Implicit Surfaces



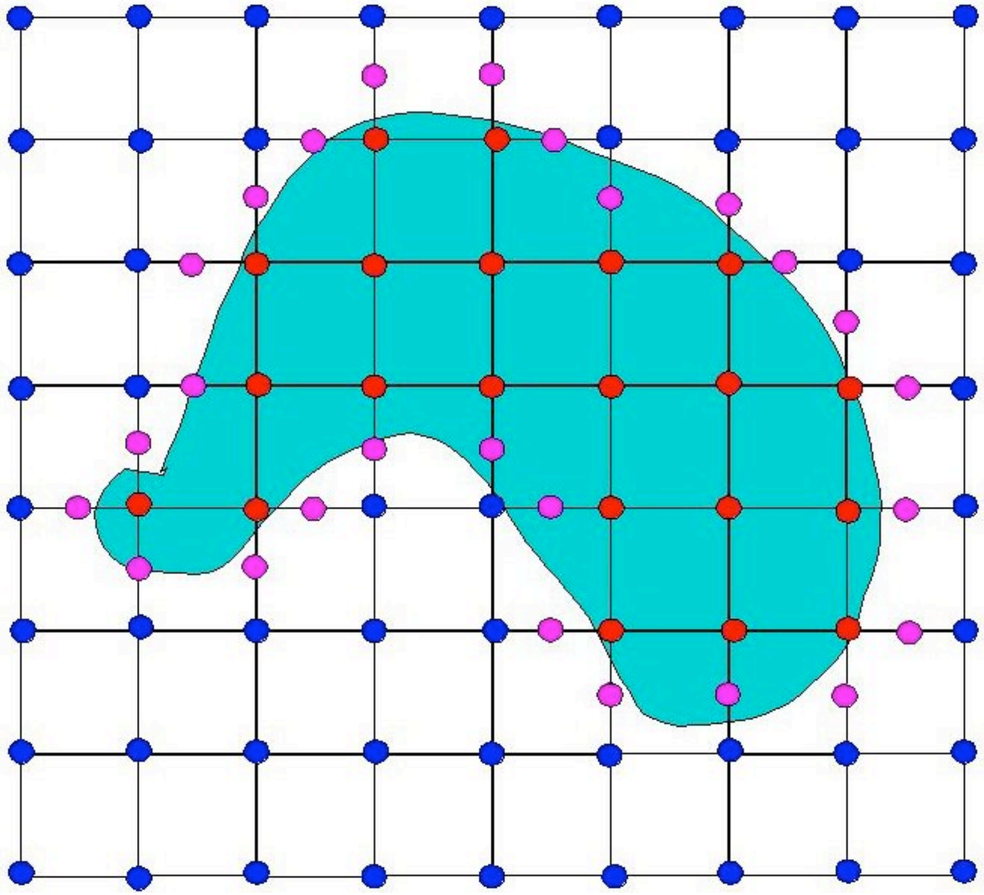
Level sets of Occupancy, SDF parameterize surface

Easy to answer questions about occupancy

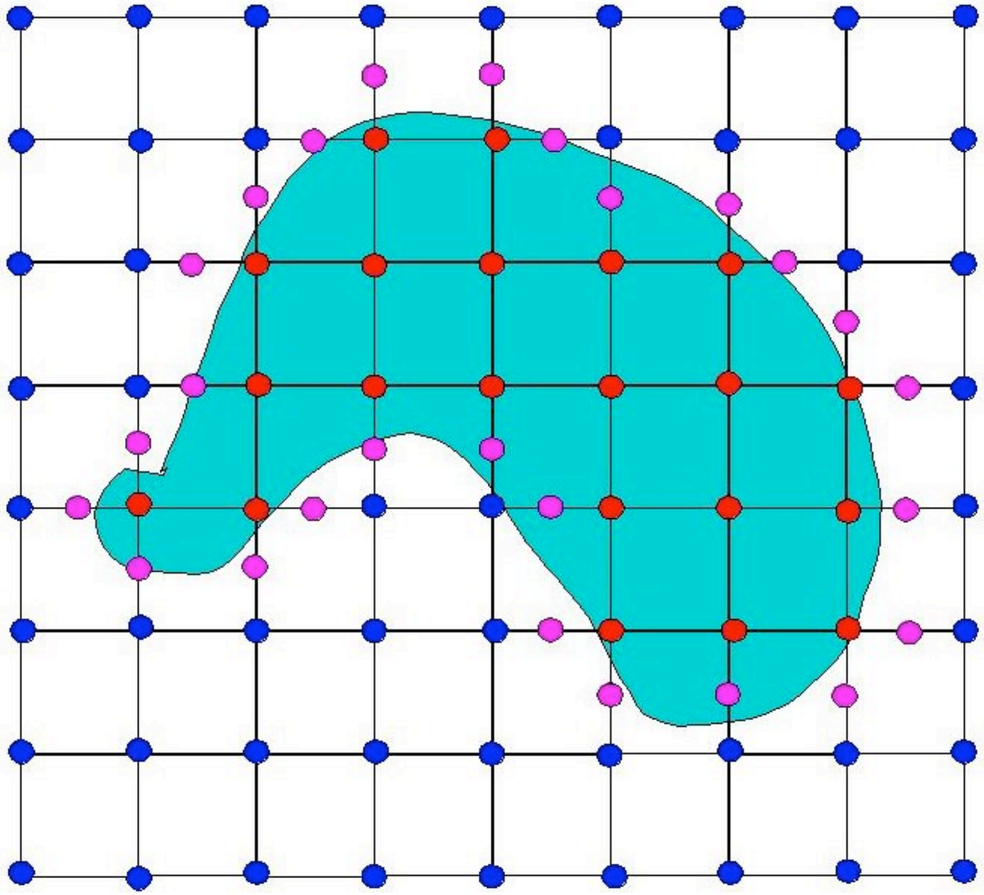
Difficult to reason about the surface: have to first **find** zero-level-set!

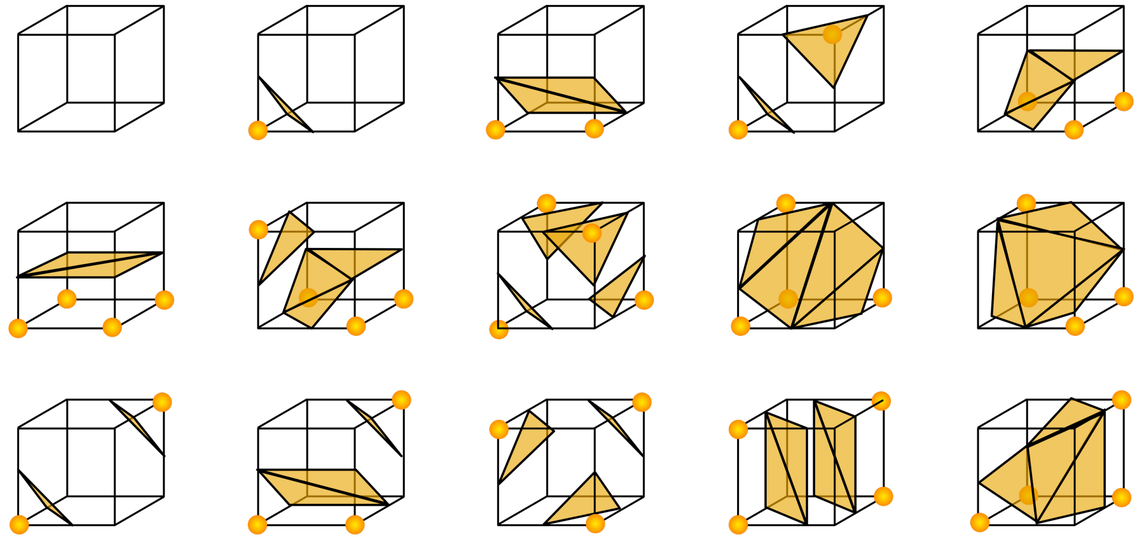
Image credits: IGR: Implicit Geometric Regularization for Learning Shapes. *Gropp et. al*

Marching Cubes: Extracting 2D Mesh from 3D Implicit Representation

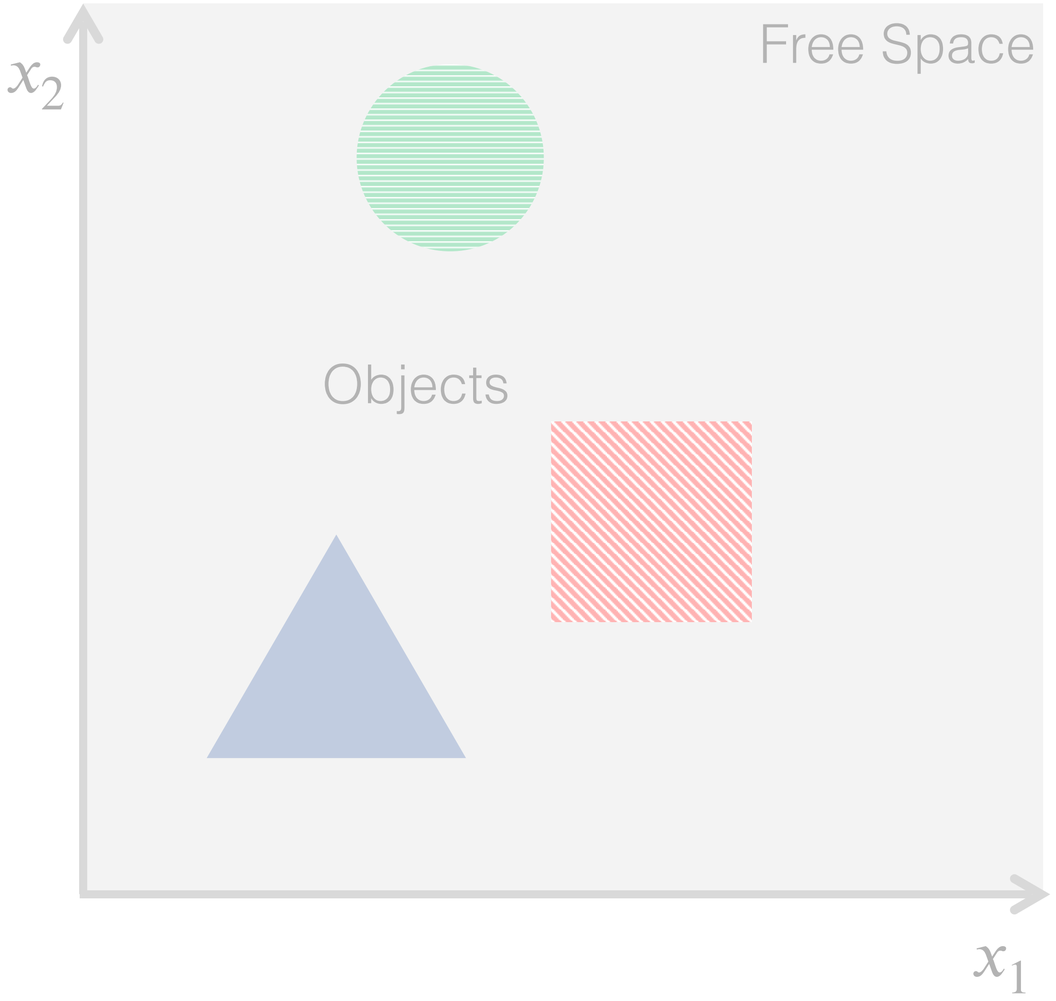


Marching Cubes: Extracting 2D Mesh from 3D Implicit Representation

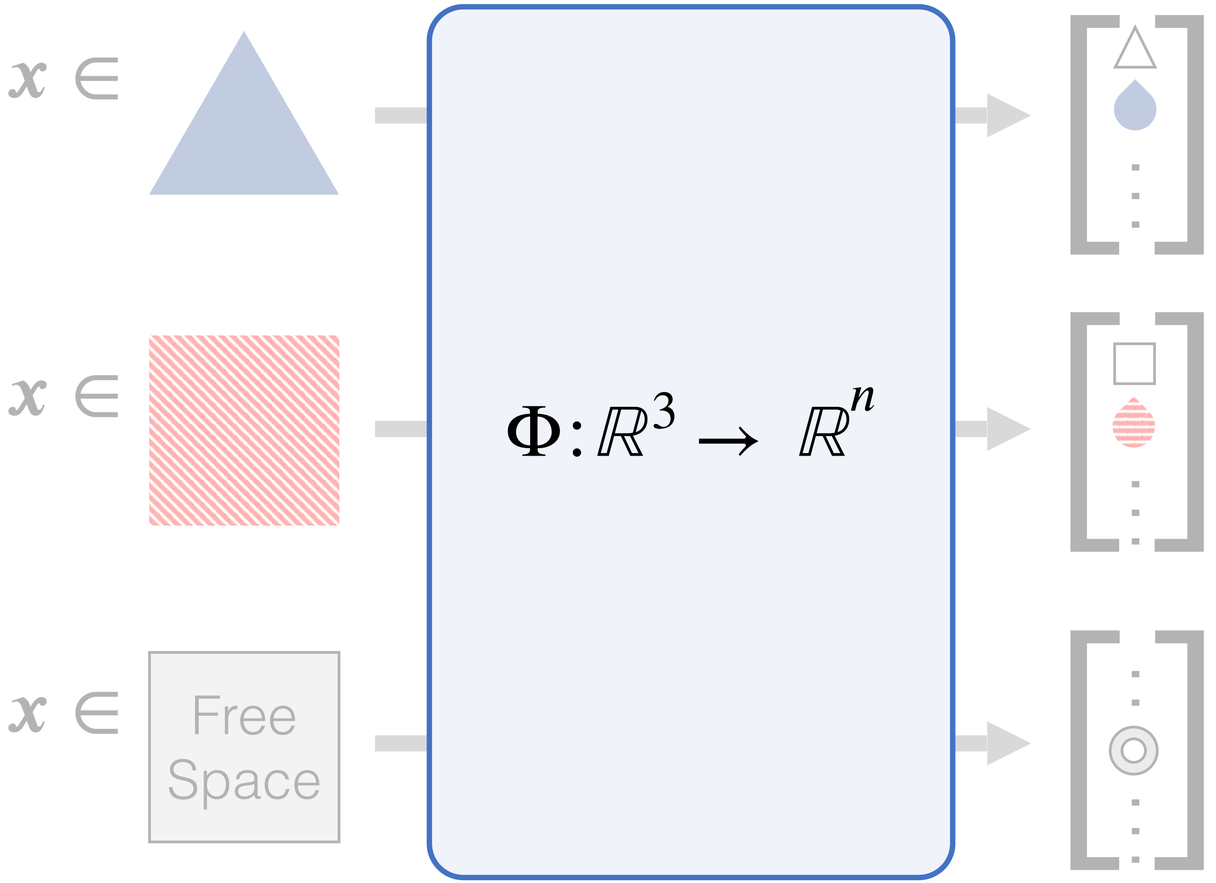




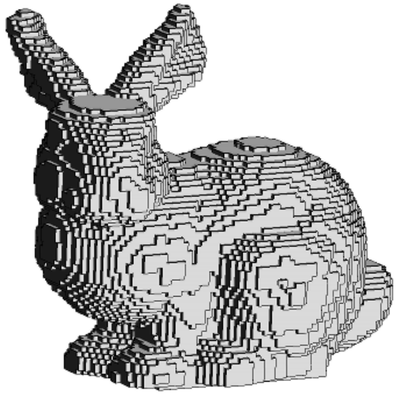
How to parameterize surfaces?

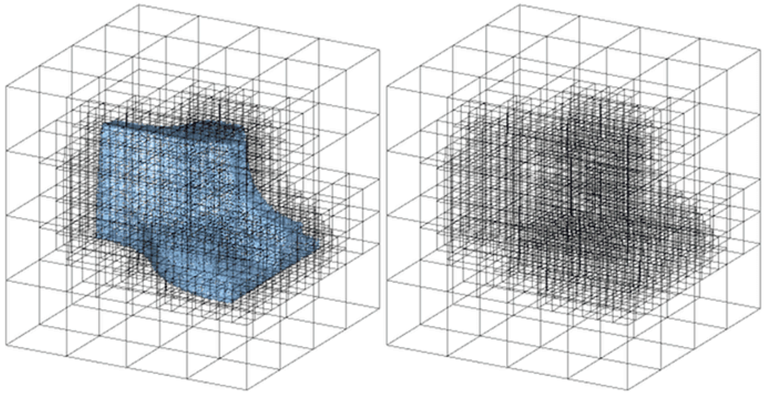


How to parameterize 3D function?

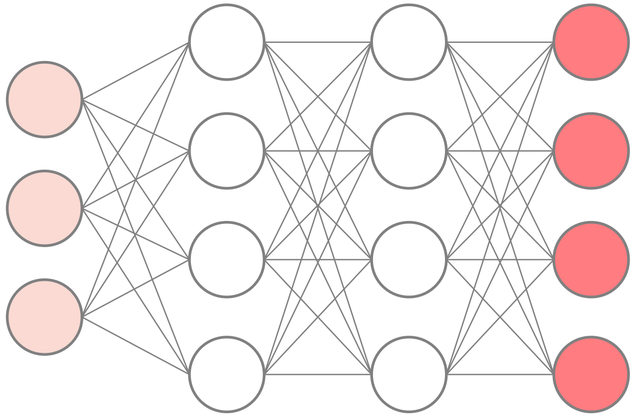


Parameterizing Fields

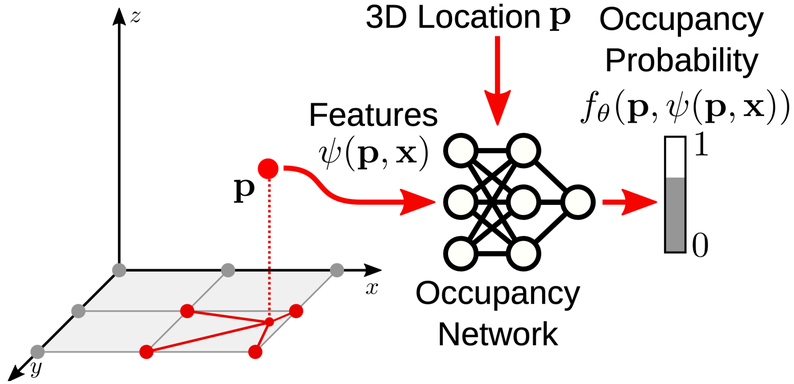


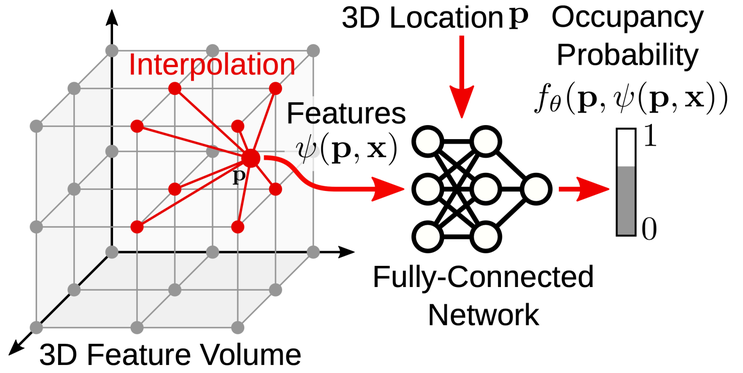


Discrete Parameterizations



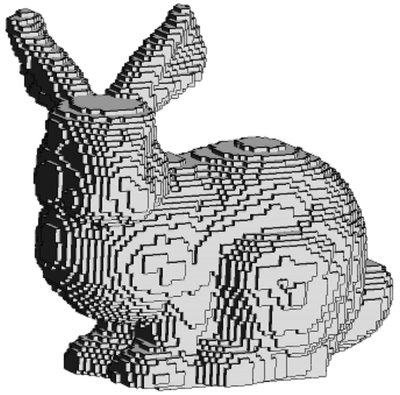
Continuous Parameterizations

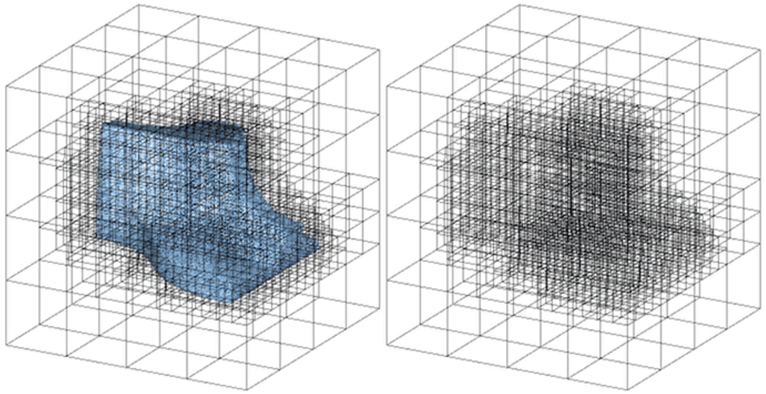


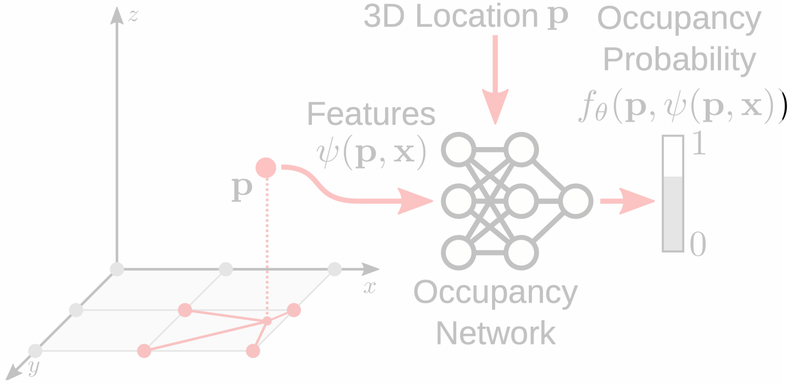


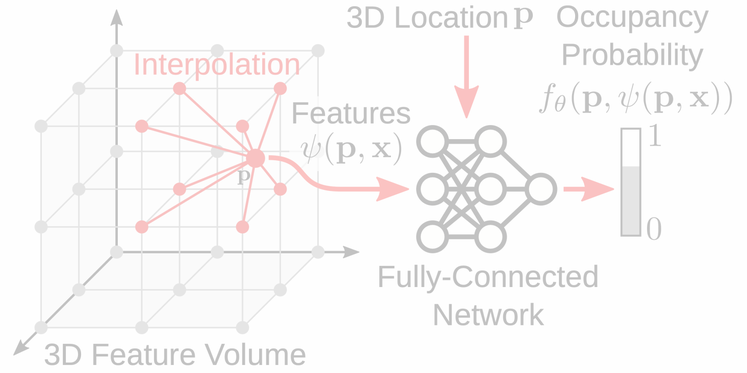
Hybrid Parameterizations

Voxel Grids







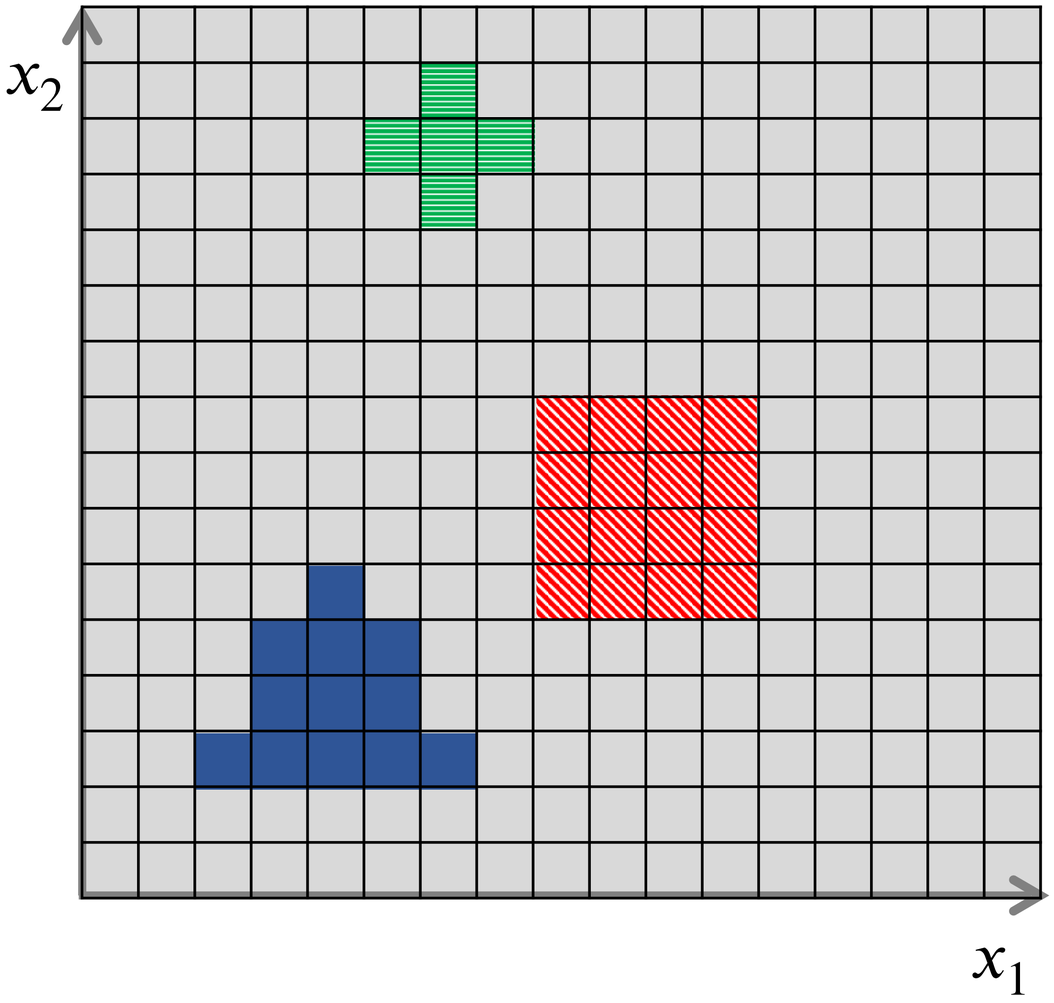


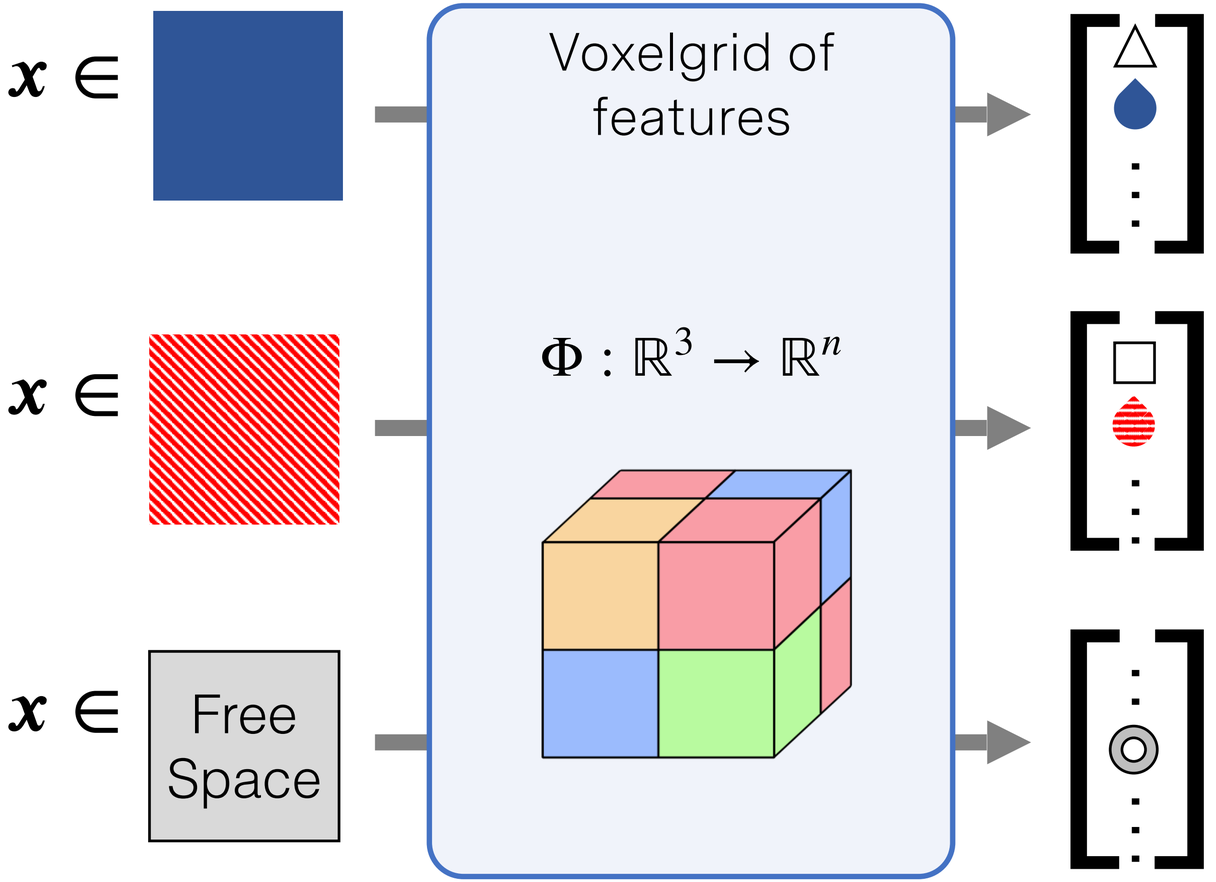
Discrete Parameterizations

Continuous Parameterizations

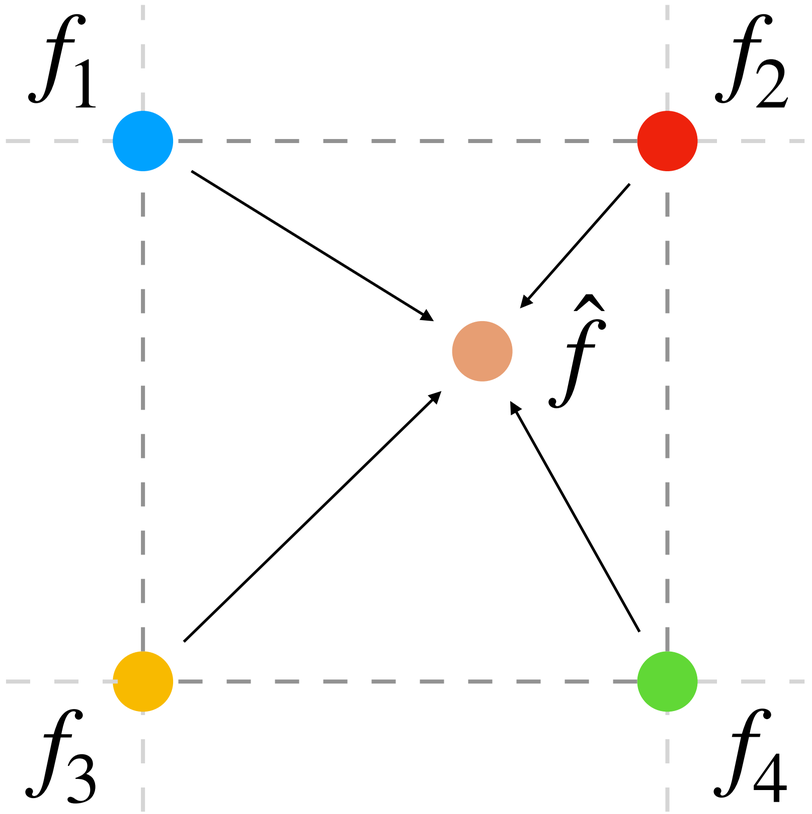
Hybrid Parameterizations

Voxel Grids


x_2
x_1


$x \in$
$x \in$
$x \in$
Free Space
Voxelgrid of features
$\Phi : \mathbb{R}^3 \rightarrow \mathbb{R}^n$

Interpolation: Querying *continuous* values



Interpolation in a 2D voxel grid.

- Only stores values at vertex locations, i.e. corners
- Values at intermediate coordinates are defined via interpolation w/ some kernel k

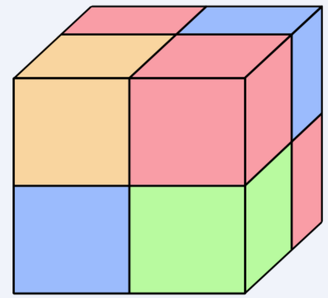
$$\hat{f}(\mathbf{x}) = \sum_{i=1}^{N} f_i k(\mathbf{x}, \mathbf{x}_i)$$

Voxel grids

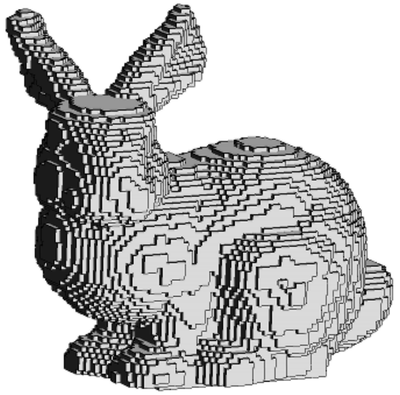
- For d spatial dimensions and resolution n, memory grows $O(n^d)$
- Fast sampling: d-linear interpolation
  = index into array 2^d times & weighted sum.
- Convenient processing as it exposes *locality.* Can easily run convolutions, nearest-neighbor lookups, etc.
- Intractable in higher dimensions :/

Voxelgrid of features

$\Phi : \mathbb{R}^3 \to \mathbb{R}^n$



Multi-Resolution Representations



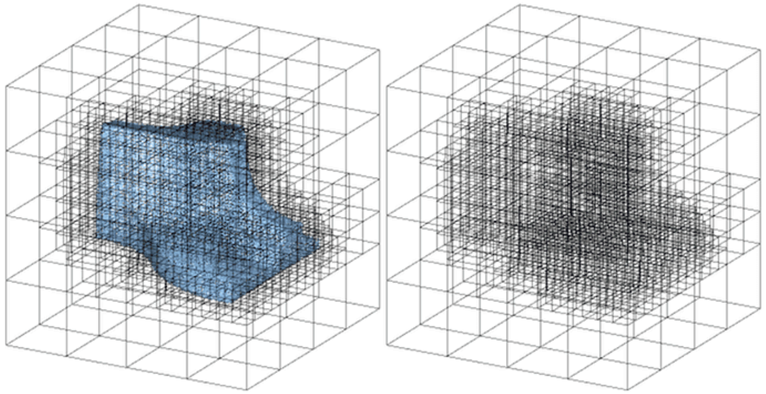
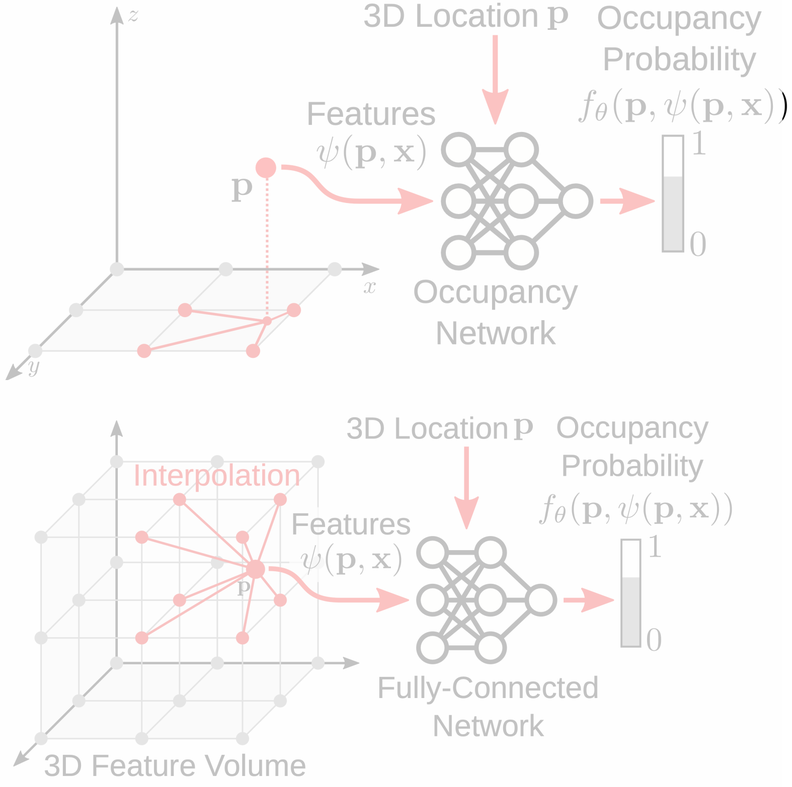


Image Source: https://doc.cgal.org/latest/Orthtree/index.html

Discrete Parameterizations

Continuous Parameterizations



Hybrid Parameterizations

Octrees

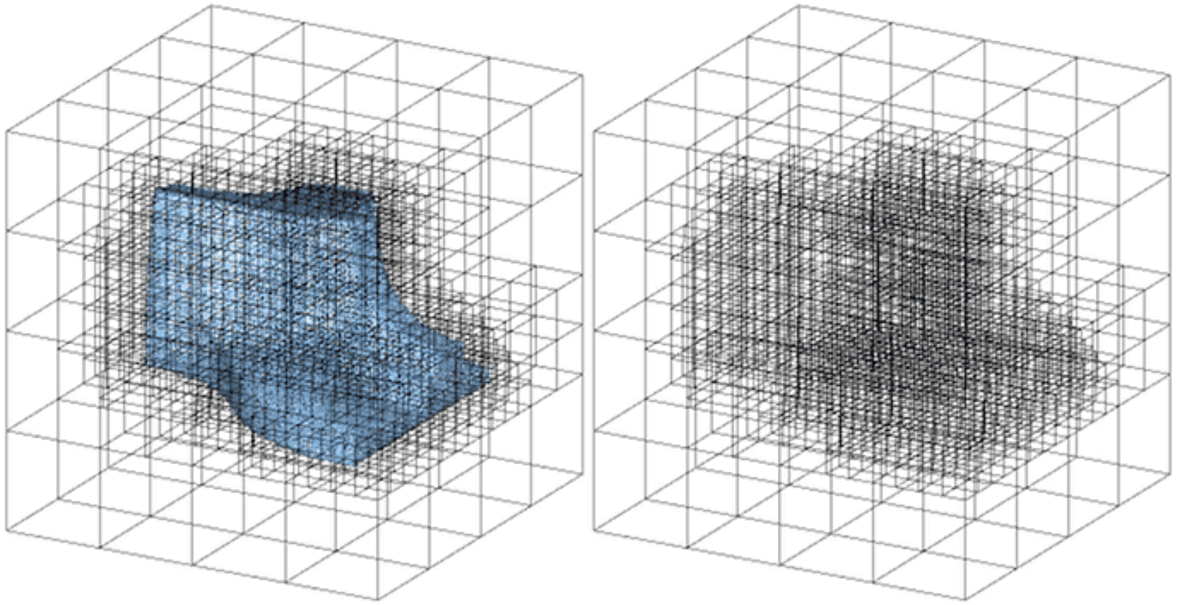


Image Source: https://doc.cgal.org/latest/Orthtree/index.html

Divide volume into 8 voxels

If any box has more than *N* points in it, divide it into 8 voxels

Do not divide voxels with zero points.

Octrees

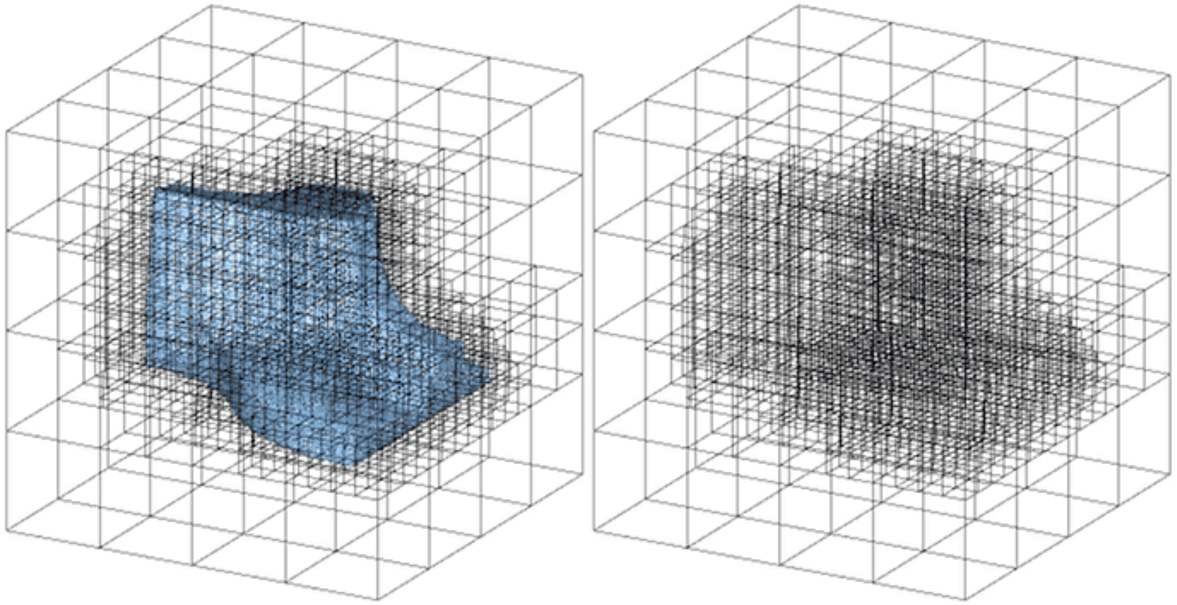


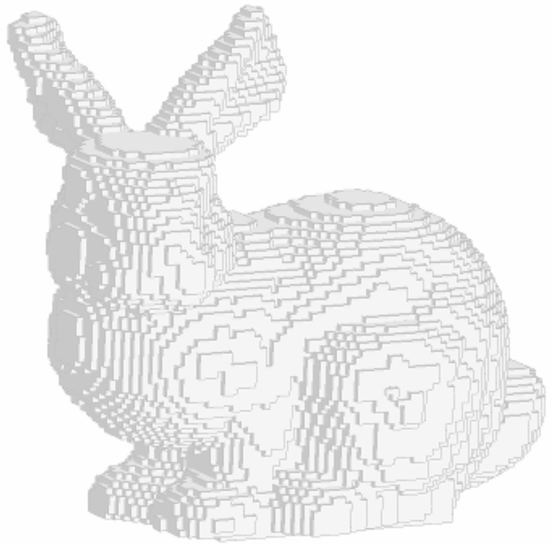
Image Source: https://doc.cgal.org/latest/Orthtree/index.html

Saves memory (less resolution in empty parts)

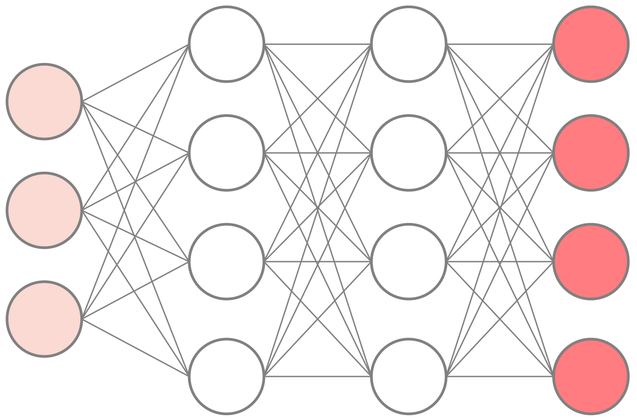
Sub-division is not differentiable: can’t easily build neural network that outputs Octree

To sample point, have to traverse multi-resolution hierarchy. Still fast, though.

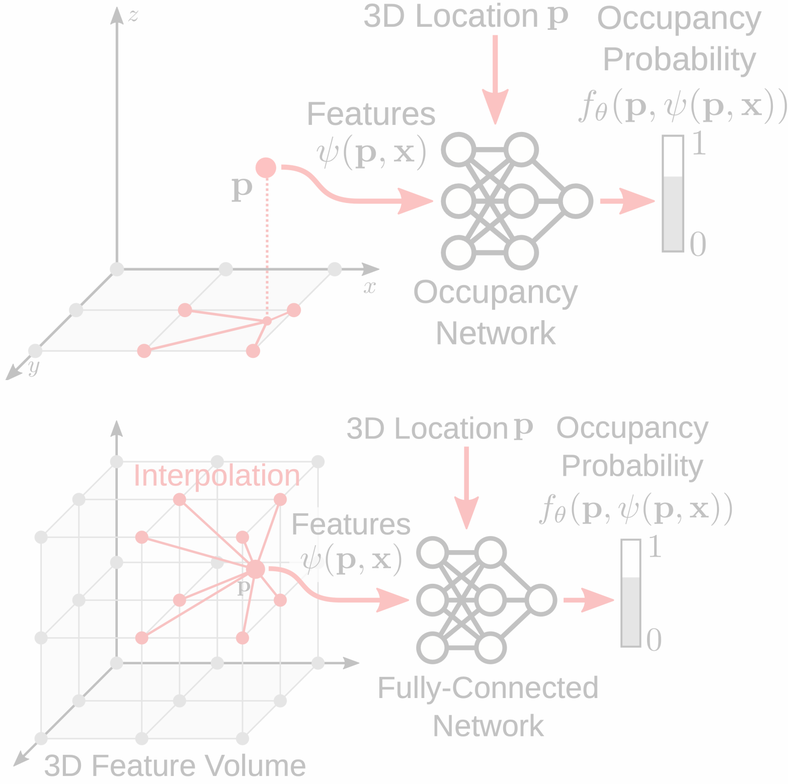
Neural Fields



Discrete Parameterizations

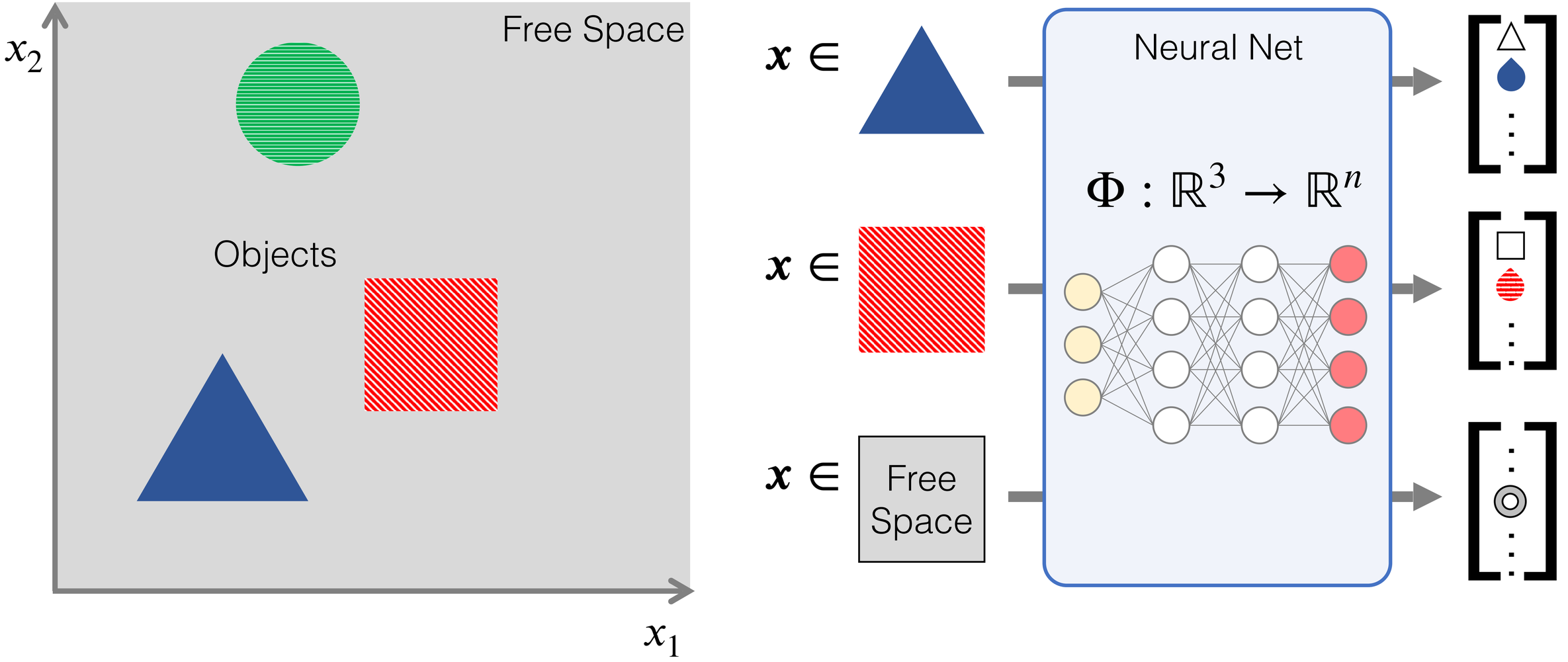


Neural Fields



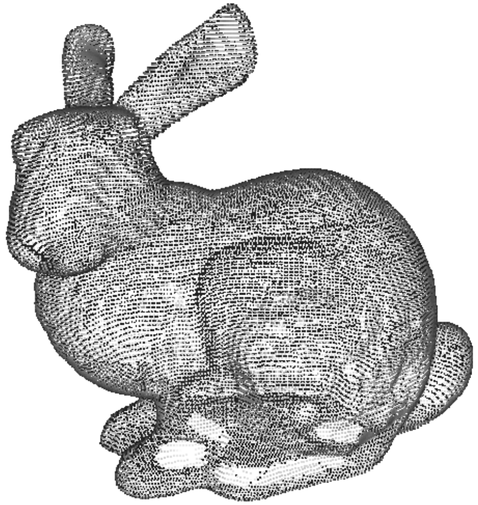
Hybrid Parameterizations

Neural fields: Parameterize Field as Neural Network!

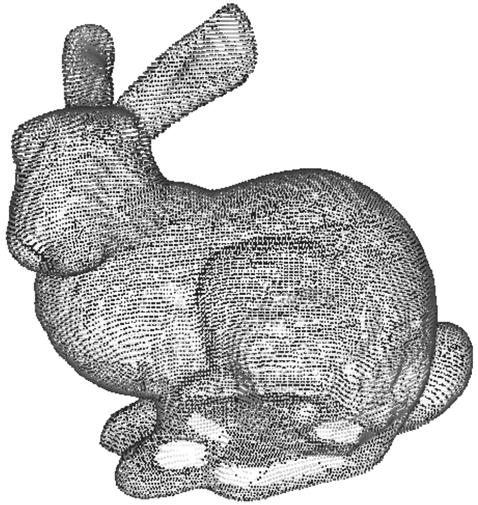


Occupancy Networks: Learning 3D Reconstruction in Function Space, Mescheder et al.
IM-Net: Learning Implicit Fields for Generative Shape Modeling, Chen et al.
DeepSDF: Learning Continuous Signed Distance Functions for Shape Representation, Park et al. 2019
Sitzmann et al: Scene Representation Networks: Continuous 3D-Structure-Aware Neural Scene Representations, NeurIPS 2019.

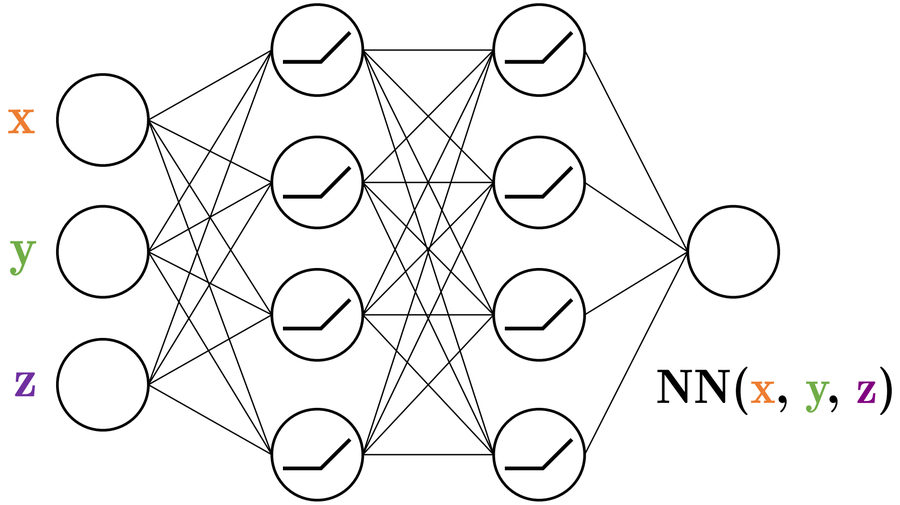
Shapes



Shapes



ReLU MLP



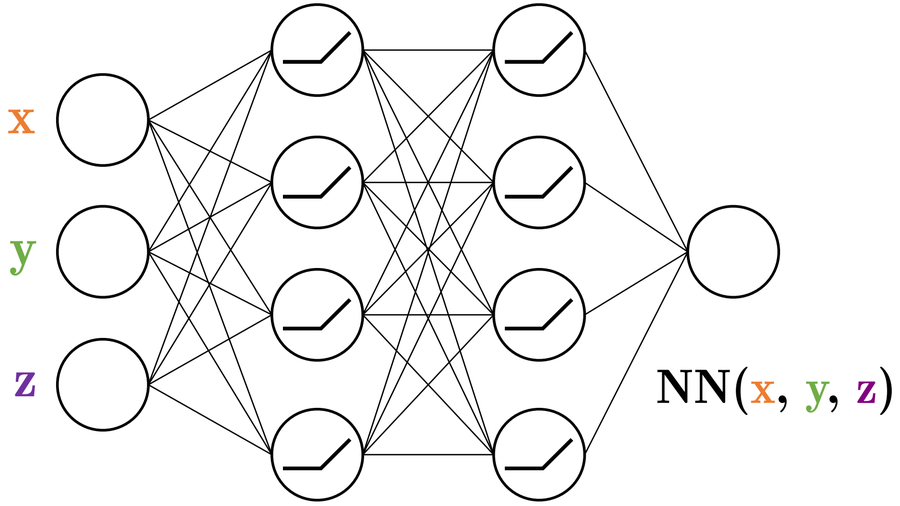


Mescheder et al. 2018
Park et al. 2018
Chen et al. 2018

Shapes



ReLU MLP



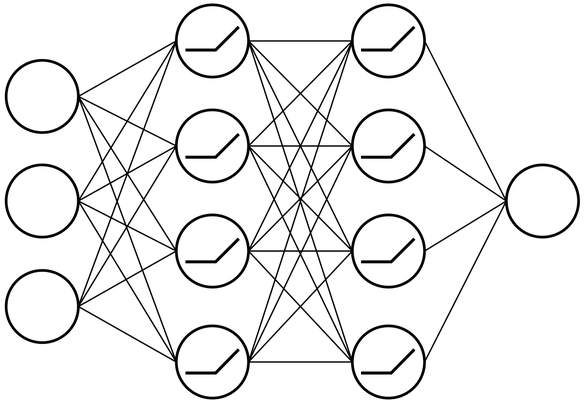


Mescheder et al. 2018
Park et al. 2018
Chen et al. 2018

Shapes



ReLU MLP





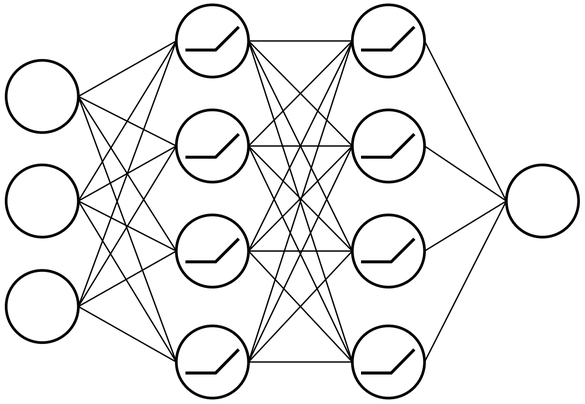
Images

Shapes





ReLU MLP







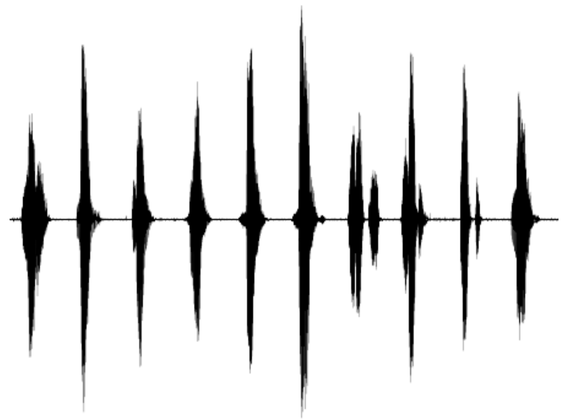
Images

Shapes

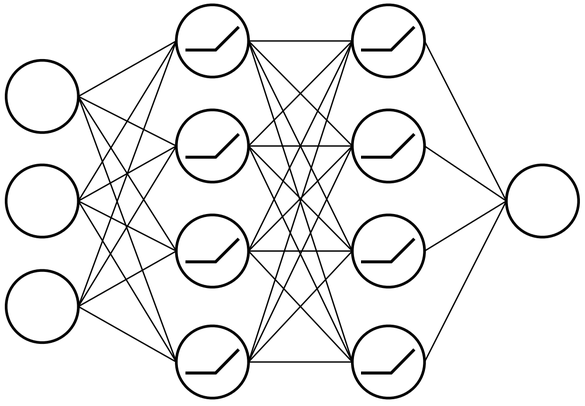
Audio





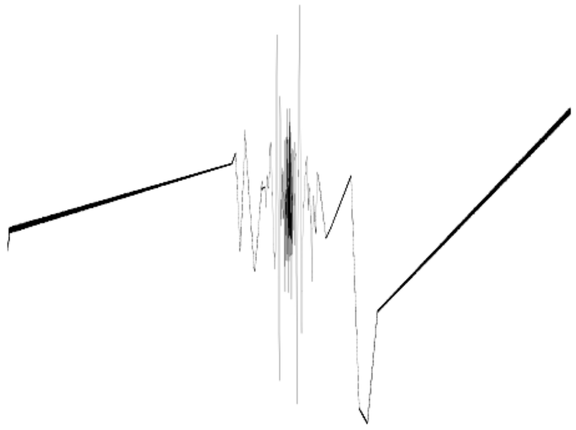


ReLU MLP









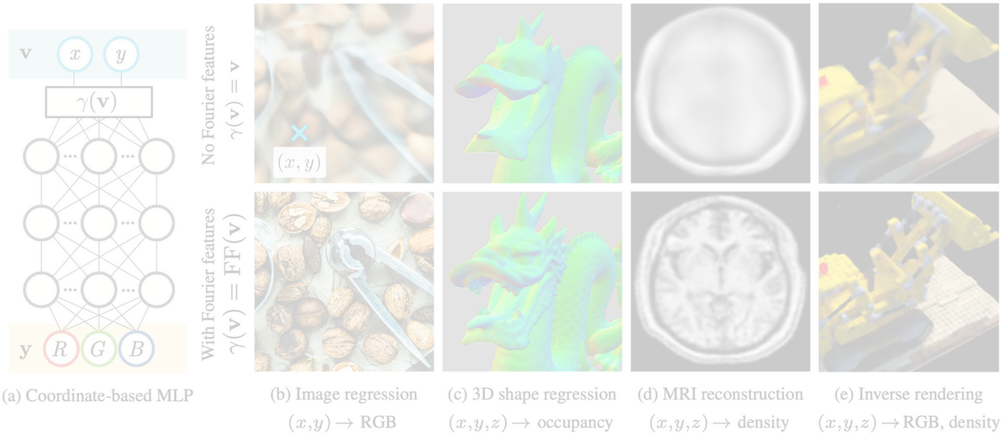
SIREN: Neural Implicit Representations With Periodic Activation Functions

Sitzmann & Martel et al. NeurIPS 2020



Fourier Features Let Networks Learn High Frequency Functions in Low Dimensional Domains

Tancik, Srinivasan, Mildenhall NeurIPS 2020



(a) Coordinate-based MLP

(b) Image regression $(x,y) \rightarrow$ RGB

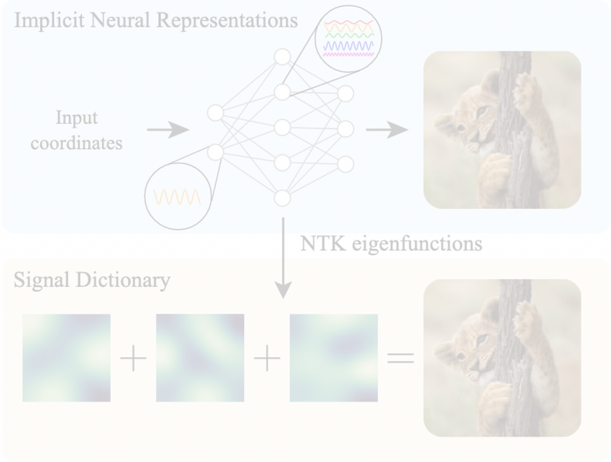
(c) 3D shape regression $(x,y,z) \rightarrow$ occupancy

(d) MRI reconstruction $(x,y,z) \rightarrow$ density

(e) Inverse rendering $(x,y,z) \rightarrow$ RGB, density

A Structured Dictionary Perspective on Implicit Neural Representations

Yüce & Ortiz-Jiménez et al. CVPR 2022



Gaussian Activated Neural Radiance Fields for High Fidelity Reconstruction & Pose Estimation

Chng et al. ECCV 2022

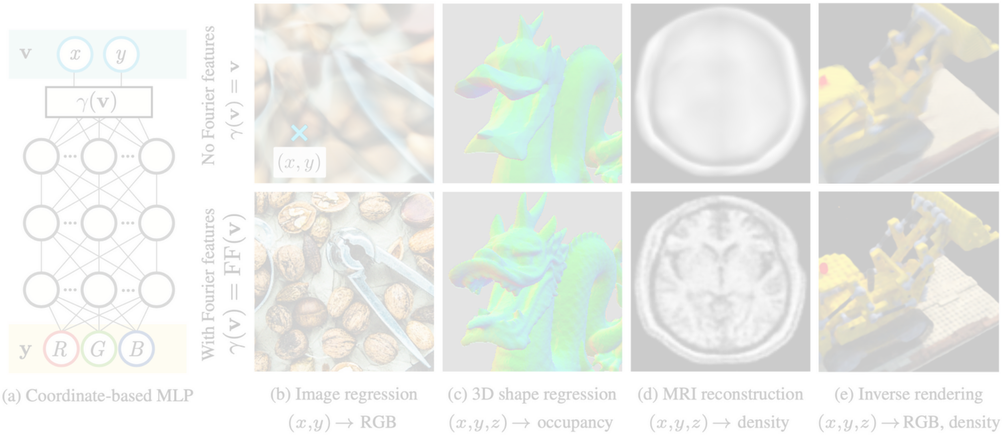
SIREN: Neural Implicit Representations With Periodic Activation Functions

Sitzmann & Martel et al. NeurIPS 2020



Fourier Features Let Networks Learn High Frequency Functions in Low Dimensional Domains

Tancik, Srinivasan, Mildenhall NeurIPS 2020



(a) Coordinate-based MLP

(b) Image regression $(x,y) \rightarrow$ RGB

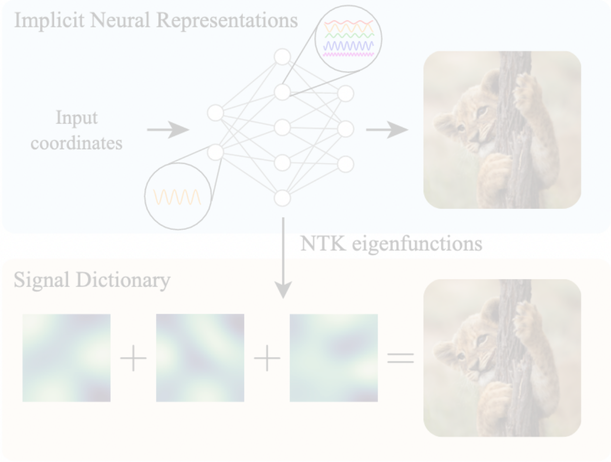
(c) 3D shape regression $(x,y,z) \rightarrow$ occupancy

(d) MRI reconstruction $(x,y,z) \rightarrow$ density

(e) Inverse rendering $(x,y,z) \rightarrow$ RGB, density

A Structured Dictionary Perspective on Implicit Neural Representations

Yüce & Ortiz-Jiménez et al. CVPR 2022



Gaussian Activated Neural Radiance Fields for High Fidelity Reconstruction & Pose Estimation

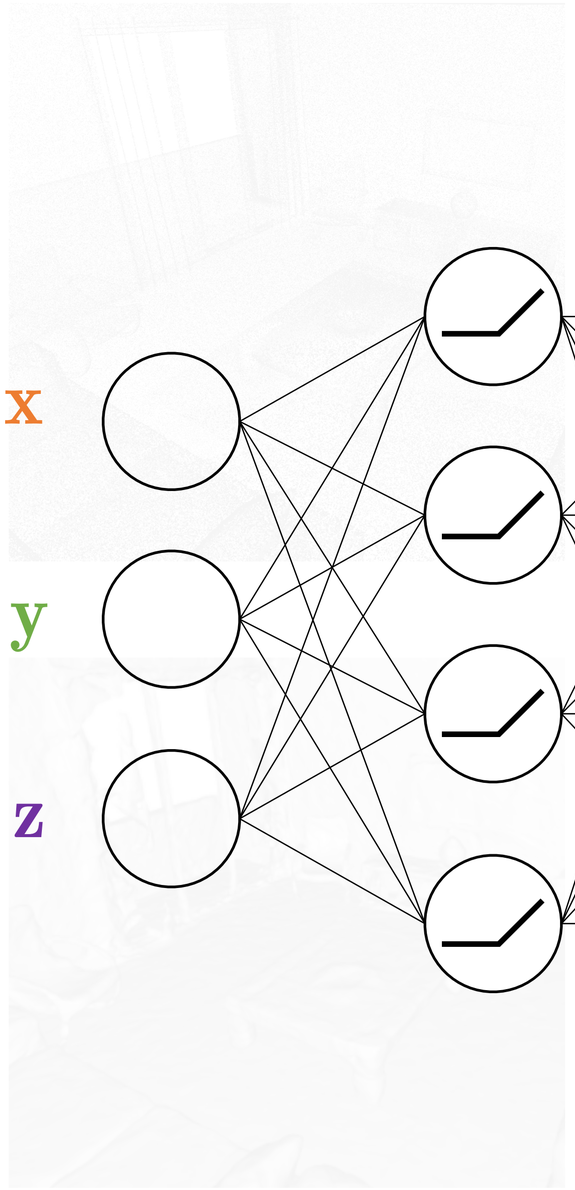
Chng et al. ECCV 2022

Images

Shapes

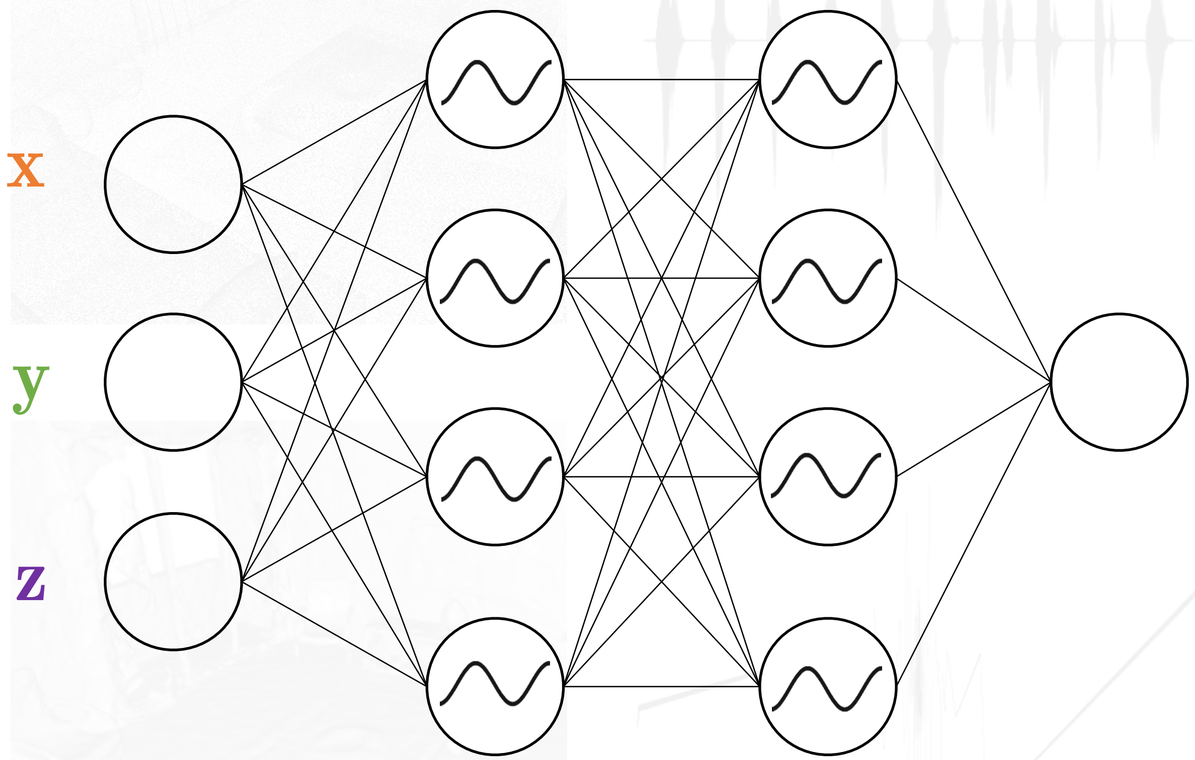
Audio

Quantities defined by a differential equation



Sitzmann et al.: Implicit neural representations with periodic activation functions, NeurIPS 2020

SIREN: Sinusoidal Representation Networks



Sitzmann et al.: Implicit neural representations with periodic activation functions, NeurIPS 2020

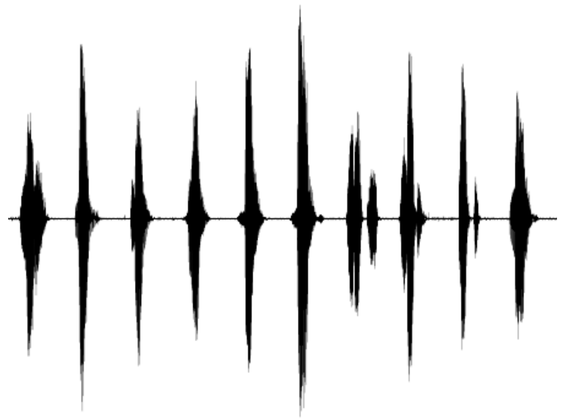
Images

Shapes

Audio

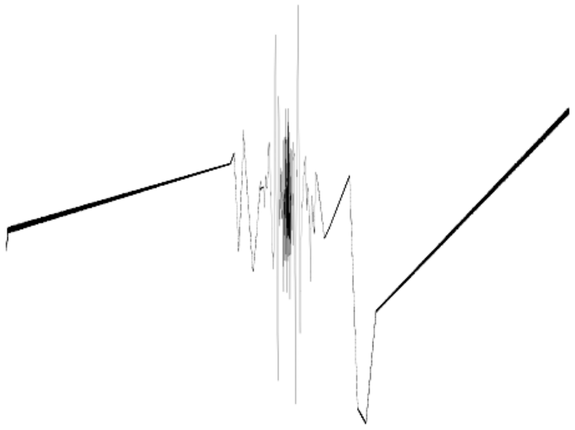












Images



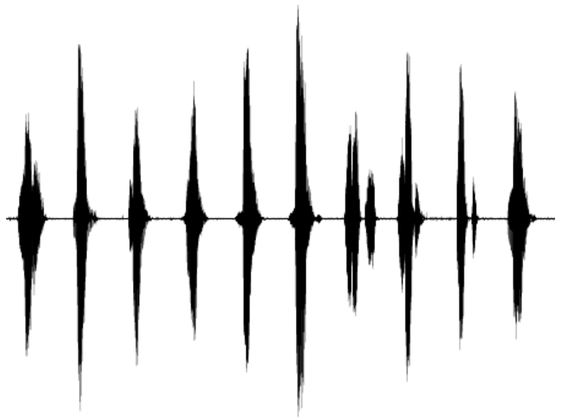


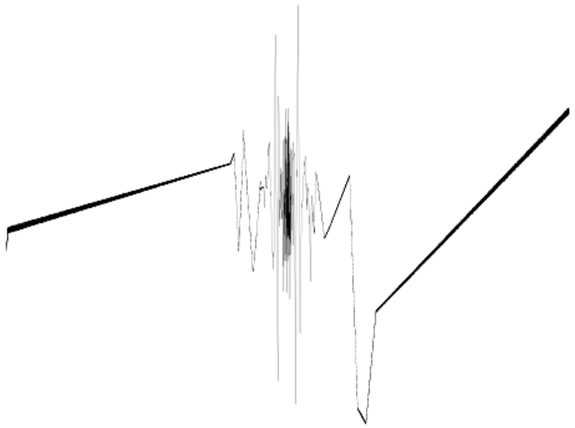
Shapes



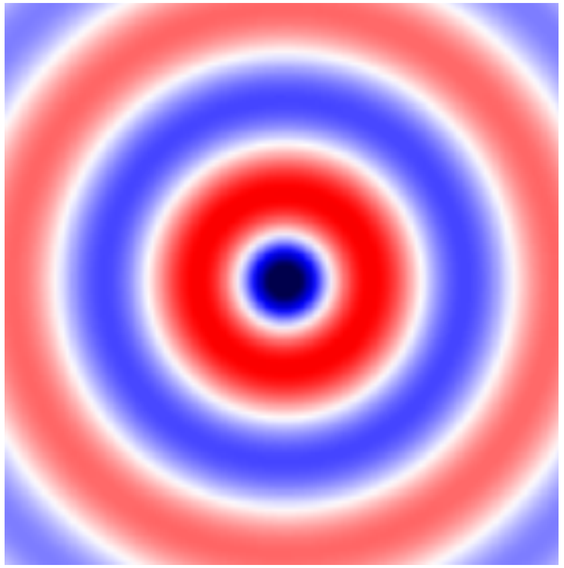


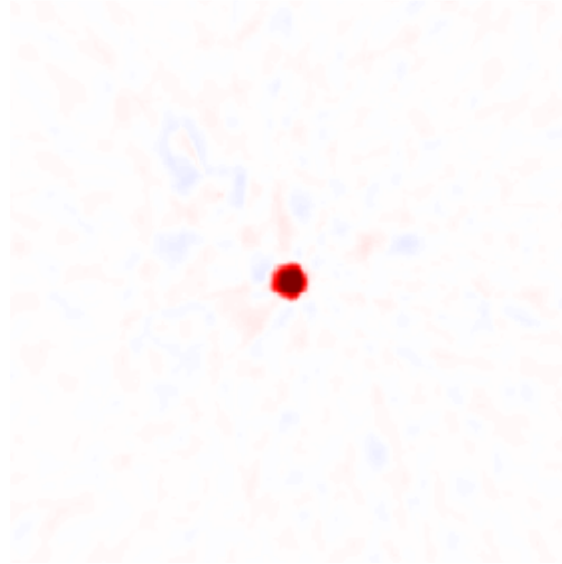
Audio





Quantities defined by a differential equation





Images

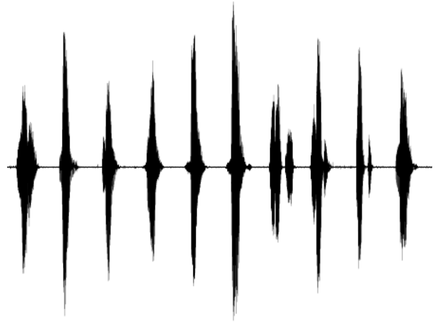
Shapes

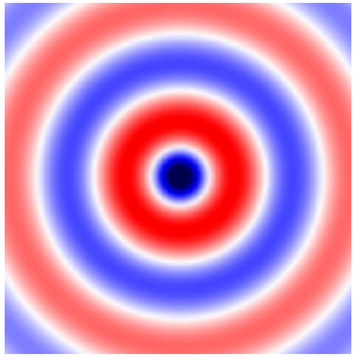
Audio

Quantities defined by a differential equation

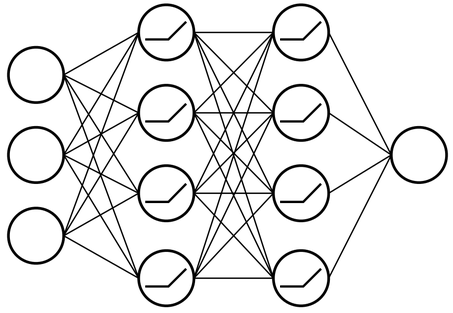






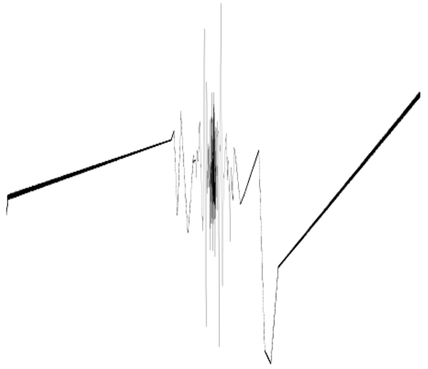


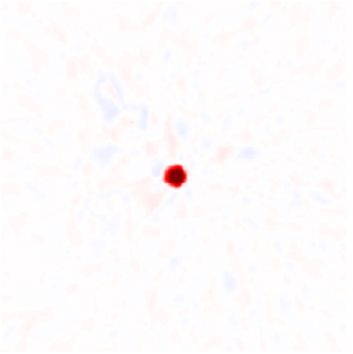
ReLU MLP



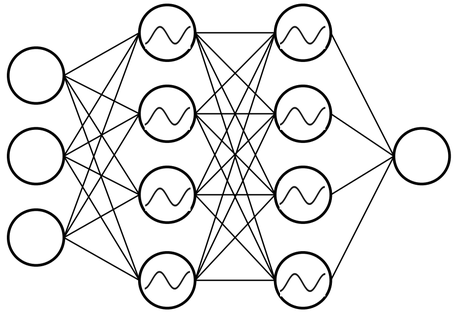








SIREN



Images

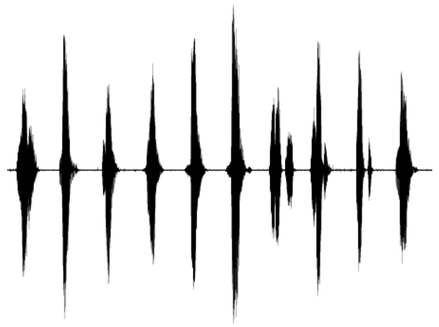
Shapes

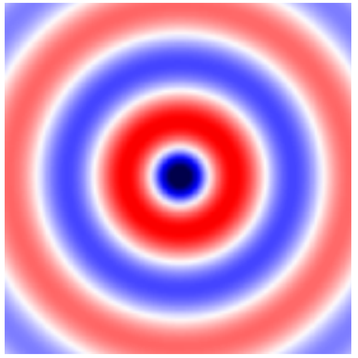
Audio

Quantities defined by a differential equation

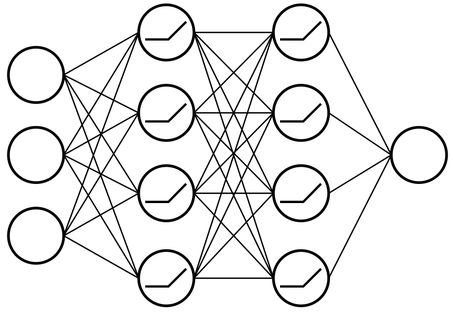






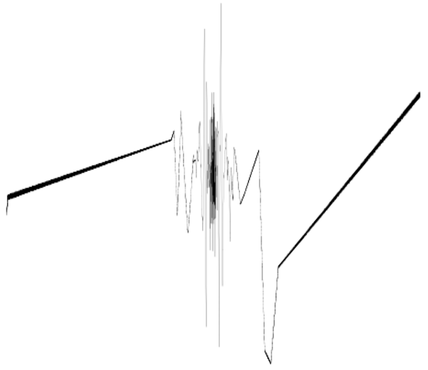


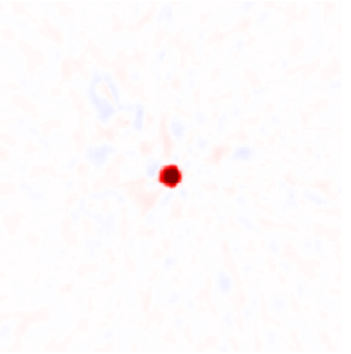
ReLU MLP



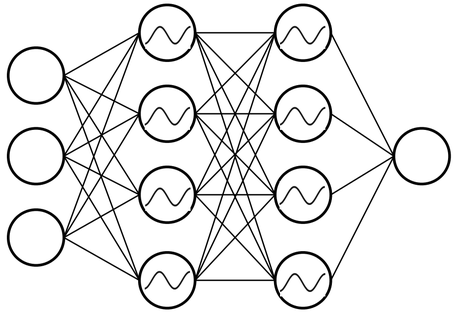






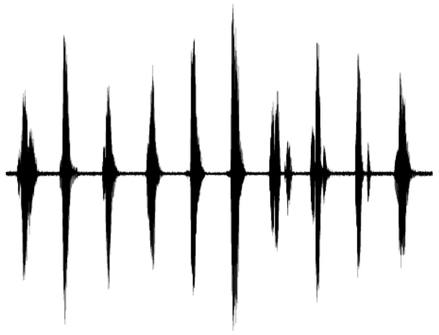


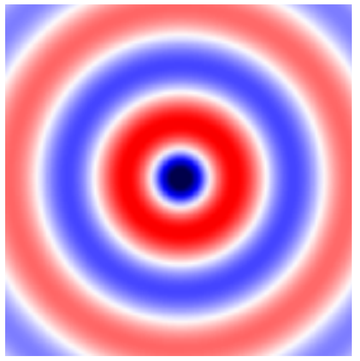
SIREN





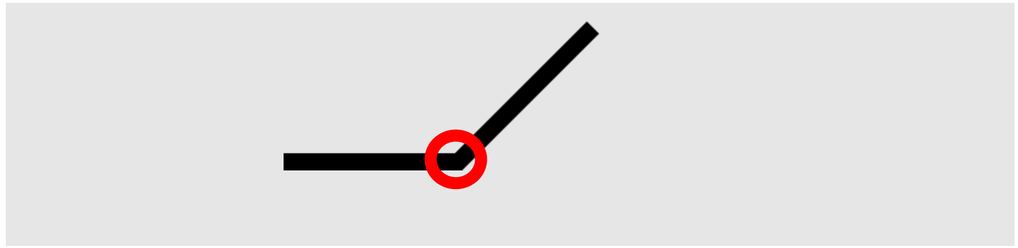


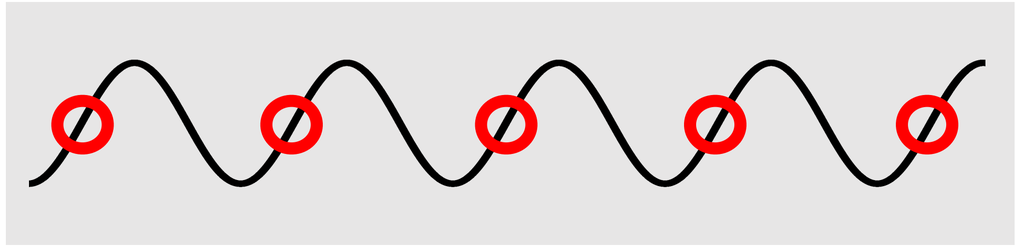




Locality

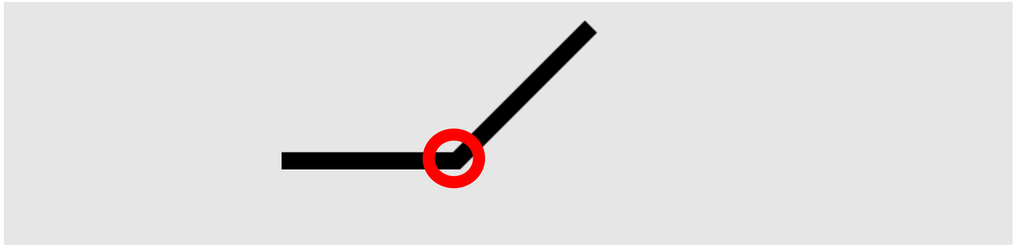
$\delta/\delta x$ $\delta^2/\delta x^2$

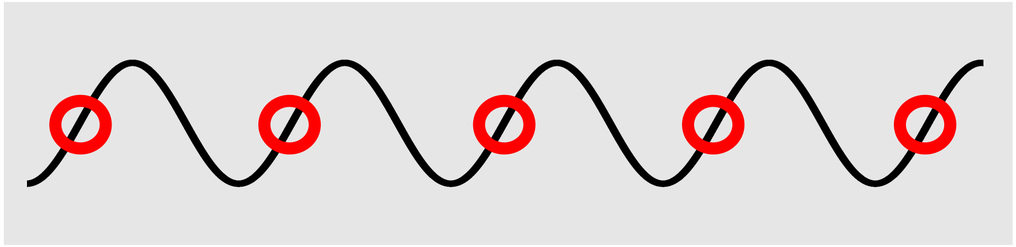




Periodicity allows SIREN to replicate activations across input domain

Locality

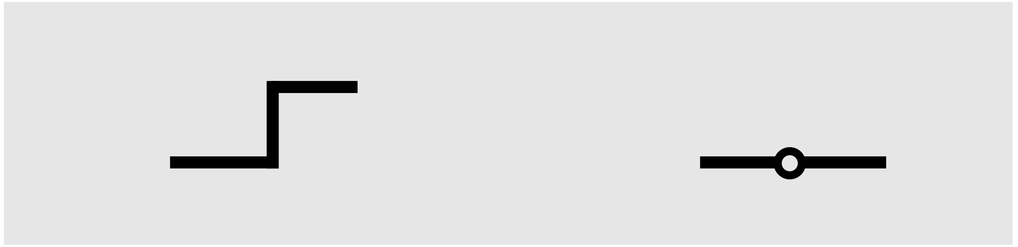


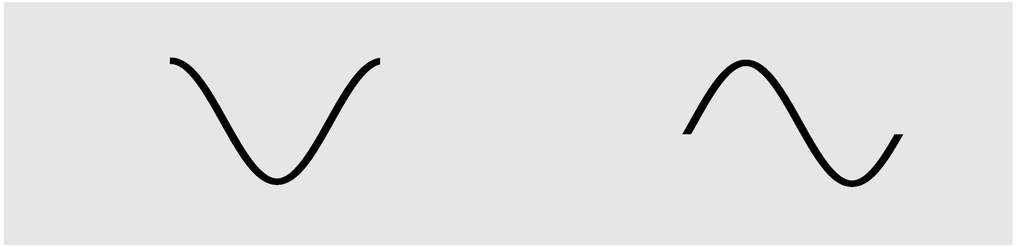


Periodicity allows SIREN to replicate activations across input domain

Derivatives

$\delta/\delta x$ $\delta^2/\delta x^2$



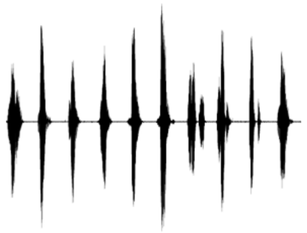


All derivatives exist, are nonzero and bounded by 1

Images



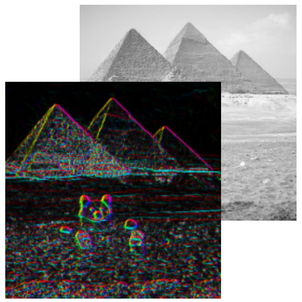
Audio



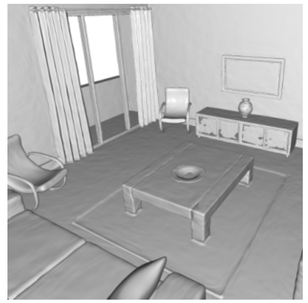
Videos



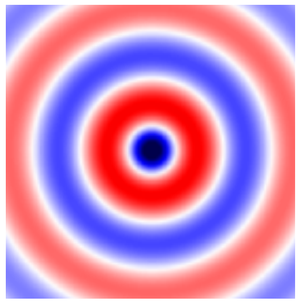
Poisson equation



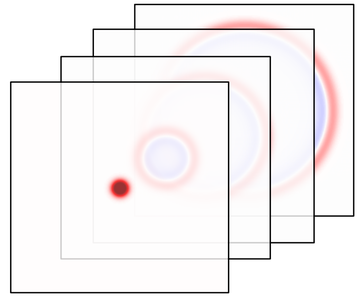
Eikonal equation



Helmholtz equation



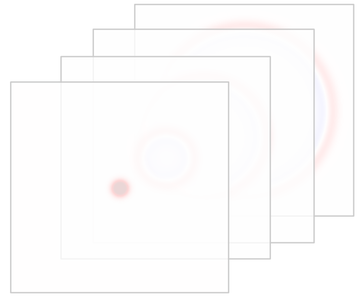
Wave equation



Images | Audio | Videos | Poisson equation | Eikonal equation | Helmholtz equation | Wave equation

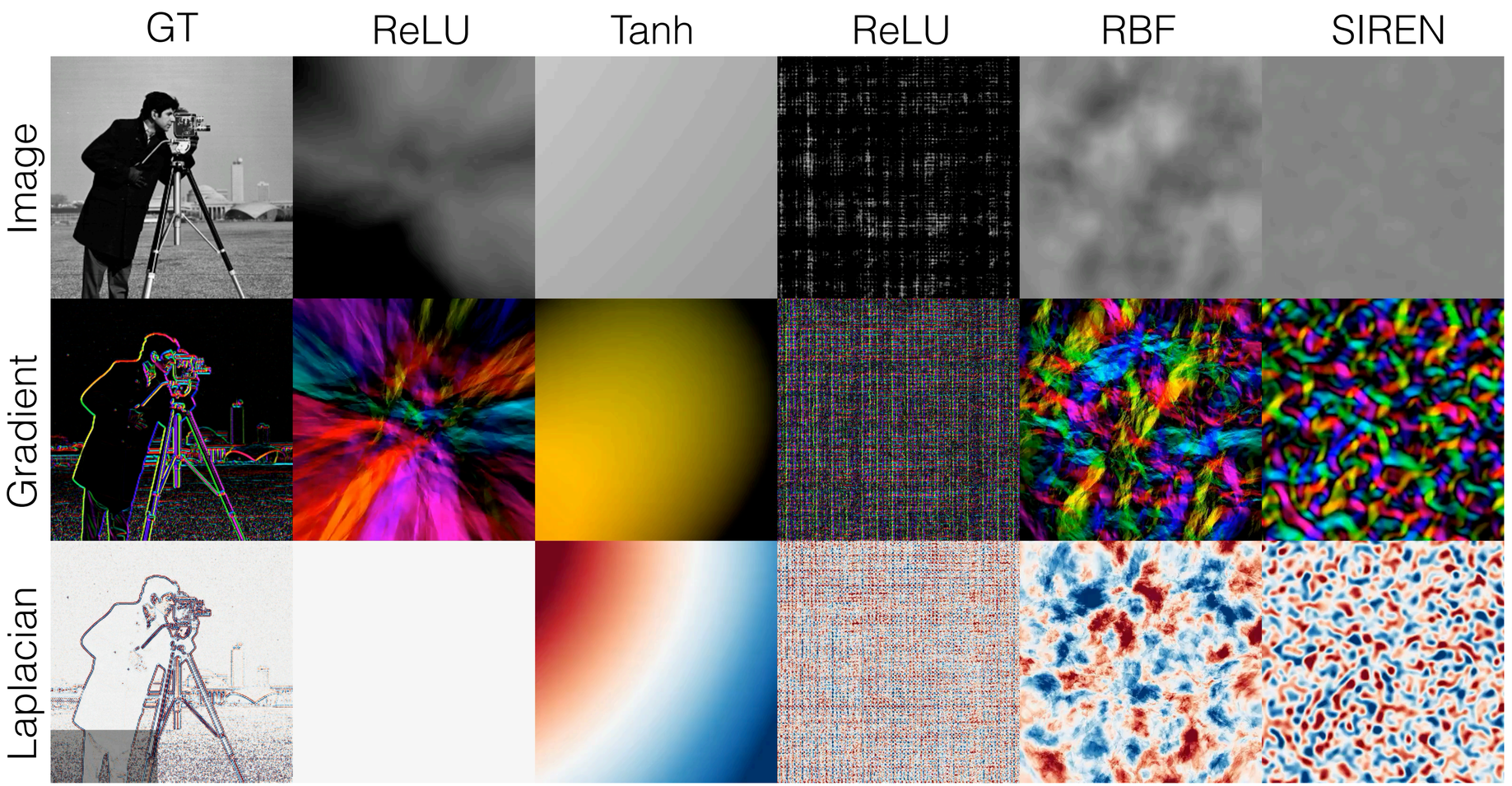


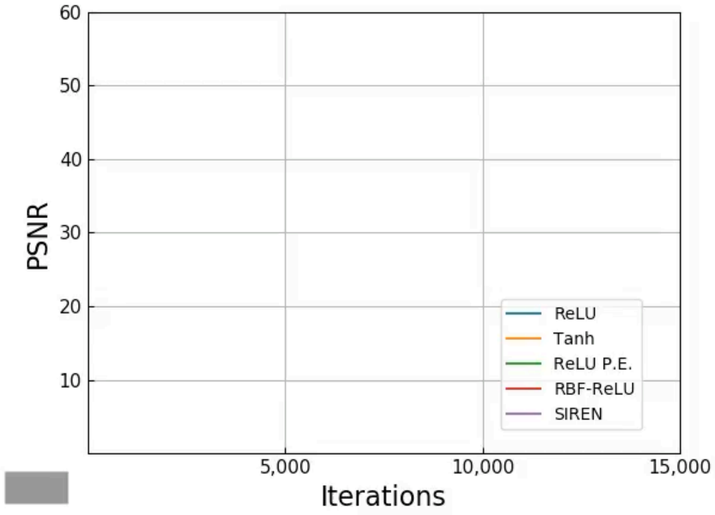




Input	Output supervised by	Implicit Formulation Find $\boldsymbol{\Phi}$ that minimizes $\mathscr{L}$
$\mathbf{x} \in \mathbb{R}^2$	$f(\mathbf{x}) \in \mathbb{R}^3$	$\mathcal{L}_{\text{img}} = \int_{\Omega} \lVert \Phi(\mathbf{x}) - f(\mathbf{x}) \rVert \, \mathrm{d}\mathbf{x}$
spatial coords.	*RGB values*	

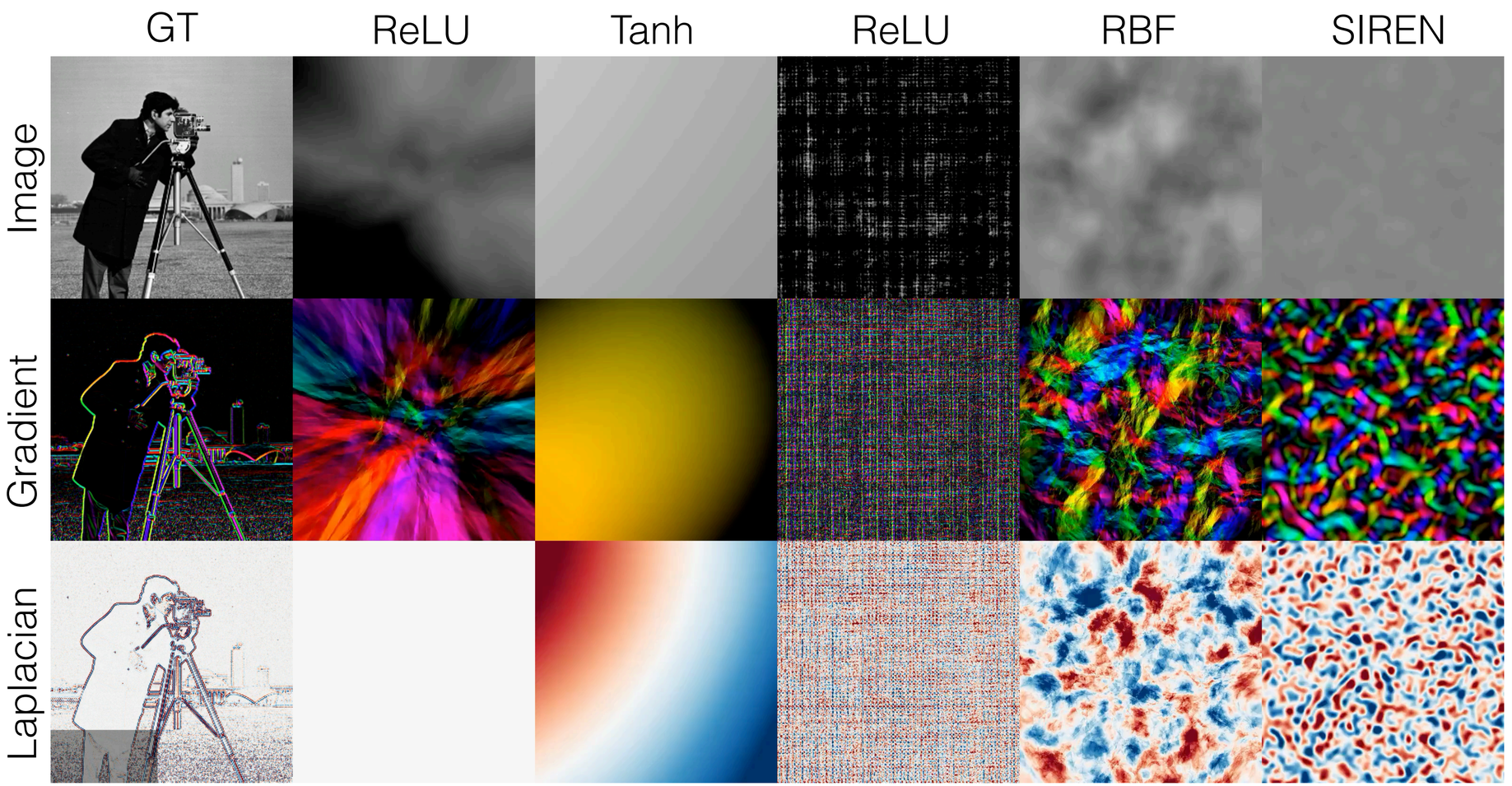
Representing Images

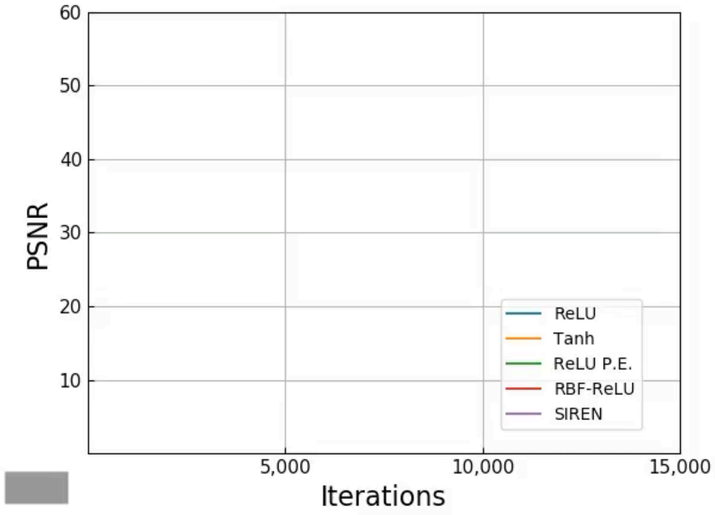




[1] Nerf [Mildenhall et al. 2020]

Representing Images

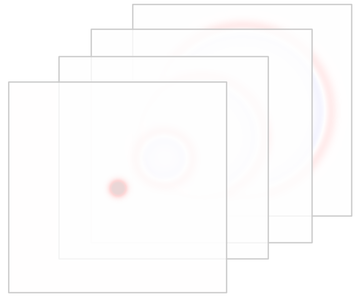




[1] Nerf [Mildenhall et al. 2020]

Images | Audio | Videos | Poisson equation | Eikonal equation | Helmholtz equation | Wave equation



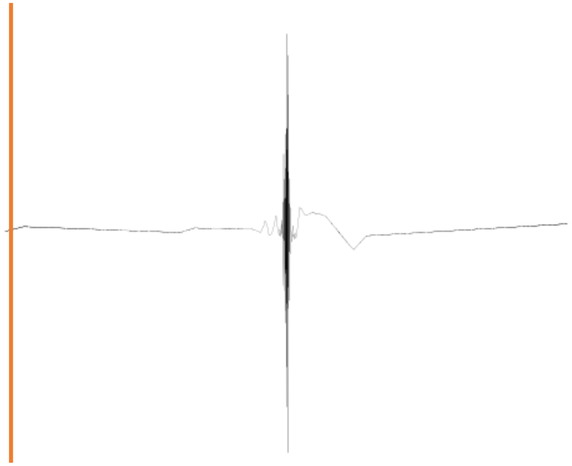
Input	Output supervised by	Implicit Formulation Find $\boldsymbol{\Phi}$ that minimizes $\mathscr{L}$
$t \in \mathbb{R}$	$f(t) \in \mathbb{R}$	$\mathcal{L}_{\text{audio}} = \int_{\Omega} \lVert \Phi(t) - f(t) \rVert \, \mathrm{d}t$
time point	*amplitude*	

Representing Audio – Music

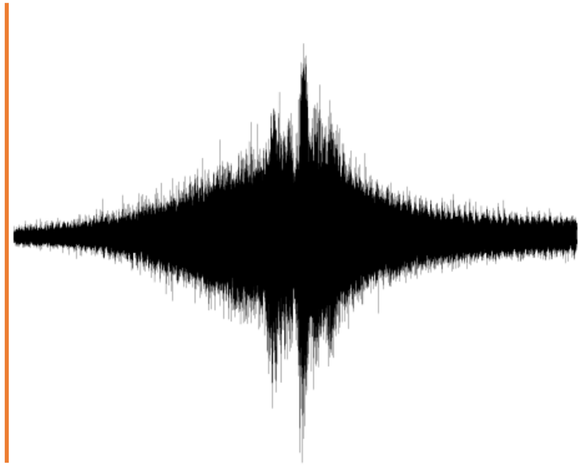
Ground Truth



ReLU MLP



ReLU w/ positional encoding



SIREN

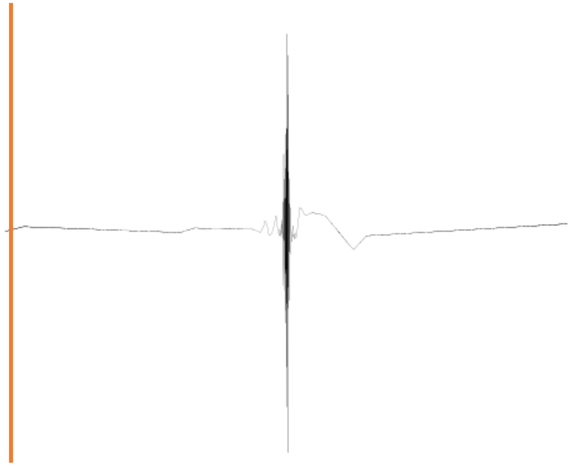


Representing Audio – Music

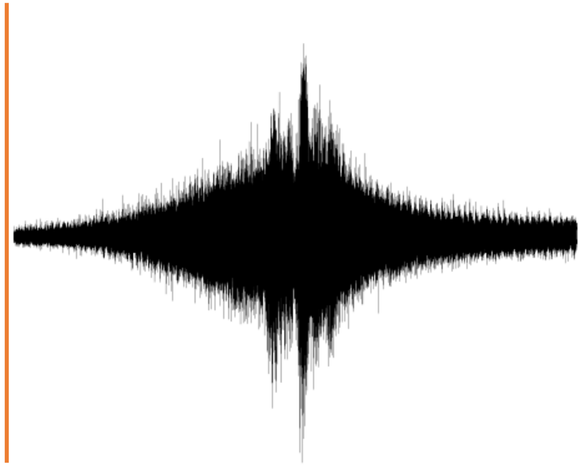
Ground Truth



ReLU MLP



ReLU w/ positional encoding



SIREN

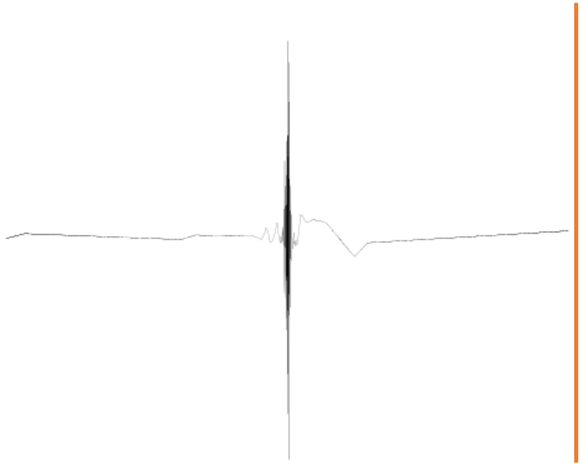


Representing Audio – Music

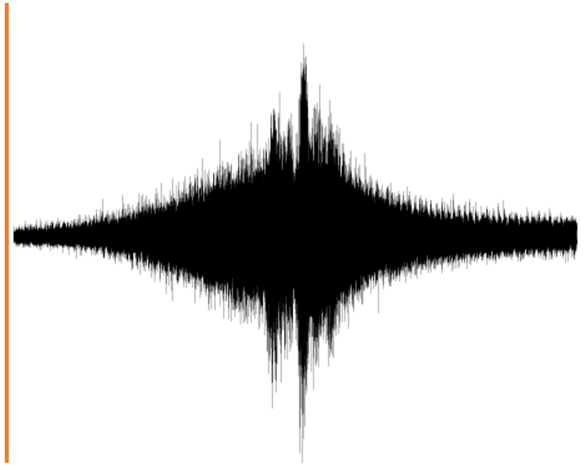
Ground Truth



ReLU MLP



ReLU w/ positional encoding



SIREN

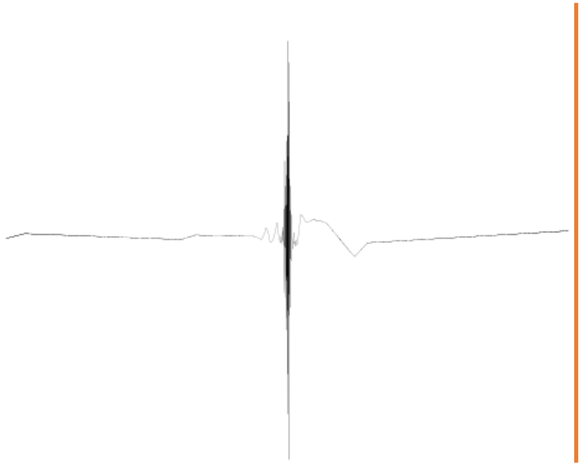


Representing Audio – Music

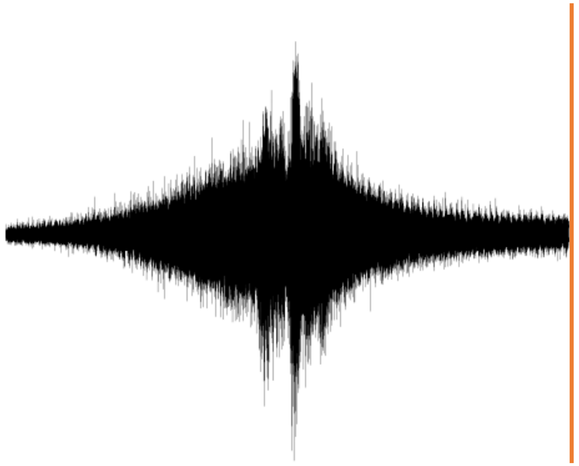
Ground Truth



ReLU MLP



ReLU w/ positional encoding



SIREN

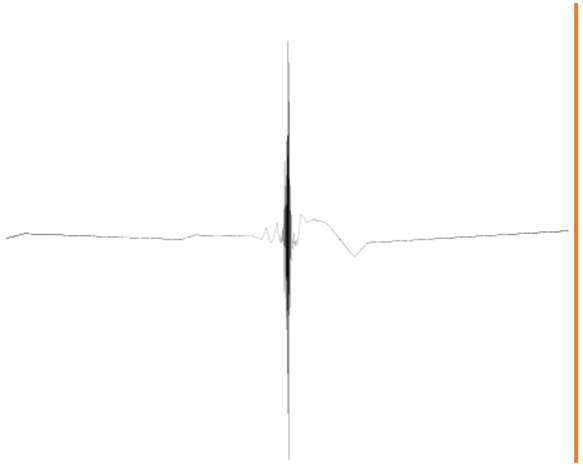


Representing Audio – Music

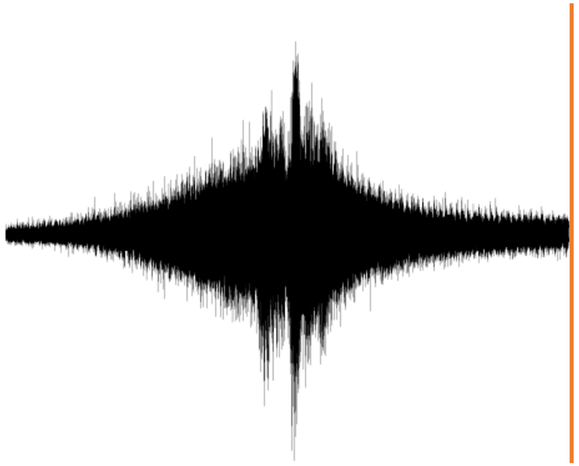
Ground Truth



ReLU MLP



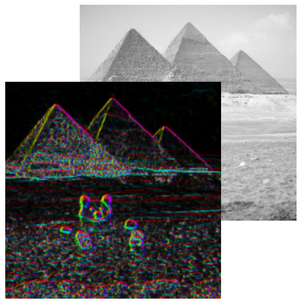
ReLU w/ positional encoding

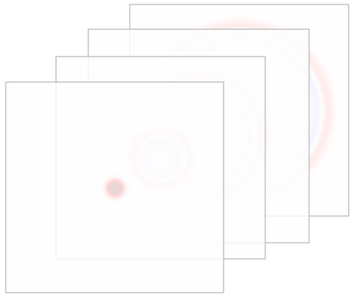


SIREN



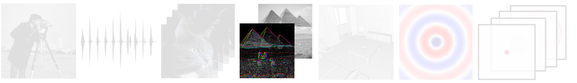
Images | Audio | Videos | Poisson equation | Eikonal equation | Helmholtz equation | Wave equation

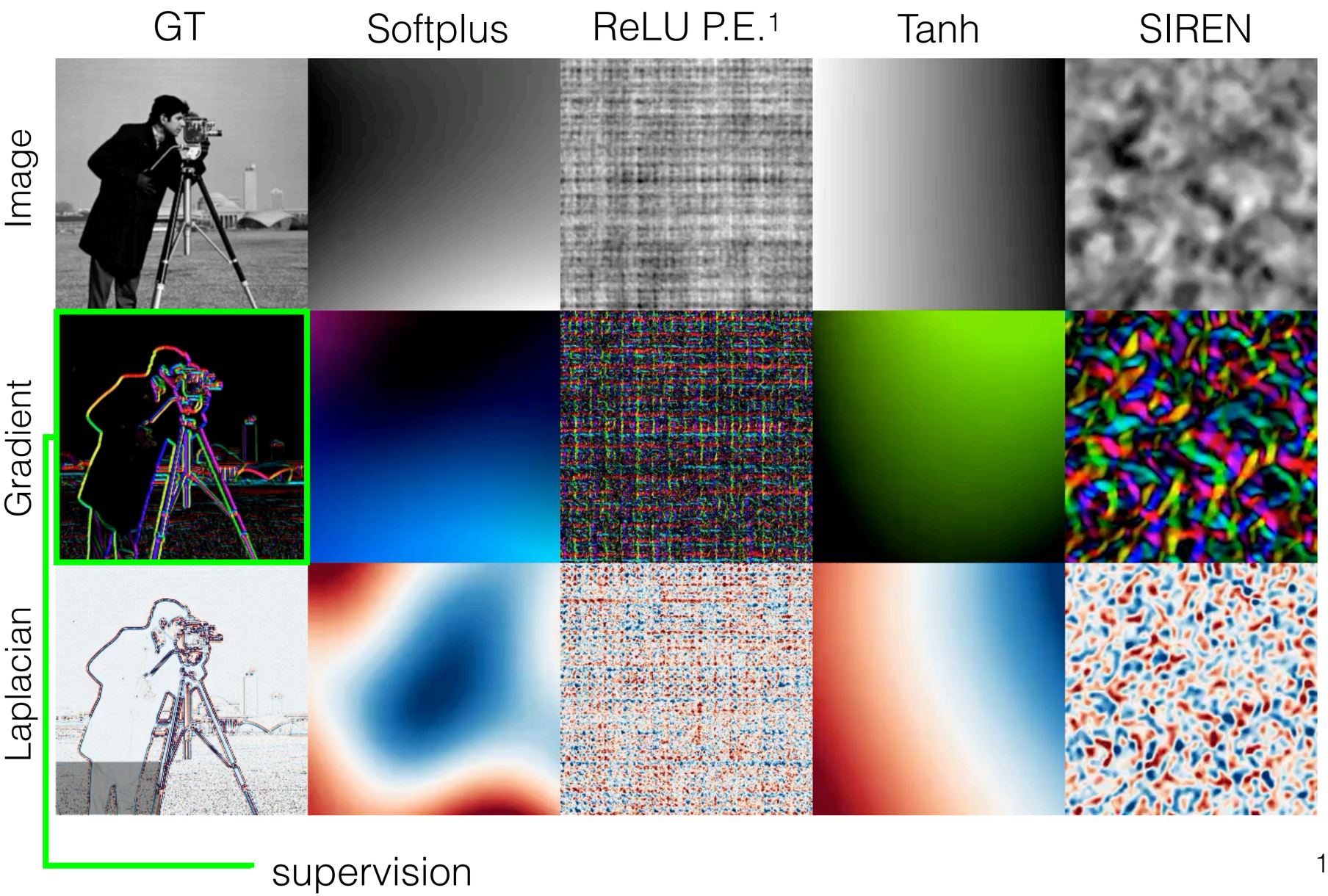



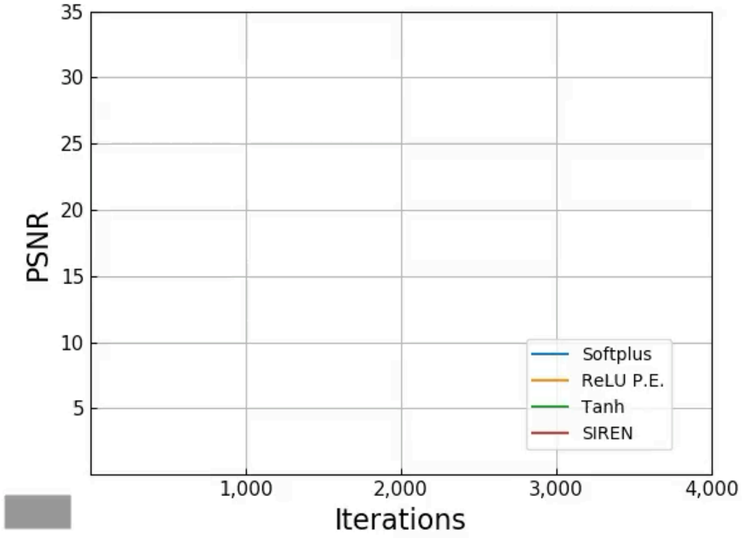


Input	Output supervised by	Implicit Formulation Find $\boldsymbol{\Phi}$ that minimizes $\mathscr{L}$
$\mathbf{x} \in \mathbb{R}^2$	$f(\mathbf{x}) \in \mathbb{R}^2$	$\mathcal{L}_{\text{Poisson}} = \int_{\Omega} \|\nabla\Phi(\mathbf{x}) - \nabla f(\mathbf{x})\| \, \mathrm{d}\mathbf{x}$
spatial coord.	*gray level*	

Poisson's Equation

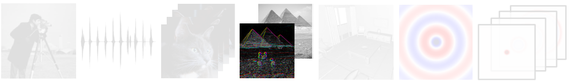


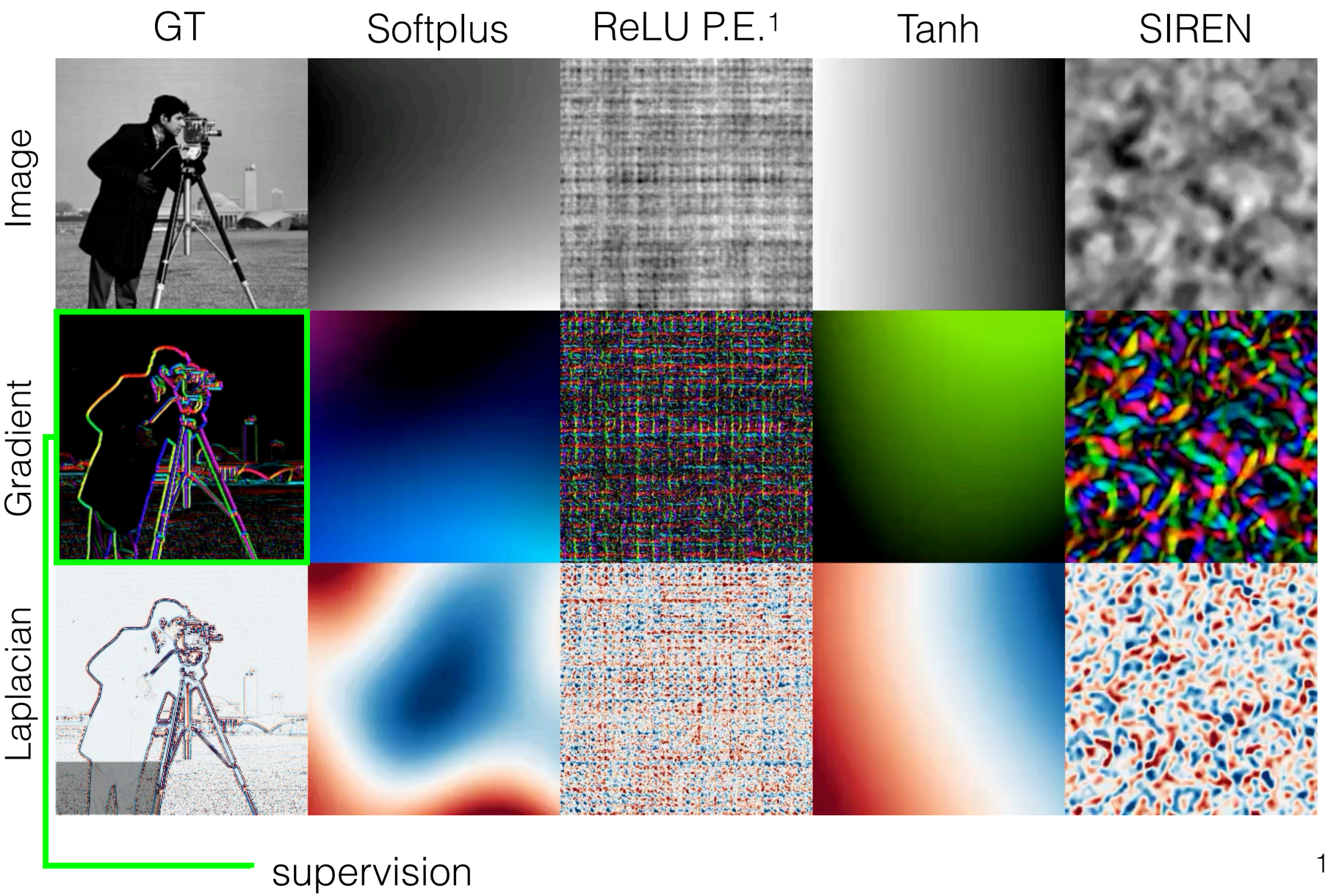


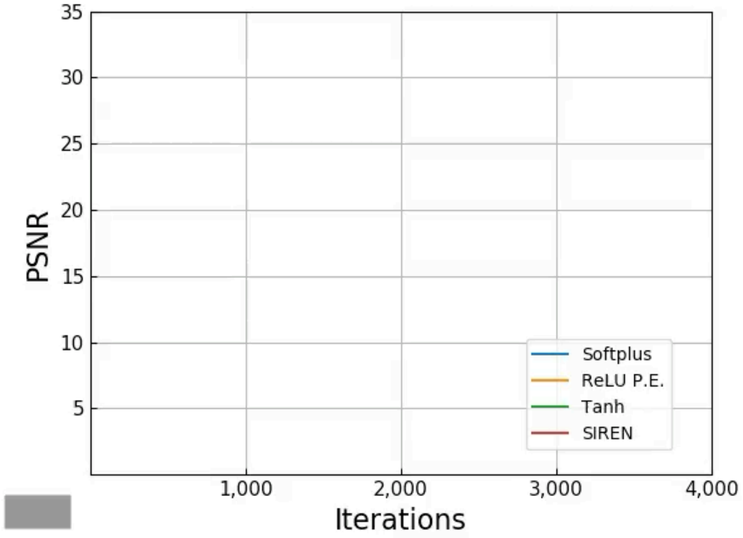


[1] Nerf [Mildenhall et al. 2020]

Poisson's Equation





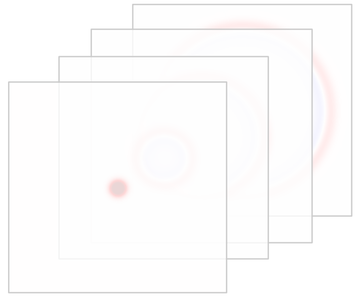


[1] Nerf [Mildenhall et al. 2020]

Images | Audio | Videos | Poisson equation | Eikonal equation | Helmholtz equation | Wave equation







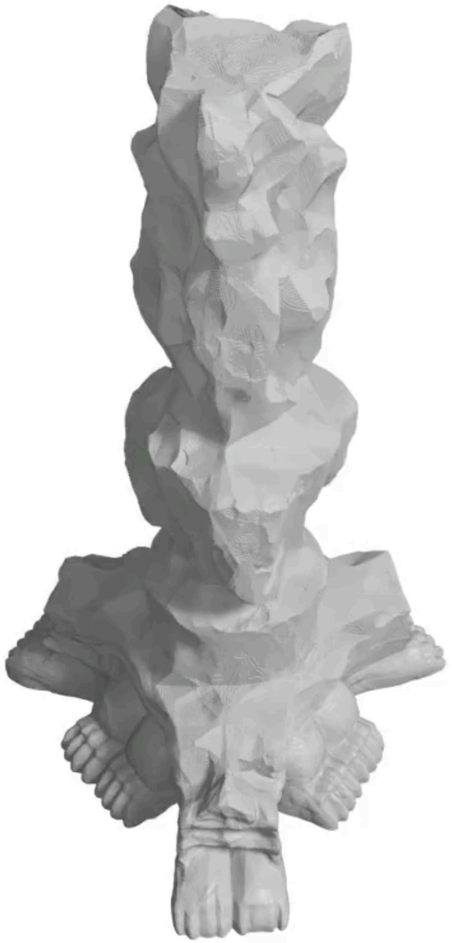
Input	Output supervised by	Implicit Formulation Find $\boldsymbol{\Phi}$ that minimizes $\mathscr{L}$
$\mathbf{x} \in \mathbb{R}^3$	$f(\mathbf{x}) \in \mathbb{R}$	$\mathcal{L}_{\text{Eikonal}} = \int_{\Omega_0} \lvert\Phi(\mathbf{x})\rvert + (1 - \langle\nabla\Phi(\mathbf{x}), \nabla f(\mathbf{x})\rangle)\, \mathrm{d}\mathbf{x}$ $+ \int_{\Omega} \lVert\lvert\nabla\Phi(\mathbf{x})\rvert - 1\rVert\, \mathrm{d}\mathbf{x}$
spatial coord.	*signed distance*	

Implicit Geometric Regularization for Learning Shapes [Gropp et al. 202

3D Shapes - solving the Eikonal equation



ReLU



SIREN

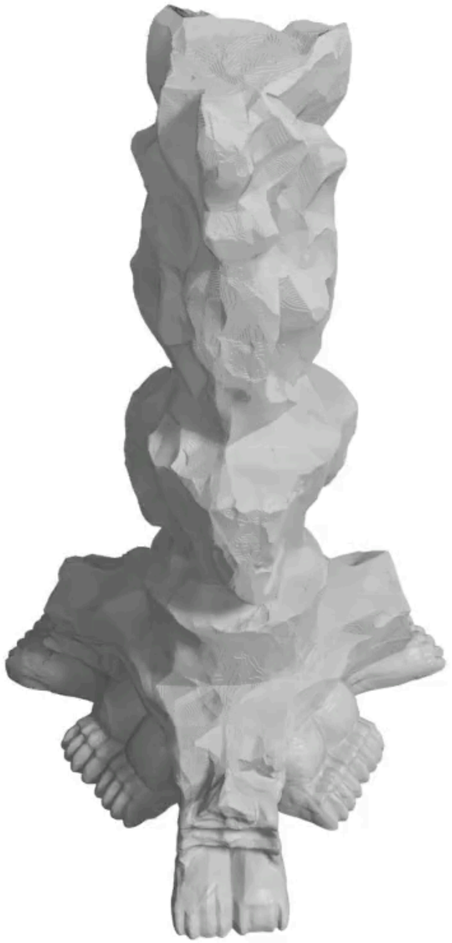


5 layers, 256 hidden units

3D Shapes - solving the Eikonal equation



ReLU



SIREN



5 layers, 256 hidden units

SIREN



ReLU

SIREN

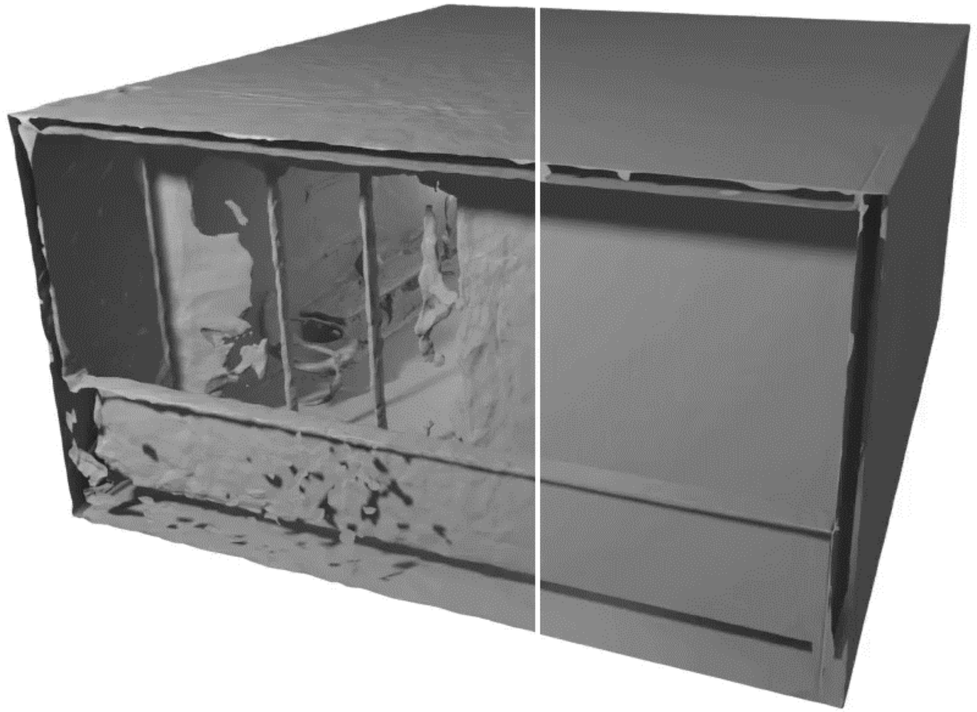
SIREN



ReLU

SIREN

SIREN



~1MB compared to
110MB of full mesh!

ReLU

SIREN

Background: Kernel Regression

Given "training" set $\{(x_i, y_i)\}_i$, a kernel function makes predictions on a point $\mathbf{x}$ by interpolating labels $\mathbf{y}_i$ in the training set according to pairwise weights between $\mathbf{x}_i$ to $\mathbf{x}$ as measured by a kernel function:

$$f(\mathbf{x}) = \sum_{i=1}^{n} (\mathbf{K}^{-1}\mathbf{y})_i k(\mathbf{x}_i, \mathbf{x})$$

Where $\mathbf{K}_{ij} = k(\mathbf{x}_i, \mathbf{x}_j)$, k is kernel function with $k \geq 0$.

The Neural Tangent Kernel

$$f(\mathbf{x}) = \sum_{i=1}^{n} (\mathbf{K}^{-1}\mathbf{y})_i k(\mathbf{x}_i, \mathbf{x})$$

It turns out (Jacot et al. 2018) that in the limit of an infinitely wide MLP $f(x)$ with initialization θ_0, test-time predictions are made according to a kernel:

$$k_{NTK}(\mathbf{x}_1, \mathbf{x}_2) = \langle \nabla_\theta f_{\theta_0}(\mathbf{x}_1), \nabla_\theta f_{\theta_0}(\mathbf{x}_2) \rangle$$

The Neural Tangent Kernel

$$f(\mathbf{x}) = \sum_{i=1}^{n} (\mathbf{K}^{-1}\mathbf{y})_i k(\mathbf{x}_i, \mathbf{x})$$

It turns out (Jacot et al. 2018) that in the limit of an infinitely wide MLP $f(x)$ with initialization θ_0, test-time predictions are made according to a kernel:

$$k_{NTK}(\mathbf{x}_1, \mathbf{x}_2) = \langle \nabla_\theta f_{\theta_0}(\mathbf{x}_1), \nabla_\theta f_{\theta_0}(\mathbf{x}_2) \rangle$$

In other words: “Neural” implicit representations interpolate the training set according to a relatively simple rule. No “intelligence”, no AI, no magic.

The Neural Tangent Kernel

$$f(\mathbf{x}) = \sum_{i=1}^{n} (\mathbf{K}^{-1}\mathbf{y})_i k(\mathbf{x}_i, \mathbf{x})$$

It turns out (Jacot et al. 2018) that in the limit of an infinitely wide MLP $f(x)$ with initialization θ_0, test-time predictions are made according to a kernel:

$$k_{NTK}(\mathbf{x}_1, \mathbf{x}_2) = \langle \nabla_\theta f_{\theta_0}(\mathbf{x}_1), \nabla_\theta f_{\theta_0}(\mathbf{x}_2) \rangle$$

What we can do is design the kernel and pick the space of $(\mathbf{x}, \mathbf{y})$,
the space in which our neural network will interpolate.

The Neural Tangent Kernel

$$f(\mathbf{x}) = \sum_{i=1}^{n} (\mathbf{K}^{-1}\mathbf{y})_i k(\mathbf{x}_i, \mathbf{x})$$

It turns out (Jacot et al. 2018) that in the limit of an infinitely wide MLP $f(x)$ with initialization θ_0, test-time predictions are made according to a kernel:

$$k_{NTK}(\mathbf{x}_1, \mathbf{x}_2) = \langle \nabla_\theta f_{\theta_0}(\mathbf{x}_1), \nabla_\theta f_{\theta_0}(\mathbf{x}_2) \rangle$$

We do **not** know of an equivalent rule for transformers to date - work in progress!

The Neural Tangent Kernel

$$f(\mathbf{x}) = \sum_{i=1}^{n} (\mathbf{K}^{-1}\mathbf{y})_i k(\mathbf{x}_i, \mathbf{x})$$

It turns out (Jacot et al. 2018) that in the limit of an infinitely wide MLP $f(x)$ with initialization θ_0, test-time predictions are made according to a kernel:

$$k_{NTK}(\mathbf{x}_1, \mathbf{x}_2) = \langle \nabla_\theta f_{\theta_0}(\mathbf{x}_1), \nabla_\theta f_{\theta_0}(\mathbf{x}_2) \rangle$$

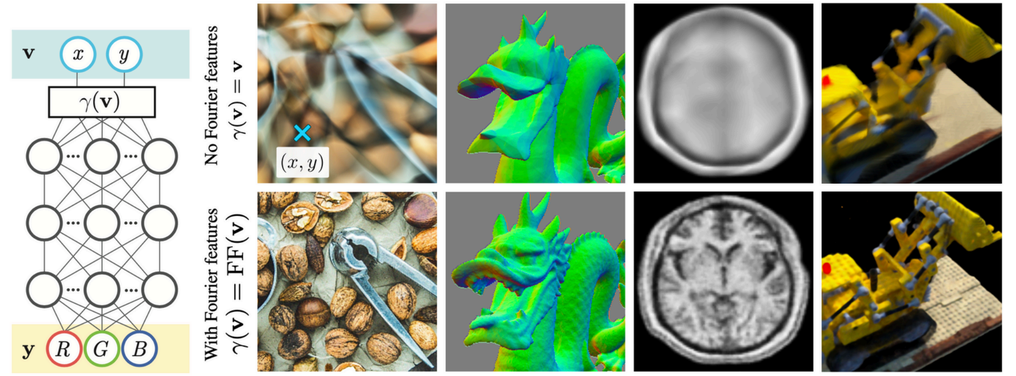
What does this mean for neural fields?

SIREN: Neural Implicit Representations With Periodic Activation Functions

Sitzmann & Martel et al. NeurIPS 2020

Fourier Features Let Networks Learn High Frequency Functions in Low Dimensional Domains

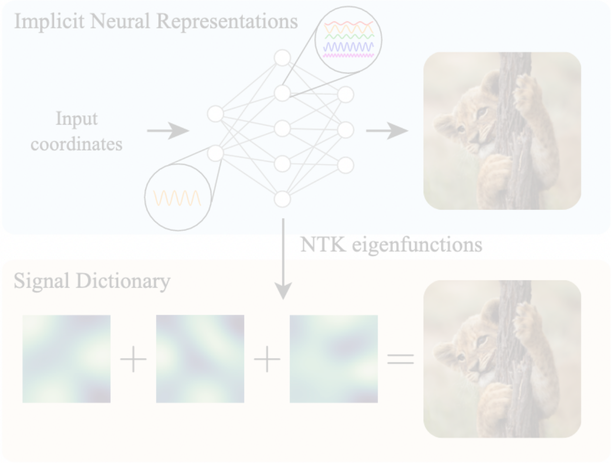
Tancik, Srinivasan, Mildenhall et al, NeurIPS 2020



(a) Coordinate-based MLP | (b) Image regression $(x,y) \rightarrow$ RGB | (c) 3D shape regression $(x,y,z) \rightarrow$ occupancy | (d) MRI reconstruction $(x,y,z) \rightarrow$ density | (e) Inverse rendering $(x,y,z) \rightarrow$ RGB, density

A Structured Dictionary Perspective on Implicit Neural Representations

Yüce & Ortiz-Jiménez et al. CVPR 2022



Gaussian Activated Neural Radiance Fields for High Fidelity Reconstruction & Pose Estimation

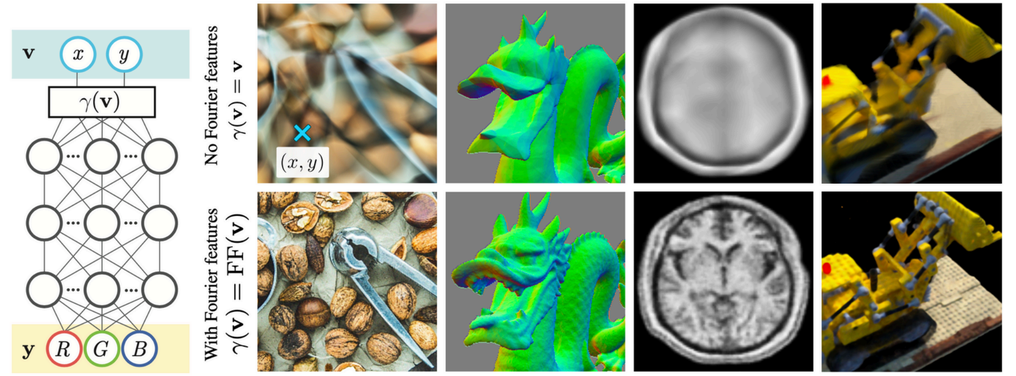
Chng et al. ECCV 2022

SIREN: Neural Implicit Representations With Periodic Activation Functions

Sitzmann & Martel et al. NeurIPS 2020

Fourier Features Let Networks Learn High Frequency Functions in Low Dimensional Domains

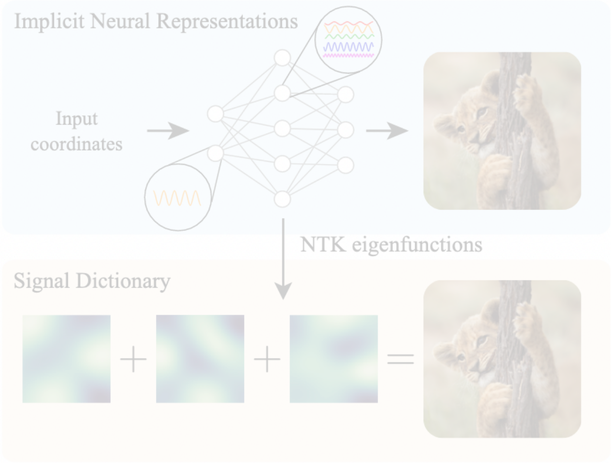
Tancik, Srinivasan, Mildenhall et al, NeurIPS 2020



(a) Coordinate-based MLP | (b) Image regression $(x,y) \rightarrow$ RGB | (c) 3D shape regression $(x,y,z) \rightarrow$ occupancy | (d) MRI reconstruction $(x,y,z) \rightarrow$ density | (e) Inverse rendering $(x,y,z) \rightarrow$ RGB, density

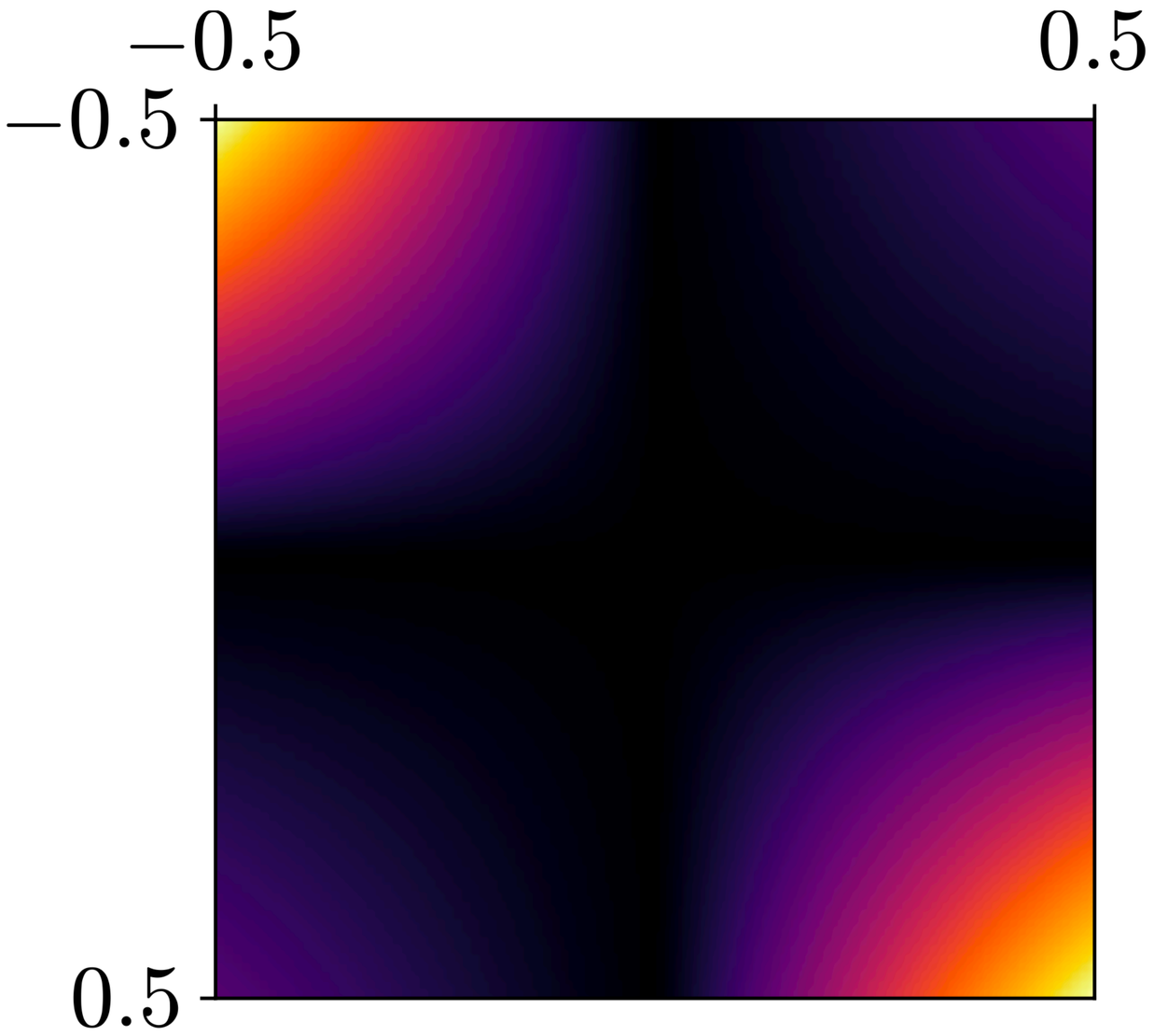
A Structured Dictionary Perspective on Implicit Neural Representations

Yüce & Ortiz-Jiménez et al. CVPR 2022



Gaussian Activated Neural Radiance Fields for High Fidelity Reconstruction & Pose Estimation

Chng et al. ECCV 2022

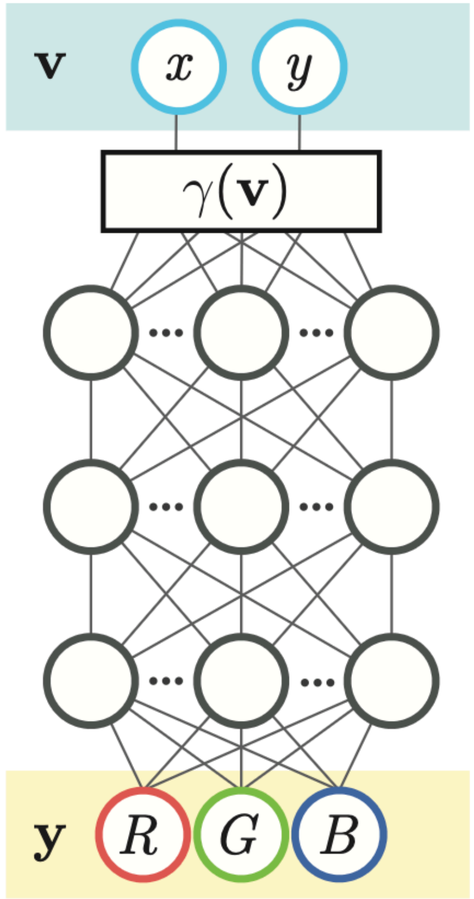


NTK kernel $k_{NTK}(x_i, x_j)$ for a 4-layer ReLU MLP with one scalar input.

From “Fourier Features Let Networks Learn High Frequency Functions in Low Dimensional Domains”, Tancik, Srinivasan, Mildenhall et al.

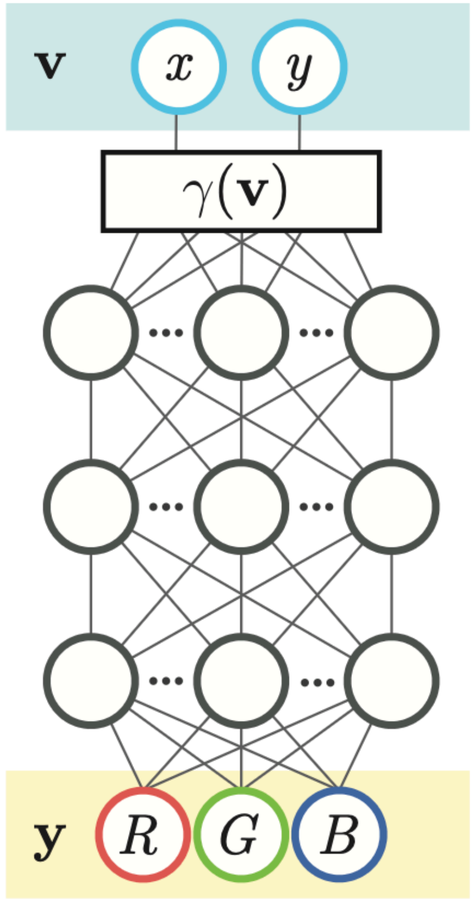
NTK of ReLU neural net is quite non-local.

Feature Embeddings



Reconstructing continuous distributions of 3D protein…, Zhong et al.; Neural Radiance Fields, Mildenhall et al.

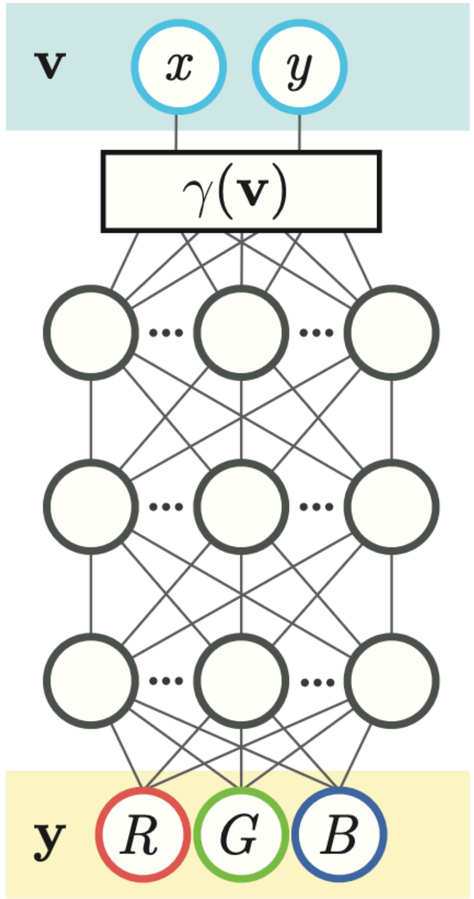
Feature Embeddings



$$\gamma(\mathbf{x}) = [\gamma_1(\mathbf{x}), \ldots, \gamma_n(\mathbf{x})]$$

Reconstructing continuous distributions of 3D protein…, Zhong et al.; Neural Radiance Fields, Mildenhall et al.

Feature Embeddings



$$\gamma(\mathbf{x}) = [\gamma_1(\mathbf{x}), \ldots, \gamma_n(\mathbf{x})]$$

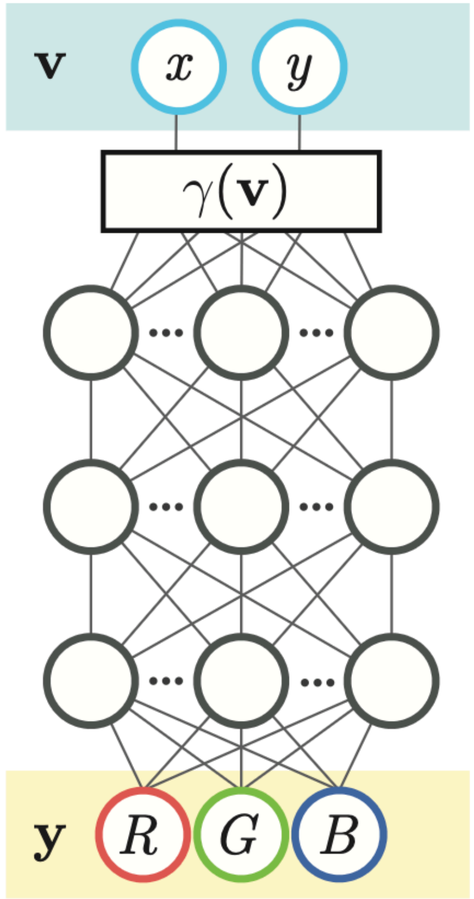
$$\gamma_{2i}(\mathbf{x}) = \sin(2^{i-1}\pi x_i)$$

$$\gamma_{(2i+1)}(\mathbf{x}) = \cos(2^{i-1}\pi x_i)$$

Sinusoidal Embeddings
Zhong et al. ICLR 2020
Mildenhall et al., ECCV 2020

Reconstructing continuous distributions of 3D protein…, Zhong et al.; Neural Radiance Fields, Mildenhall et al.

Feature Embeddings



$$\gamma(\mathbf{x}) = [\gamma_1(\mathbf{x}), \ldots, \gamma_n(\mathbf{x})]$$

$$\gamma_{2i}(\mathbf{x}) = \sin(2^{i-1}\pi x_i)$$
$$\gamma_{(2i+1)}(\mathbf{x}) = \cos(2^{i-1}\pi x_i)$$

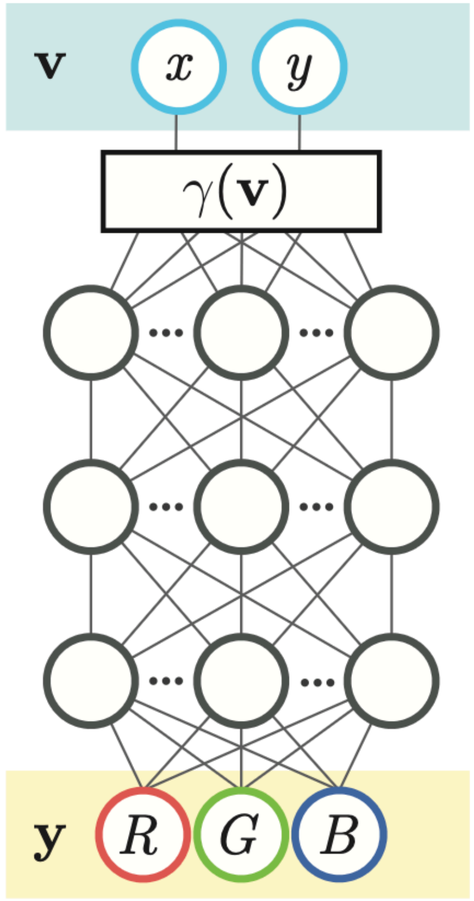
Sinusoidal Embeddings
Zhong et al. ICLR 2020
Mildenhall et al., ECCV 2020

$$\gamma(\mathbf{x}) = \exp(-\frac{\|t - x\|^2}{2\sigma^2})$$

Gaussian Embeddings
Zheng et al., arXiv 2021

Reconstructing continuous distributions of 3D protein…, Zhong et al.; Neural Radiance Fields, Mildenhall et al.

Feature Embeddings



$$\gamma(\mathbf{x}) = [\gamma_1(\mathbf{x}), \ldots, \gamma_n(\mathbf{x})]$$

$$\gamma_{2i}(\mathbf{x}) = \sin(2^{i-1}\pi x_i)$$
$$\gamma_{(2i+1)}(\mathbf{x}) = \cos(2^{i-1}\pi x_i)$$

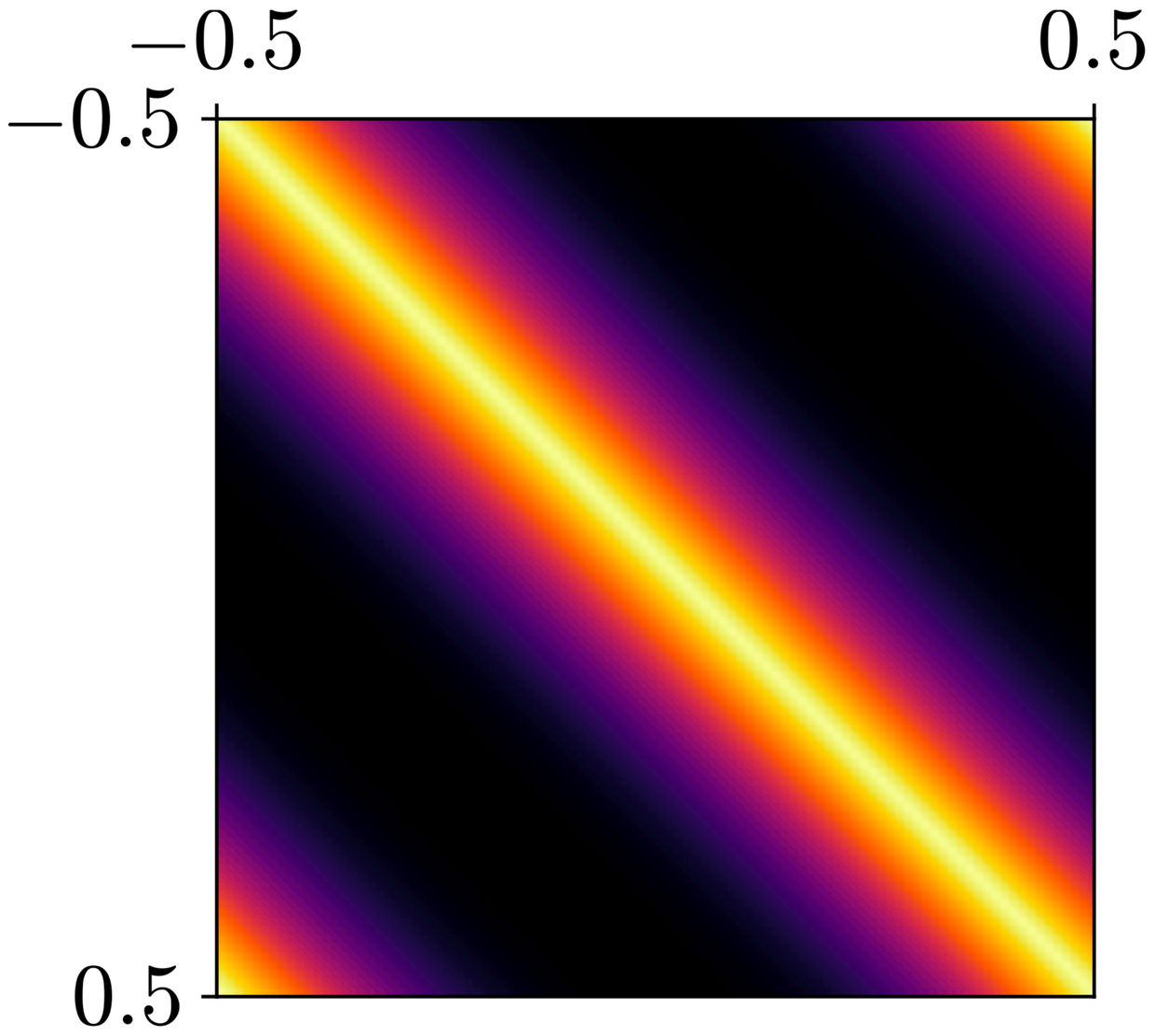
Sinusoidal Embeddings
Zhong et al. ICLR 2020
Mildenhall et al., ECCV 2020

$$\gamma(\mathbf{x}) = \exp(-\frac{\|t - x\|^2}{2\sigma^2})$$

Gaussian Embeddings
Zheng et al., arXiv 2021

• • •

Reconstructing continuous distributions of 3D protein…, Zhong et al.; Neural Radiance Fields, Mildenhall et al.



NTK kernel $k_{NTK}(\gamma(x_i), \gamma(x_j))$ for a 4-layer ReLU MLP with one scalar input and **Sinusoidal Feature Mapping**

$$\gamma(x) = [\cos(2\pi x), \sin(2\pi x)].$$

From “Fourier Features Let Networks Learn High Frequency Functions in Low Dimensional Domains”, Tancik, Srinivasan, Mildenhall et al.

NTK is local (note diagonal).

SIREN



ReLU

SIREN

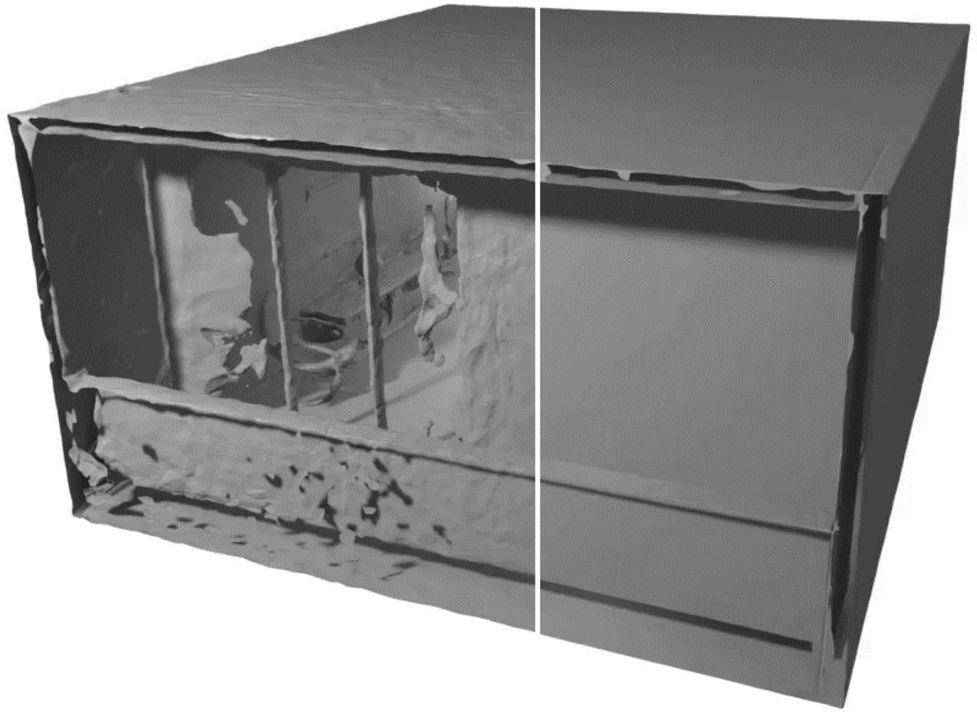
SIREN



ReLU

SIREN

SIREN

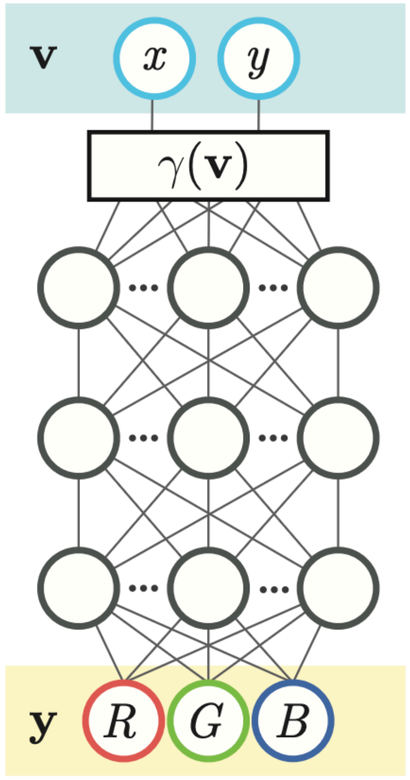


~1MB compared to
110MB of full mesh!

ReLU

SIREN

Fourier Features



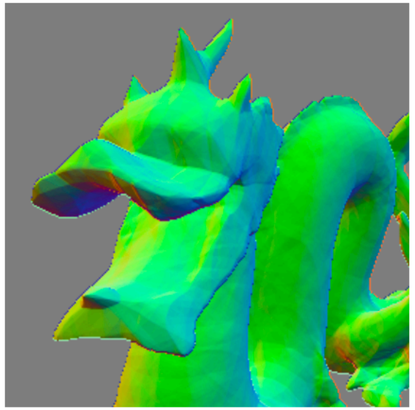
(a) Coordinate-based MLP

No Fourier features $\gamma(\mathbf{v}) = \mathbf{v}$

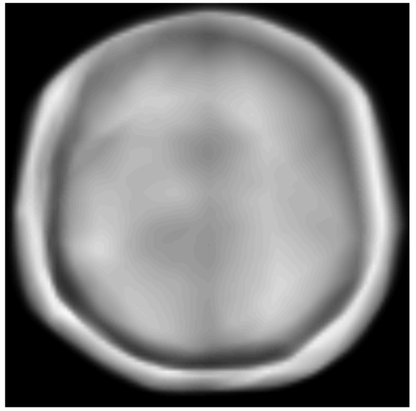
With Fourier features $\gamma(\mathbf{v}) = \mathrm{FF}(\mathbf{v})$



(b) Image regression
$(x,y) \rightarrow$ RGB



(c) 3D shape regression
$(x,y,z) \rightarrow$ occupancy

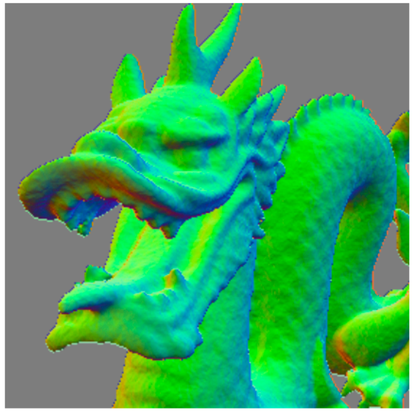


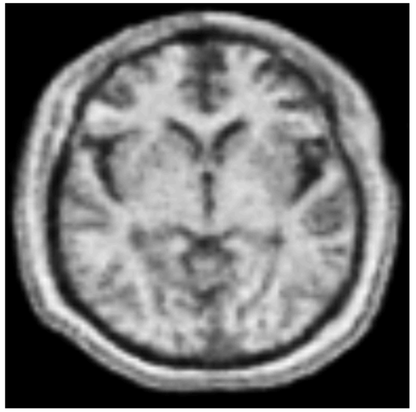
(d) MRI reconstruction
$(x,y,z) \rightarrow$ density

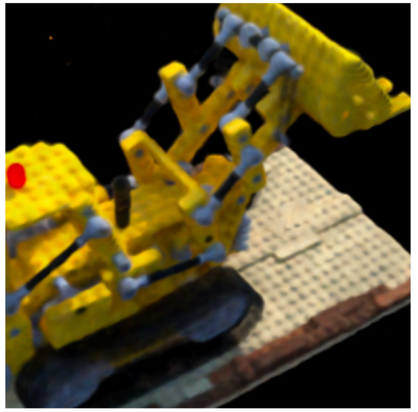


(e) Inverse rendering
$(x,y,z) \rightarrow$ RGB, density









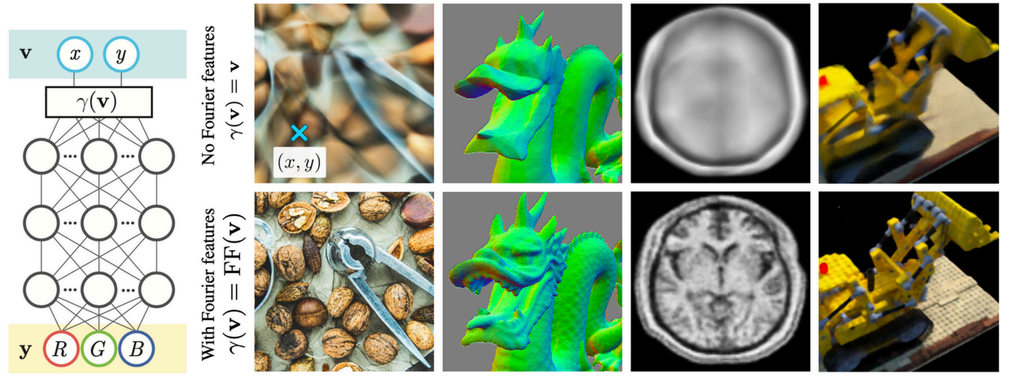
SIREN: Neural Implicit Representations With Periodic Activation Functions

Sitzmann & Martel et al. NeurIPS 2020



Fourier Features Let Networks Learn High Frequency Functions in Low Dimensional Domains

Tancik, Srinivasan, Mildenhall NeurIPS 2020



(a) Coordinate-based MLP

(b) Image regression $(x,y) \rightarrow$ RGB

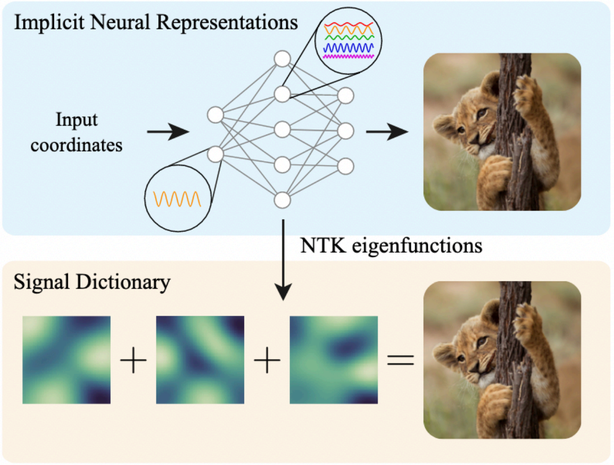
(c) 3D shape regression $(x,y,z) \rightarrow$ occupancy

(d) MRI reconstruction $(x,y,z) \rightarrow$ density

(e) Inverse rendering $(x,y,z) \rightarrow$ RGB, density

A Structured Dictionary Perspective on Implicit Neural Representations

Yüce & Ortiz-Jiménez et al. CVPR 2022



Gaussian Activated Neural Radiance Fields for High Fidelity Reconstruction & Pose Estimation

Chng et al. ECCV 2022



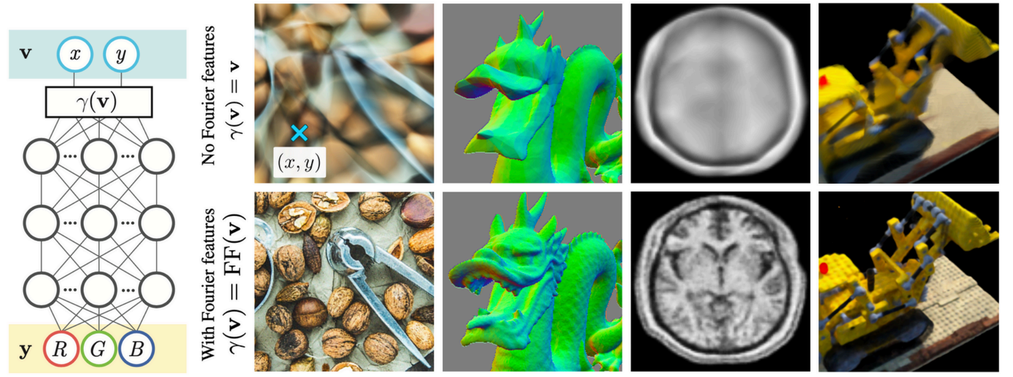
SIREN: Neural Implicit Representations With Periodic Activation Functions

Sitzmann & Martel et al. NeurIPS 2020



Fourier Features Let Networks Learn High Frequency Functions in Low Dimensional Domains

Tancik, Srinivasan, Mildenhall NeurIPS 2020



(a) Coordinate-based MLP

(b) Image regression $(x,y) \rightarrow$ RGB

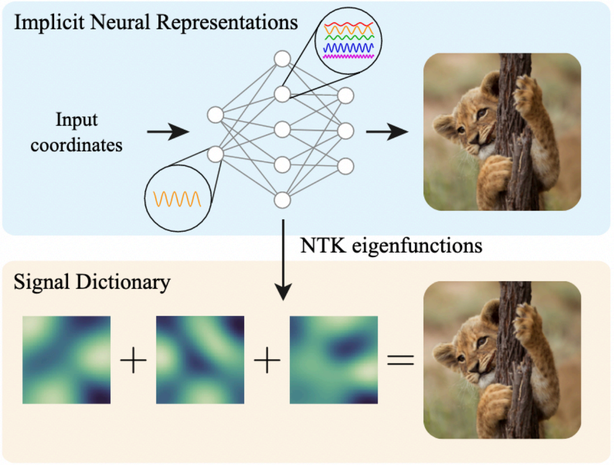
(c) 3D shape regression $(x,y,z) \rightarrow$ occupancy

(d) MRI reconstruction $(x,y,z) \rightarrow$ density

(e) Inverse rendering $(x,y,z) \rightarrow$ RGB, density

A Structured Dictionary Perspective on Implicit Neural Representations

Yüce & Ortiz-Jiménez et al. CVPR 2022



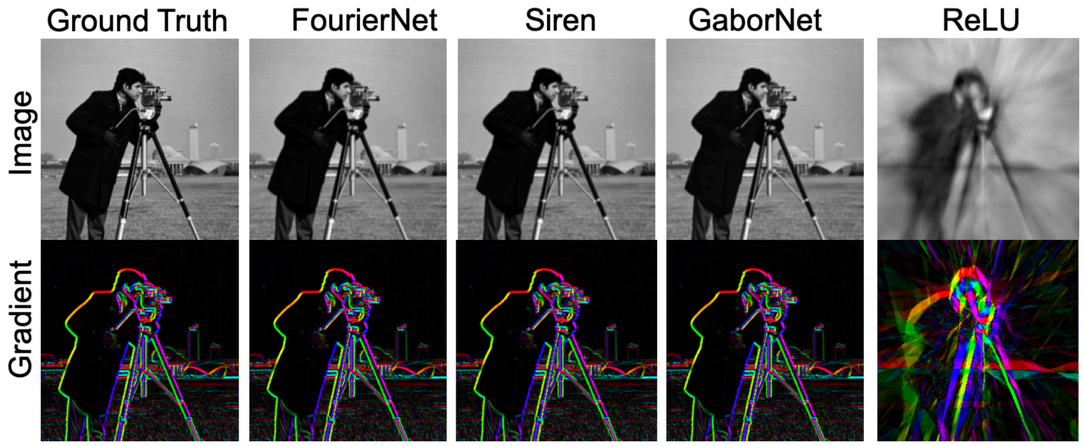
Gaussian Activated Neural Radiance Fields for High Fidelity Reconstruction & Pose Estimation

Chng et al. ECCV 2022

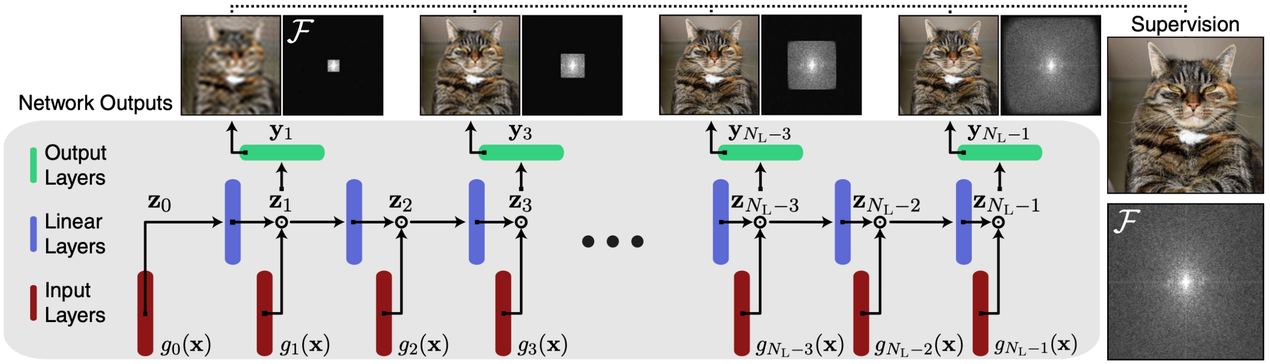


Neural (?) Fields with analytical (but not tractably computable) Fourier spectrum!

Multiplicative Filter Networks
Fathony et al. ICLR 2021

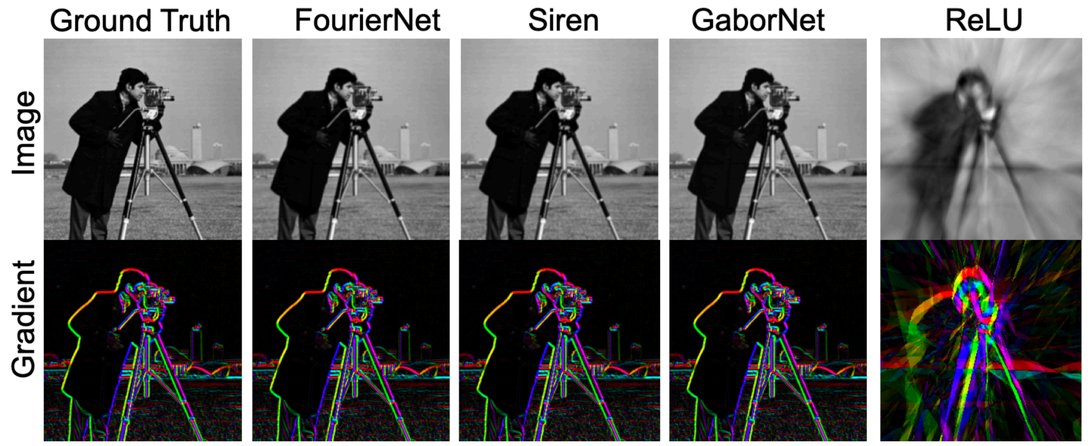


BACON: Band-limited coordinate networks for multiscale scene representation
Lindell et al. CVPR 2022

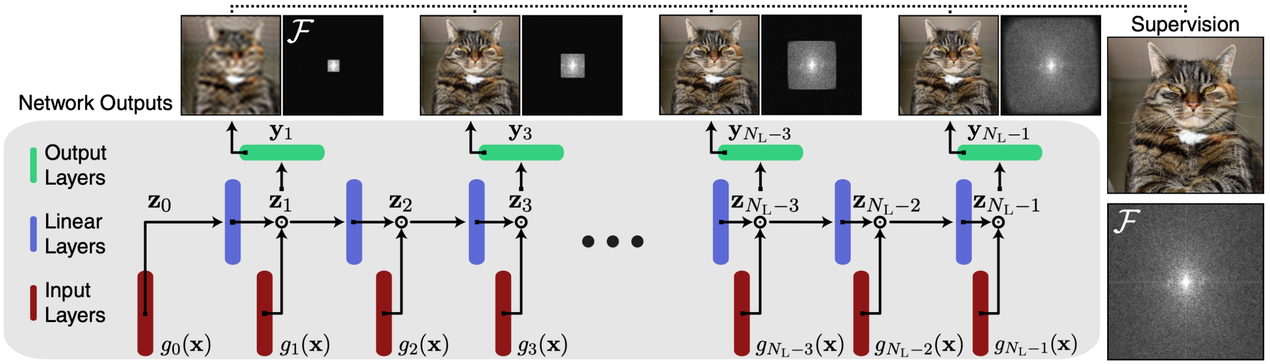


Neural (?) Fields with analytical (but not tractably computable) Fourier spectrum!

Multiplicative Filter Networks
Fathony et al. ICLR 2021



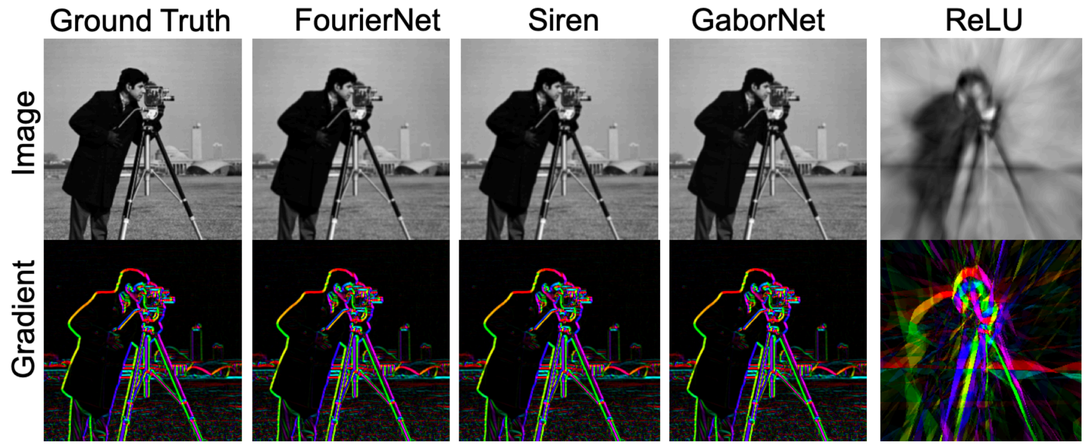
BACON: Band-limited coordinate networks for multiscale scene representation
Lindell et al. CVPR 2022



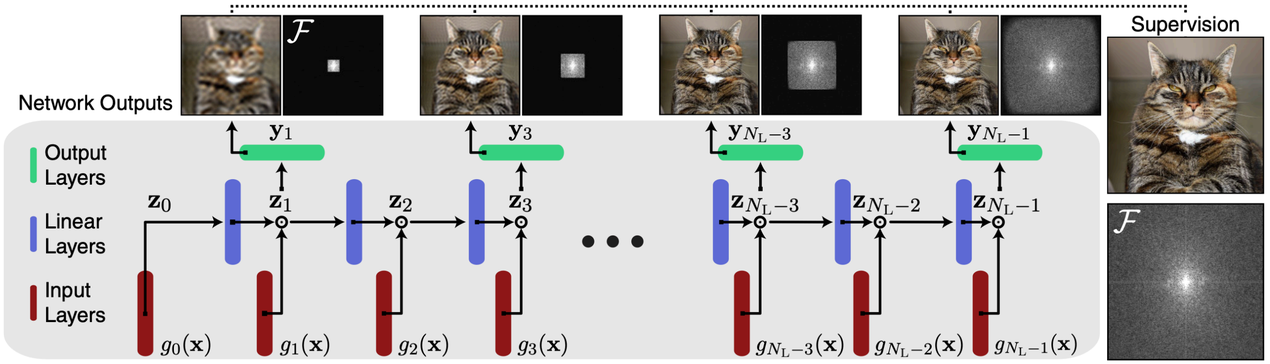
• • •

Neural (?) Fields with analytical (but not tractably computable) spectrum!

Multiplicative Filter Networks
Fathony et al. ICLR 2021



BACON: Band-limited coordinate networks for multiscale scene representation
Lindell et al. CVPR 2022



arXiv:2111.11426v4 [cs.CV] 5 Apr 2022

EUROGRAPHICS 2022
D. Meneveaux and G. Patanè
(Guest Editors)

Volume 41 (2022), Number 2
STAR – State of The Art Report

Neural Fields in Visual Computing and Beyond

Yiheng Xie[1,2] Towaki Takikawa[3,4] Shunsuke Saito[5] Or Litany[4] Shiqin Yan[1] Numair Khan[1] Federico Tombari[6,7]
James Tompkin[1] Vincent Sitzmann[8†] Srinath Sridhar[1†]

[1]Brown University [2]Unity Technologies [3]University of Toronto [4]NVIDIA [5]Meta Reality Labs Research [6]Google [7]Technical University of Munich
[8]Massachusetts Institute of Technology [†]*Equal advising*

https://neuralfields.cs.brown.edu/

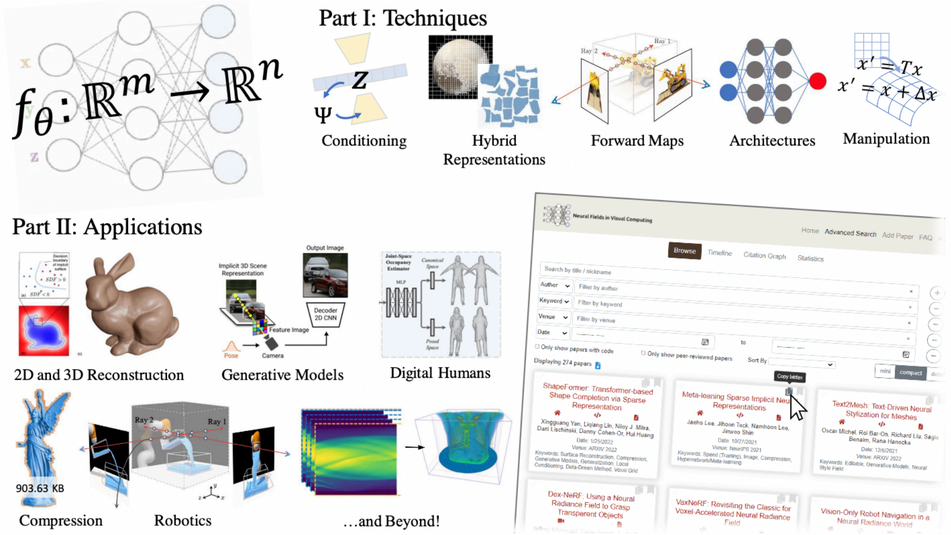
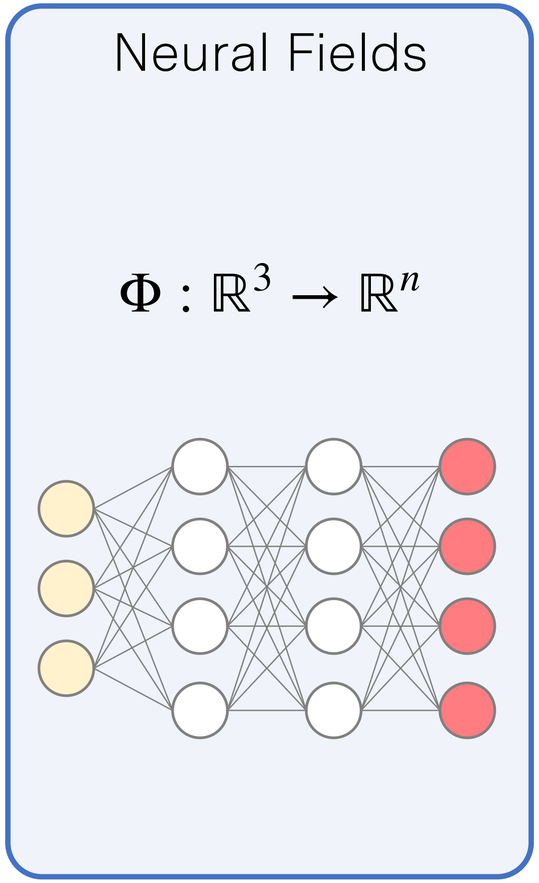


Figure 1: **Contribution of this report.** Following a survey of over 250 papers, we provide a review of (**Part I**) techniques in neural fields such as prior learning and conditioning, representations, forward maps, architectures, and manipulation, and of (**Part II**) applications in visual computing including 2D image processing, 3D scene reconstruction, generative modeling, digital humans, compression, robotics, and beyond. This report is complemented by a community-driven website with search, filtering, bibliographic, and visualization features.

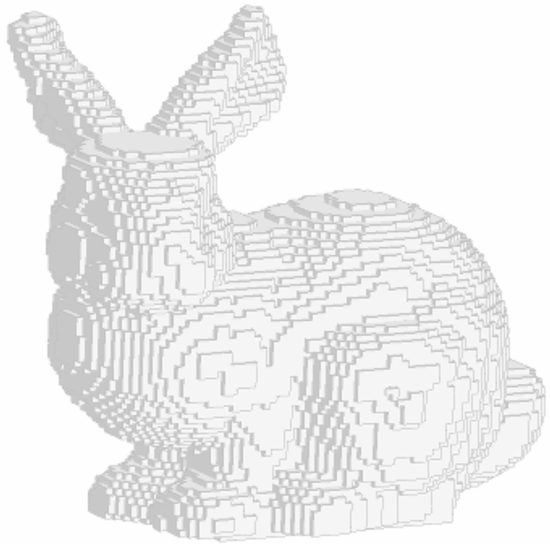
Neural Fields

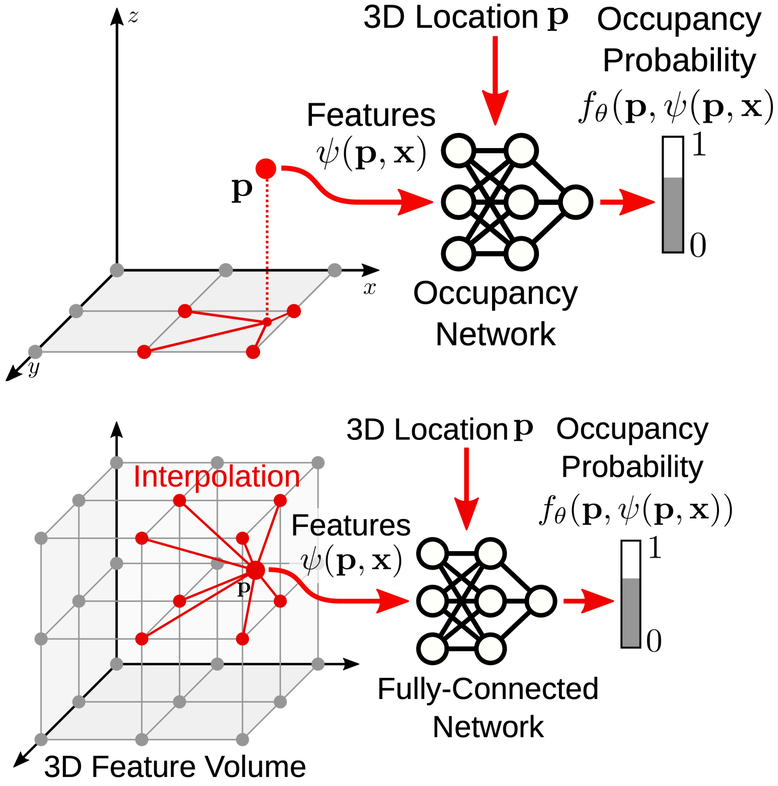
- Storage memory **does not grow with spatial resolution or number of spatial dimensions**.
- **Slow** sampling: Each query takes a forward pass through the neural net. Means GPU-memory intensive forward passes.
- Does **not** expose locality: Can't identify set of parameters / direction in parameter space that encodes particular spatial location.
- **Inconvenient processing**: cannot run convolutions.
- But: automatic adaptive resolution. Over optimization, will converge to assign more compute to higher-frequency areas.



Note: Despite their name, has nothing to do with Machine Learning itself.
It's as "smart" as a voxelgrid.

Hybrid discrete / continuous reps



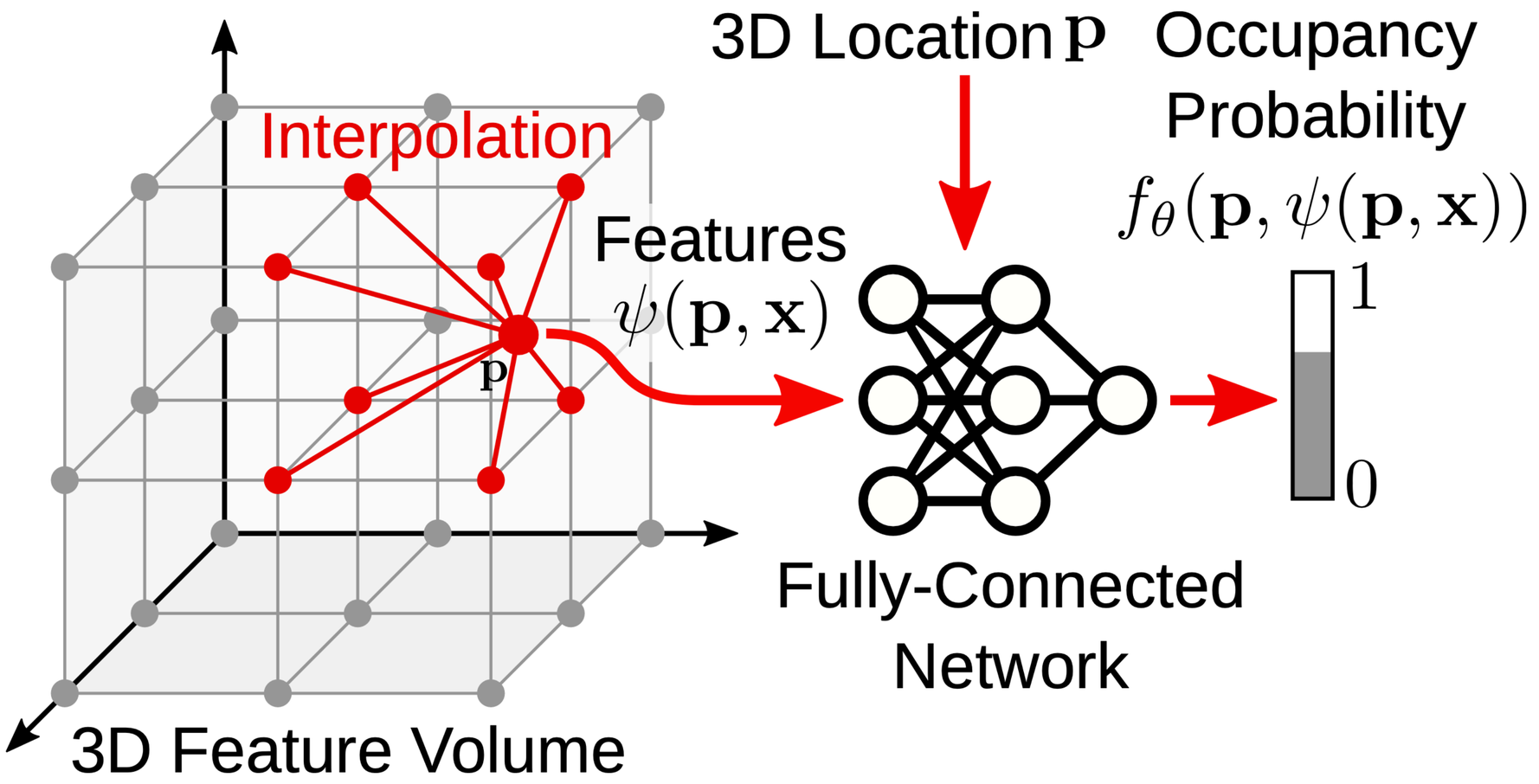


Discrete Parameterizations

Continuous Parameterizations

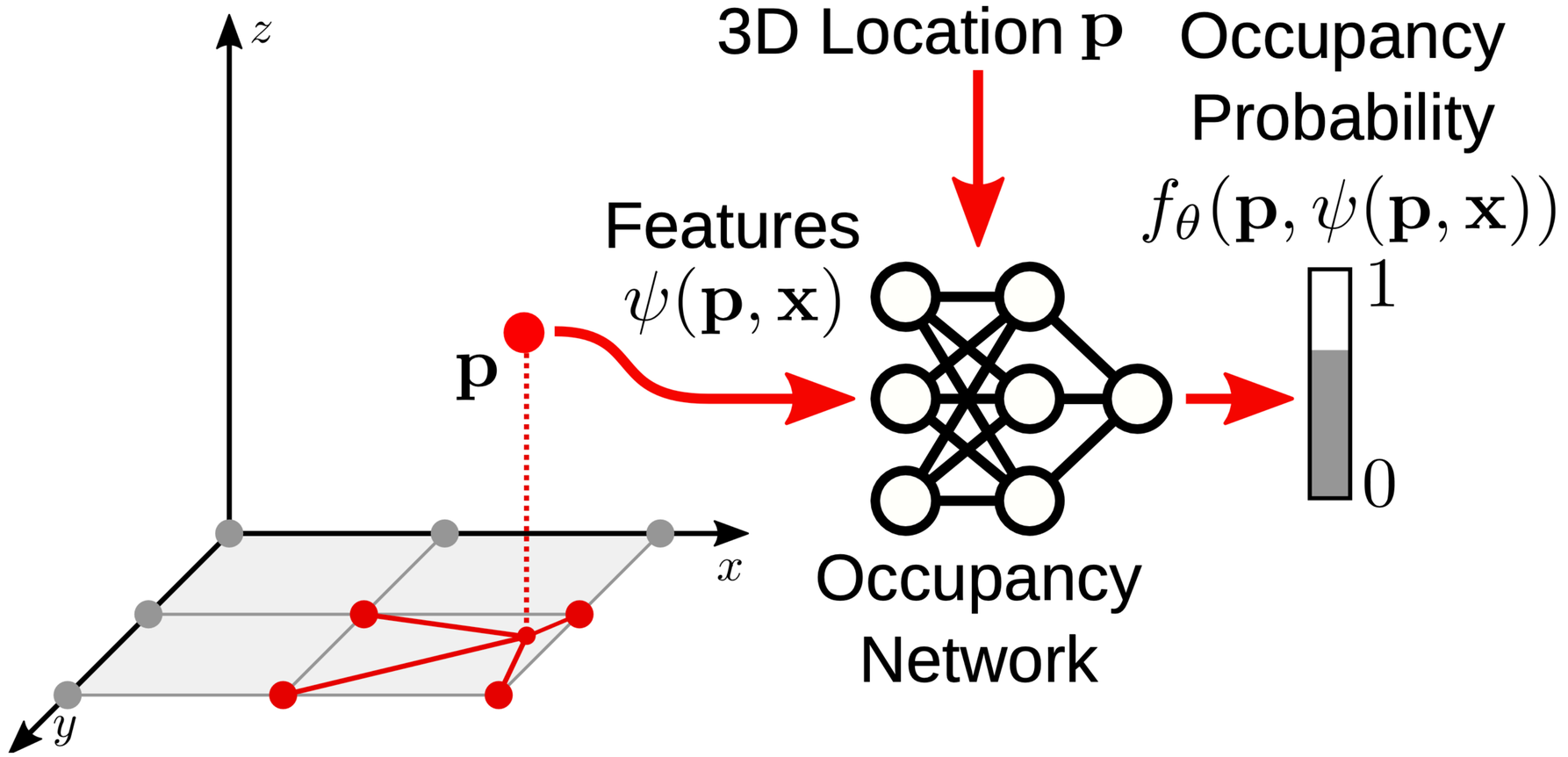
Hybrid Parameterizations

Basic idea: Use Neural Net as Interpolation Kernel



Convolutional Occupancy Networks [Peng et al. 2020]
Local Implicit Grid Representations for 3D Scenes [Jiang et al. 2020]
Implicit Functions in Feature Space for 3D Shape Reconstruction and Completion [Chibane et al. 2020]
Deep Local Shapes: Learning Local SDF Priors for Detailed 3D Reconstruction [Chabra et al. 2020]

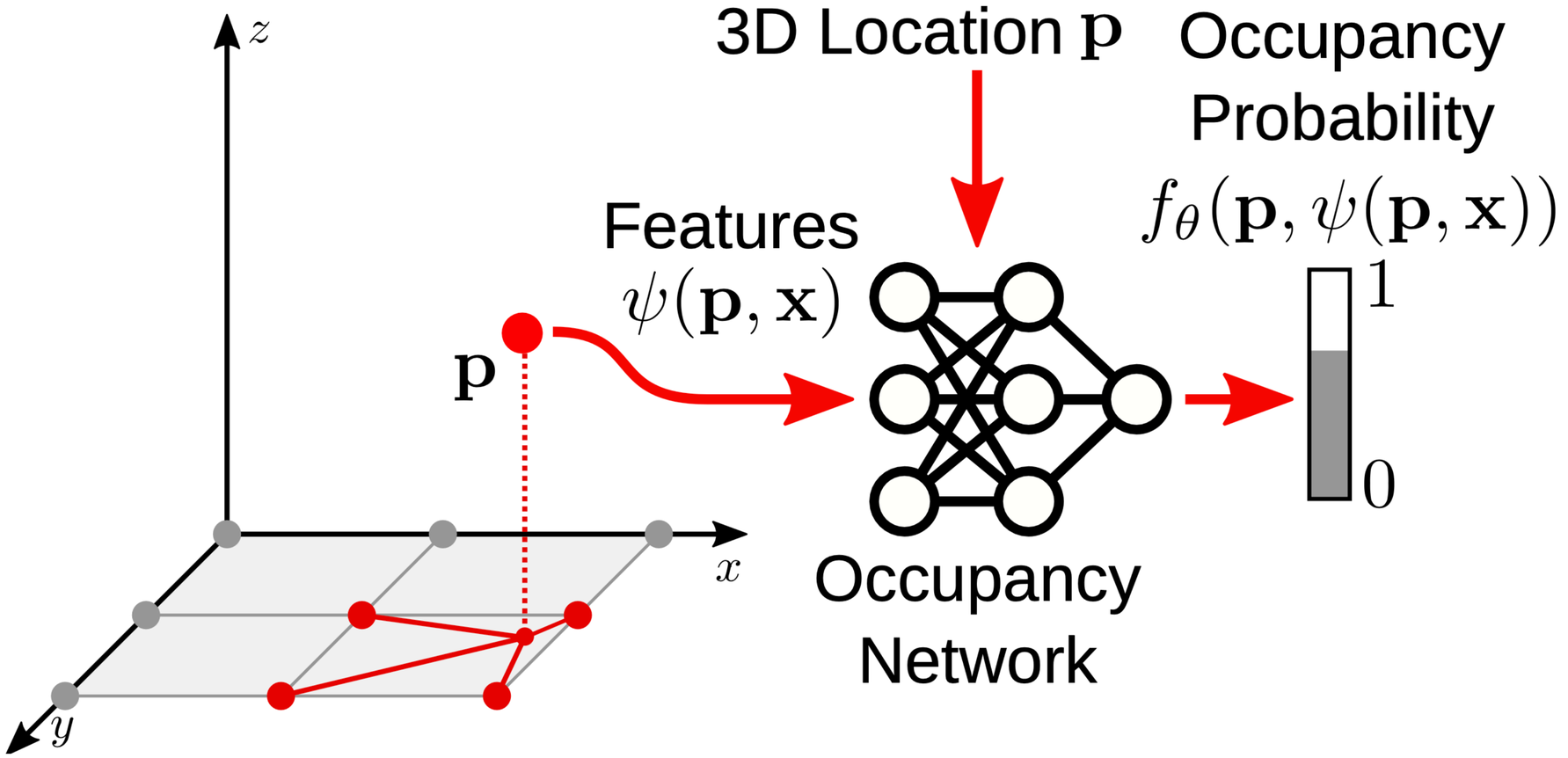
Ground plan & Orthographic Projection



Any assumptions made on scene? Any limitations?

Convolutional Occupancy Networks [Peng et al. 2020]

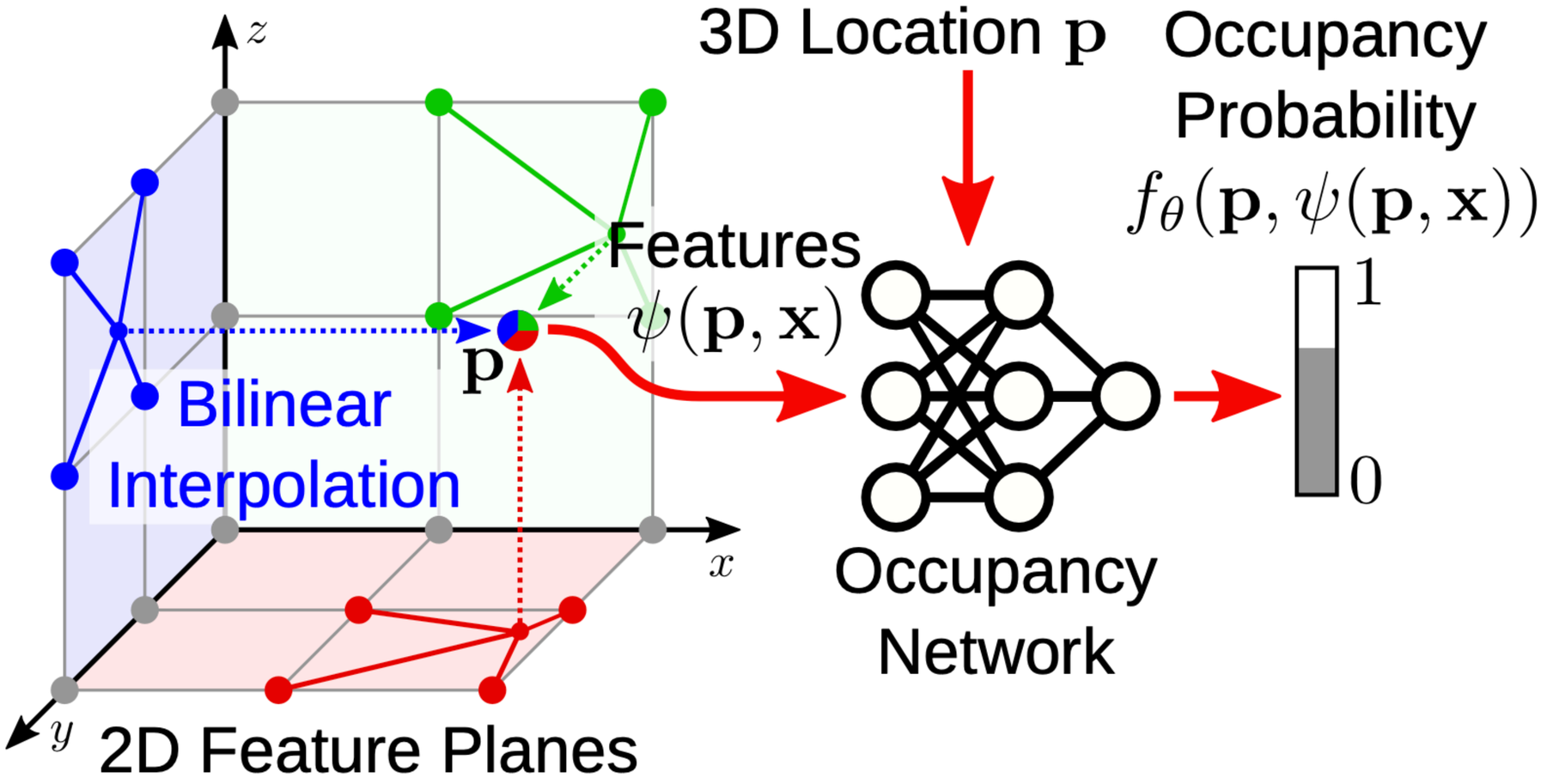
Ground plan & Orthographic Projection



No. You can still represent *any scene* - some info stored in NN, some in grid.

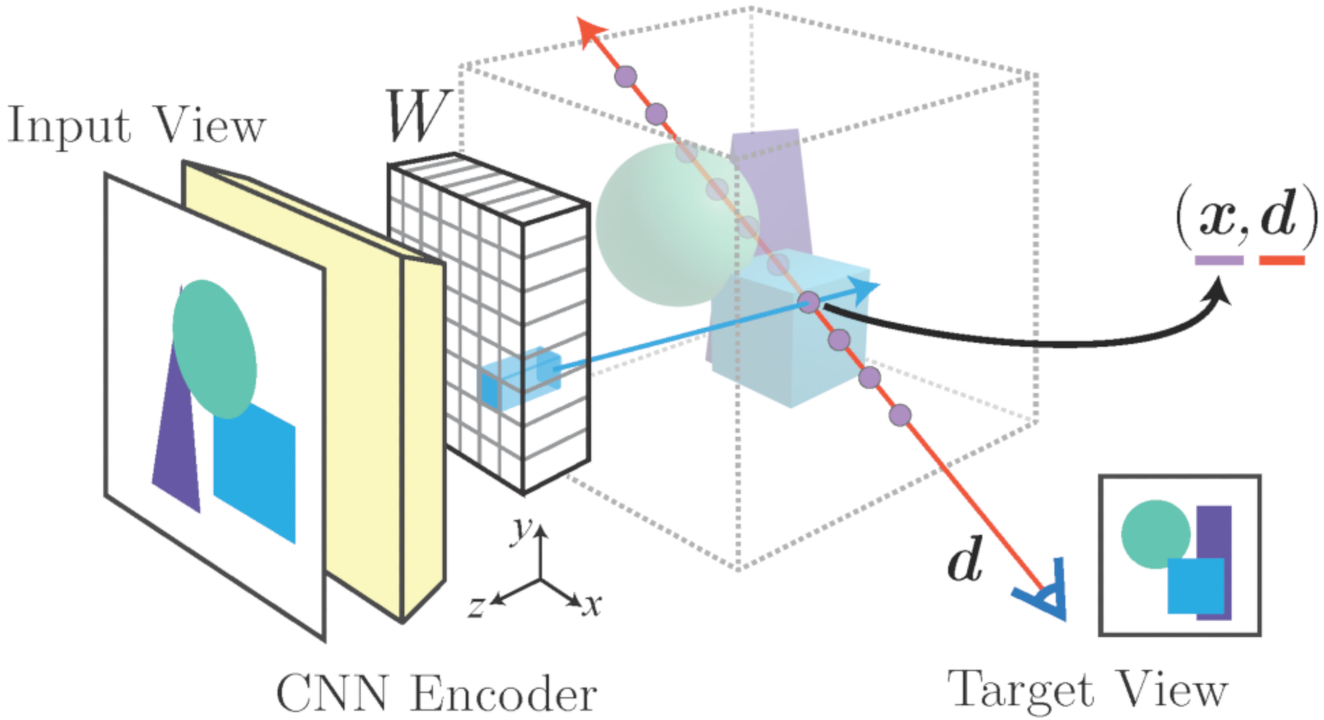
Convolutional Occupancy Networks [Peng et al. 2020]

Tri-Planes and Orthographic Projection



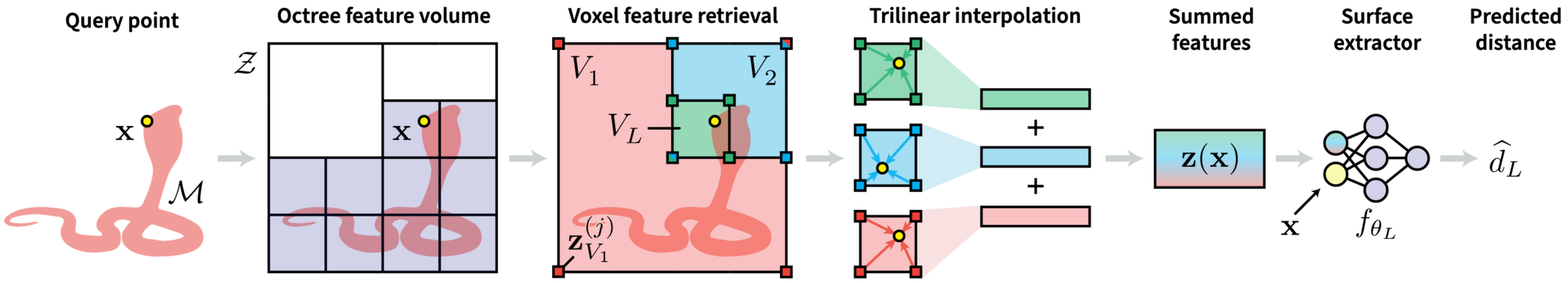
Convolutional Occupancy Networks [Peng et al. 2020]

Image Grids & Perspective Projection



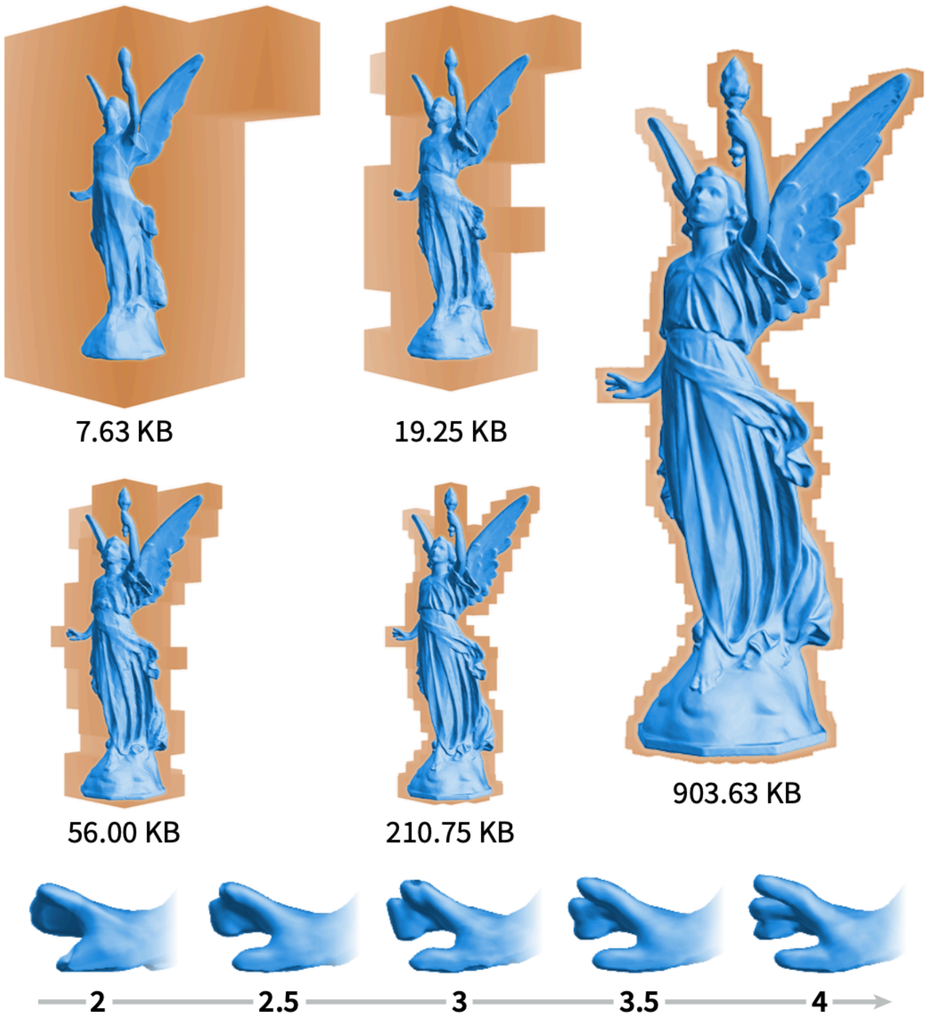
PixelNeRF (Yu et al. 2020)

Multi-scale Voxel Grids



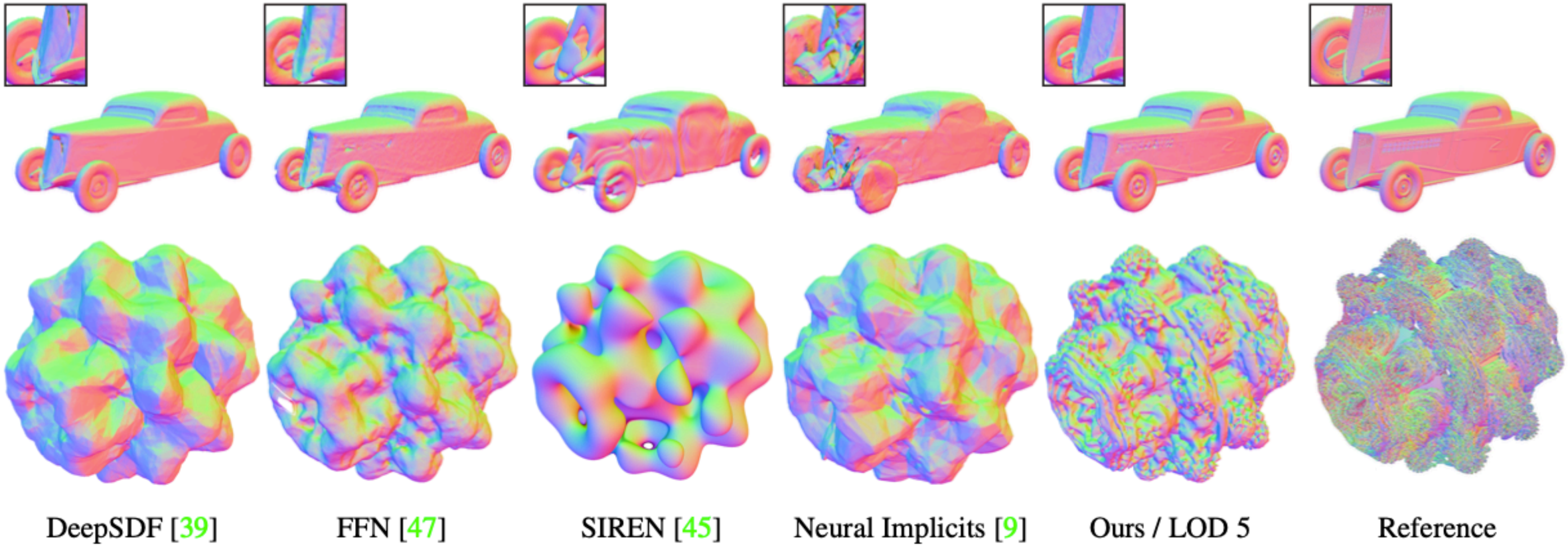
Neural Geometric Level of Detail: Real-time Rendering with Implicit 3D Shapes, Takikawa, Litalien et al. 2021

Multi-scale Voxel Grids



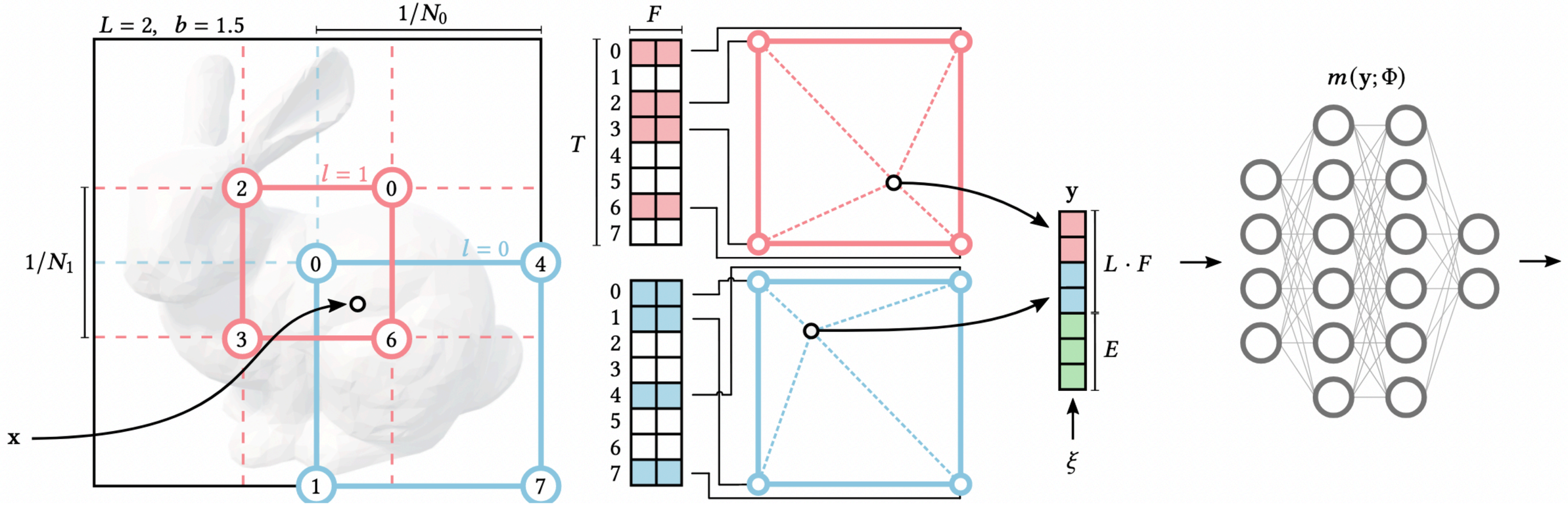
Neural Geometric Level of Detail: Real-time Rendering with Implicit 3D Shapes, Takikawa, Litalien et al. 2021

Multi-scale Voxel Grids



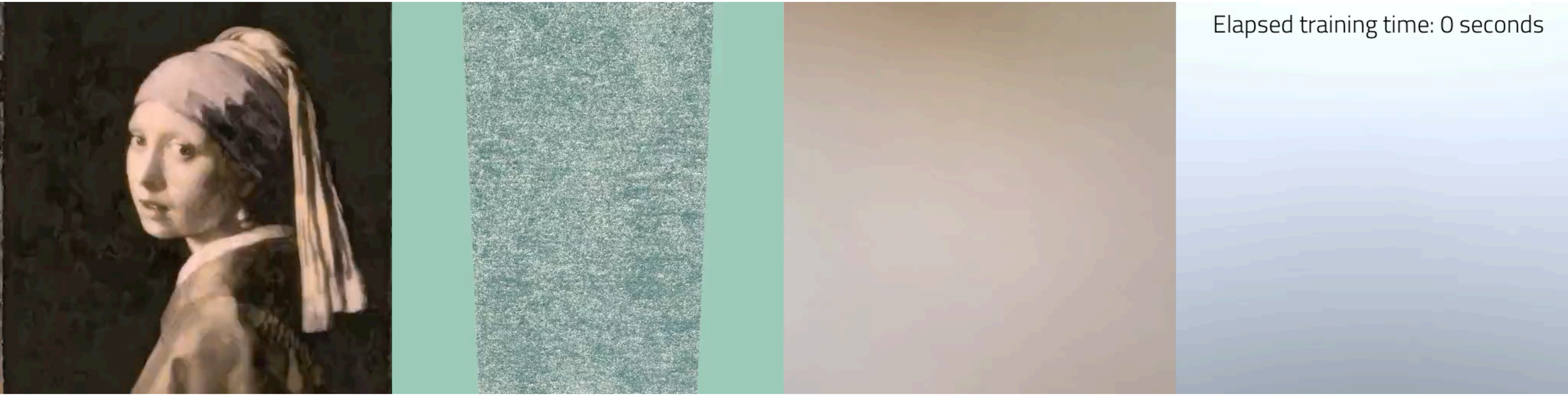
Neural Geometric Level of Detail: Real-time Rendering with Implicit 3D Shapes, Takikawa, Litalien et al. 2021

Multi-scale Voxel Grids + Hash Table



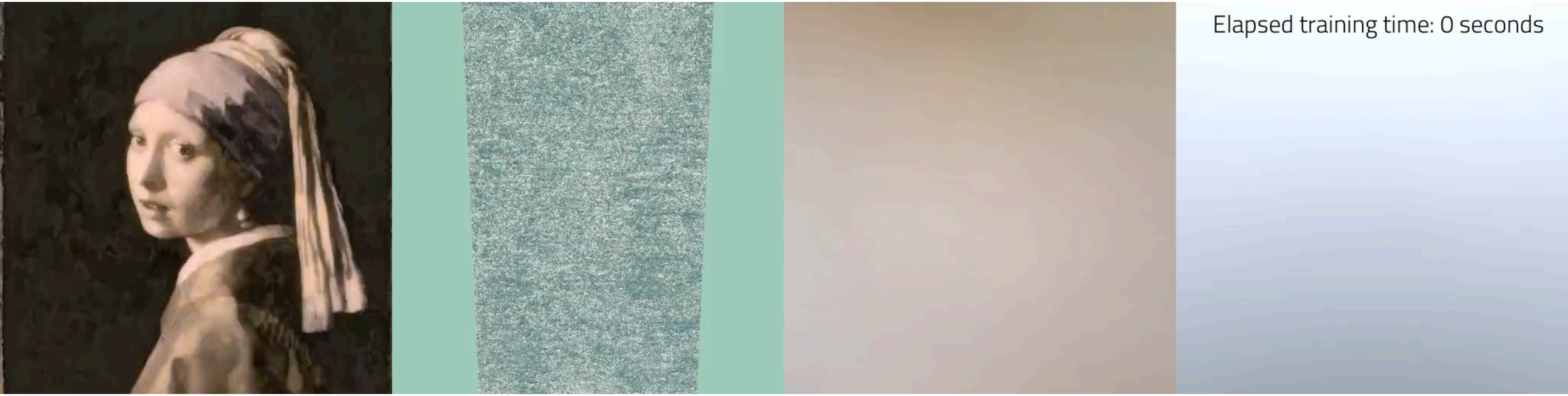
Instant Neural Graphics Primitives with a Multiresolution Hash Encoding, Müller et al. 2022

Multi-scale Voxel Grids + Hash Table



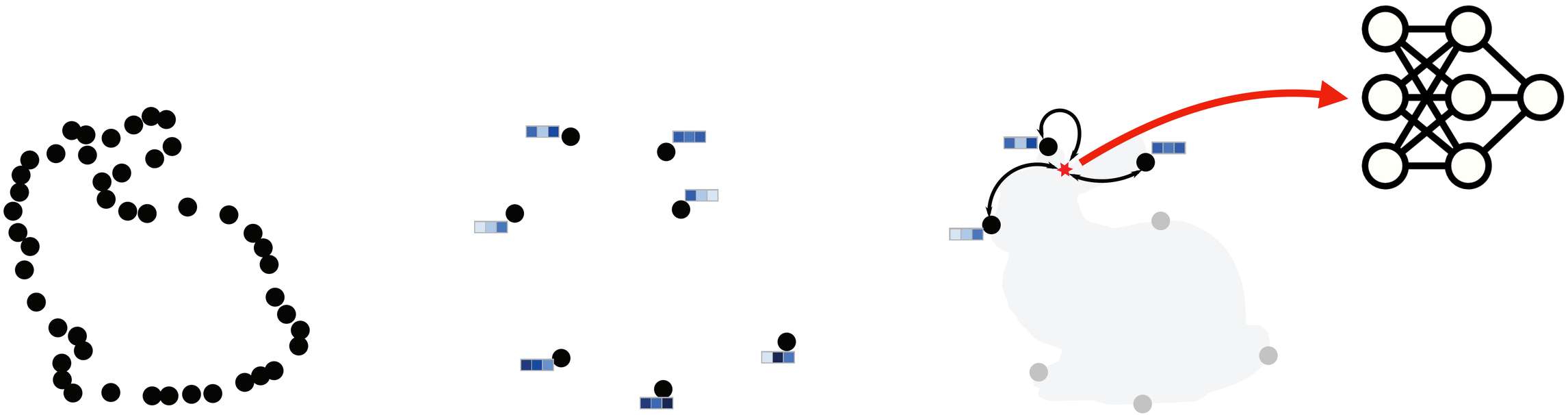
Instant Neural Graphics Primitives with a Multiresolution Hash Encoding, Müller et al. 2022

Multi-scale Voxel Grids + Hash Table



Instant Neural Graphics Primitives with a Multiresolution Hash Encoding, Müller et al. 2022

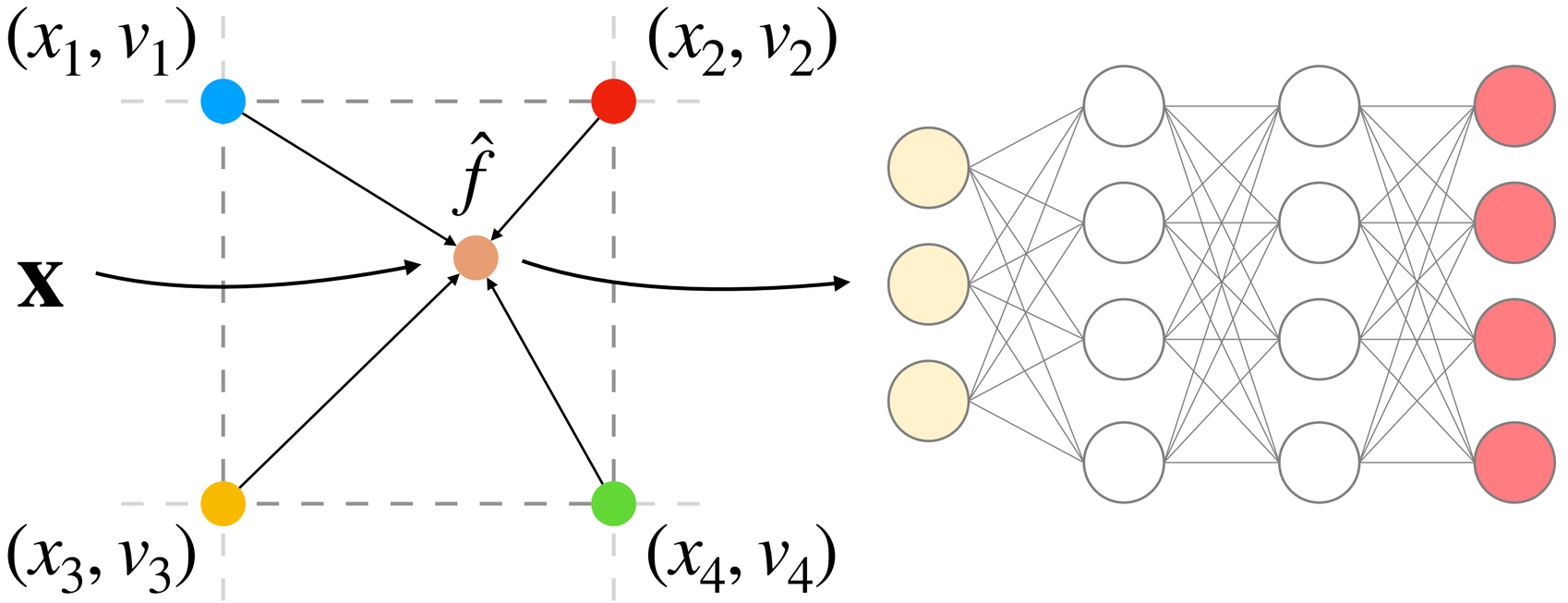
Point Cloud & Nearest Neighbor



Hybrid representations can be used to decode a surface representation into a volume representation!

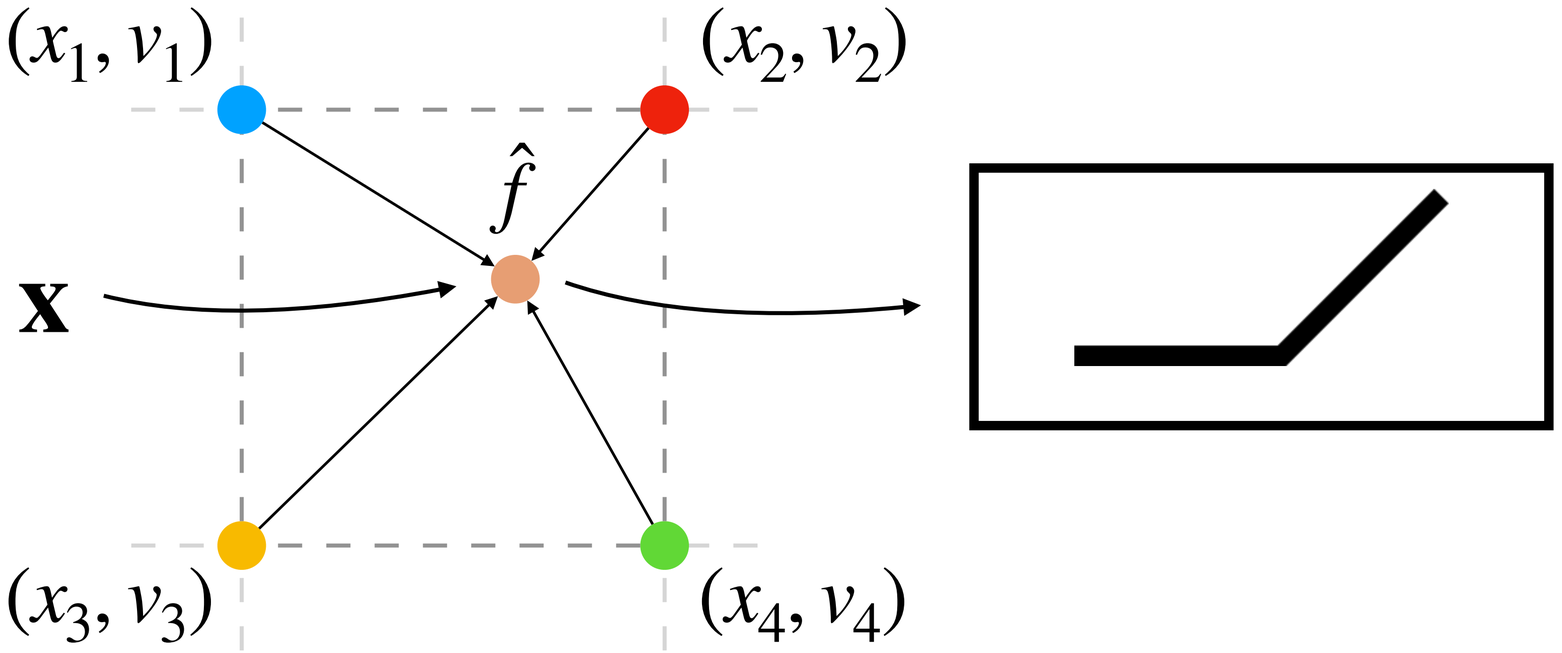
AirNet (Giebenhain et al. 2021)

ReLU Fields: How much MLP do you need?



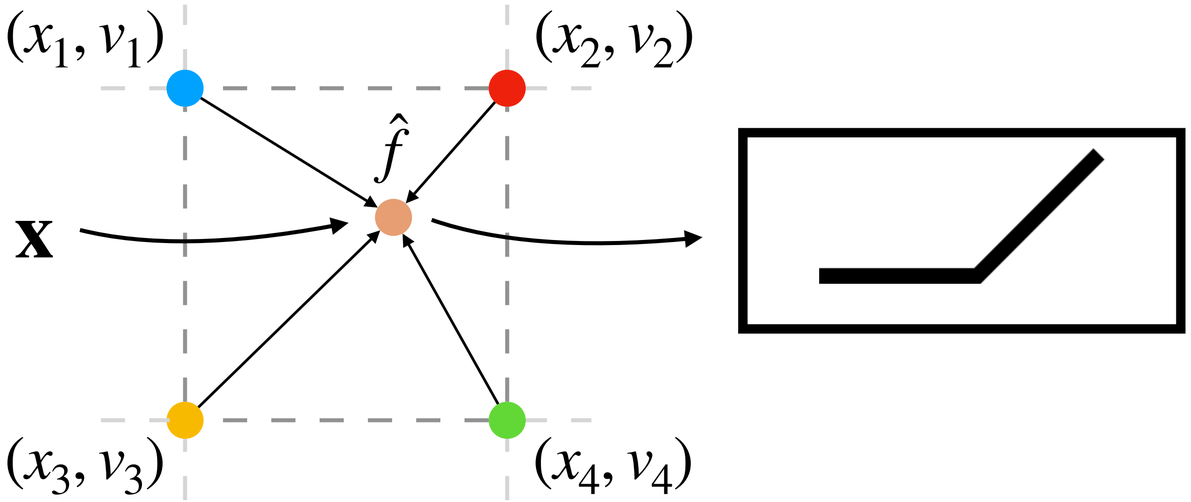
ReLU Fields: The Little Non-linearity That Could (Karnewar et al. 2022)

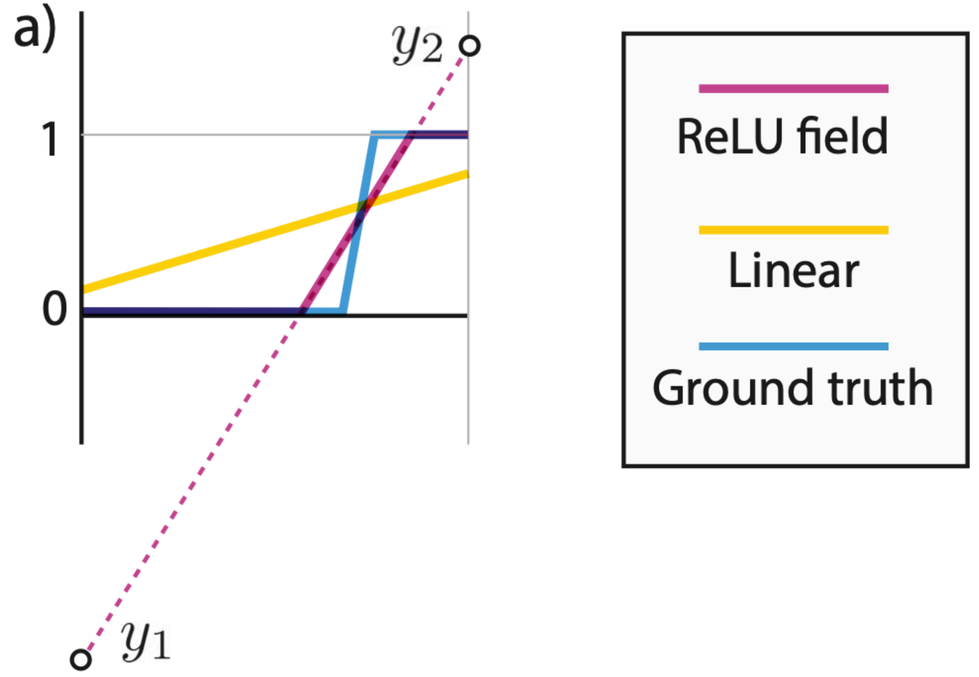
ReLU Fields: How much MLP do you need?



ReLU Fields: The Little Non-linearity That Could (Karnewar et al. 2022)

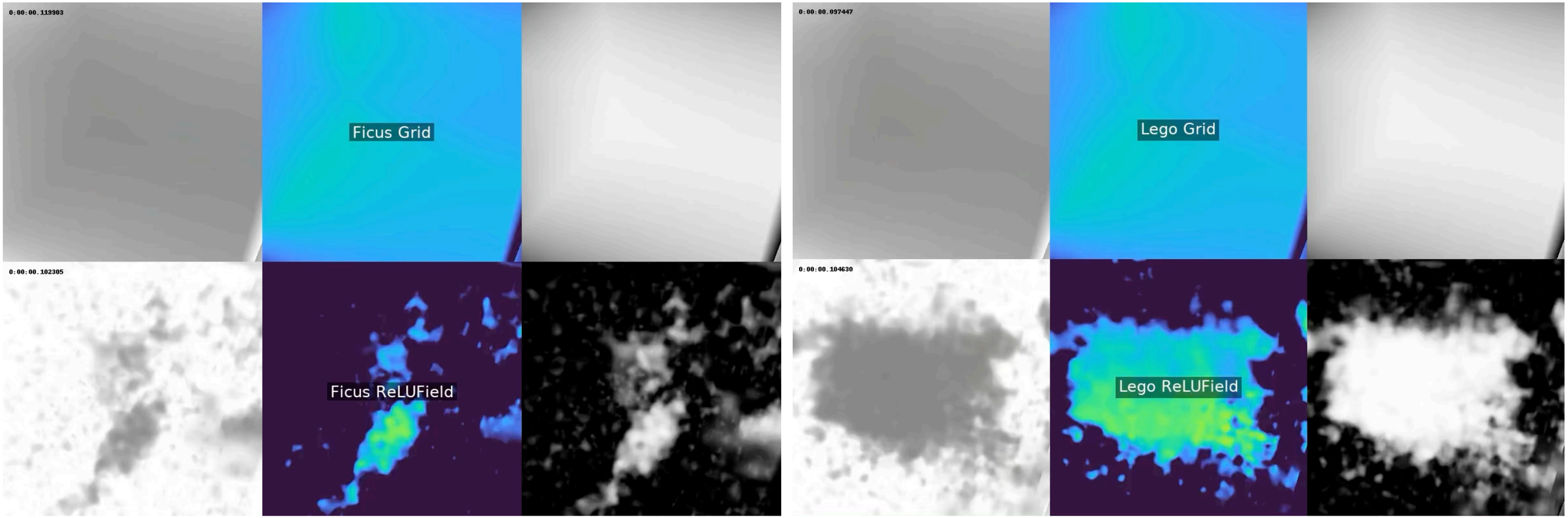
ReLU Fields: How much MLP do you need?





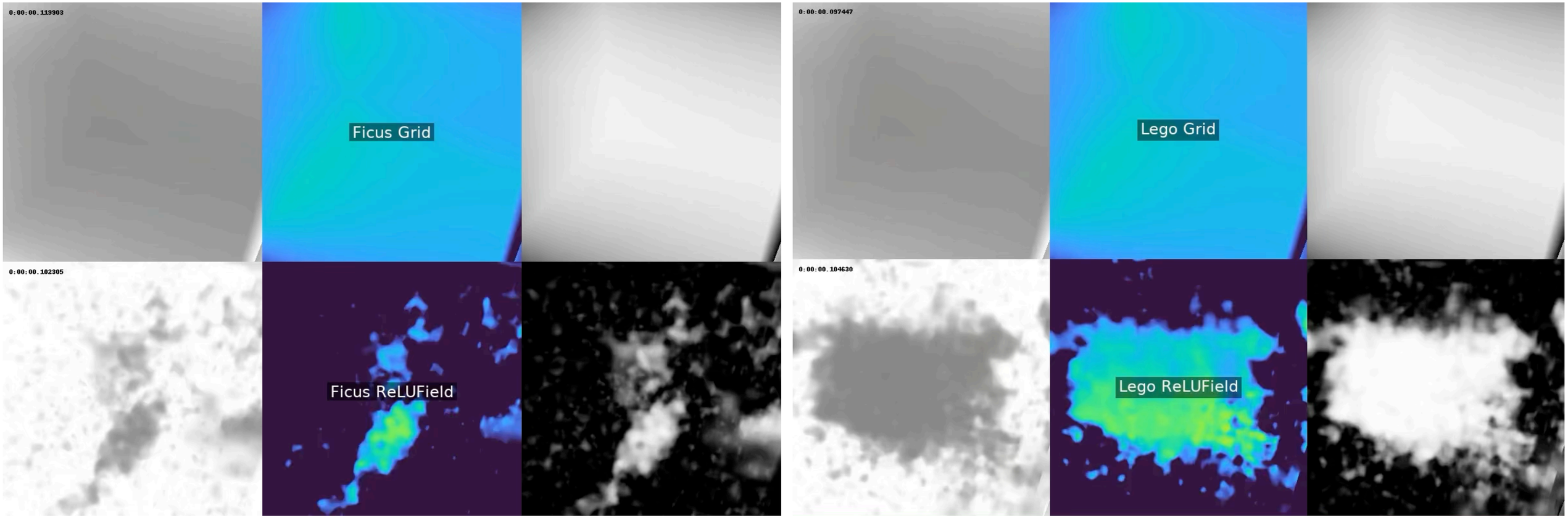
ReLU Fields: The Little Non-linearity That Could (Karnewar et al. 2022)

ReLU Fields: How much MLP do you need?



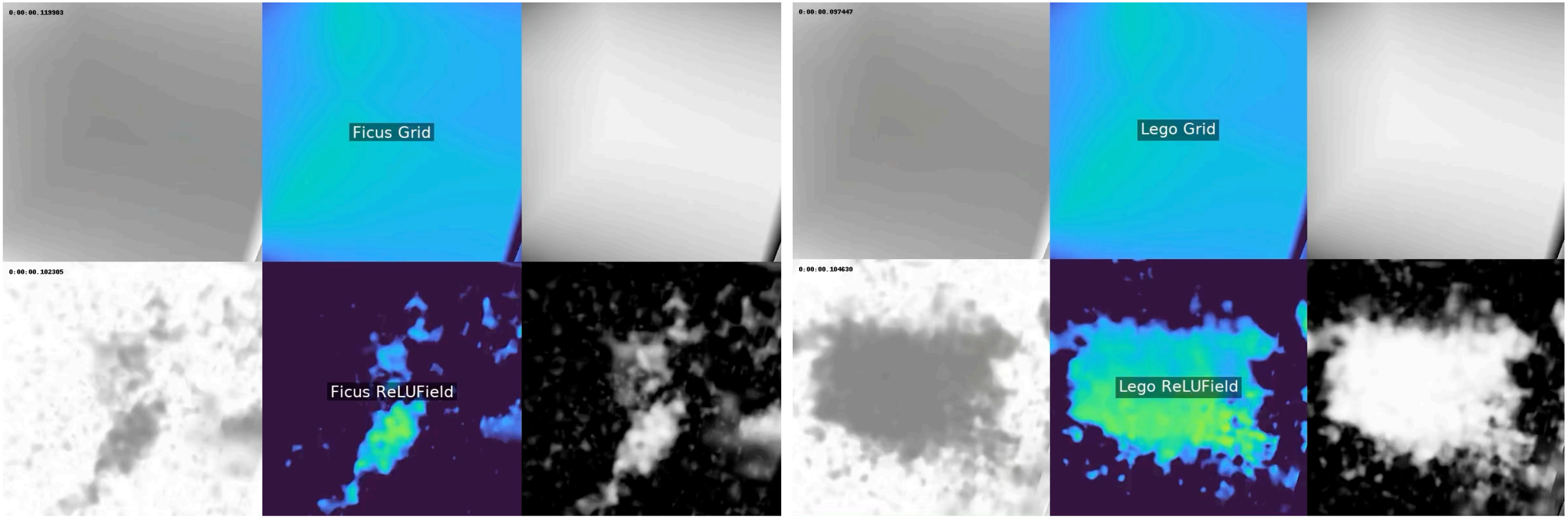
ReLU Fields: The Little Non-linearity That Could (Karnewar et al. 2022)

ReLU Fields: How much MLP do you need?



ReLU Fields: The Little Non-linearity That Could (Karnewar et al. 2022)

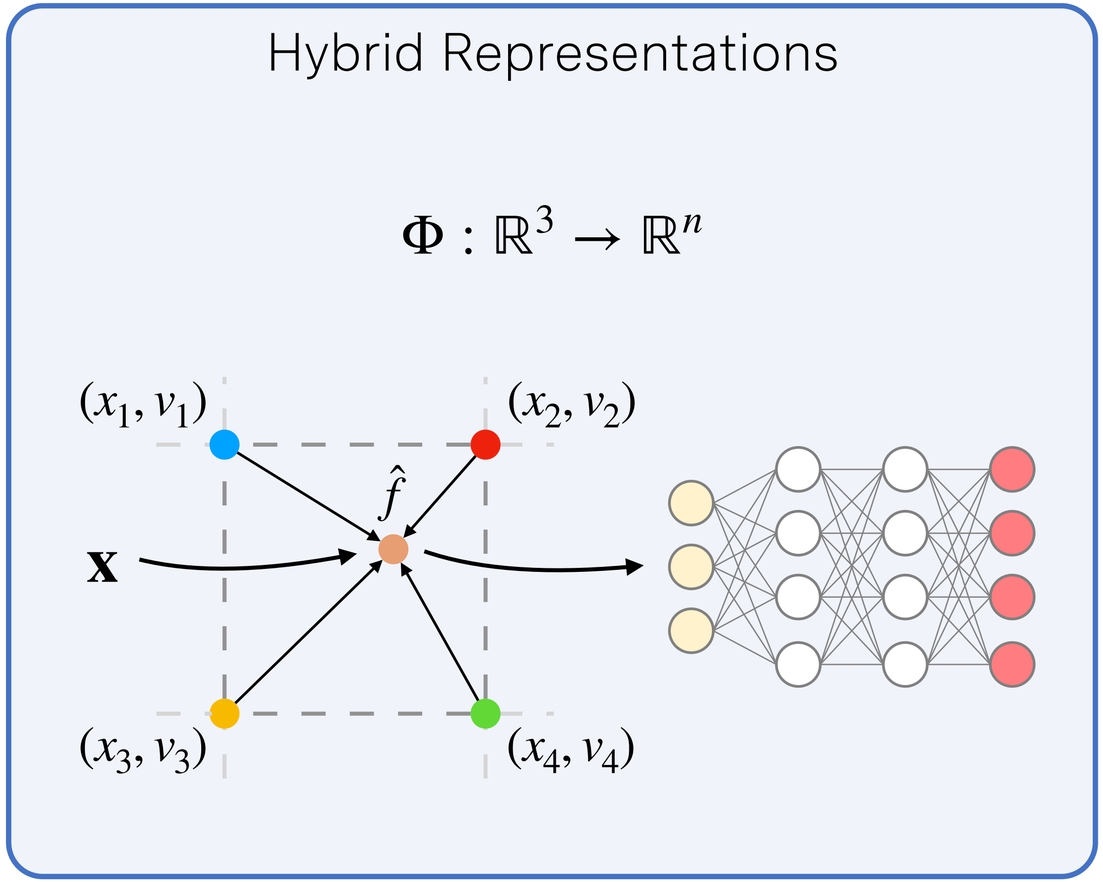
ReLU Fields: How much MLP do you need?



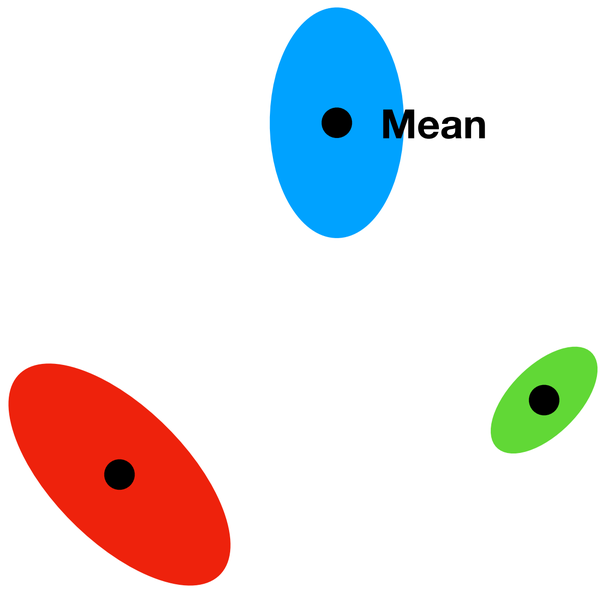
ReLU Fields: The Little Non-linearity That Could (Karnewar et al. 2022)

Hybrid Representations

- Very natural to trade off memory with compute.
- Neural network can locally “super resolve” discrete representation: Basically acts as interpolation kernel.
- Can be the best of both worlds: Fast and “continuous”.

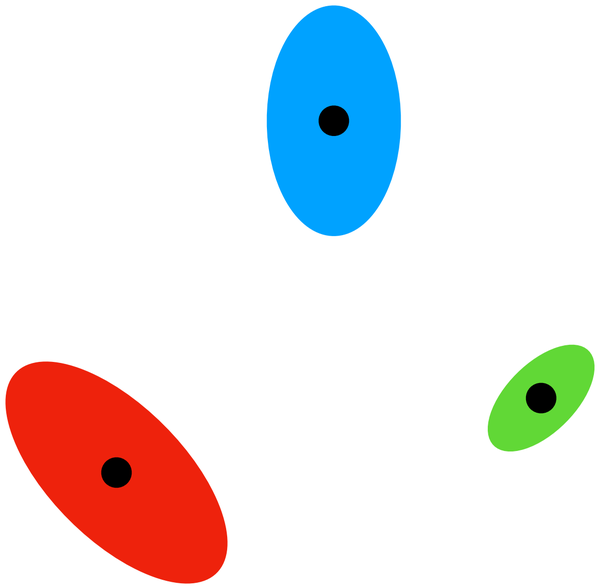


Primitive-based representations



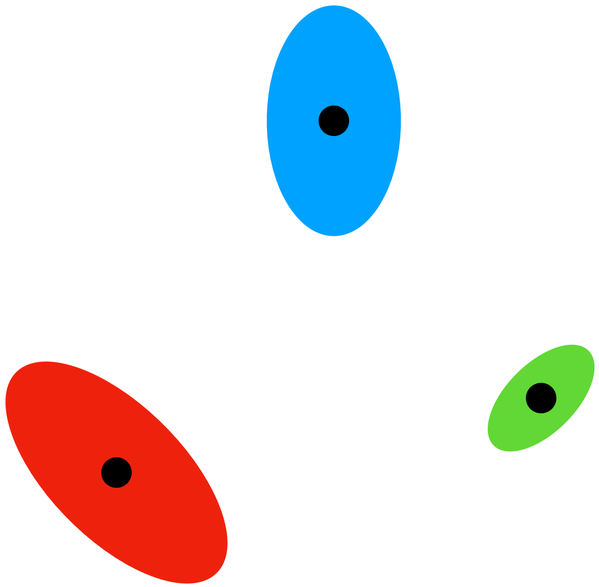
3D Gaussian Blobs floating in Space

Primitive-based representations



3D Gaussian Blobs floating in Space

Primitive-based representations

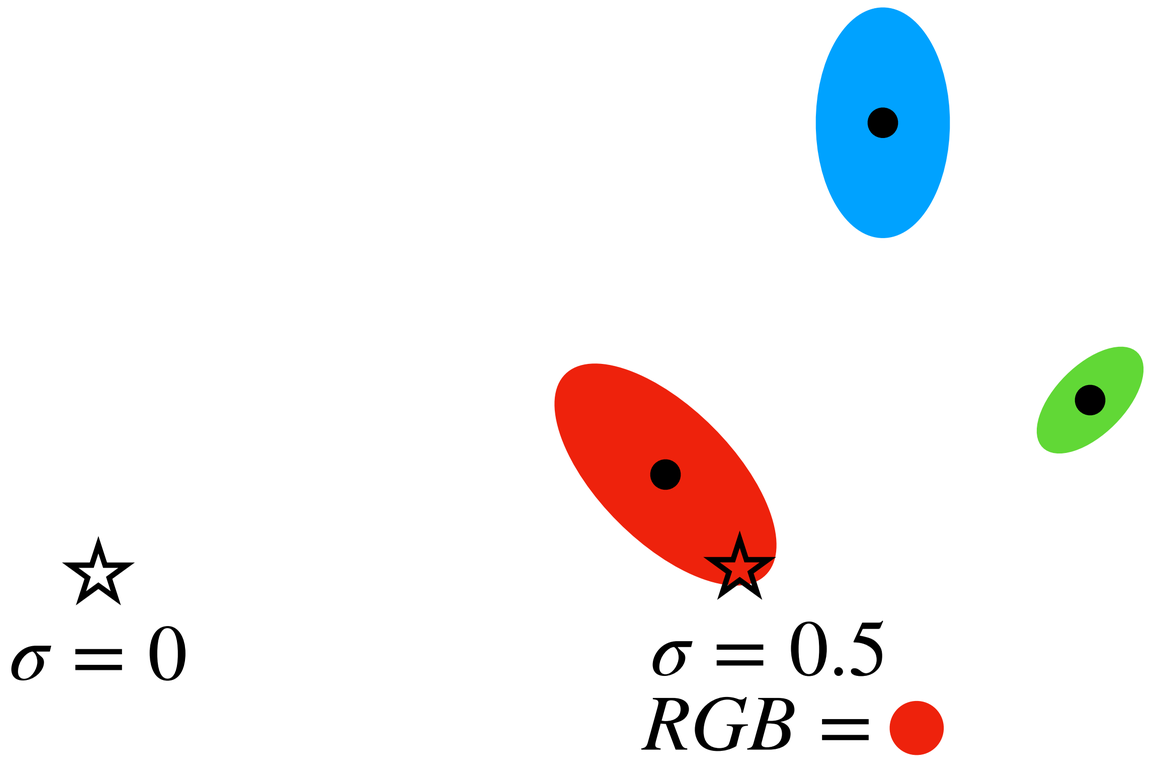


3D Gaussian Blobs floating in Space

☆

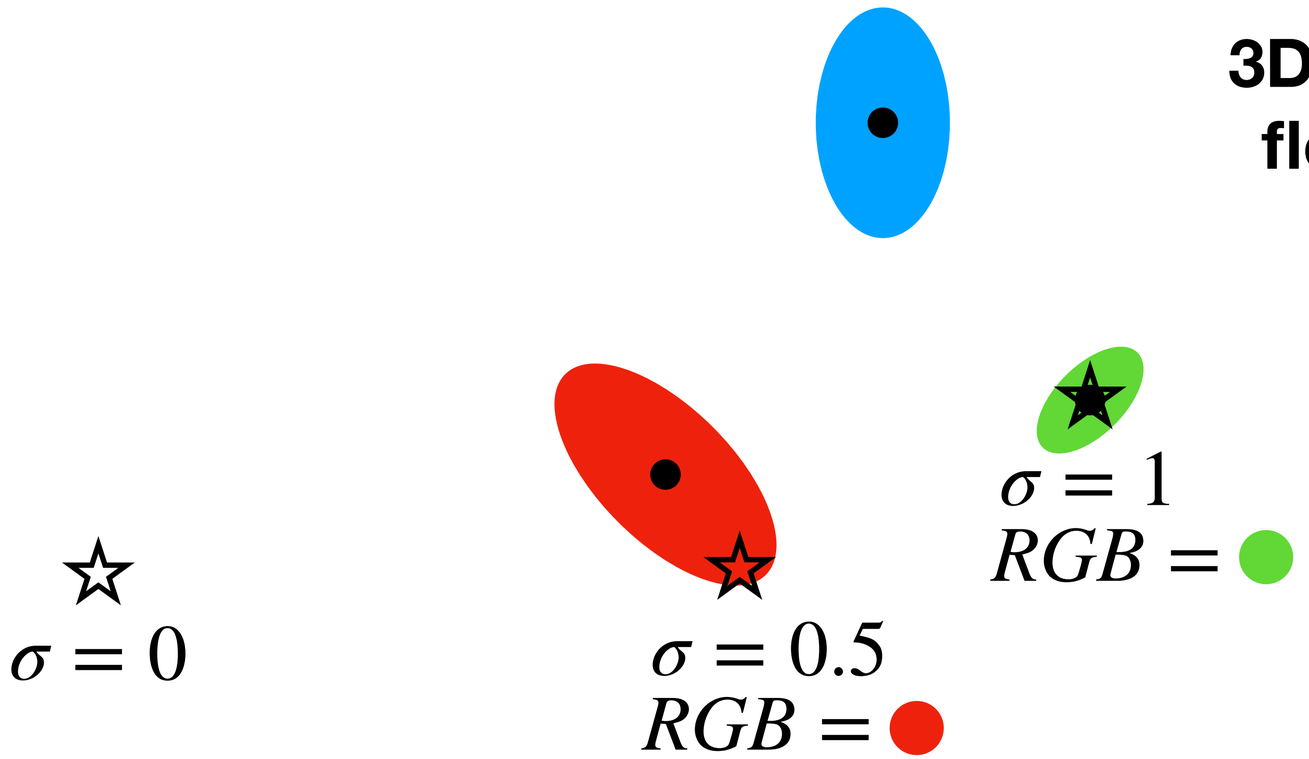
$\sigma = 0$

Primitive-based representations



3D Gaussian Blobs floating in Space

Primitive-based representations



Anisotropic Volumetric 3D Gaussians



Final Rendering

3D Gaussian Visualization

Kerblk, Kopanas et al, 3D Gaussian Splatting for Real-Time Radiance Field Rendering

Anisotropic Volumetric 3D Gaussians



Final Rendering

3D Gaussian Visualization

Kerblk, Kopanas et al, 3D Gaussian Splatting for Real-Time Radiance Field Rendering

How can we represent unbounded scenes?



Image Source: Wikipedia

How can we represent unbounded scenes?

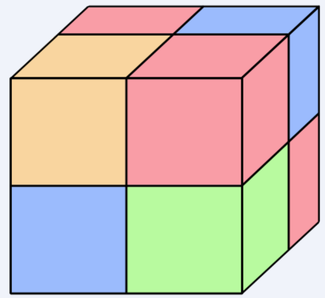


Image Source: Wikipedia

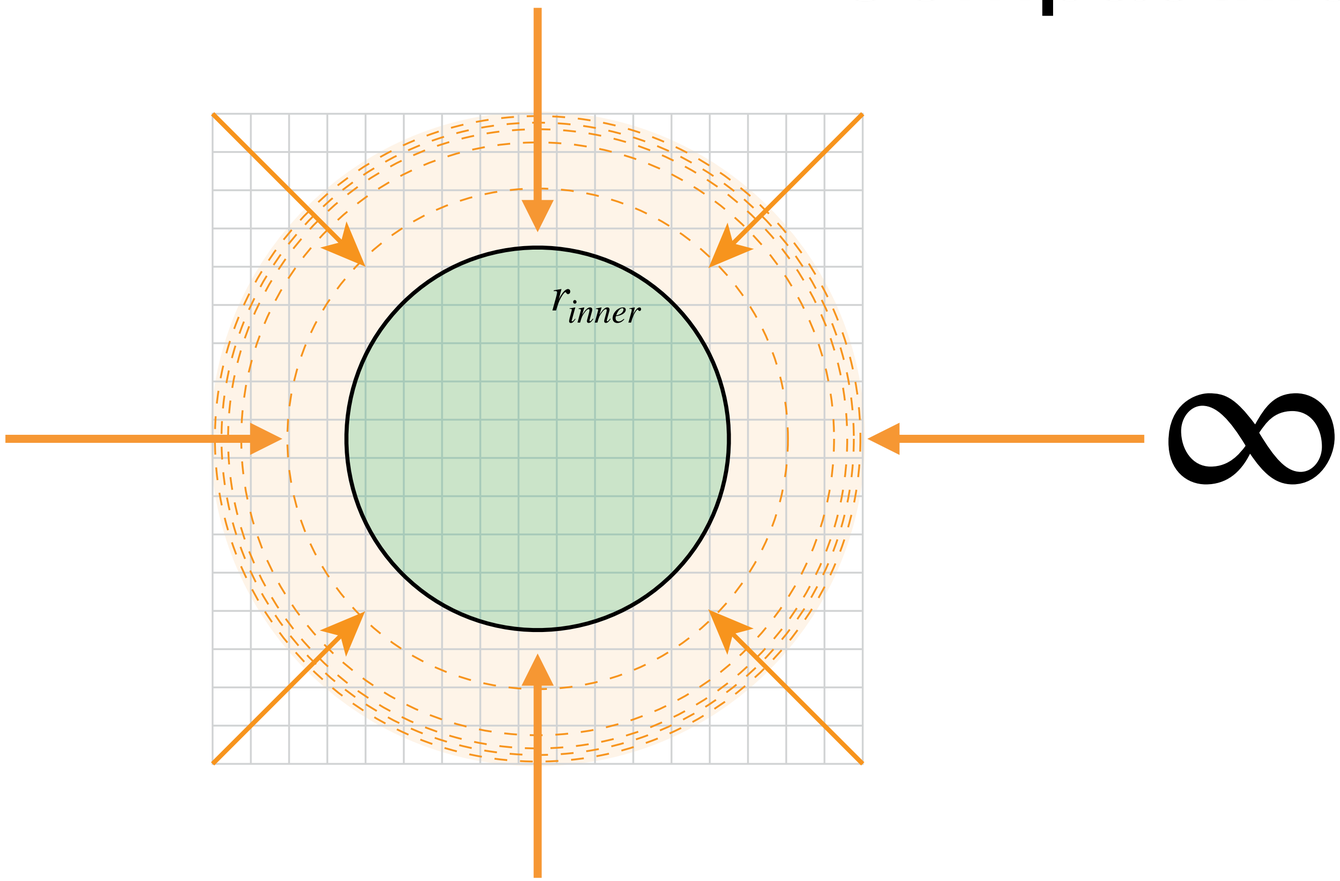


Scene Representation

$$\Phi : \mathbb{R}^3 \rightarrow \mathbb{R}^n$$



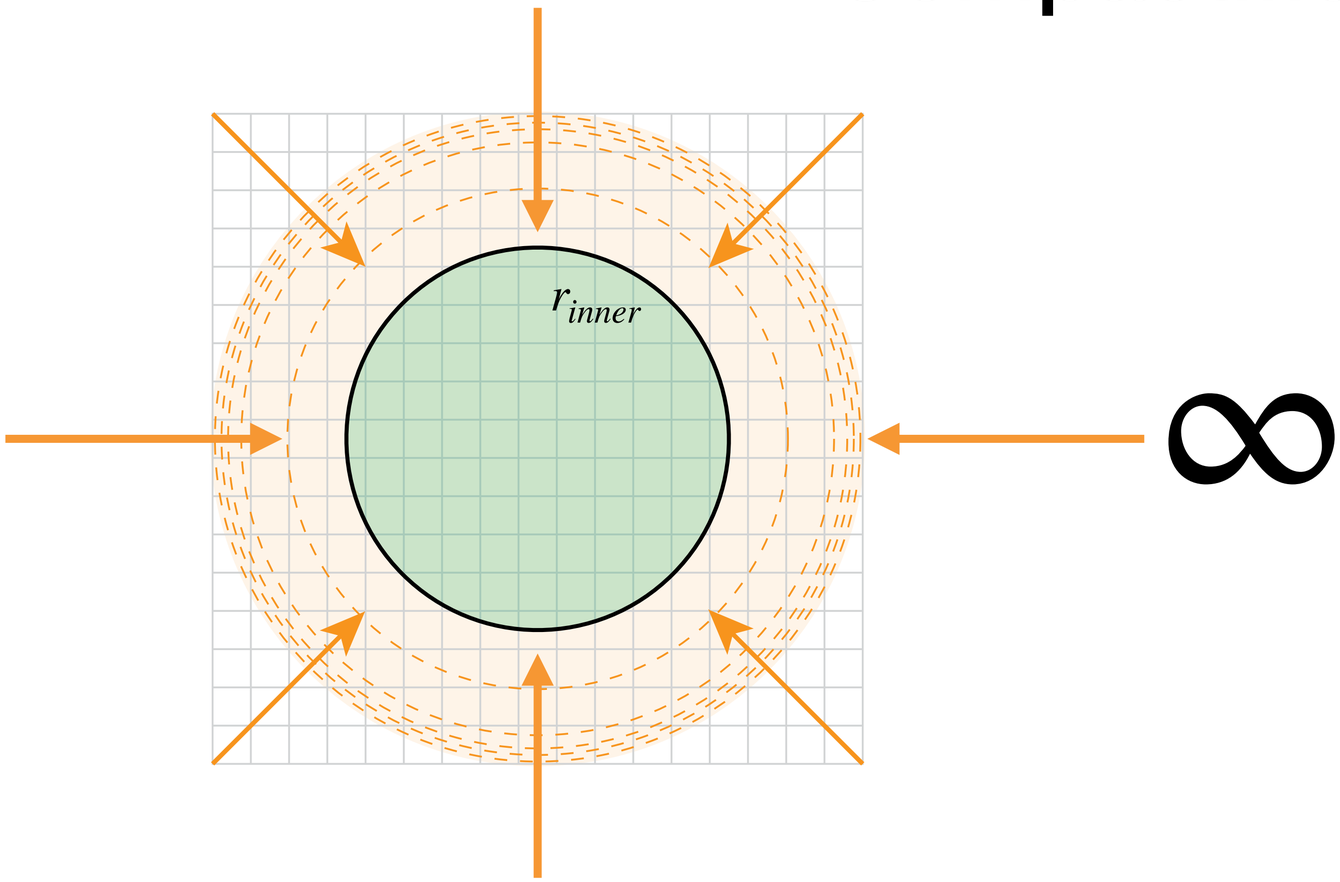
Parameterizing Unbounded Scenes: Compactification



Source: “Mip-NeRF 360: Unbounded Anti-Aliased Neural Radiance Fields”, Barron et al., CVPR 2022

Come up with nonlinear mapping that “squashes” space, such that infinity is mapped to some **finite** value. Aligns with intuition “further away, less resolution”.

Parameterizing Unbounded Scenes: Compactification



Source: “Mip-NeRF 360: Unbounded Anti-Aliased Neural Radiance Fields”, Barron et al., CVPR 2022

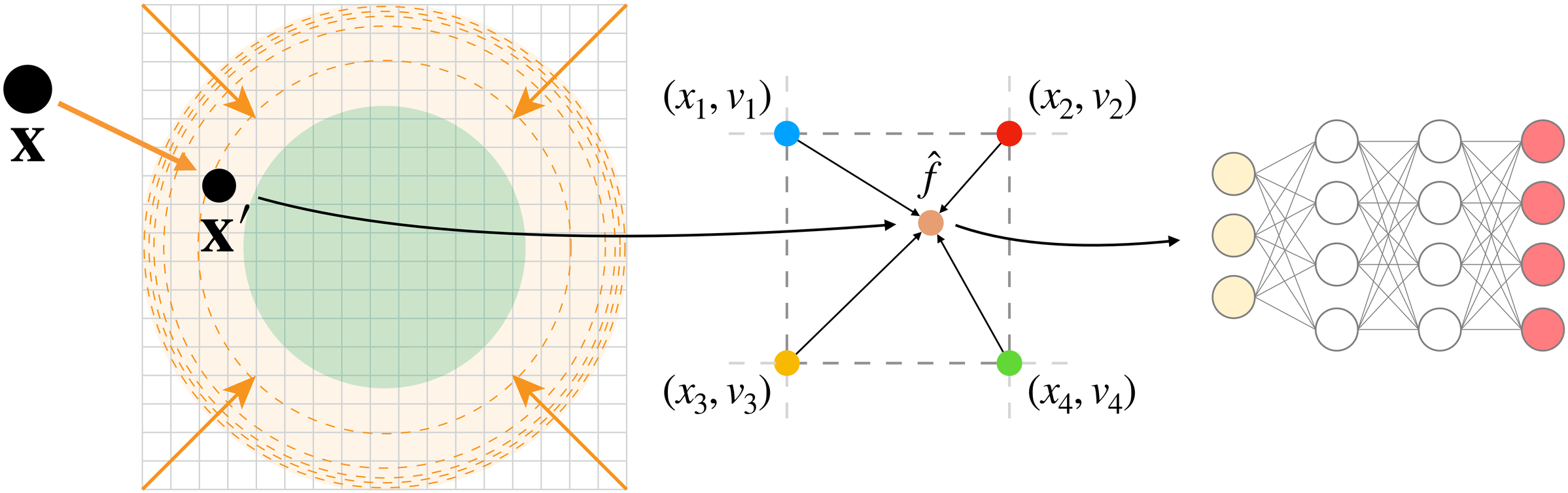
$$\mathbf{x}' = \frac{(1+k) - k/||\mathbf{u}||}{\mathbf{u}/||\mathbf{u}||} r_{inner}$$

$$\mathbf{u} = \frac{\mathbf{x}}{r_{inner}}$$

Example Compactification function from “Mip-NeRF 360: Unbounded Anti-Aliased Neural Radiance Fields”, Barron et al., CVPR 2022

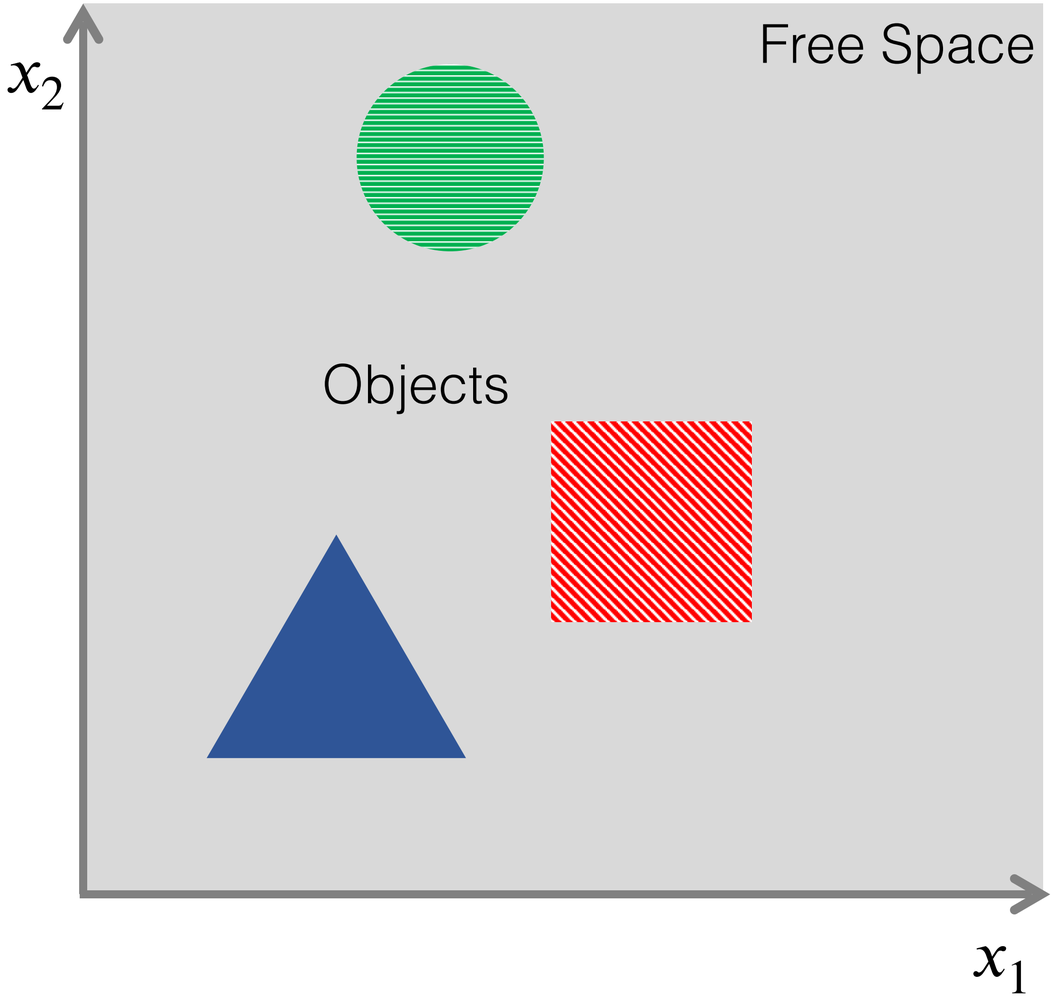
Come up with nonlinear mapping that “squashes” space, such that infinity is mapped to some **finite** value. Aligns with intuition “further away, less resolution”.

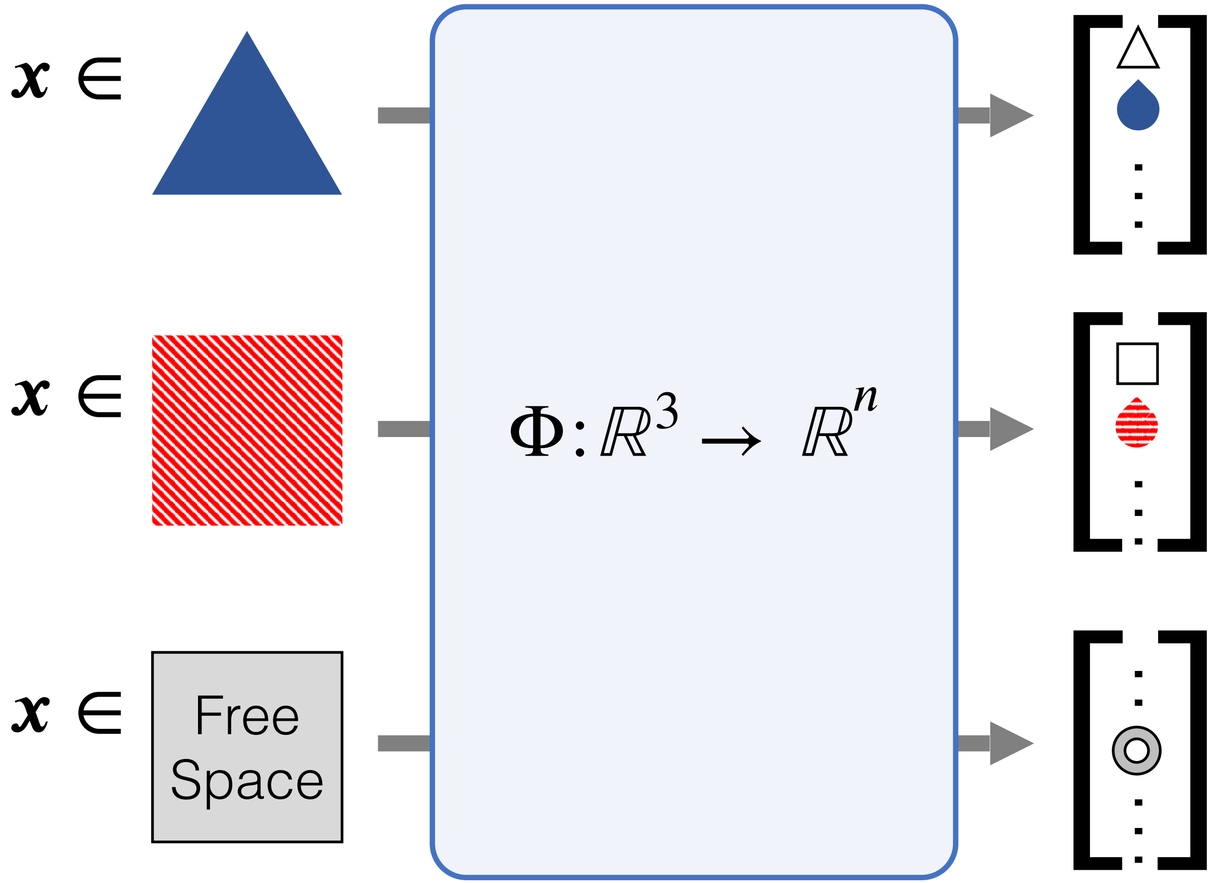
Parameterizing Unbounded Scenes: Compactification



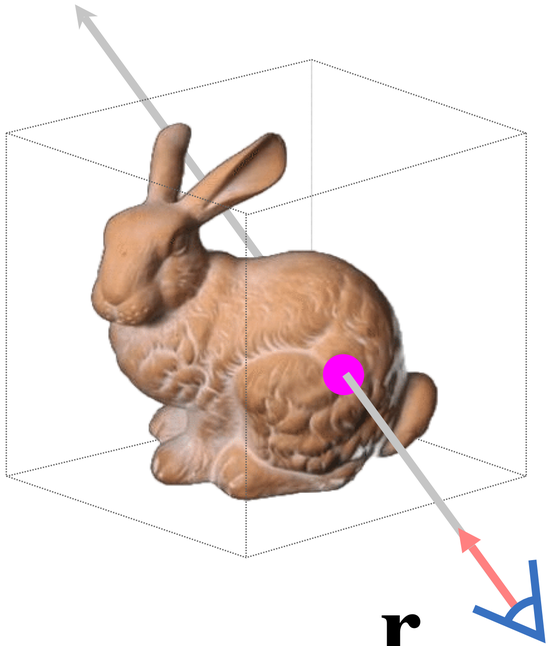
Source: “Mip-NeRF 360: Unbounded Anti-Aliased Neural Radiance Fields”, Barron et al., CVPR 2022

Do we have to parameterize 3D Scene as 3D Function?





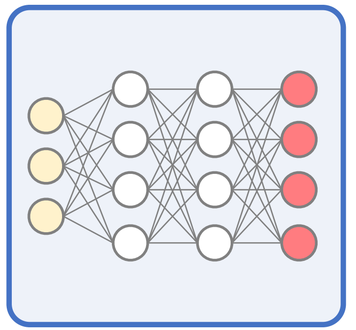
3D-structured Representation



$\mathbf{r}$

Camera Ray

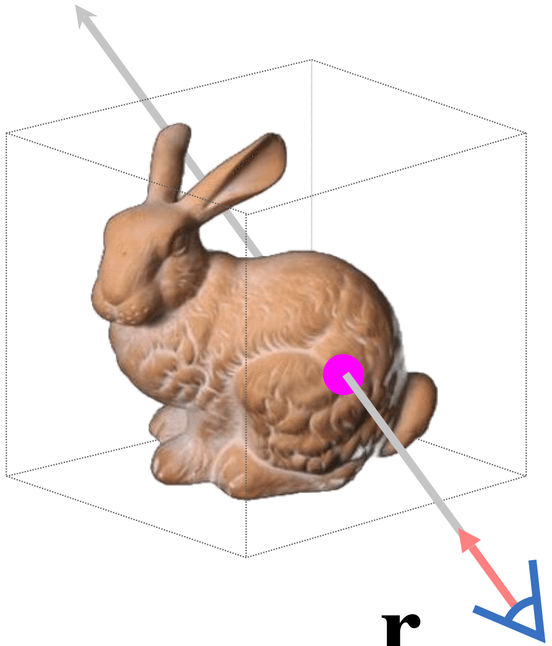




$\Phi^{3D} : \mathbb{R}^3 \rightarrow \mathbb{R}^n$

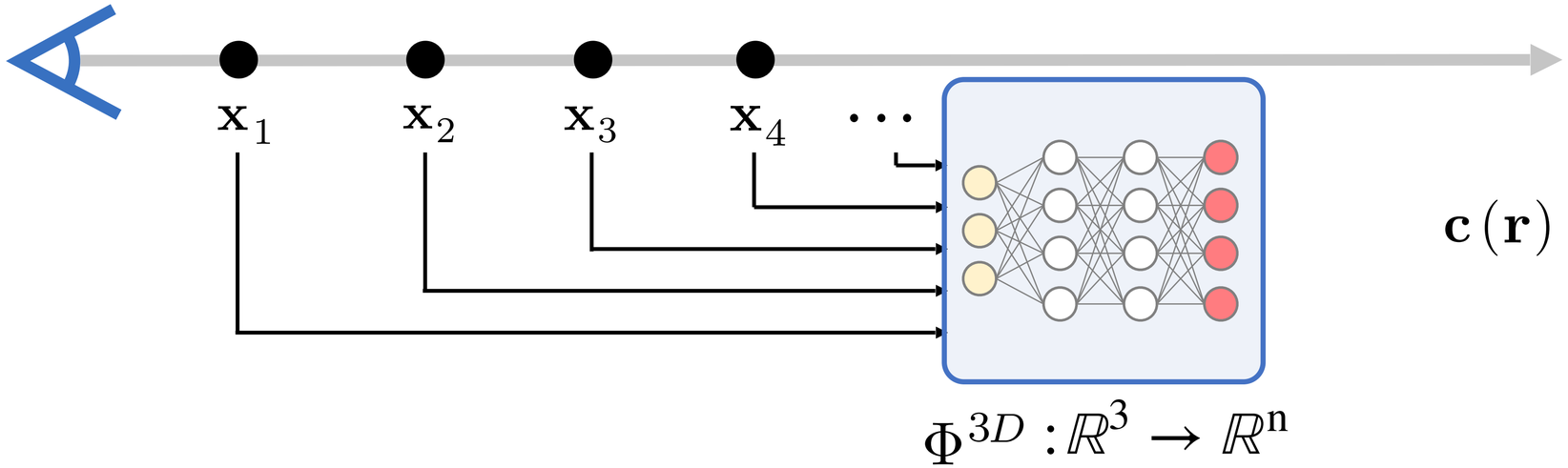
$\mathbf{c}(\mathbf{r}) =$ 

3D-structured Representation

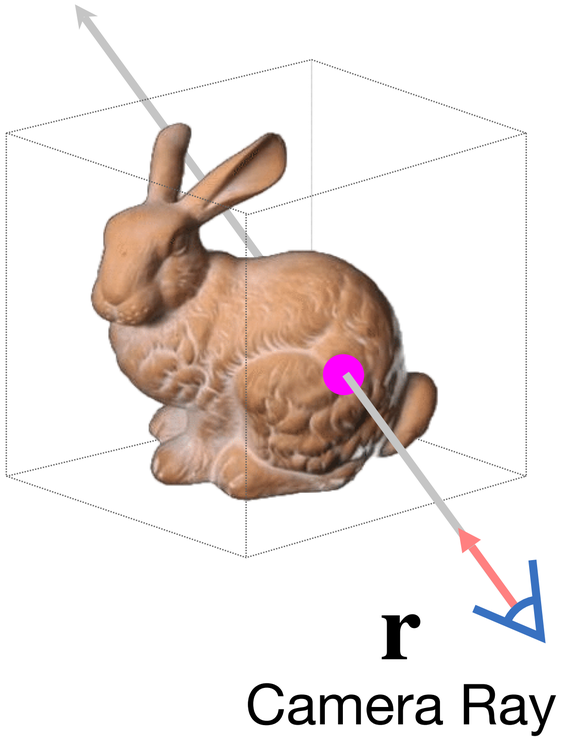


$\mathbf{r}$

Camera Ray

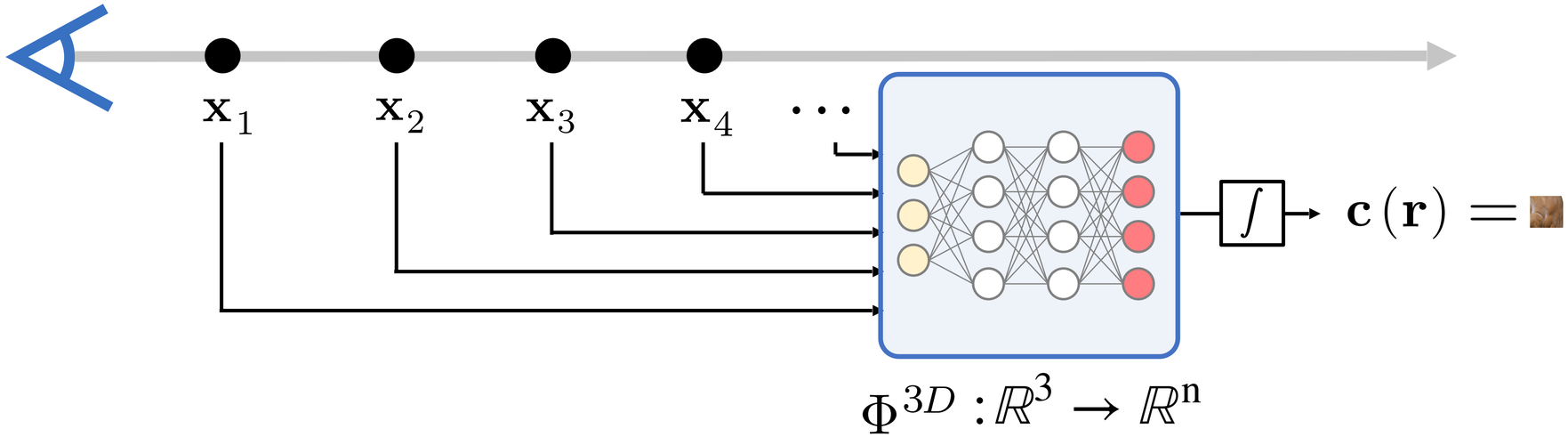


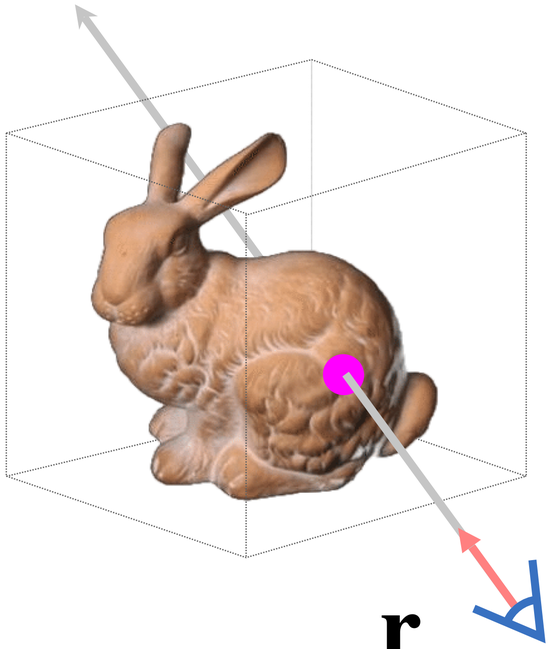
3D-structured Representation



$\mathbf{r}$

Camera Ray





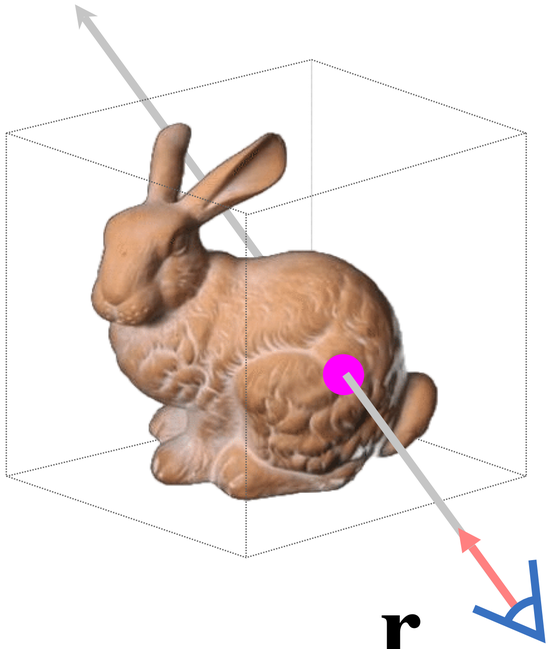
$\mathbf{r}$

Camera Ray

Light Field

$\mathbf{c}(\mathbf{r}) =$

[Adelson et al. 1991, Levoy et al. 1996, Gortler et al. 1996]



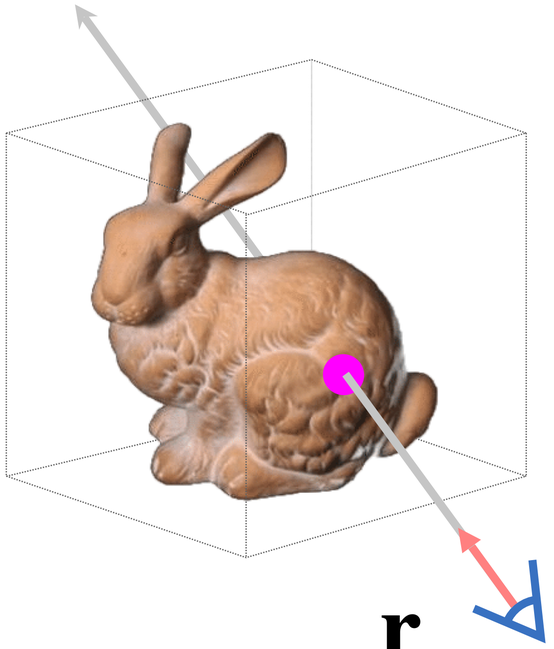
$\mathbf{r}$

Camera Ray

Light Field Networks

$\mathbf{c}(\mathbf{r}) =$

Light Field Networks



$\mathbf{r}$

Camera Ray

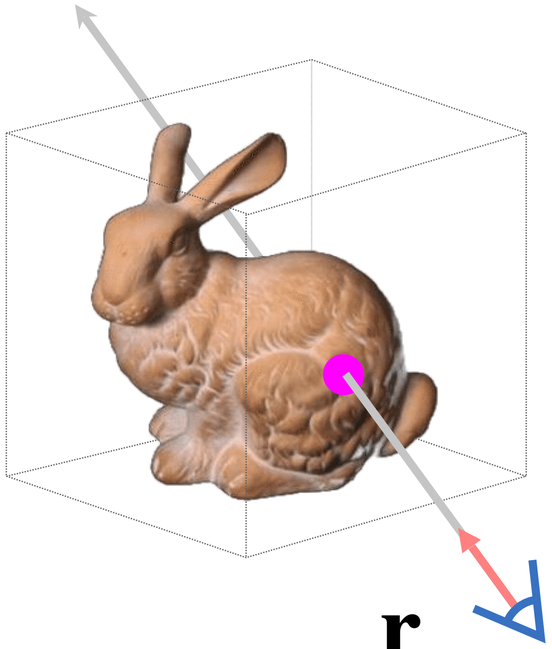
$\mathbf{x}_1$ $\mathbf{x}_2$ $\mathbf{x}_3$ $\mathbf{x}_4$ $\cdots$

$\Phi^{3D} : R^3 \rightarrow R^n$

$\int$

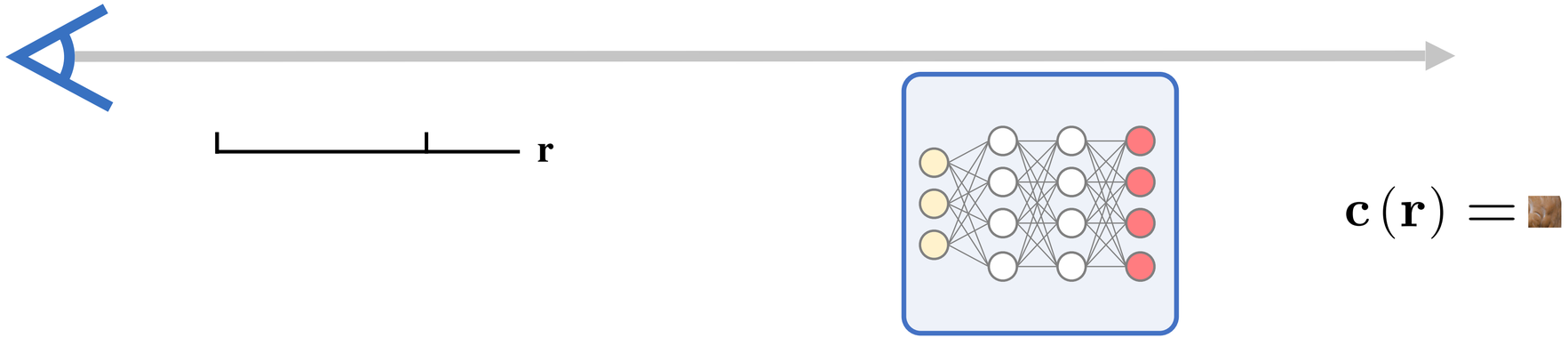
$\mathbf{c}(\mathbf{r}) =$

Light Field Networks



$\mathbf{r}$

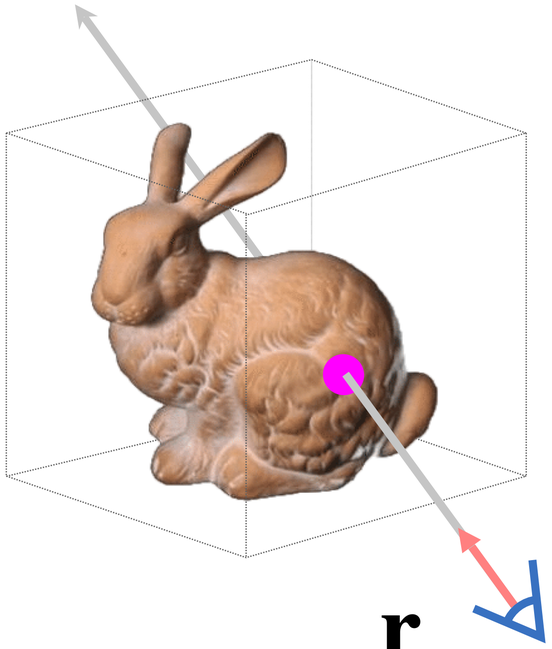
Camera Ray



$\mathbf{c}(\mathbf{r}) =$

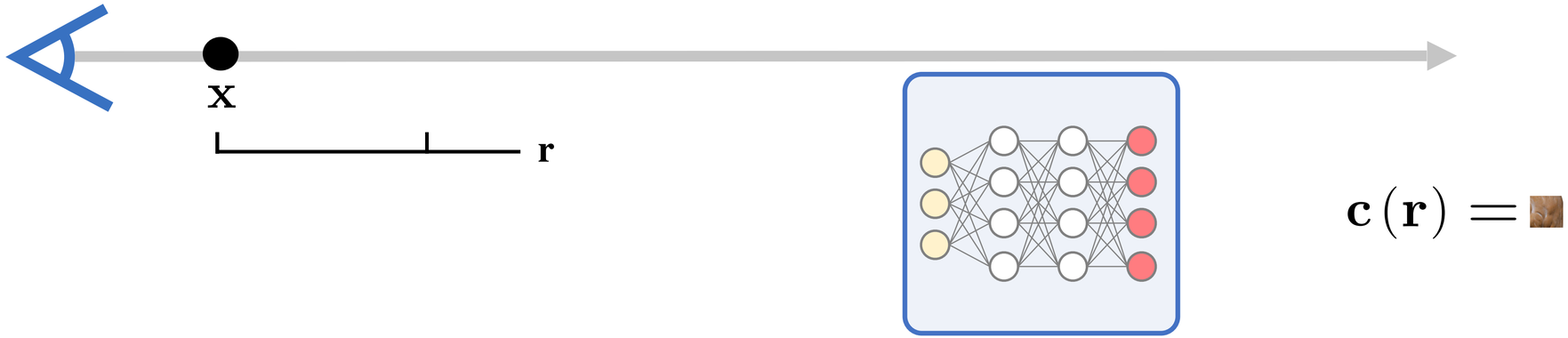
An Alternative Scene Representation

Light Field Networks



$\mathbf{r}$

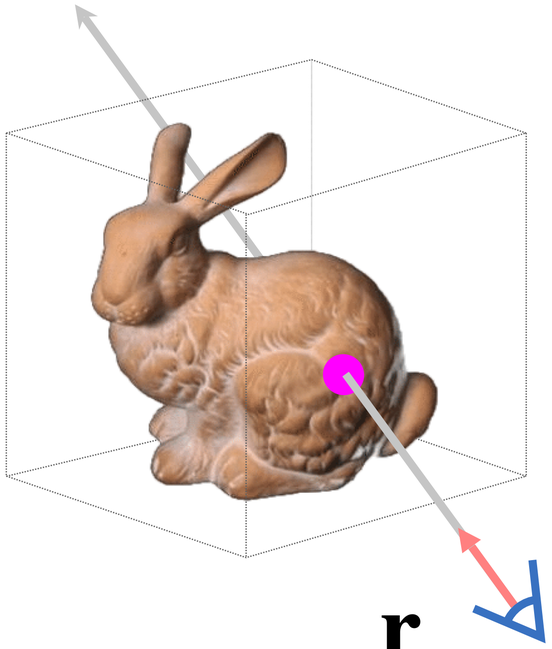
Camera Ray



$\mathbf{c}(\mathbf{r}) =$

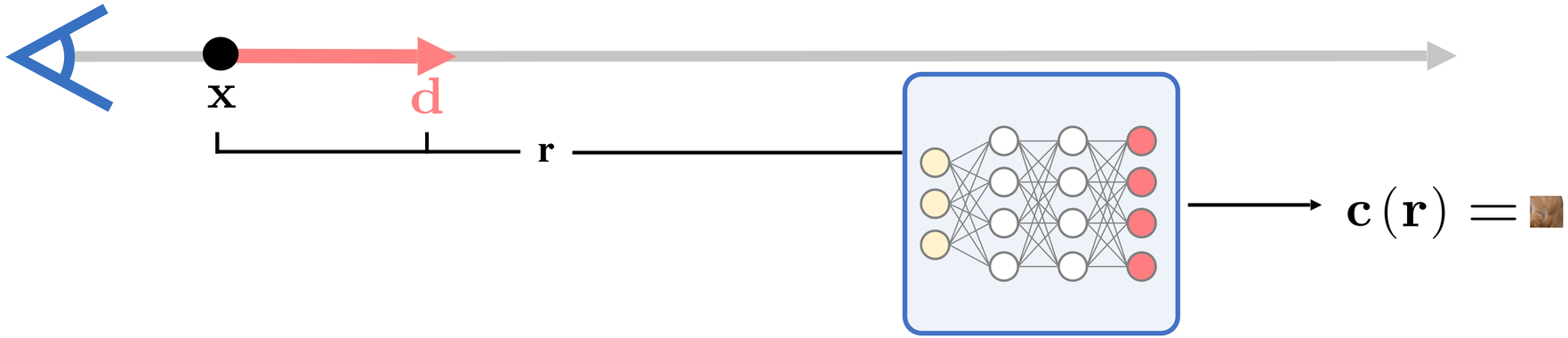
An Alternative Scene Representation

Light Field Networks



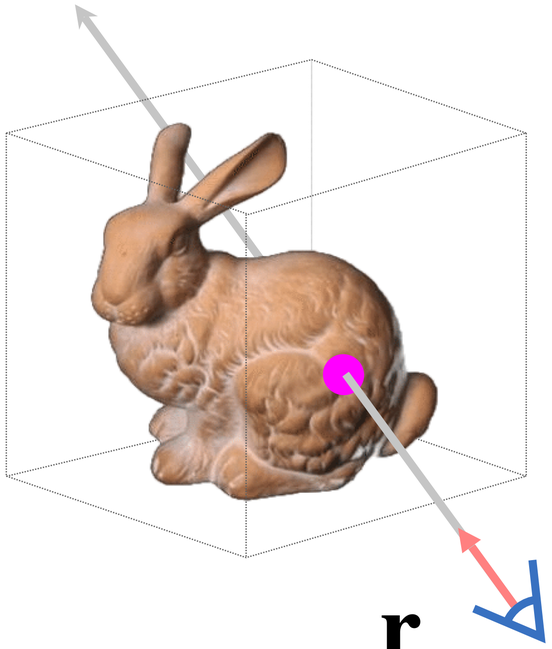
r

Camera Ray



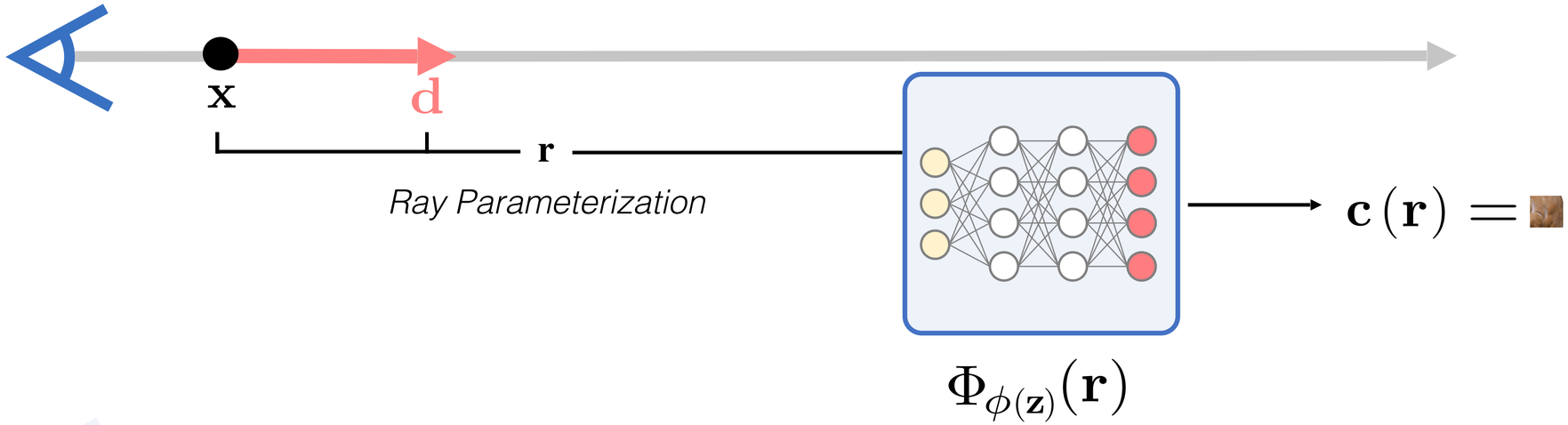
An Alternative Scene Representation

Light Field Networks



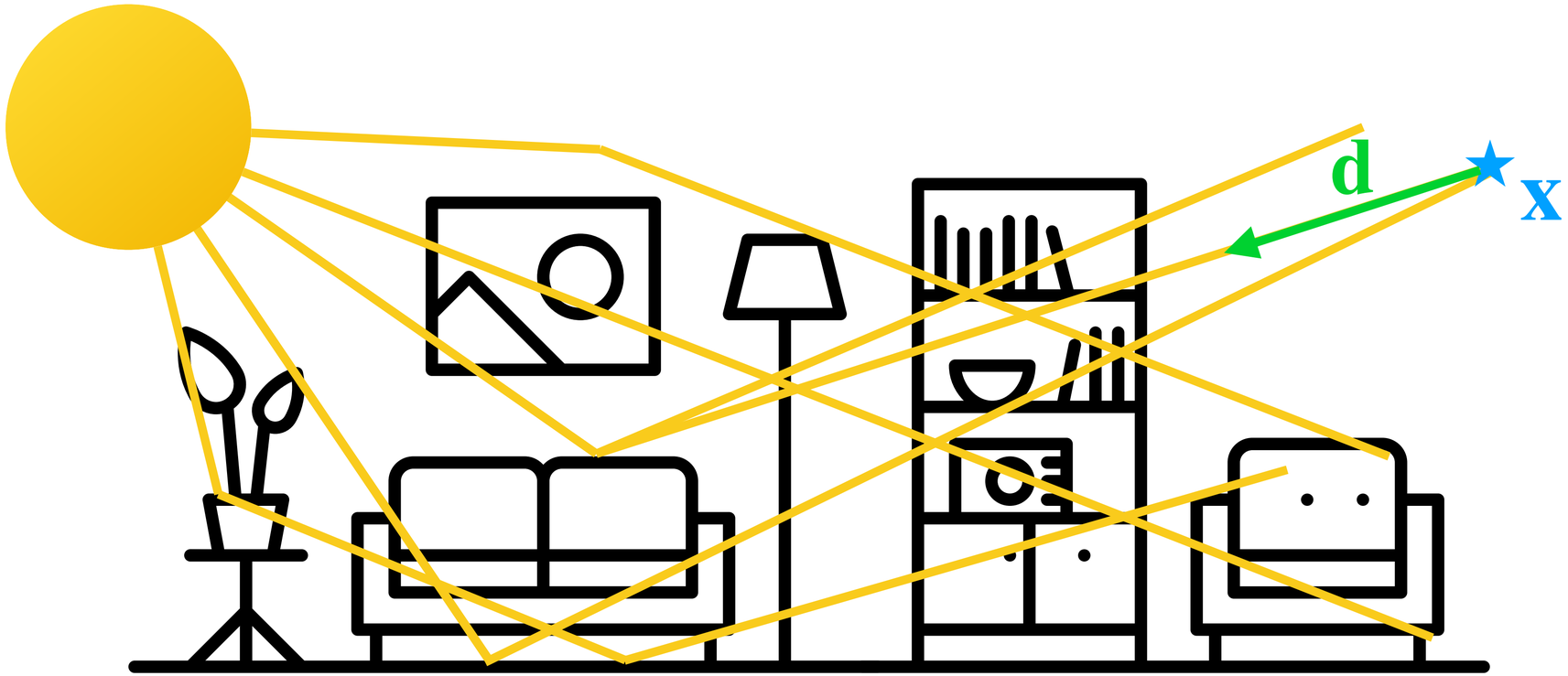
$\mathbf{r}$

Camera Ray



An Alternative Scene Representation

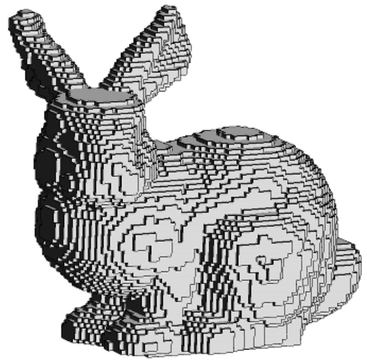
Just because Scene is 3D, don't have to store information in 3D data structure…

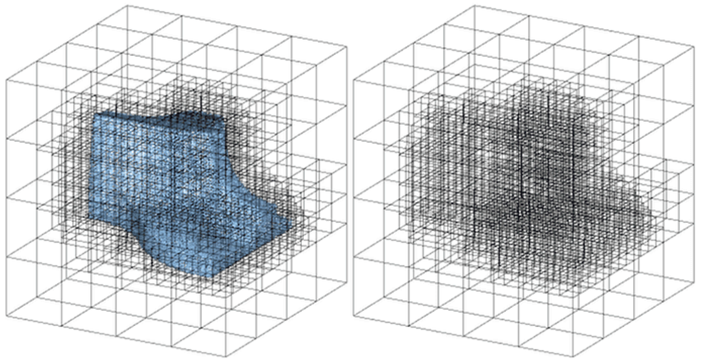


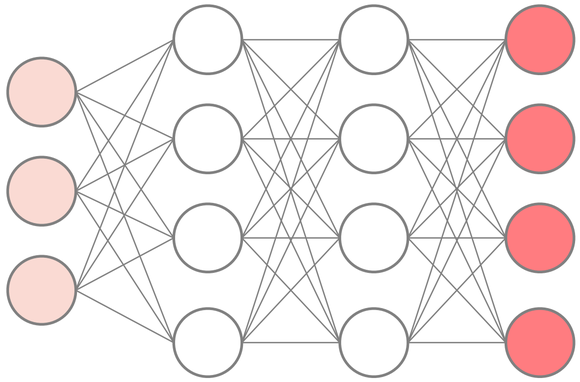
$$LF : \mathbb{R}^3 \times \mathbb{S}^2 \rightarrow \mathbb{R}^3, \quad LF(\mathbf{x}, \mathbf{d}) = \mathbf{c}$$

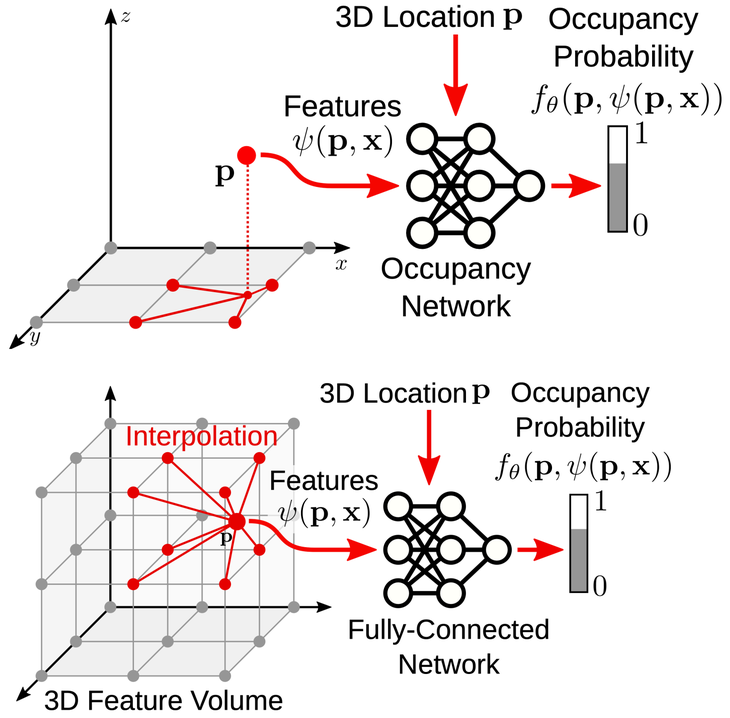
Light Field is 5-dimensional! What parameterizations are possible…?

Summary









- No “Artificial Intelligence”, “understanding” or “Machine Learning” when fitting a single representation to a single 3D thing, even if there are neural networks in them.
- It becomes *a lot* more interesting when considering *inference,* i.e., we want representation to be output of an encoder! More on that later in “Prior-based reconstruction”.