UNIT 8

Neural Fields

Johannes Brandstetter, Markus Holzleitner
Institute for Machine Learning

JOHANNES KEPLER
UNIVERSITY LINZ

Institute for
Machine Learning

Copyright statement:

Field (physics)

- A field (in physics) is an assignment of a quantity (scalar/vector) to each point in an input domain, most commonly Euclidean space $\mathbb{R}^n$.
- Not to be confused with the mathematical field ("Körper" in German).
- **Neural field**: A field that is parameterized by a neural network.
- Neural fields assign scalar or vector valued quantities to each point of an input domain:

$$\Psi(\boldsymbol{x}) : \mathbb{R}^3 \to \mathbb{R}^3 \qquad \text{vector field}$$

$$\Psi(\boldsymbol{x}) : \mathbb{R}^3 \to \mathbb{R} \qquad \text{scalar field}$$

Example of neural fields

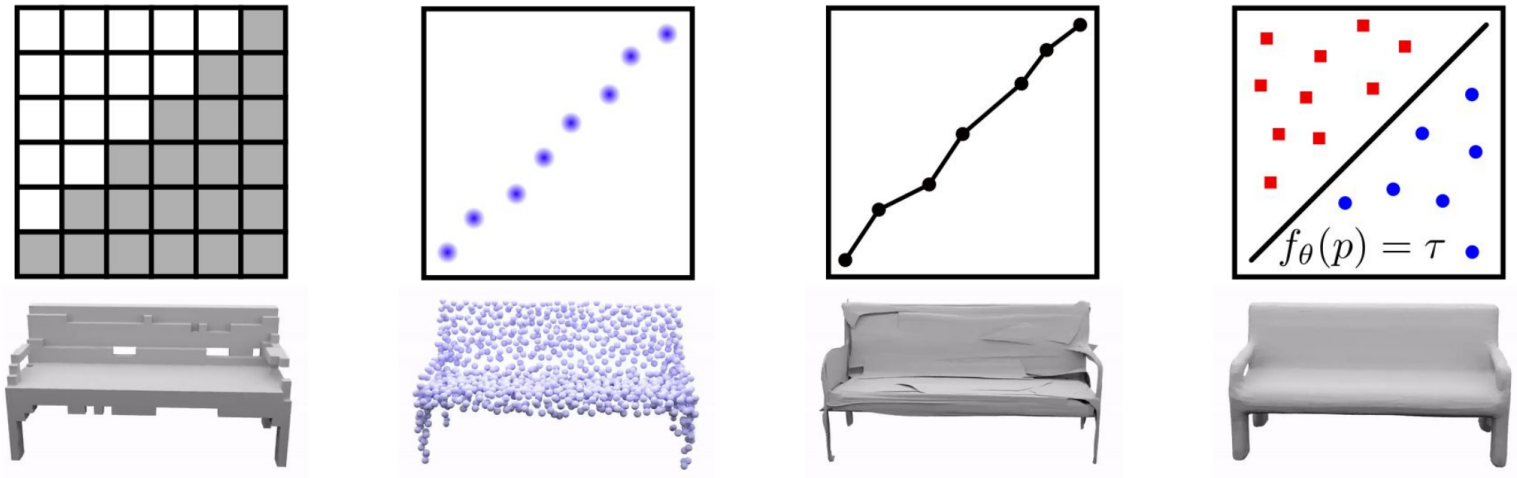
- Images: $\Psi(\boldsymbol{x}) : \mathbb{R}^2 \rightarrow \mathbb{R}^3$ (pixel coordinates $\Rightarrow$ rgb values)
- Geometries: $\Psi(\boldsymbol{x}) : \mathbb{R}^2/\mathbb{R}^3 \rightarrow \mathbb{R}$ (volumetric coordinates $\Rightarrow$ signed-distance function)
- Geometries: $\Psi(\boldsymbol{x}) : \mathbb{R}^2/\mathbb{R}^3 \rightarrow$ Probability$[0, 1]$ (volumetric coordinates $\Rightarrow$ occupancy field)
- 3D scenes: $\Psi(\boldsymbol{x}) : \mathbb{R}^5 \rightarrow \mathbb{R}^4$ (volumetric cooridnates + direction $\Rightarrow$ density, rgb values)

Advantages

- Neural fields are:
  - compact representations (memory efficient)
  - differentiable
  - “easy” to optimize
  - easy to scale to higher dimensions
  - multimodal
- Neural fields allow for:
  - Real-time rendering
  - Memory-efficient storage
  - Resolution-invariant representation
  - ...

Example: 3D geometries

- Voxel representation
- Point representation
- Mesh representation
- Occupancy neural field (Mescheder et al)



Occupancy neural fields

- An occupancy field is defined by a continuous function that maps a 3D coordinate $\mathbf{x} \in \mathbb{R}^2/\mathbb{R}^3$ to a scalar value:
  - $\Psi(\boldsymbol{x}) : \mathbb{R}^2/\mathbb{R}^3 \rightarrow [0, 1]$
  - The output is the probability of the occupancy.
  - A value of 1 indicates the point is inside the object, while 0 indicates it is in empty space.
- The actual physical surface $\mathcal{S}$ of the object is implicitly defined as an **isosurface** (or level set) where the probability equals a specific threshold (usually 0.5).

Training occupancy neural fields

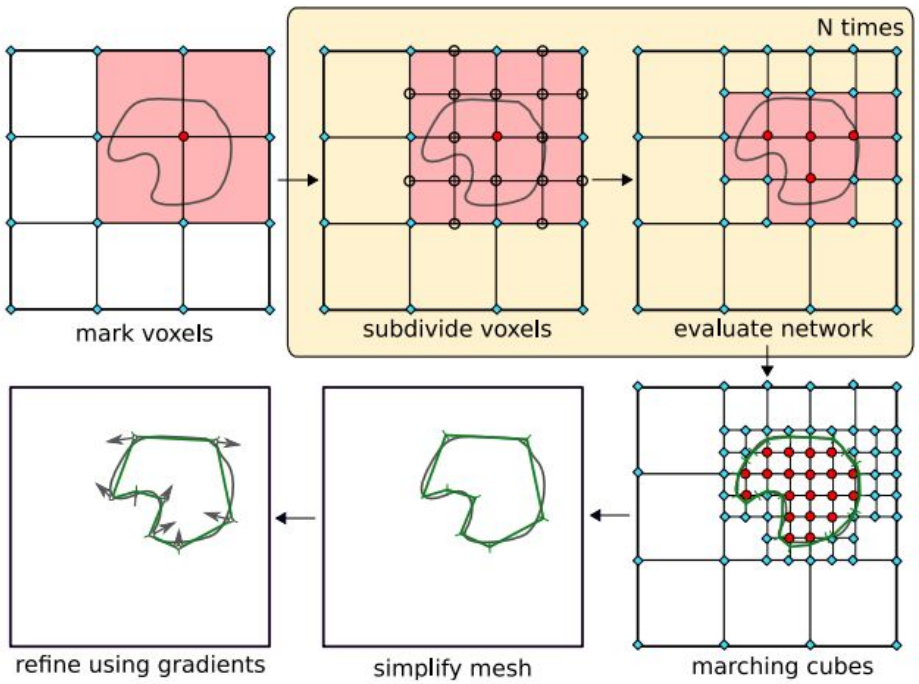
- **Cross entropy loss**: To train the network, we use a set of sampled points $\boldsymbol{x}_i$ for which the ground truth occupancy $\boldsymbol{o}_i \in \{0, 1\}$ is known. The training typically employs a Binary Cross-Entropy (BCE) loss.
  - For each batch B of sampled points, we get:

    $$\mathcal{L}_\theta = \sum_i^B \boldsymbol{o}_i \log \Psi(\boldsymbol{x}_i) + (1 - \boldsymbol{o}_i) \log(1 - \Psi(\boldsymbol{x}_i))$$

  - Occupancy fields are often preferred for complex topologies because they are easier to train on raw point clouds or noisy data where exact distance values (needed for **signed distance fields**) are hard to calculate.

Extracting a surface

- The **Marching Cubes algorithm** is a used to extract a 2D surface mesh from a 3D scalar field.
- It is the industry standard for turning **implicit representations** (like the occupancy field) into **explicit representations**.



Signed distance (neural) fields

- A Signed Distance Field (SDF) encodes not just the **state of a point** (inside or outside) but also its **proximity to the boundary**.
- In a neural SDF, a network learns a continuous mapping from a spatial coordinate to its signed distance to the nearest surface.
- A neural SDF is defined by a function $\Psi(\boldsymbol{x})$ that maps a coordinate $\boldsymbol{x}$ (2D/3D) to a scalar d: $\Psi(\boldsymbol{x}) = d$
  - $d > 0$: The point is outside the object
  - $d < 0$: The point is inside the object
  - $d = 0$: The point is exactly on the surface

The Eikonal equation

- For a function to be a valid distance field, it must satisfy the Eikonal equation.
- This equation states that the **magnitude of the gradient of the distance field must be equal to 1** almost everywhere, ensuring that the field changes at a constant rate relative to Euclidean distance:
  - If $\Psi(\boldsymbol{x}) = d(\boldsymbol{x})$ is the distance to the nearest point on a surface $\mathcal{S}$, then for any point $\boldsymbol{x}$ not on the surface, the direction of the steepest increase in distance is directly away from the surface:

$$\|\nabla_{\boldsymbol{x}} \Psi(\boldsymbol{x})\| = 1$$

Physics-informed Eikonal loss

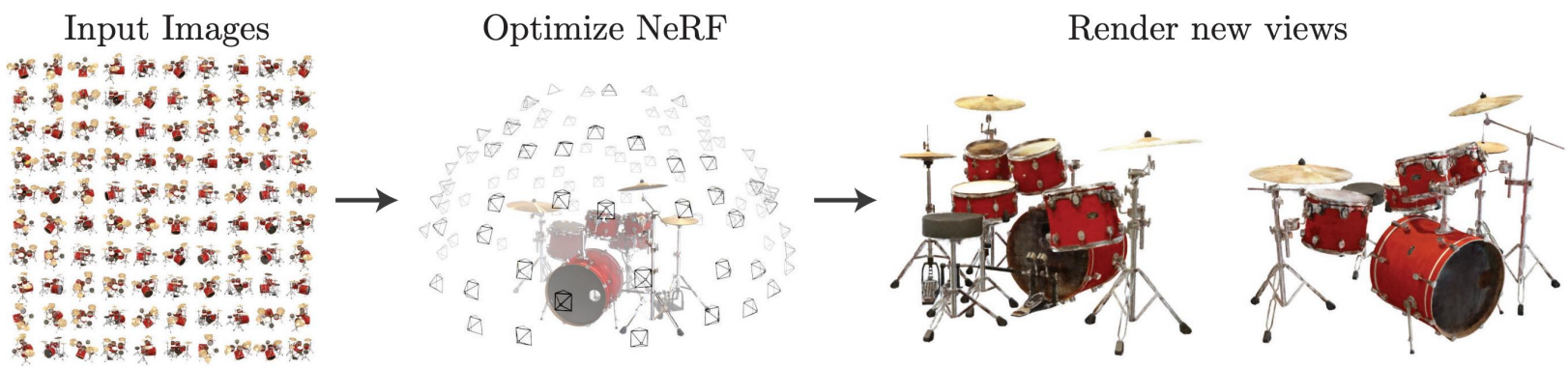
- The physics-informed Eikonal loss forces the network to obey the underlying geometry of a true distance field:

$$\mathcal{L}_{eik} = \mathbb{E}_{\boldsymbol{x}} \left(\|\nabla_{\boldsymbol{x}} \Psi(\boldsymbol{x})\| - 1 \right)^2$$

- The loss is often combined with a point-cloud loss (forcing $\Psi(\boldsymbol{x}) = 0$ at the surface) to ensure the network learns both the shape and the correct distance scale.

Neural Radiance Fields

- **Neural Radiance Fields** (NeRFs) represent a breakthrough in 3D scene reconstruction by treating a physical space as a continuous volumetric function rather than a collection of discrete points or polygons.
- Instead of storing a “shape”, the model learns to map coordinates directly to their visual properties.

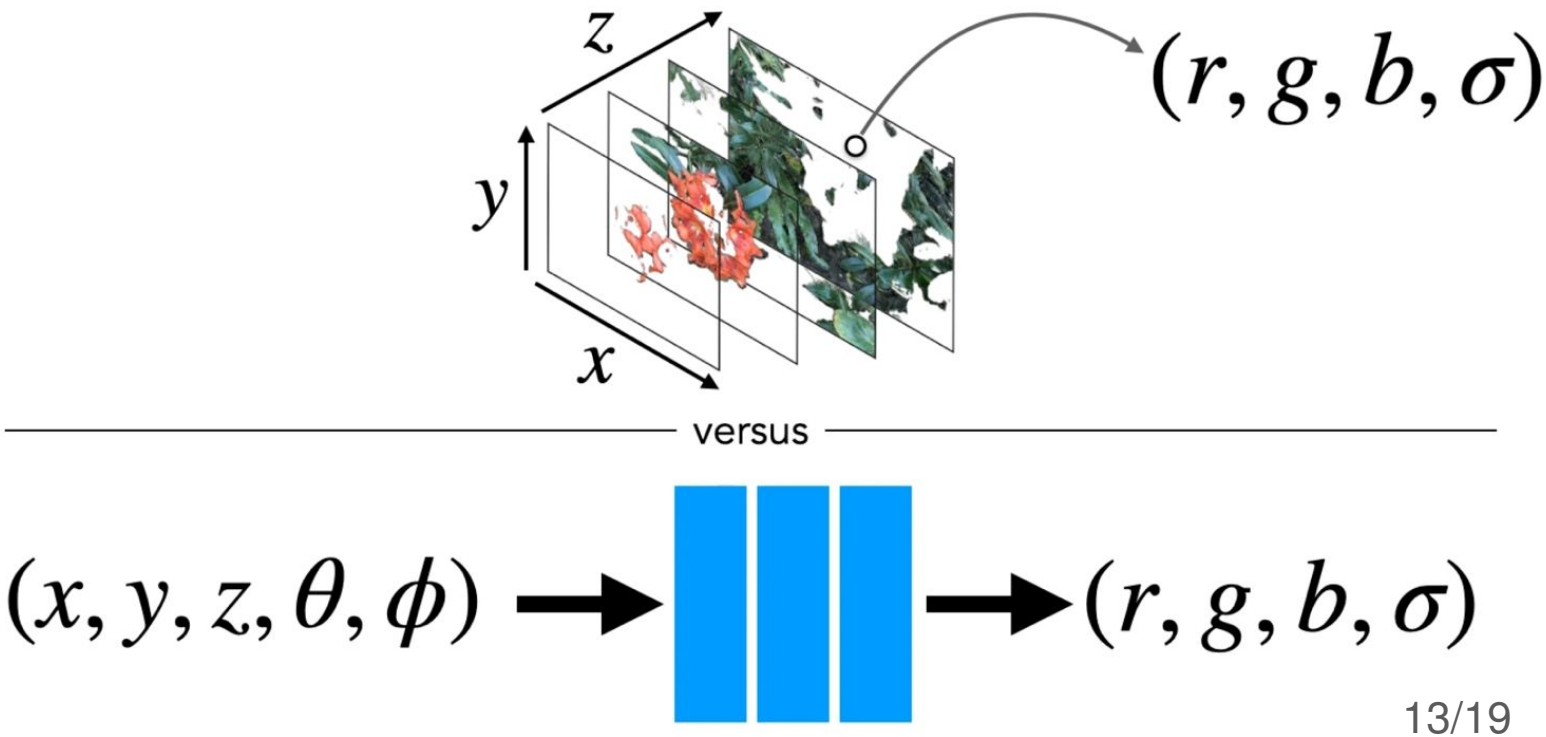


Scene representation

- Neural network replaces large N-d array:

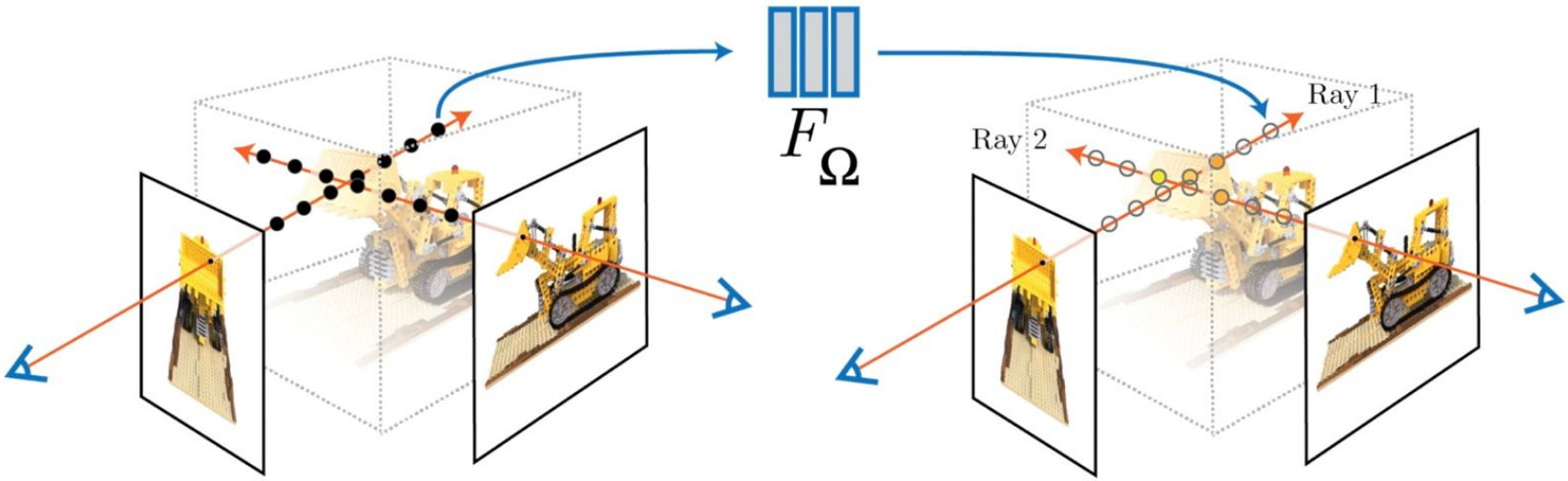
$$\Psi(\boldsymbol{x}) : \mathbb{R}^5 \rightarrow \mathbb{R}^4$$

- Neural networks outputs rgb values and absorption coefficient.



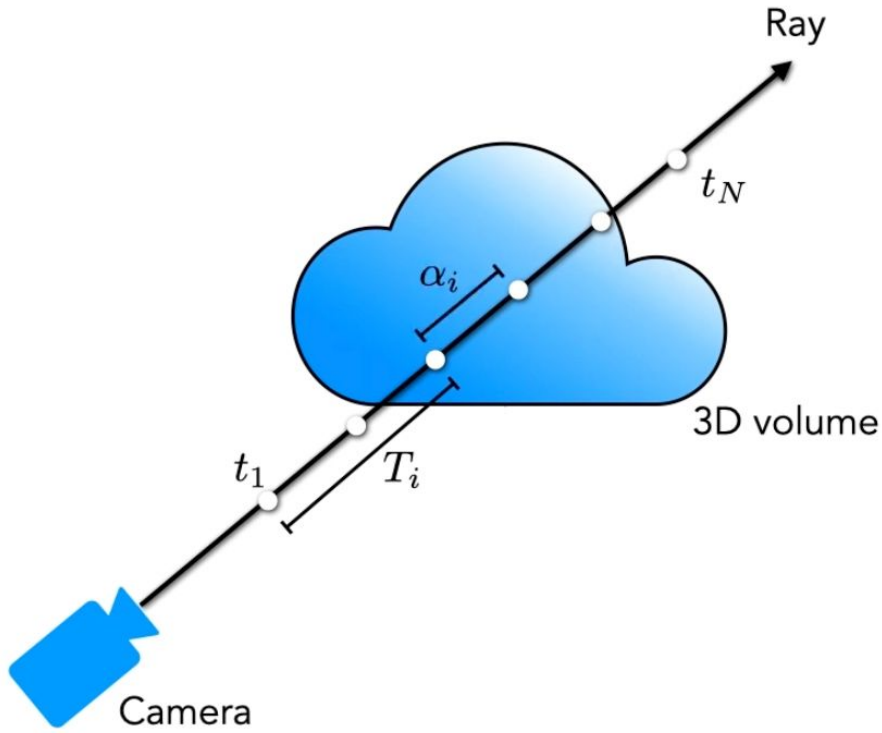
Ray tracing

- Ray tracing is a computer graphics rendering technique that simulates the physical behavior of light to create highly realistic images.



Generating views with volumetric rendering

- The color of the rendered ray C is obtained by integrating along the ray.



Volume rendering equation

- The color of the rendered ray C is obtained by integrating along the ray.
- Numerically, this means summing colors $c_i = [r, g, b]^T$ along points of the ray:

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

- How much light is blocked along the ray:

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

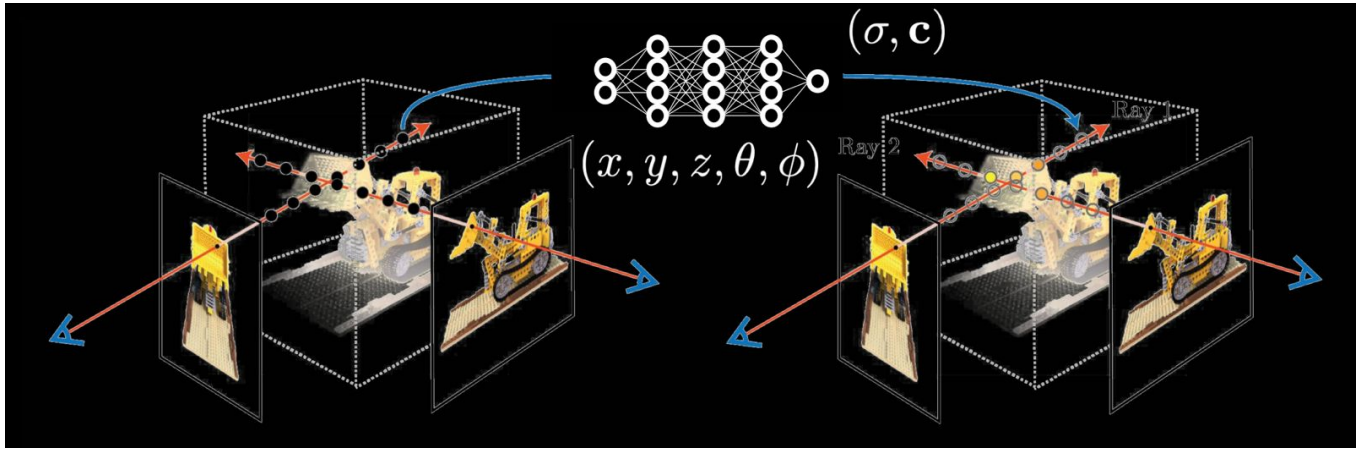
- How much light is contributed by ray segment i:

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

NeRFs loss

- Given images with known camera positions.
- We sample along rays, and optimize the absorption and radiance to minimize photometric error:

$$\min_{\sigma,c} \sum_{p} ||\text{Render}(C^p) - C^p_{\text{ground truth}}||^2$$



Relation to PINNs

- Usually, a neural network minimizes the error between its prediction and the ground truth (data loss)
- **Physics-informed neural networks (PINNs)** add a residual loss physics loss).
  - Physics losses use automatic differentiation to calculate the derivatives of the network's output with respect to its inputs
  - Derivatives need to satisfy the PDE
  - Example: The heat equation describes how temperature $u(x,t)$ changes over space x and time t: $\frac{\partial u}{\partial t} = \alpha \frac{\partial^2 u}{\partial x^2}$
  - We define a neural network $\hat{u}(x,t)$ that takes two inputs (x,t) and predicts the temperature u. Estimates of the terms $\frac{\partial \hat{u}}{\partial t}$ and $\frac{\partial^2 \hat{u}}{\partial x^2}$ are obtained by taking the derivative of the output $\hat{u}(x,t)$ w.r.t. spatial and temporal coordinates.
  - We minimize the residuals $\frac{\partial \hat{u}}{\partial t} - \alpha \frac{\partial^2 \hat{u}}{\partial x^2}$.

SDF is a physics-informed loss

- One of the core properties of an SDF is that they satisfy the Eikonal equation: $\|\nabla_{\boldsymbol{x}} \Psi(\boldsymbol{x})\| = \frac{1}{f(\boldsymbol{x})}$.
- In the special case when $f(x) = 1$, $\Psi(\boldsymbol{x})$ becomes an SDF.
- The Eikonal residual loss therefore is

$$\mathcal{L}_{eik} = \mathbb{E}_{\boldsymbol{x}} \left(\|\nabla_{\boldsymbol{x}} \Psi(\boldsymbol{x})\| - 1 \right)^2 .$$

- This loss is easy to optimize. No conflicting gradients!