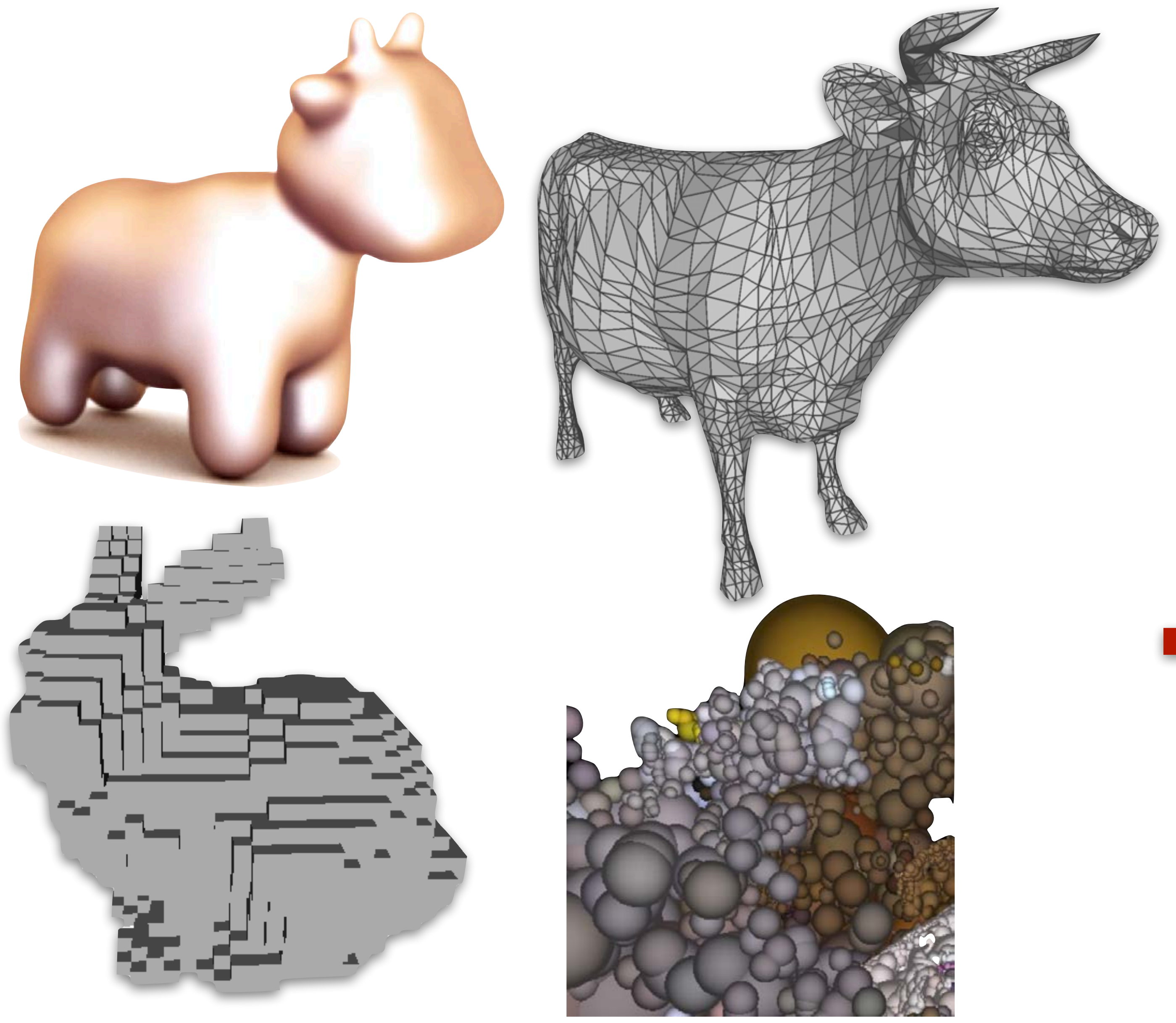
Implicit Neural Representations

Tiago Novello

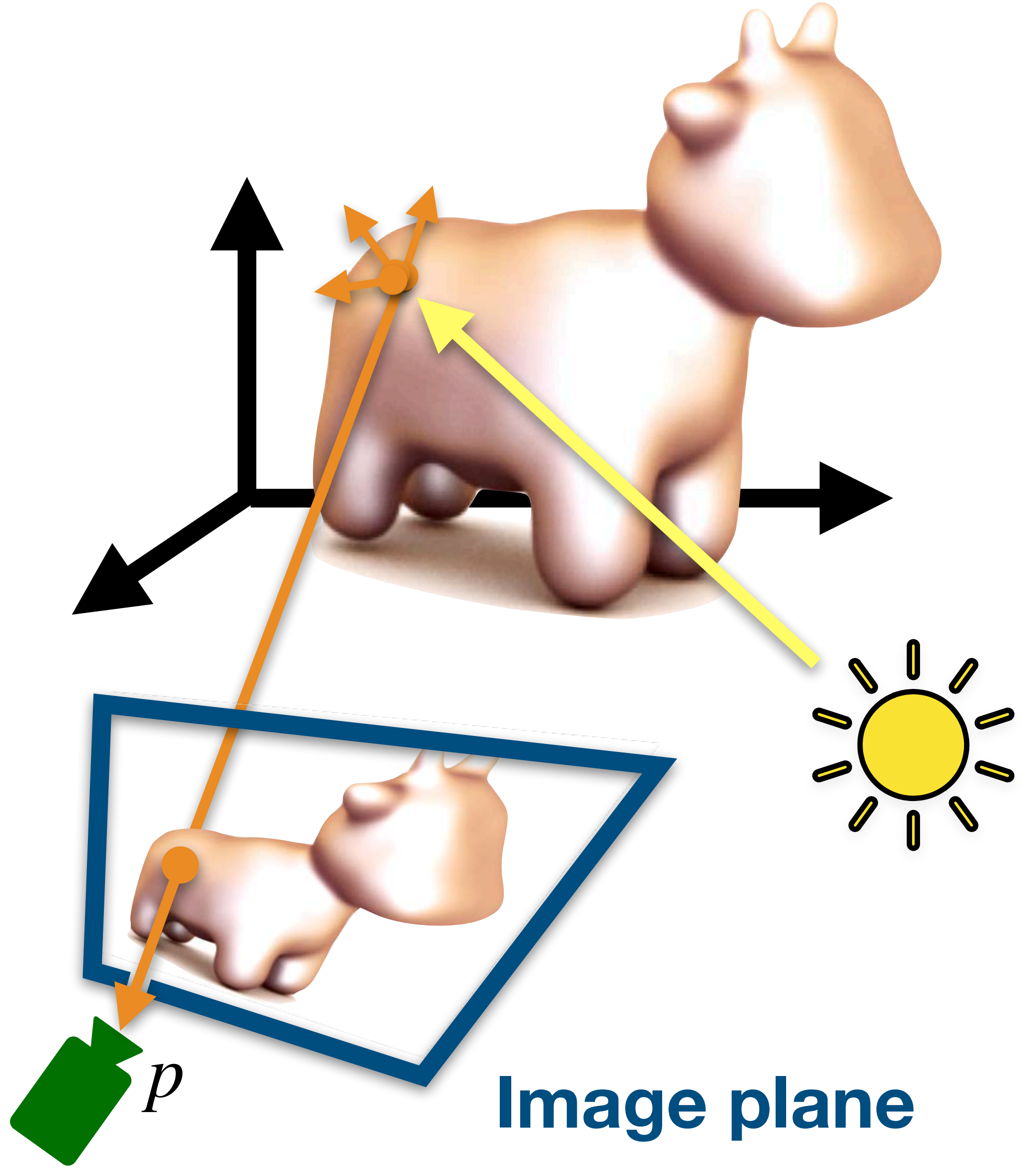
The graphics pipeline

- **Scene representation.**
  - Mesh, volume, implicits, …
  - Domain in $\mathbb{R}^3$.



Geometry processing

- **Rendering.**
  - Rasterization, ray tracing, volume ray casting.



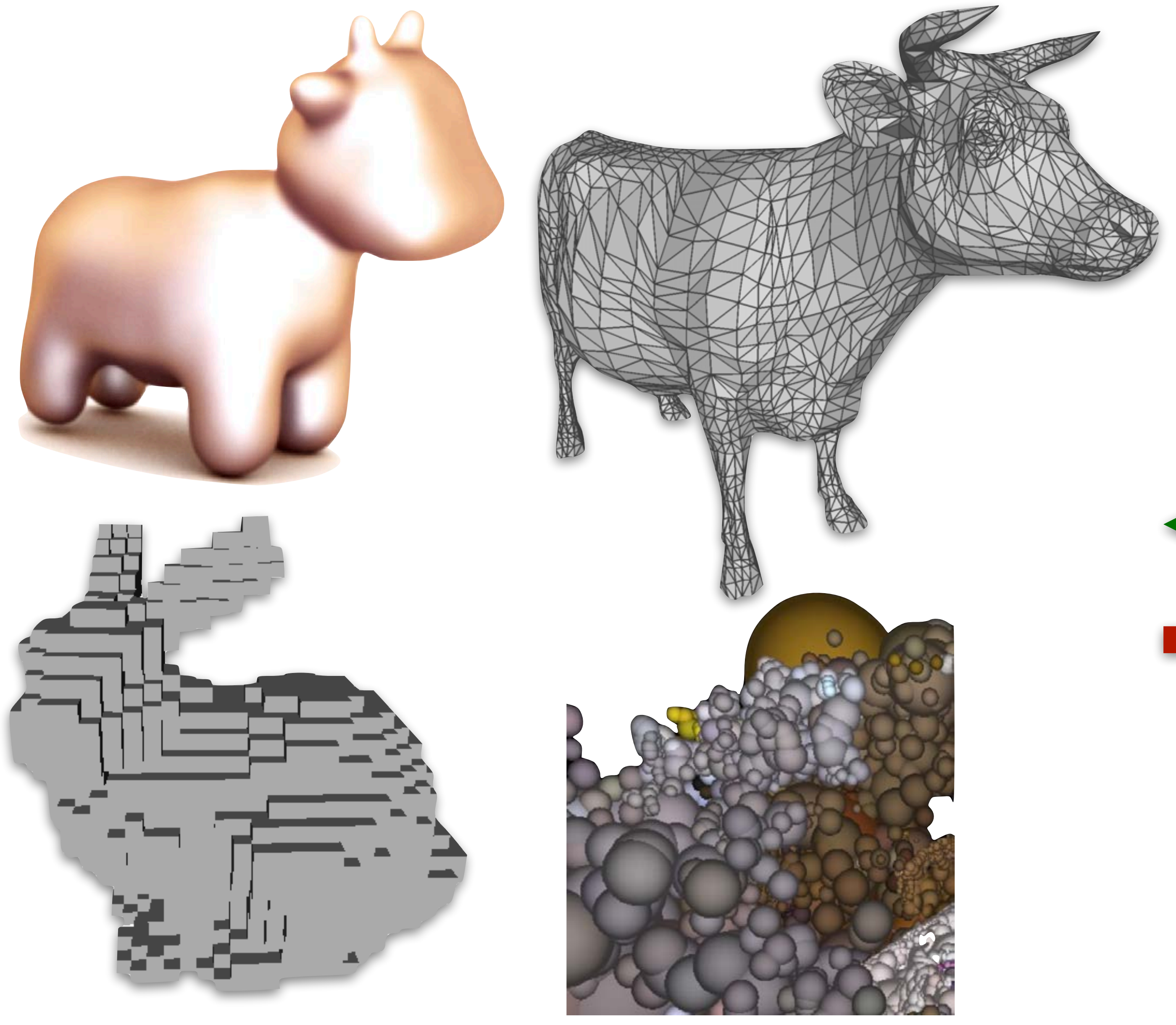
- **Rendered images.**

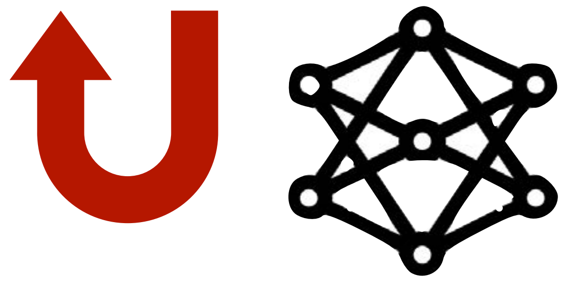
$$\mathbf{I} : \mathbb{R}^2 \to \mathscr{C}$$

Image processing

The neural graphics pipeline

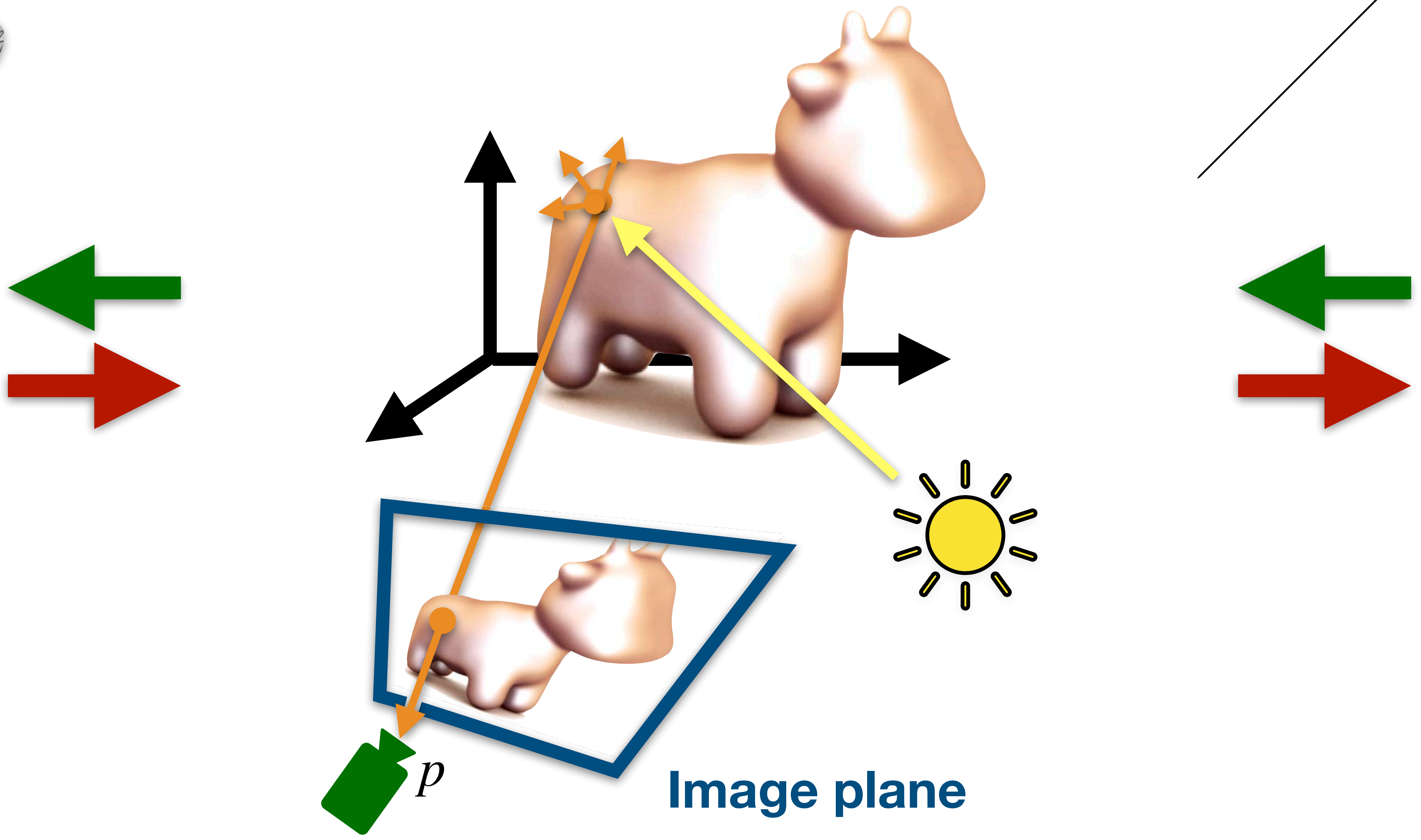
- **Diff scene representation.**
  - Mesh, volume, implicits, …
  - Domain in $\mathbb{R}^3$.





Geometry processing

- **Diff rendering.**
  - Rasterization, ray tracing, volume ray casting.



- **Rendered images.**

$\mathbf{I} : \mathbb{R}^2 \rightarrow \mathscr{C}$

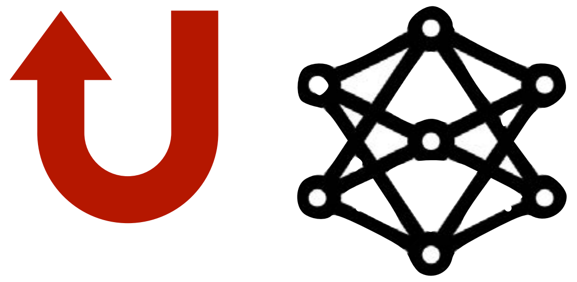


Image processing

Implicit neural representations in CG

- **Problem**: Represent a graphical object using a neural network $f : \mathbb{R}^n \to \mathbb{R}^p$.
  - $f(x) = W_d \circ f_{d-1} \circ \cdots \circ f_0(x) + b_d$
  - $f_i(x) = \sin(W_i x + b_i)$

- A **INR** is a network f where its parameters θ are implicitly defined by
  - $\mathscr{L}(\theta) = \mathscr{L}_{\textbf{data}}(\theta) + \mathscr{R}\big(\theta\big) = 0$

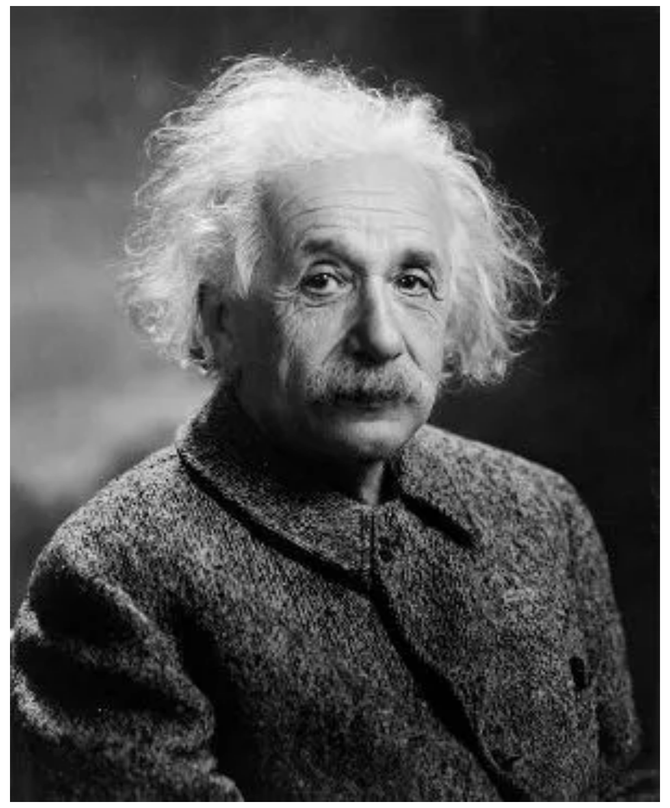
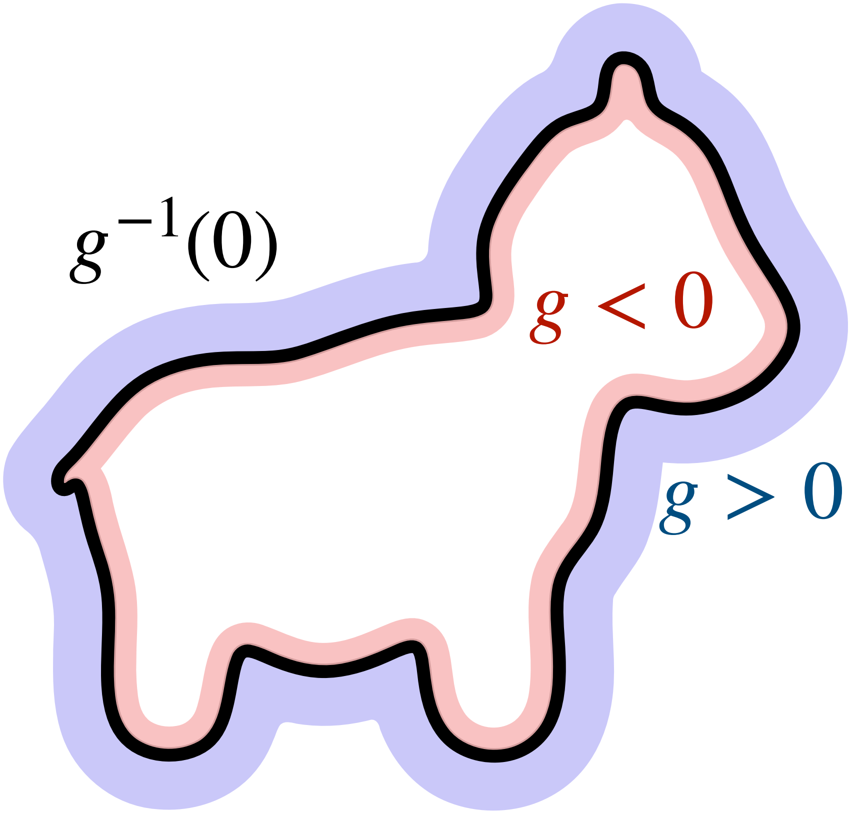
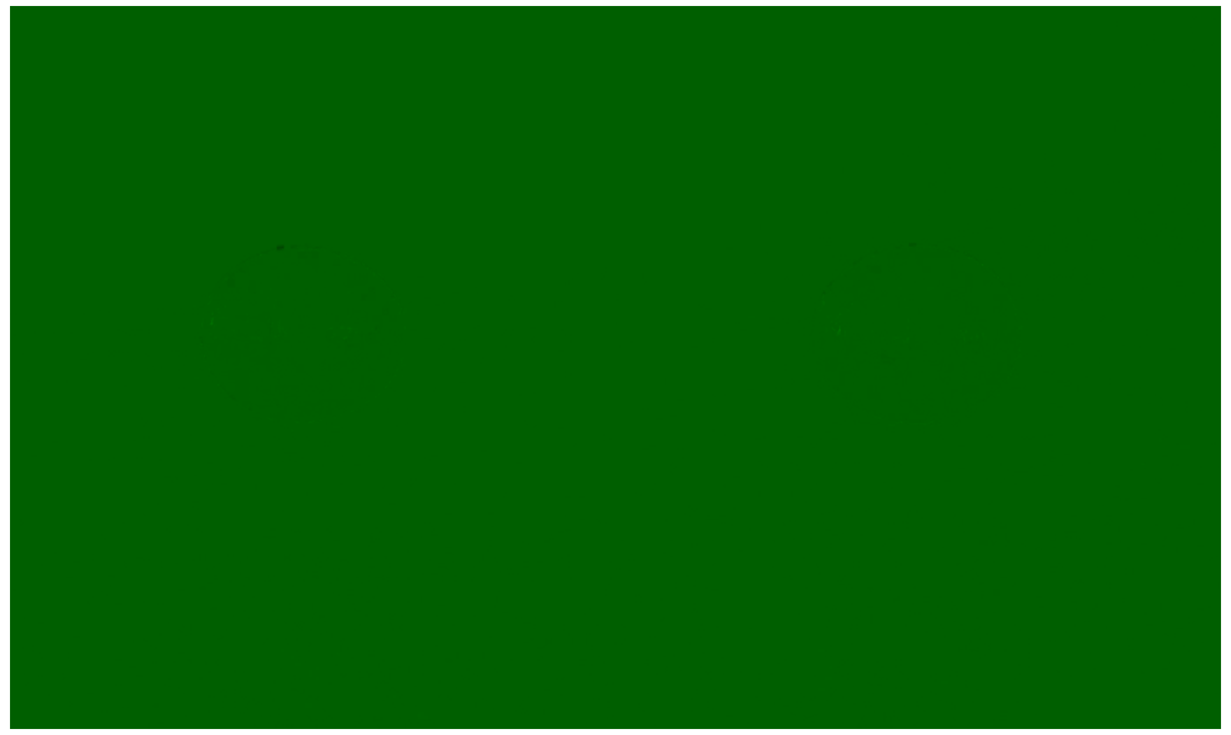


Image: $\mathbf{I} : \mathbb{R}^2 \to \mathscr{C}$

Image morphing: $f : \mathbb{R}^2 \times [0,1] \to \mathscr{C}$



Implicit surface: $g : \mathbb{R}^3 \to \mathbb{R}$



Surface evolution: $f : \mathbb{R}^3 \times [0,1] \to \mathbb{R}$

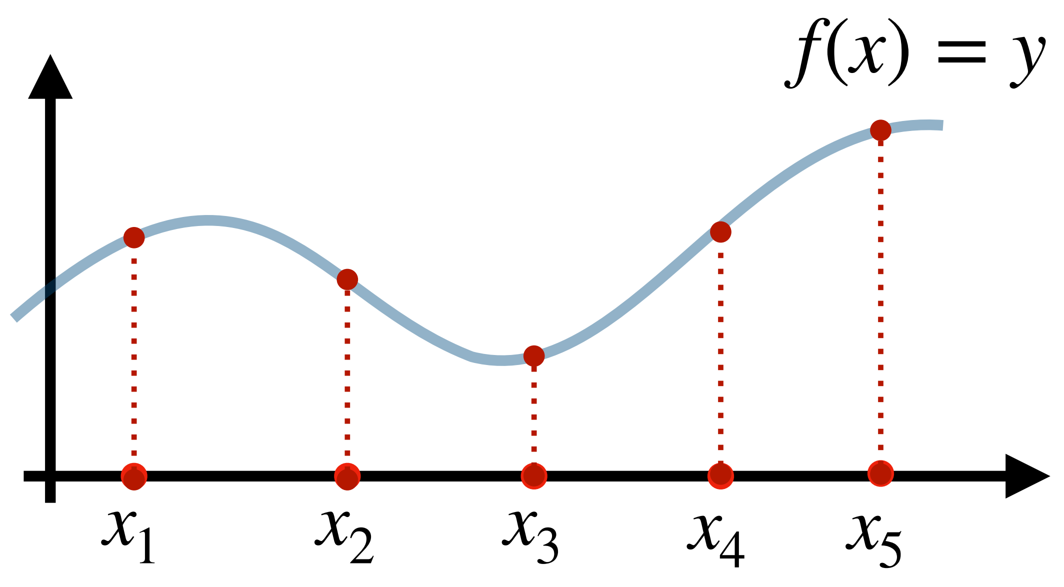
Schirmer et al. Neural Networks for Implicit Representations of 3D Scenes. SIBGRAPI. 2021.

Neural Media

- **Unisinos**
  - Luiz Schirmer
- **PUC-Rio**
  - Vinícius da Silva
  - Alberto Raposo
  - Hélio Lopes
- **IMPA**
  - Daniel Perazzo
  - Diana Aldana
  - Hallison Paz
  - Alberto Kopiler
  - Tiago Novello
  - Luiz Velho
- **U Coimbra**
  - Guilherme Schardong
  - Iurii Medvedev
  - Nuno Gonçalves

The INR pipeline

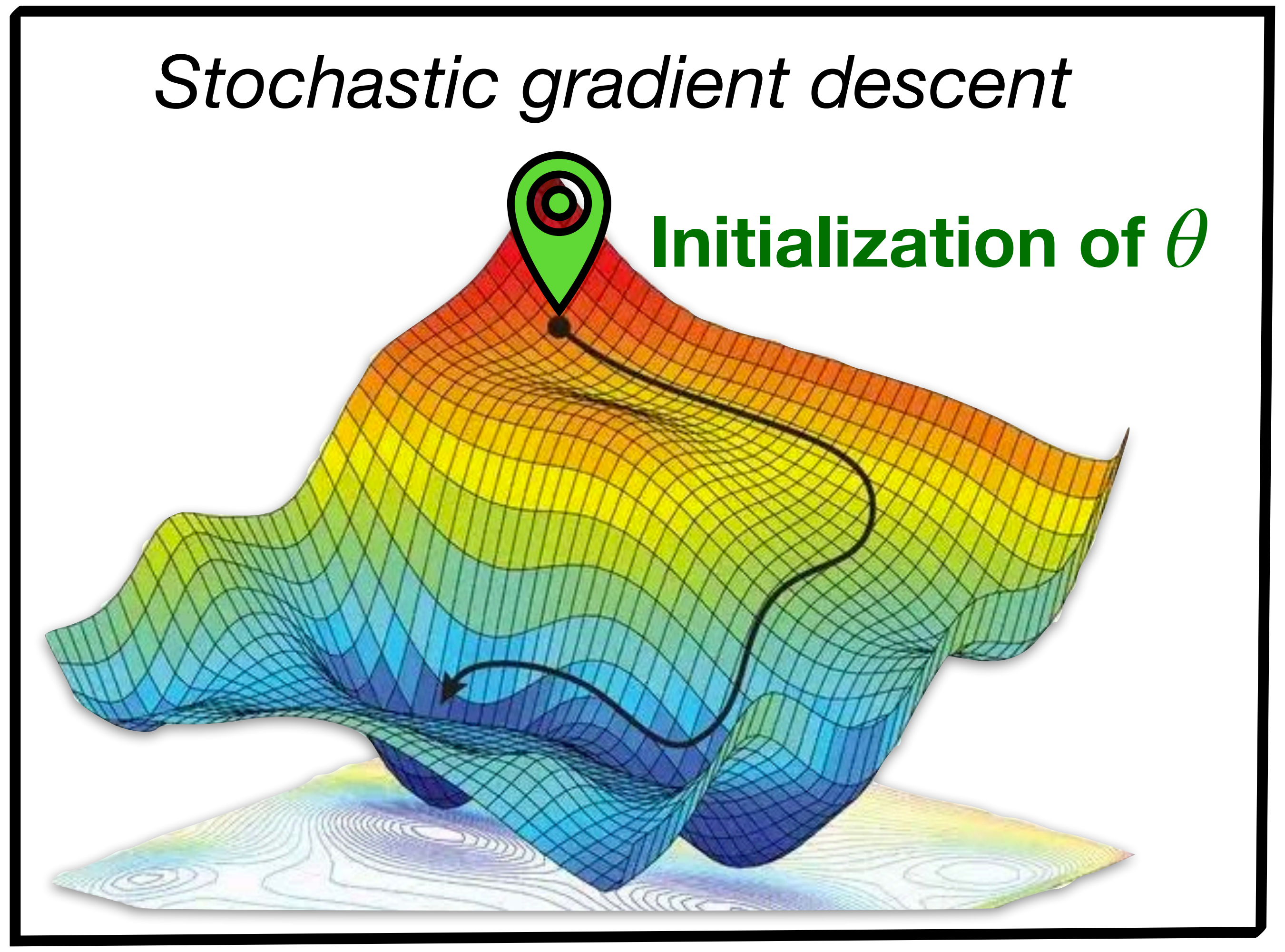
Input data



Sample $\{x_i, \mathbf{f}_i\}$ of an object $\mathbf{f} : \mathbb{R}^n \to \mathbb{R}^p$.

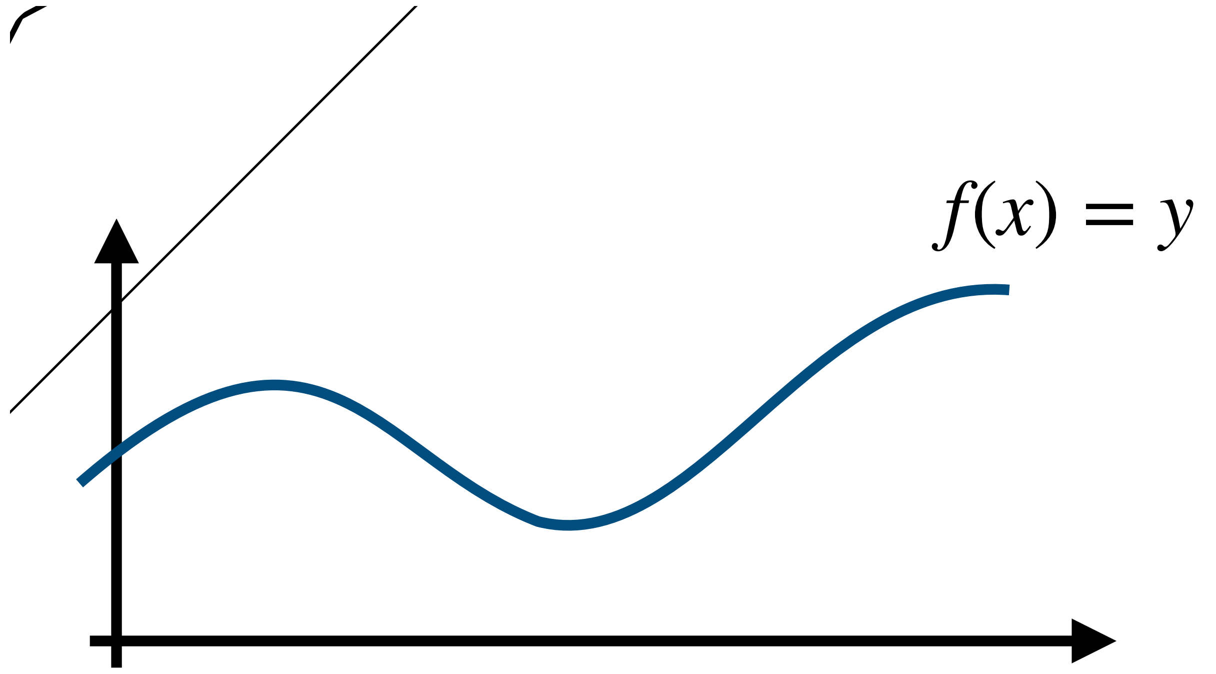
Sampling →

Training



→

Inference



Neural object $f : \mathbb{R}^n \to \mathbb{R}^p$.

- Smooth.
- Analytical derivatives.

Loss function

$$\mathscr{L}(\theta) = \sum \left(f(x_i) - \mathbf{f}_i\right)^2 + \mathscr{R}(\theta)$$

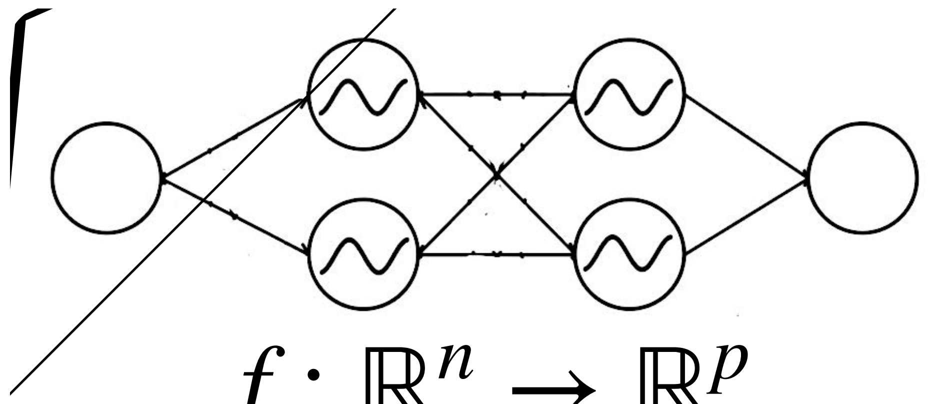
Data term

Implicit regularization

Forces f to fit a given property.

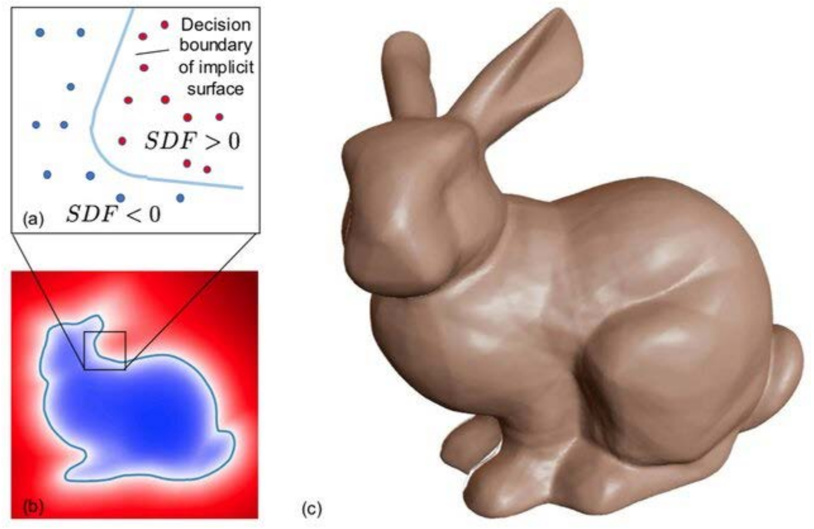
- Solution of a PDE.

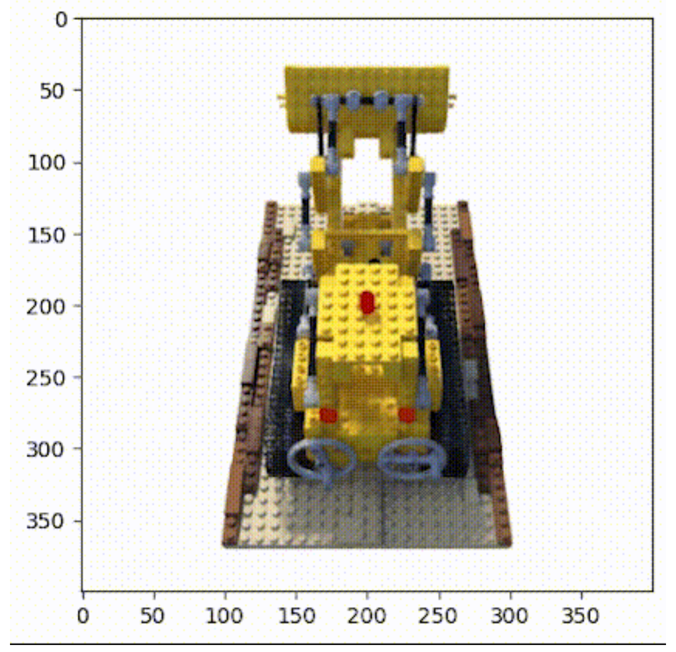
Smooth neural network



$f : \mathbb{R}^n \to \mathbb{R}^p$

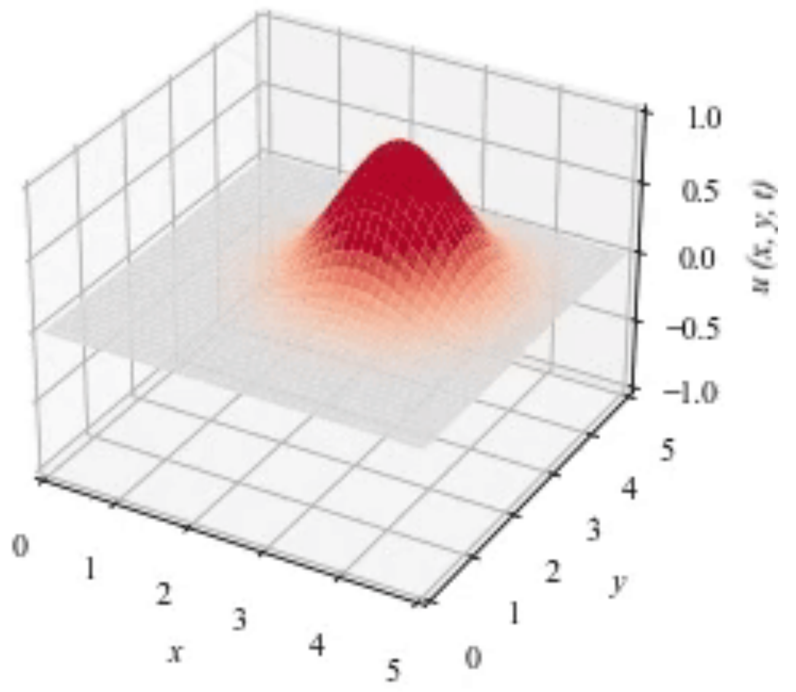
Related works





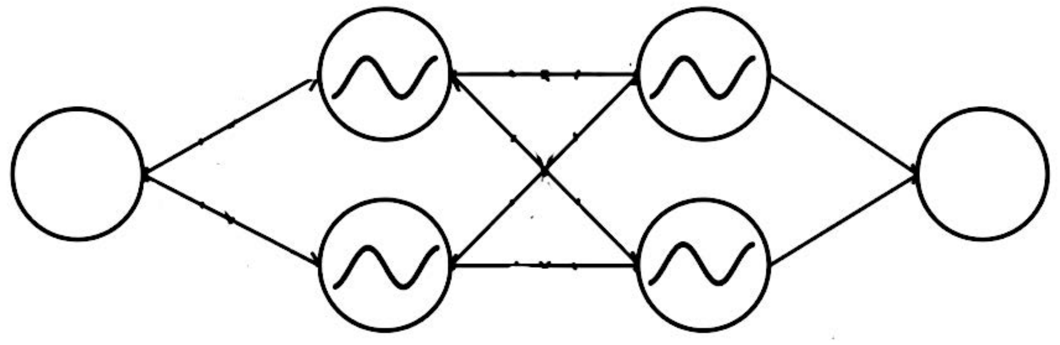
- Implicit neural representations, coordinate-based networks, neural fields, and neural implicits.

- [Mescheder et al. 2018, Occupancy Net]
- [Park et al. 2018, DeepSDF]
- [Sitzmann at al. 2019, SIREN]
- [Gropp et al. 2019, IGR]
- [Mildenhall et al. 2020, NeRF]
- ...



- Neural PDE solvers (PINNs).

- [Sirignano et al. 2018, DGM]
- [Raissi et al. 2019, PINNs]
- ...



- Sinusoidal INRs.

- [Parascandolo et al. 2016, Taming]
- [Sitzmann at al. 2019, SIREN]
- ...

Projects…

Exploring Differential Geometry in Neural Implicits

Neural Implicit Surface Evolution using Differential Equations

Neural implicit mapping via nested neighborhoods

3D scene reconstruction

Neural flows

Multiresolution sinusoidal INRs

Periodic textures

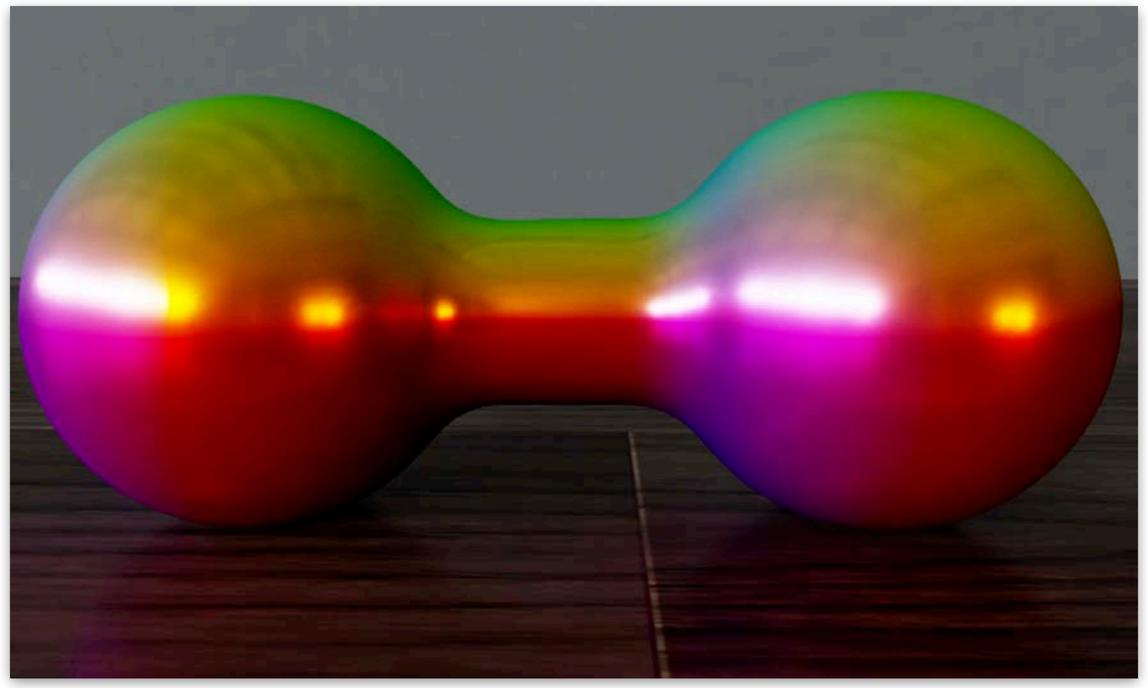
Taming the sinusoidal INRs

Projects…

Geometry processing

Exploring Differential Geometry in Neural Implicits

Neural Implicit Surface Evolution using Differential Equations



Neural implicit mapping via nested neighborhoods

3D scene reconstruction

Neural flows

Multiresolution sinusoidal INRs

Periodic textures

Taming the sinusoidal INRs

Projects…

Exploring Differential Geometry in Neural Implicits

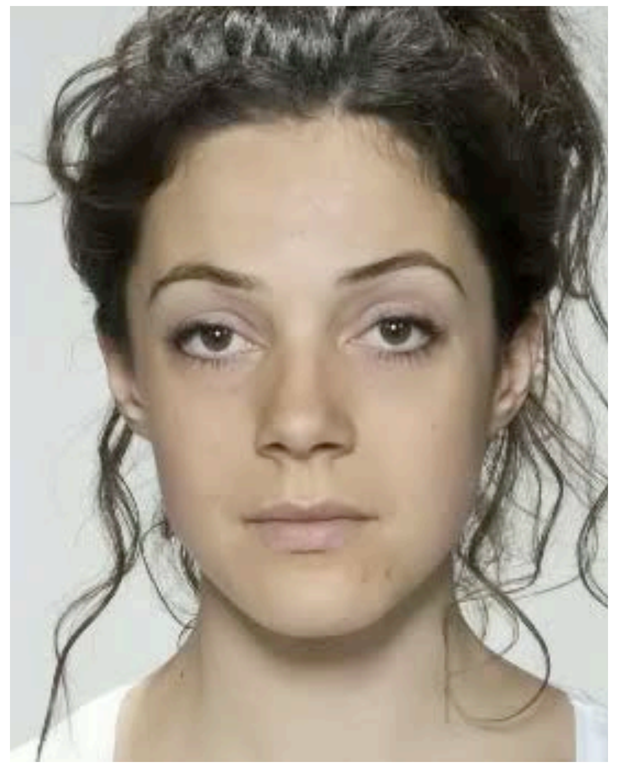
Neural Implicit Surface Evolution using Differential Equations

Neural implicit mapping via nested neighborhoods

3D scene reconstruction

Image processing

Neural flows



Multiresolution sinusoidal INRs

Periodic textures

Taming the sinusoidal INRs

Projects…

Exploring Differential Geometry in Neural Implicits

Neural Implicit Surface Evolution using Differential Equations

Neural implicit mapping via nested neighborhoods

3D scene reconstruction

Neural flows

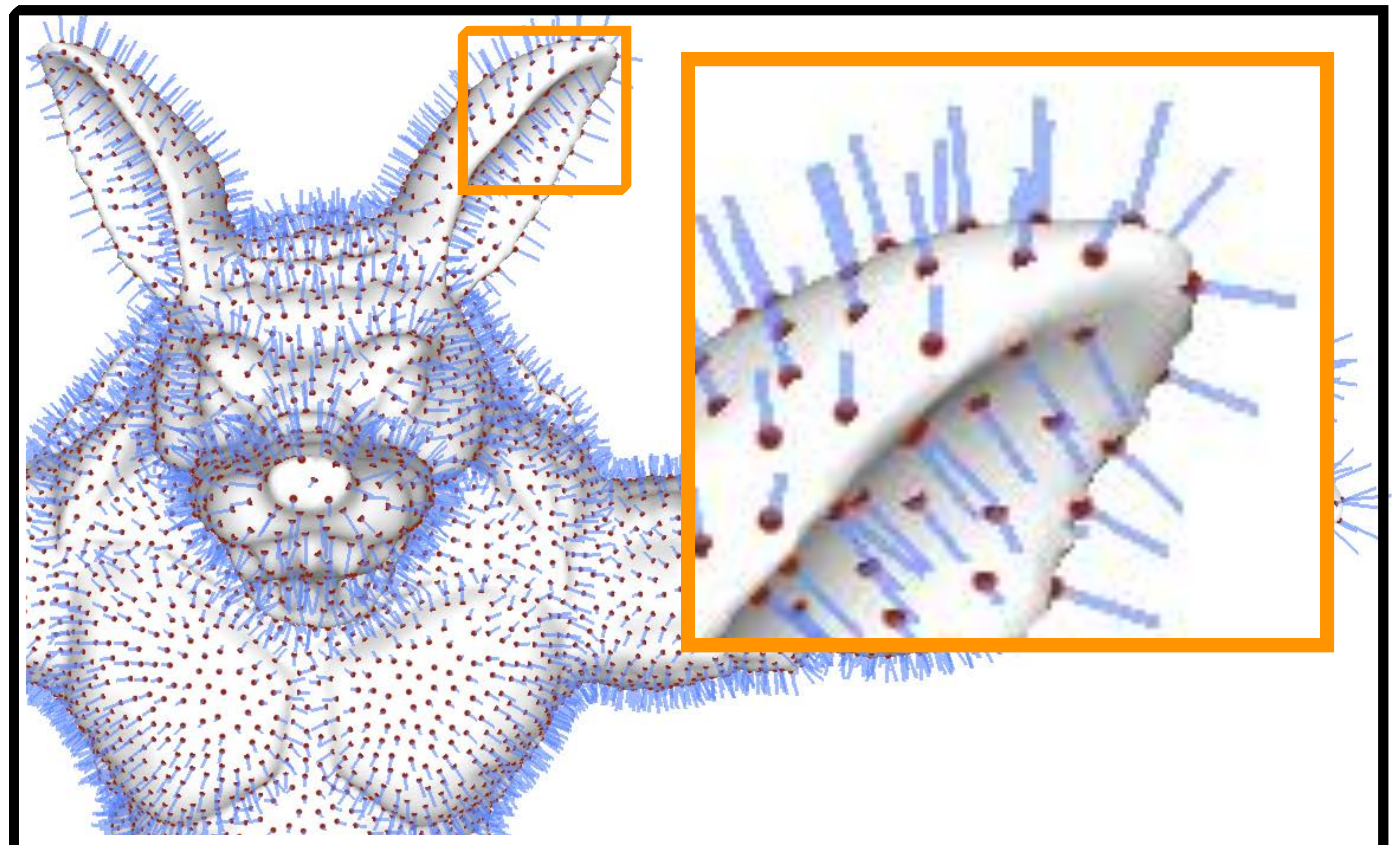
Multiresolution sinusoidal INRs

Periodic textures

Taming the sinusoidal INRs

Surfaces as level sets of INRs

Data



Sample $\{p_i, N_i\}$ of a surface S.

Sampling →

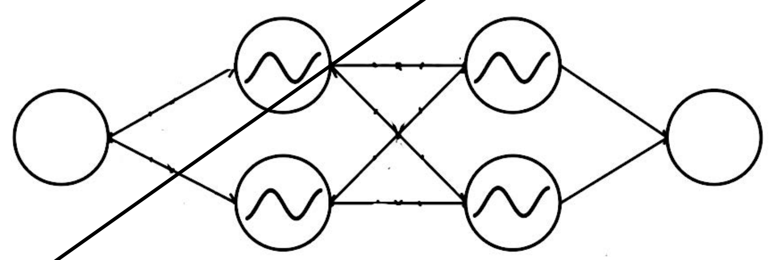
Training

Find a minimum of $\mathscr{L}$ using *gradient descent.*

Loss function

?

Smooth neural network



$g : \mathbb{R}^3 \to \mathbb{R}$

Rendering

- $S \approx g^{-1}(0)$ is **smooth.**
- Normals/curvatures in **closed form.**

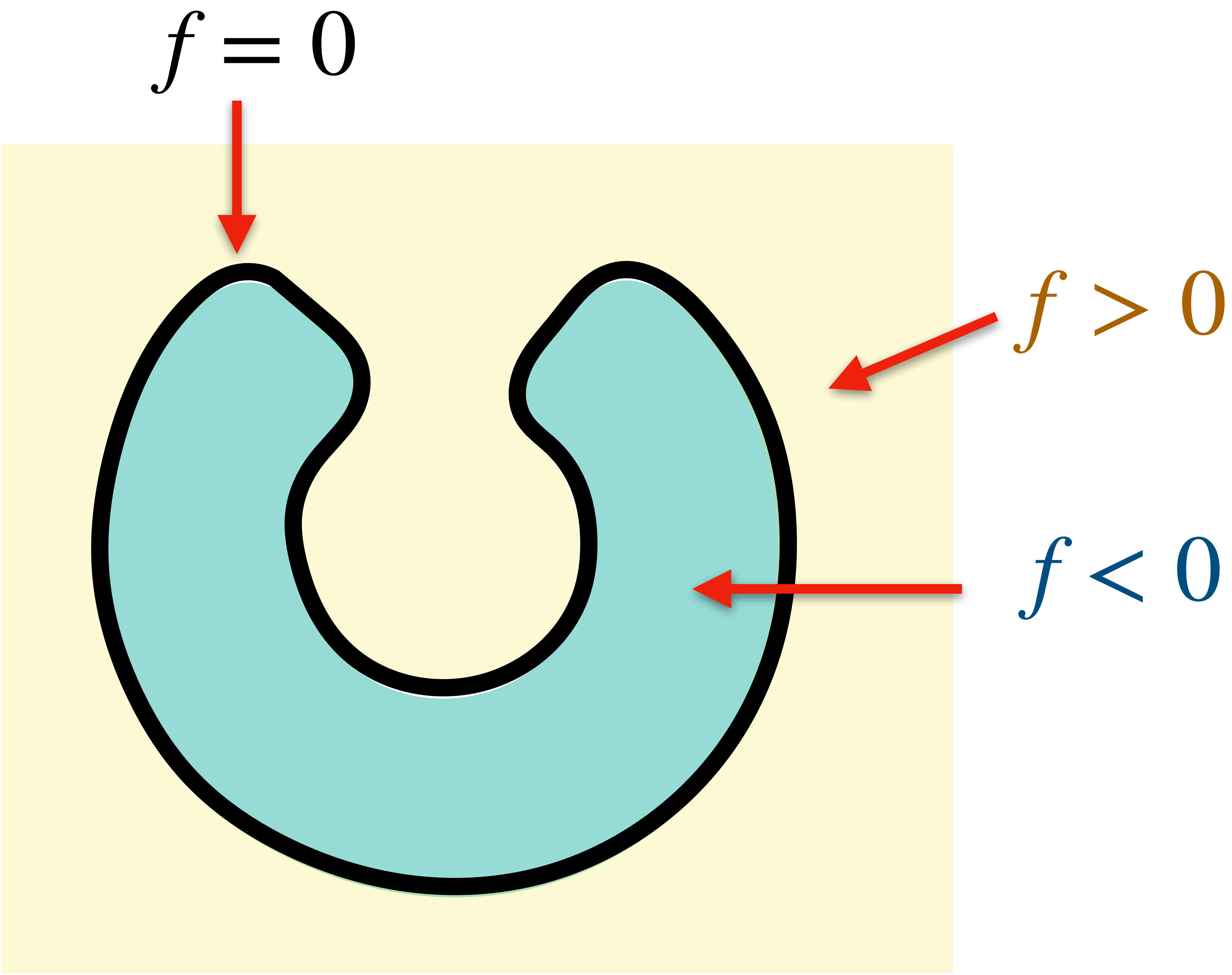
Regularization

?

Novello et al. Exploring differential geometry in neural implicits. Computers & Graphics, 2022.

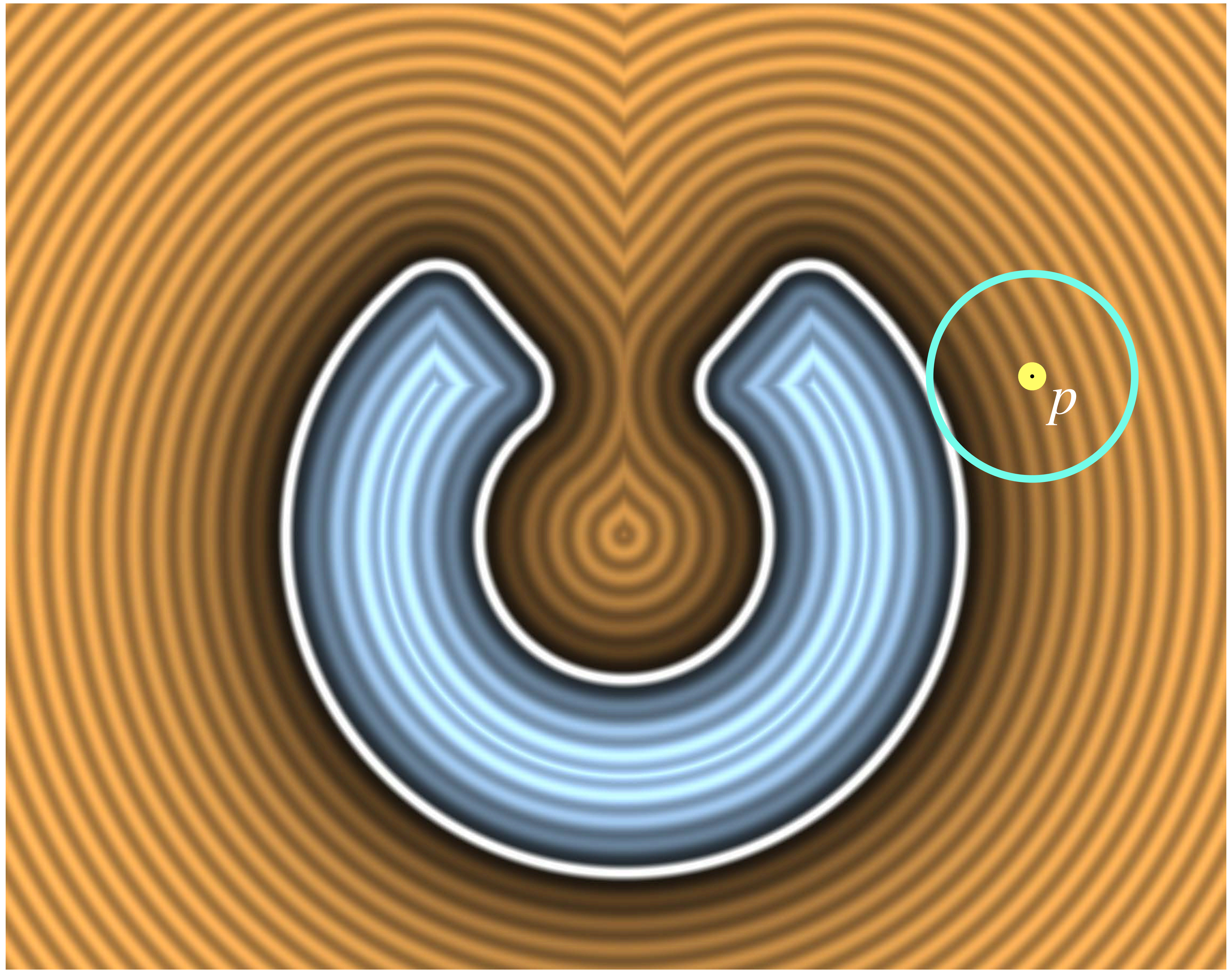
Implicit shapes

- **Problem**: Represent a surface S implicitly.
  - Find a function f such that:
    - $f(x) = 0$ **if** x **is on** S**.**
    - $f(x) > 0$ **if** x **is outside** S**.**
    - $f(x) < 0$ **if** x **is inside** S**.**
  - Thus, $S = \{x \mid f(x) = 0\}$
  - Many options for f



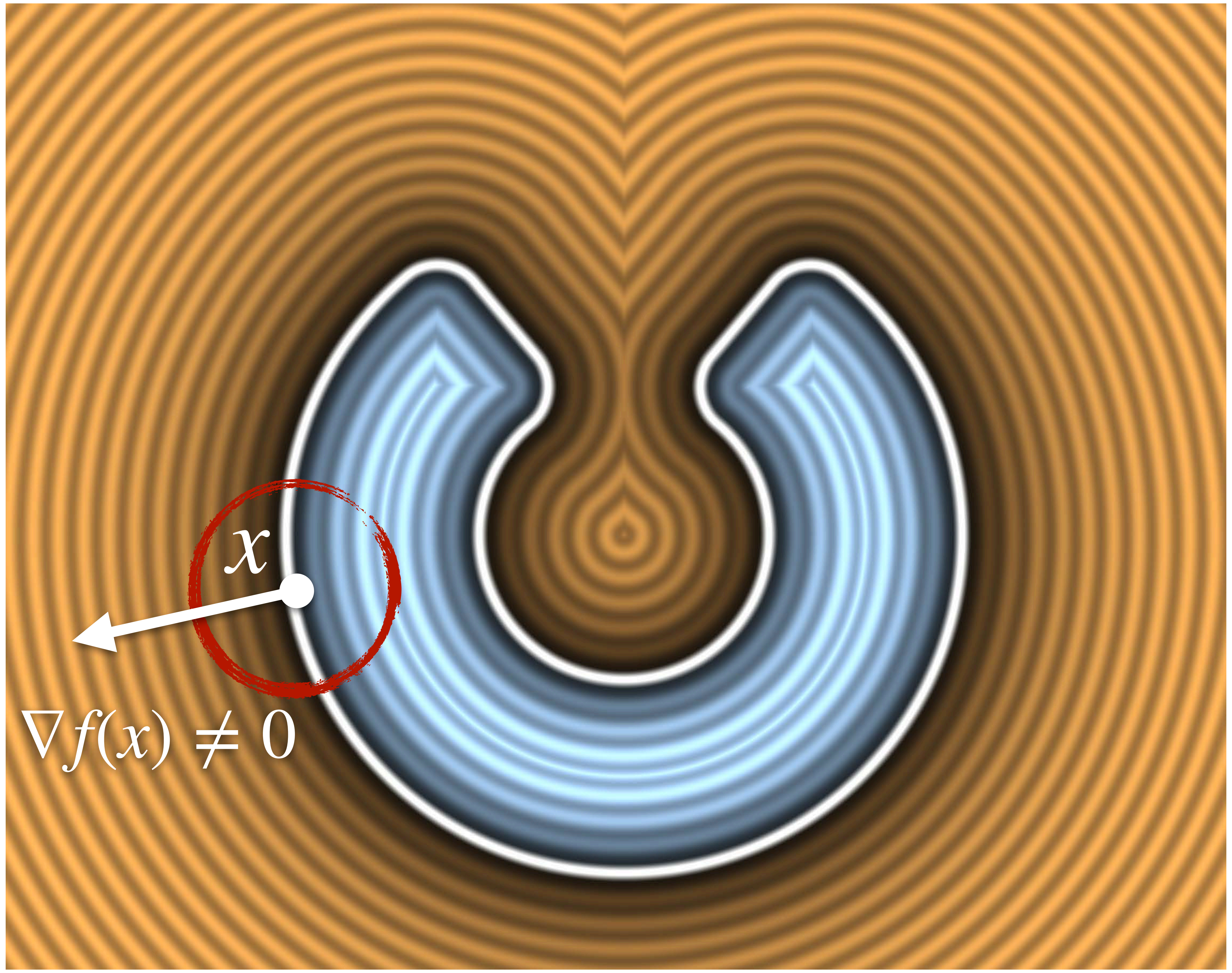
Implicit shapes

- The **signed distance function** (SDF) f of S is an important example of implicit function:
  - $f(x)$ measures the distance of each point x to S:
    - $f(x) = 0$ **if x is on S**
    - $f(x) > 0$ **if x is outside S**
    - $f(x) < 0$ **if x is inside S**



Implicit shapes

- **Implicit function theorem:**
  - For $f^{-1}(0)$ to be a surface in a **neighborhood** of a point x we need $\nabla f(x) \neq 0$.



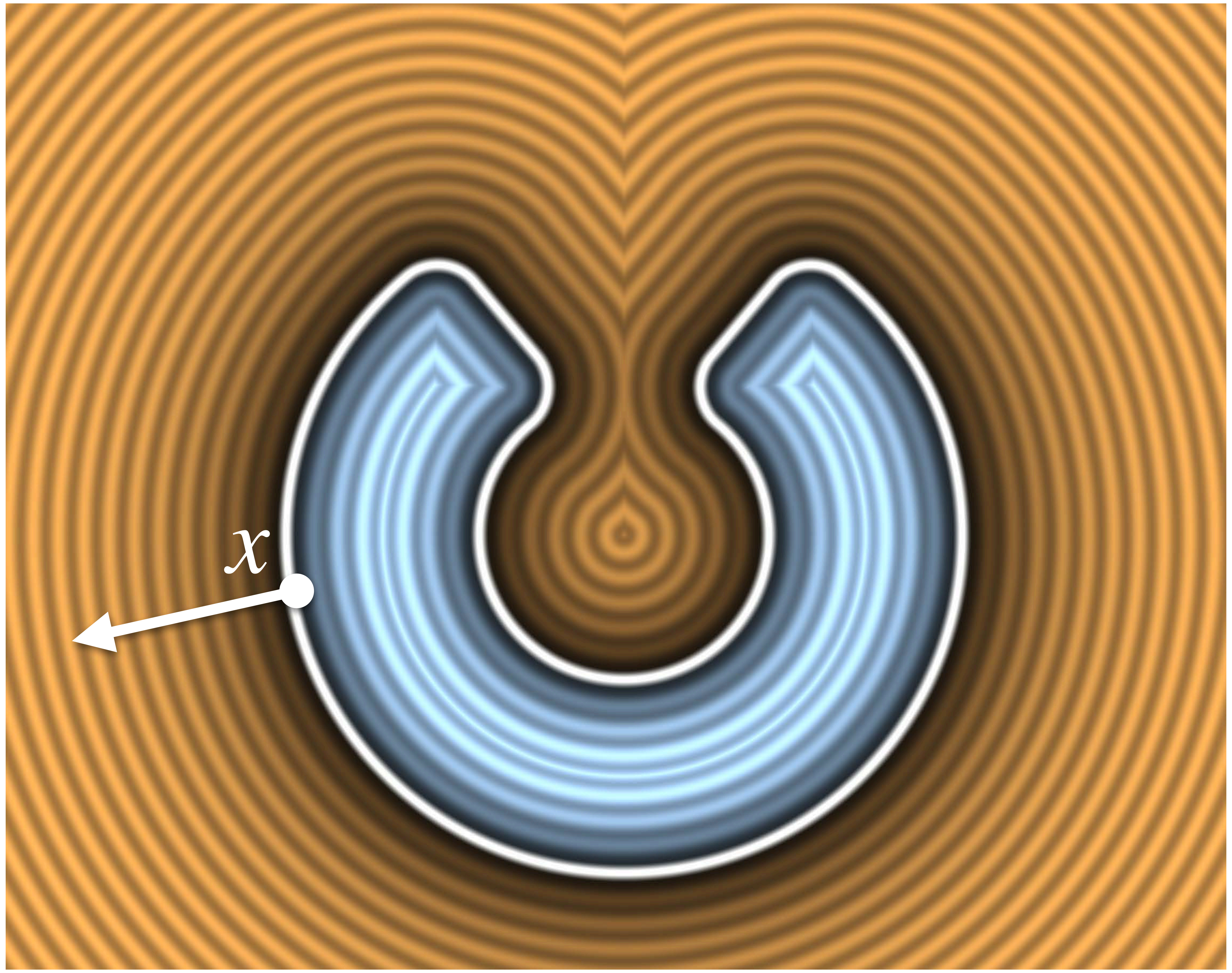
Signed distance function

- A function $g : \Omega \to \mathbb{R}$ fits the SDF of S if it satisfies the **Eikonal** eq.

$$\begin{cases} |\nabla g| = 1 & \text{in } \Omega, \\ g = 0 & \text{on } S. \end{cases}$$

- Which implies that $\dfrac{\partial g}{\partial N} = \langle \nabla g, N \rangle = 1.$

- We use these constraints to define the loss functional.



Signed distance function

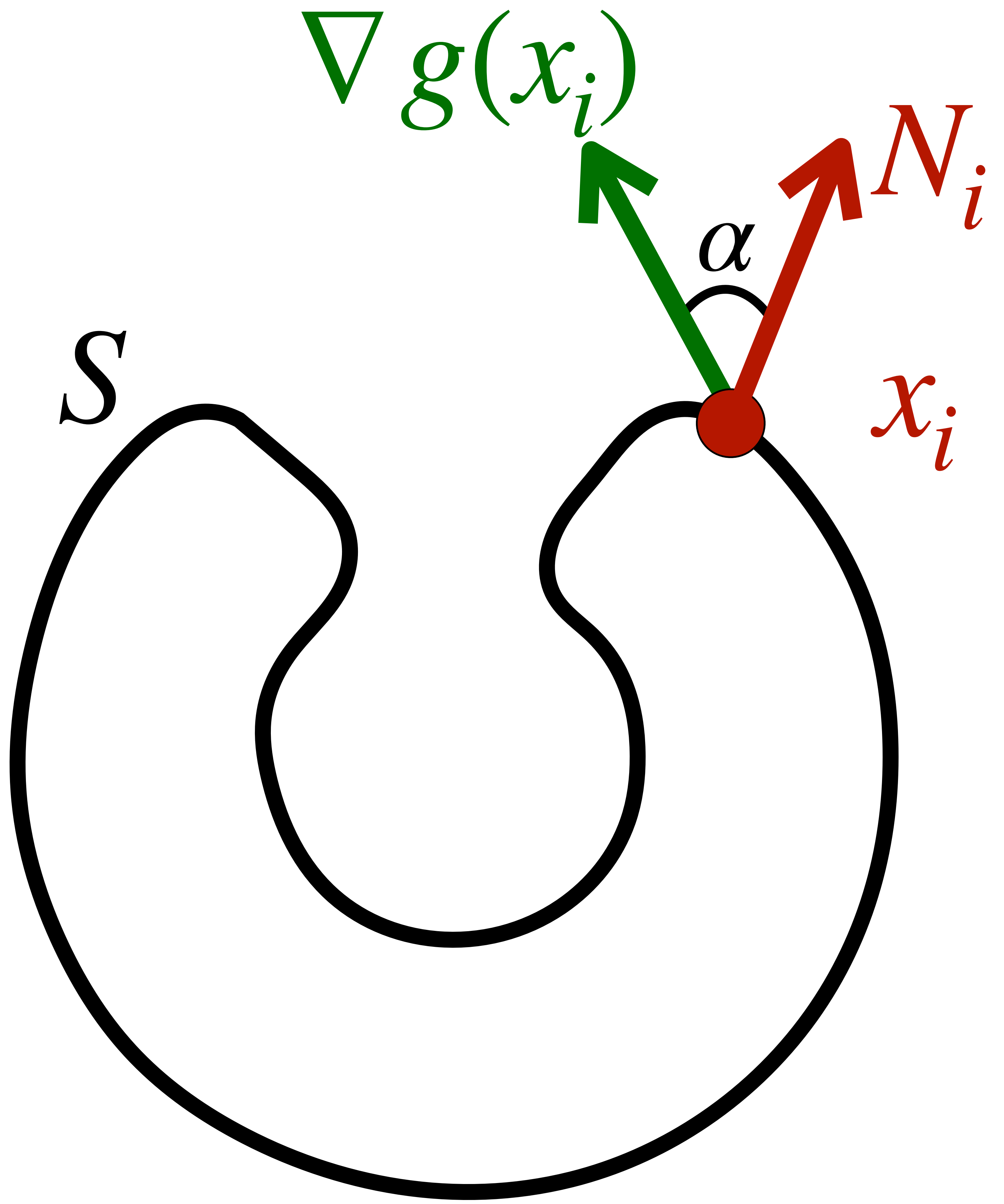
- A function $g : \Omega \to \mathbb{R}$ fits the SDF of S if it satisfies the **Eikonal** eq.

$$\begin{cases} |\nabla g| = 1 \text{ in } \Omega, \\ g = 0 \quad\quad \text{on } S. \end{cases}$$

- Which implies that $\frac{\partial g}{\partial N} = \langle \nabla g, N \rangle = 1.$

- We use these constraints to define the loss functional.

- In practice, we have a **sample** $\{x_i, N_i\}$



Surfaces as level sets of INRs

- $S \approx g^{-1}(0)$ is **smooth.**
- Normals/curvatures in **closed form.**

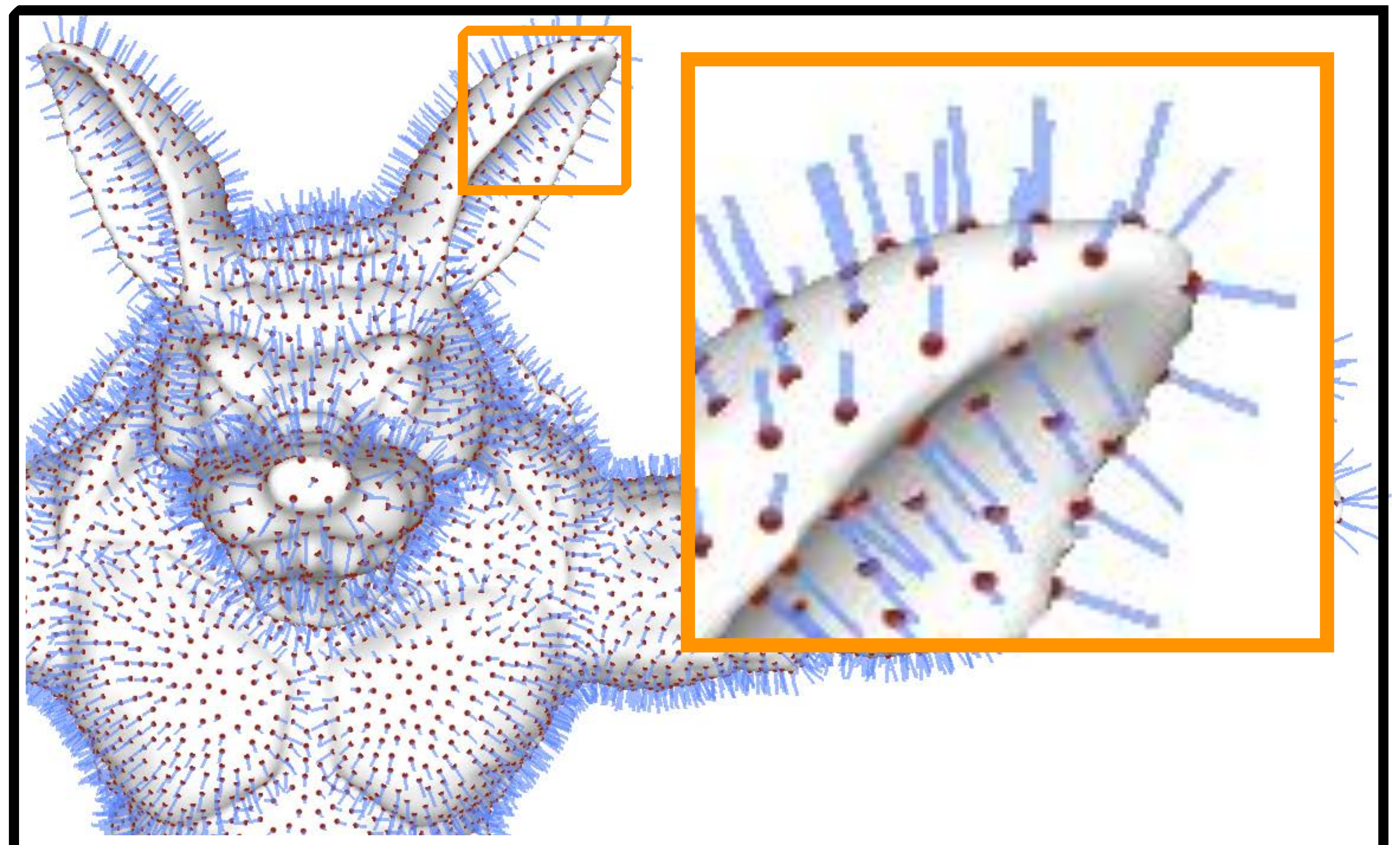
Regularization

?

Novello et al. Exploring differential geometry in neural implicits. Computers & Graphics, 2022.

Surfaces as level sets of INRs

Data



Sample $\{p_i, N_i\}$ of a surface S.

Sampling

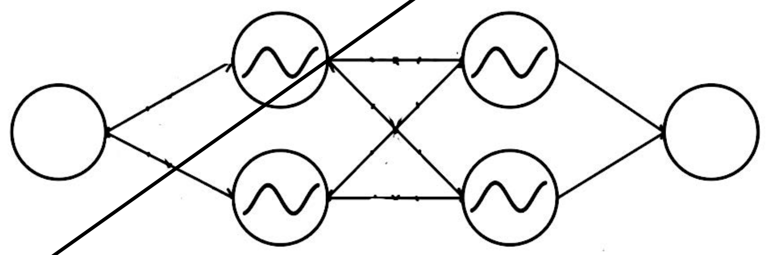
Training

Find a minimum of $\mathscr{L}$ using *gradient descent.*

Loss function

?

Smooth neural network



$g : \mathbb{R}^3 \rightarrow \mathbb{R}$

Rendering

- $S \approx g^{-1}(0)$ is **smooth.**
- Normals/curvatures in **closed form.**

Regularization

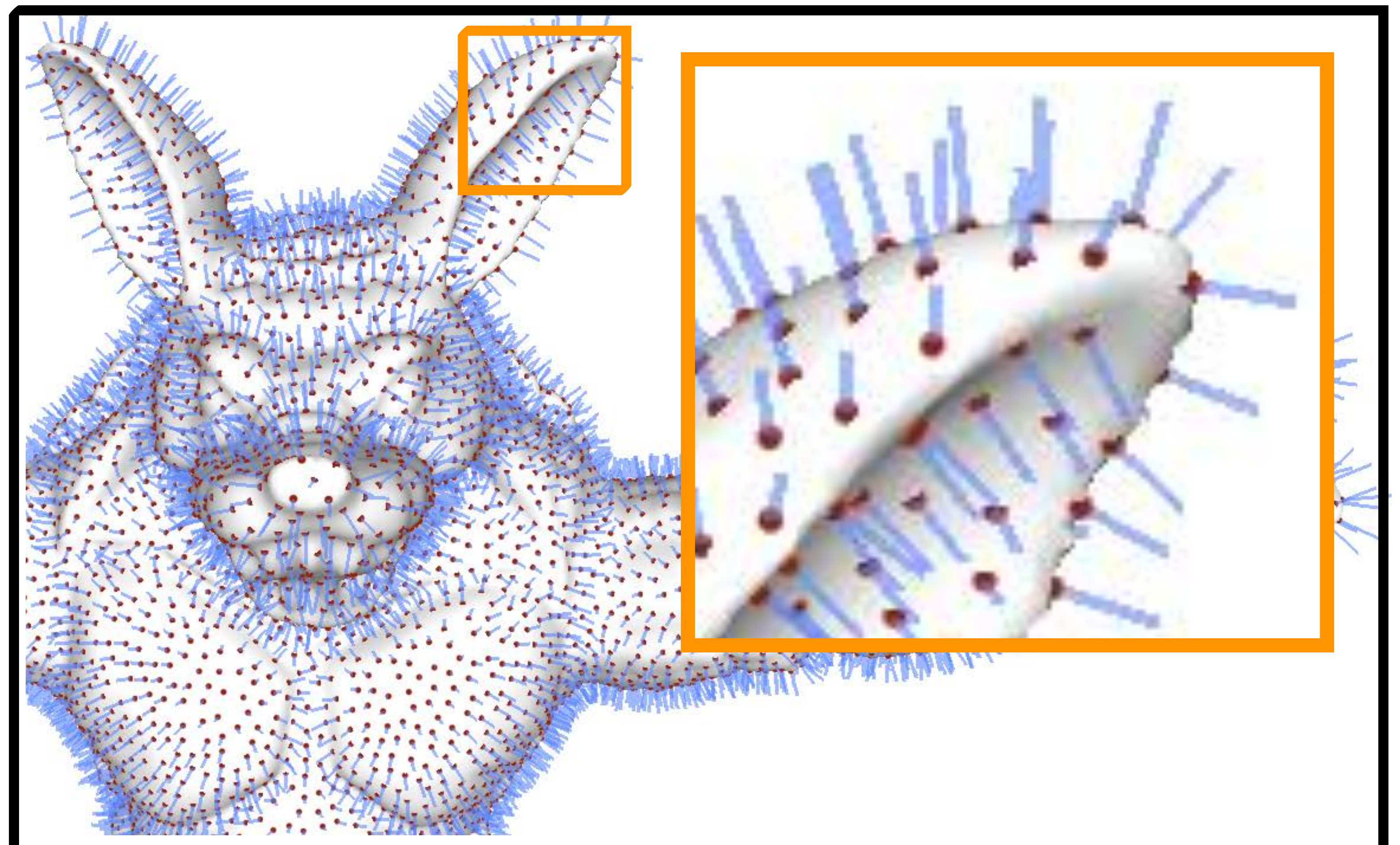
Forces f to be a SDF.

$$\begin{cases} \mathscr{F} = |\nabla g| - 1 = 0 & \textbf{in } \Omega, \\ g = 0 & \textbf{on } S \end{cases}$$

Novello et al. Exploring differential geometry in neural implicits. Computers & Graphics, 2022.

Surfaces as level sets of INRs

Data



Sample $\{p_i, N_i\}$ of a surface S.

Sampling →

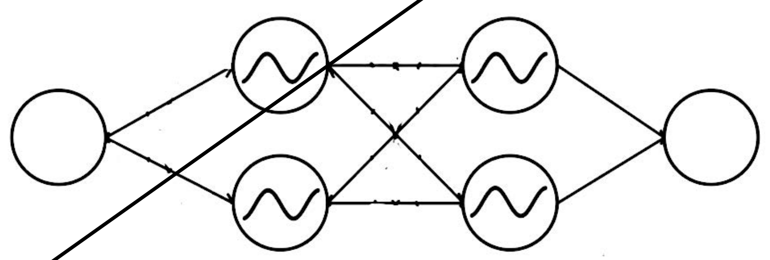
Training

Find a minimum of $\mathscr{L}$ using *gradient descent.*

Loss function

$$\mathscr{L}(\theta) = \underbrace{\sum g(p_i)^2 + \left(1 - \langle \nabla g, N_i \rangle\right)}_{\text{data term}} + \underbrace{\int_\Omega \mathscr{F}^2 dx}_{\text{Eikonal term}}$$

Smooth neural network



$g : \mathbb{R}^3 \to \mathbb{R}$

Rendering

- $S \approx g^{-1}(0)$ is **smooth.**
- Normals/curvatures in **closed form.**

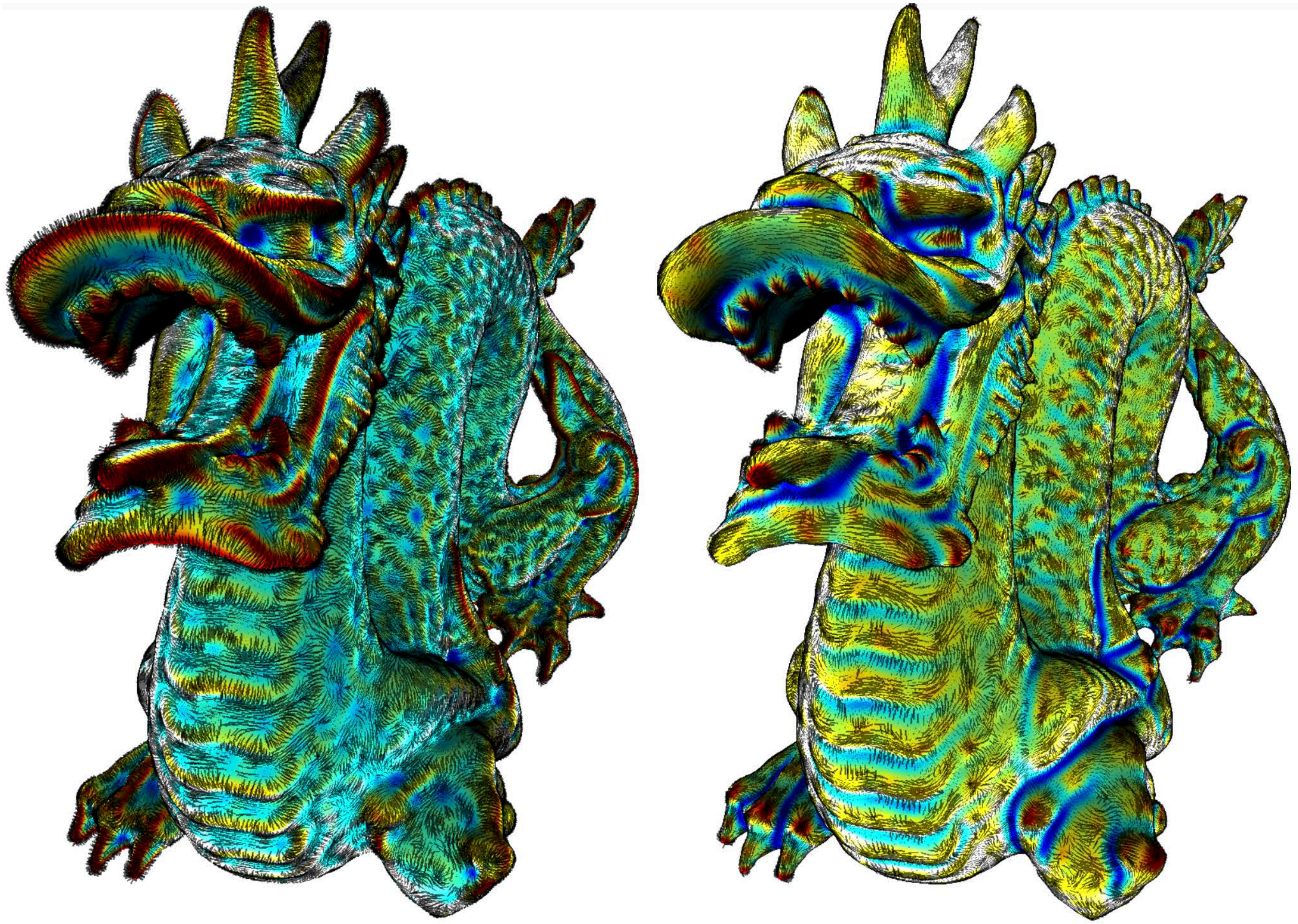
Regularization

Forces f to be a SDF.

$$\begin{cases} \mathscr{F} = |\nabla g| - 1 = 0 & \textbf{in } \Omega, \\ g = 0 & \textbf{on } S \end{cases}$$

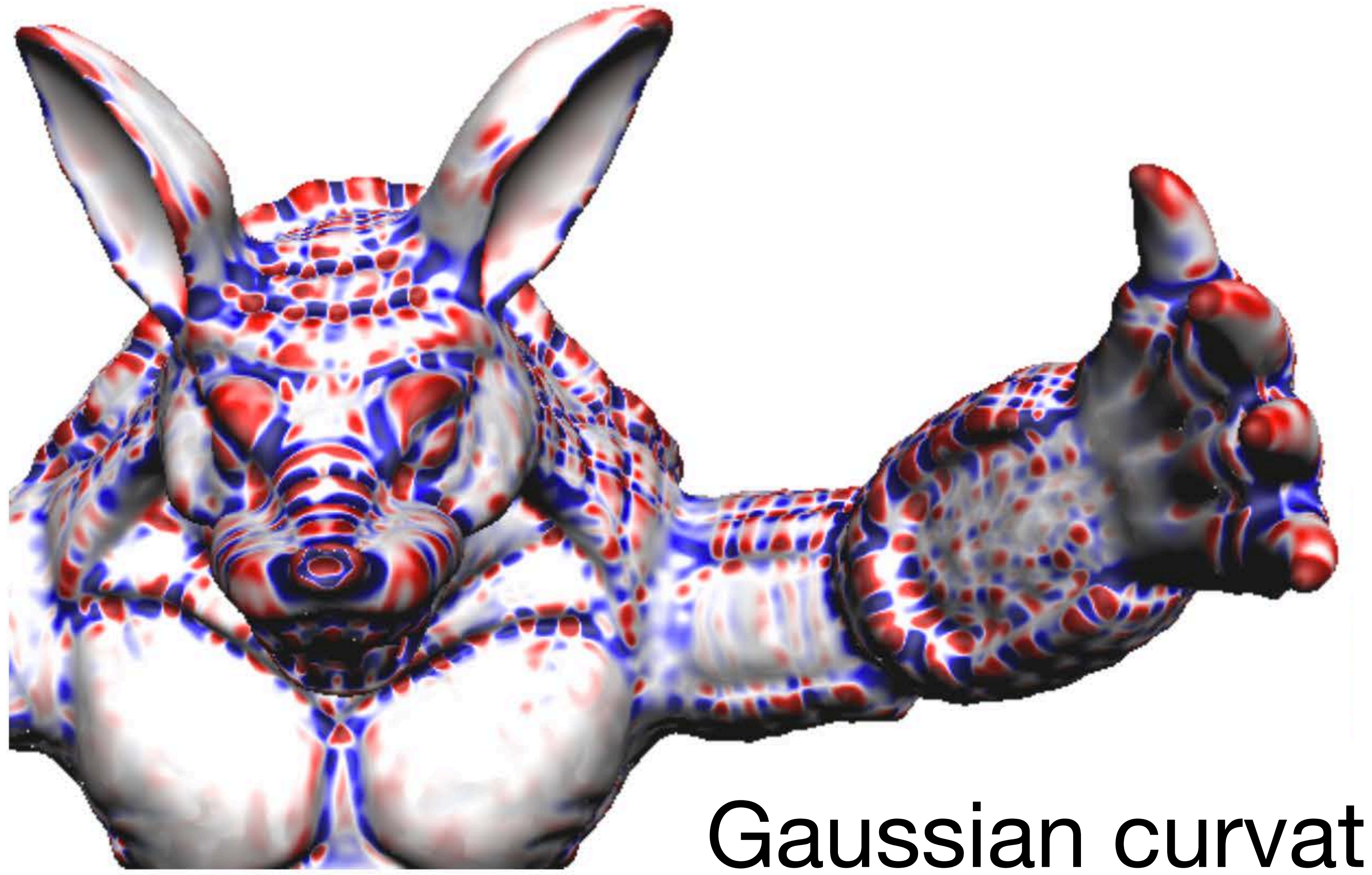
Novello et al. Exploring differential geometry in neural implicits. Computers & Graphics, 2022.

Curvatures

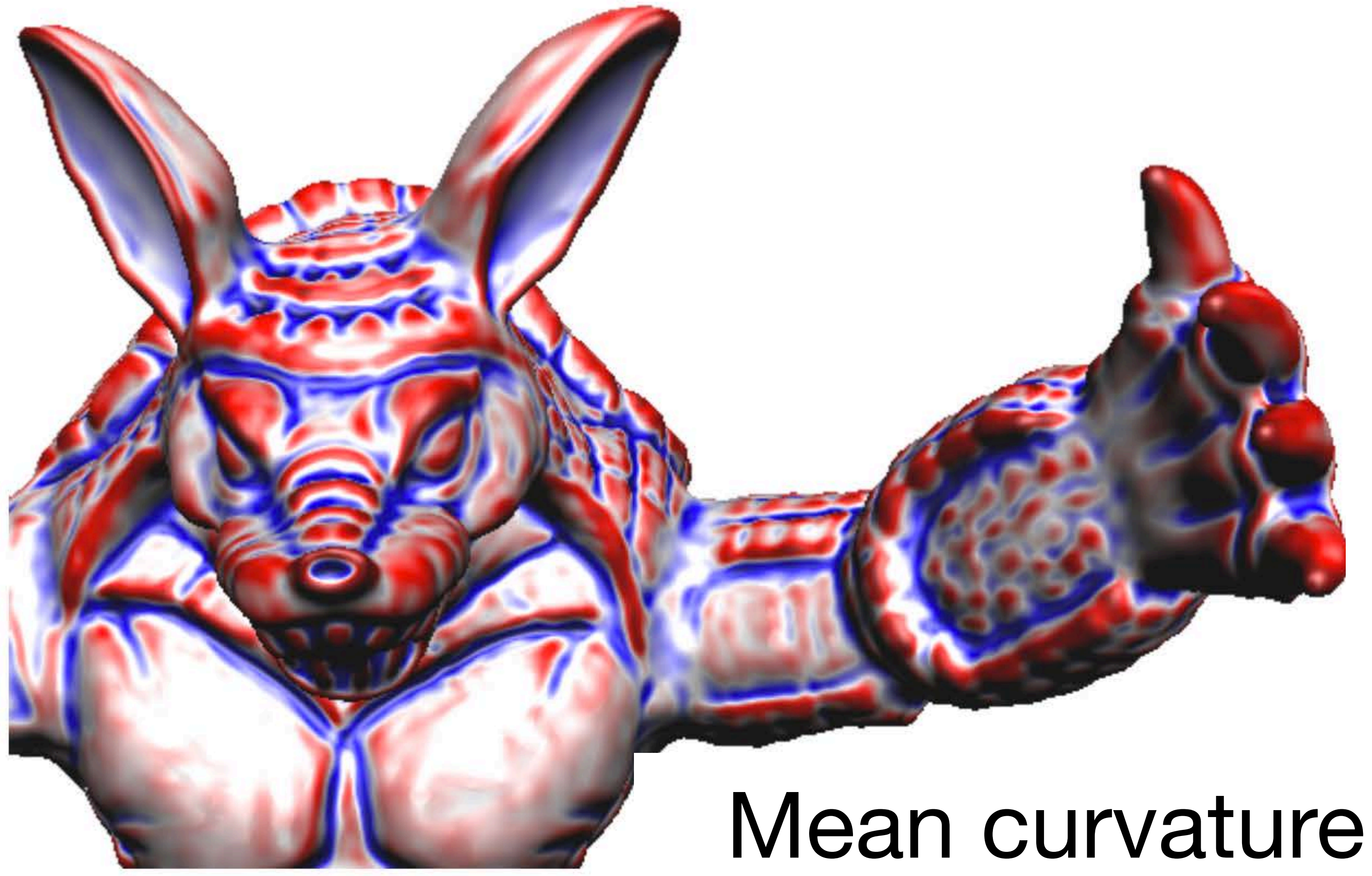


Max curvature

Min curvature



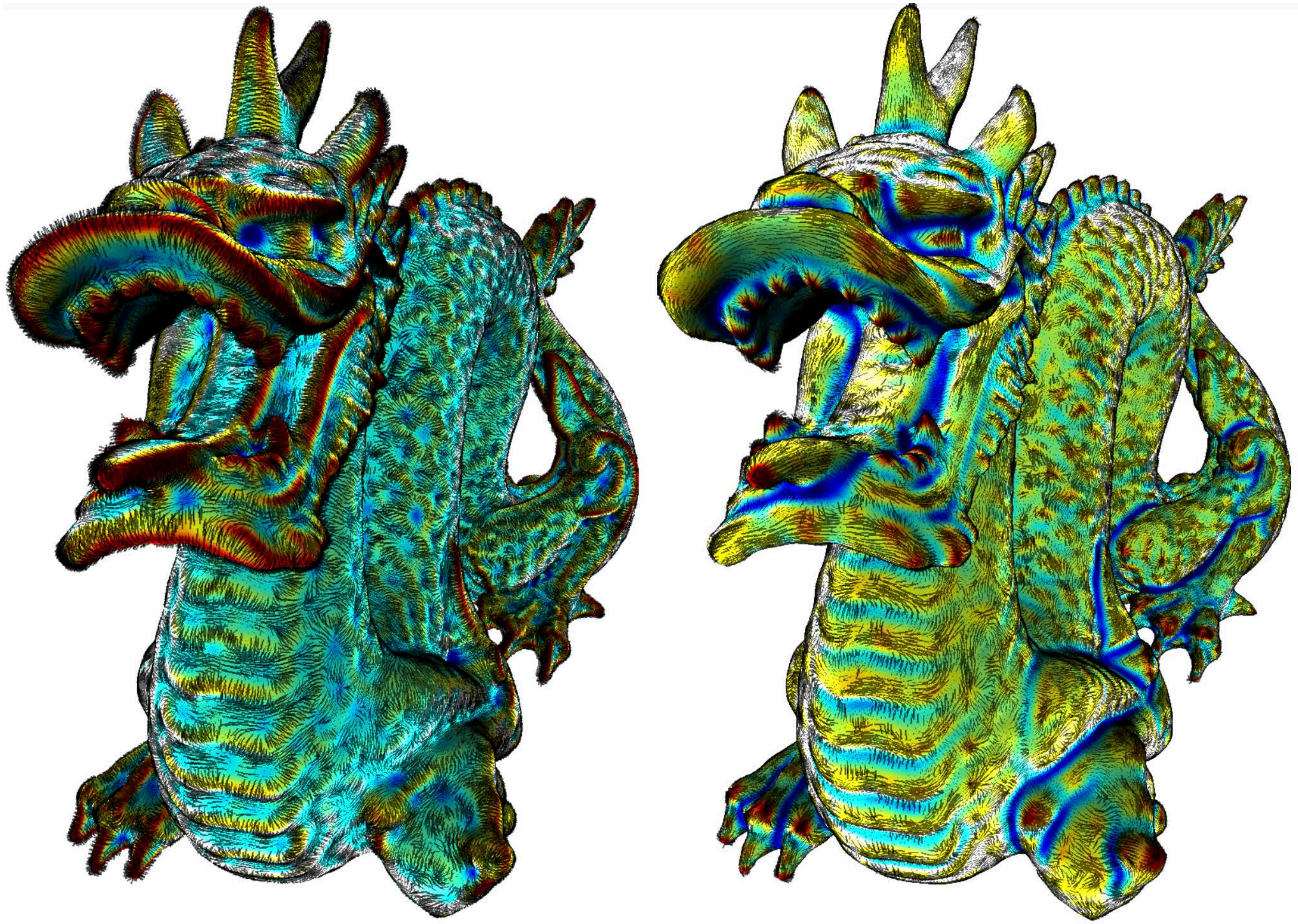
Gaussian curvature



Mean curvature

Schirmer et al. How to train your (neural) dragon. SIBGRAPI. 2023.

Curvatures



Max curvature Min curvature

Gaussian curvature Mean curvature

Future works:

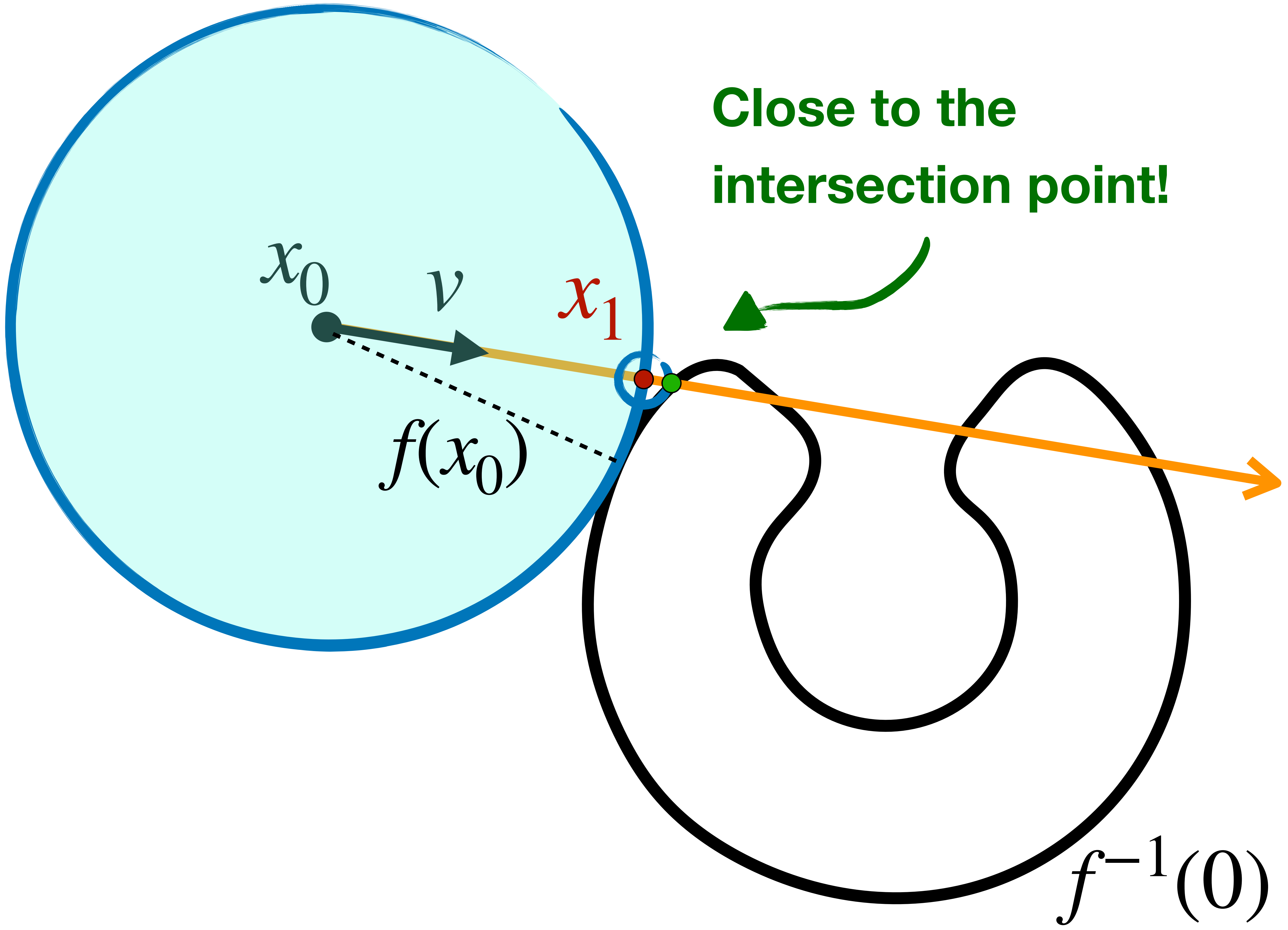
- Compute ridges (depends on the third derivative).
- Geodesics (also explore NN to model surface parametrizations).

Rendering

- **Problem**: Real-time rendering of (neural) level sets using sphere tracing.

★The intersection between $\gamma(t) = x_0 + tv$ and $f^{-1}(0)$ is approximated by iterating:

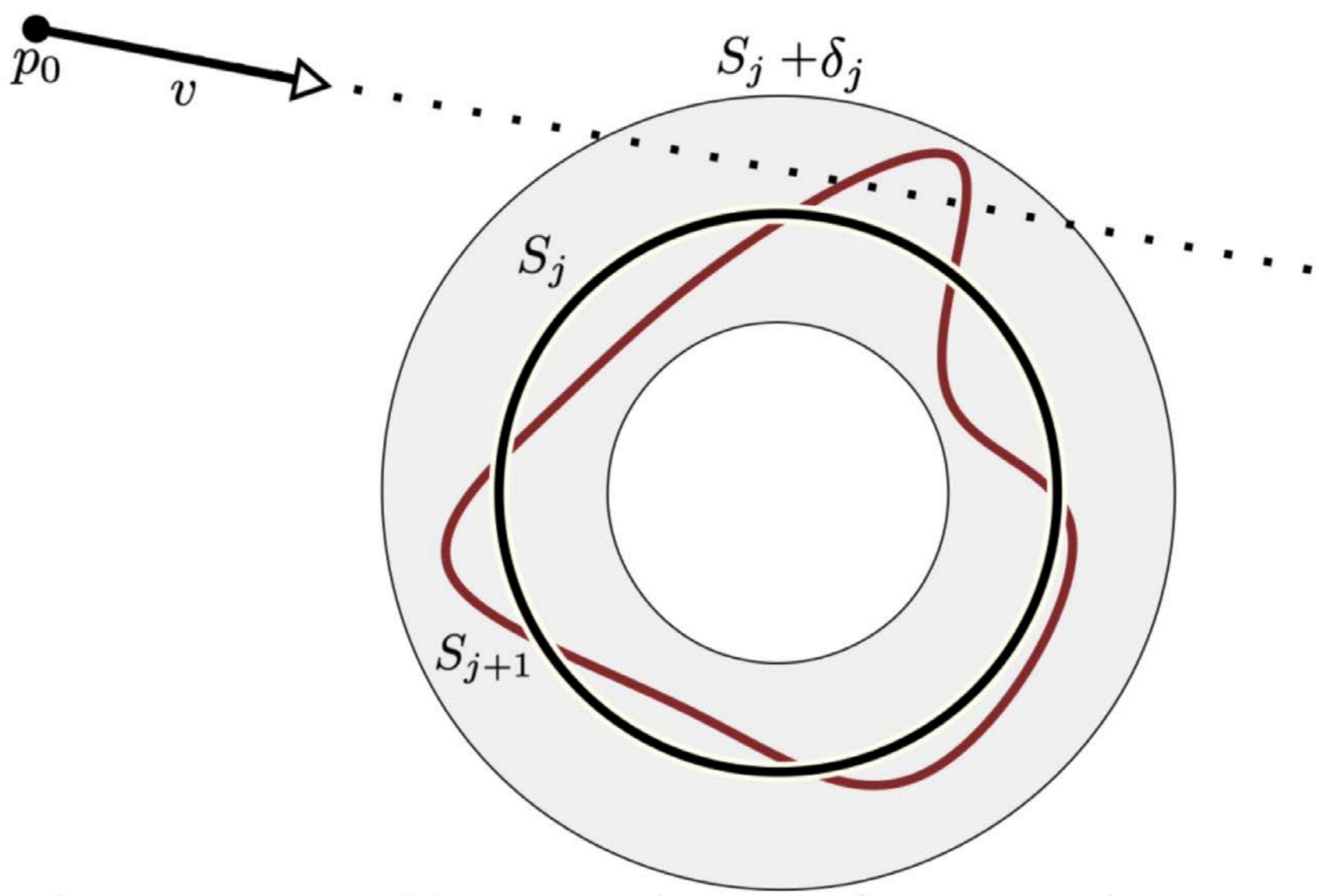
- $p_{i+1} = p_i + f(p_i)v$

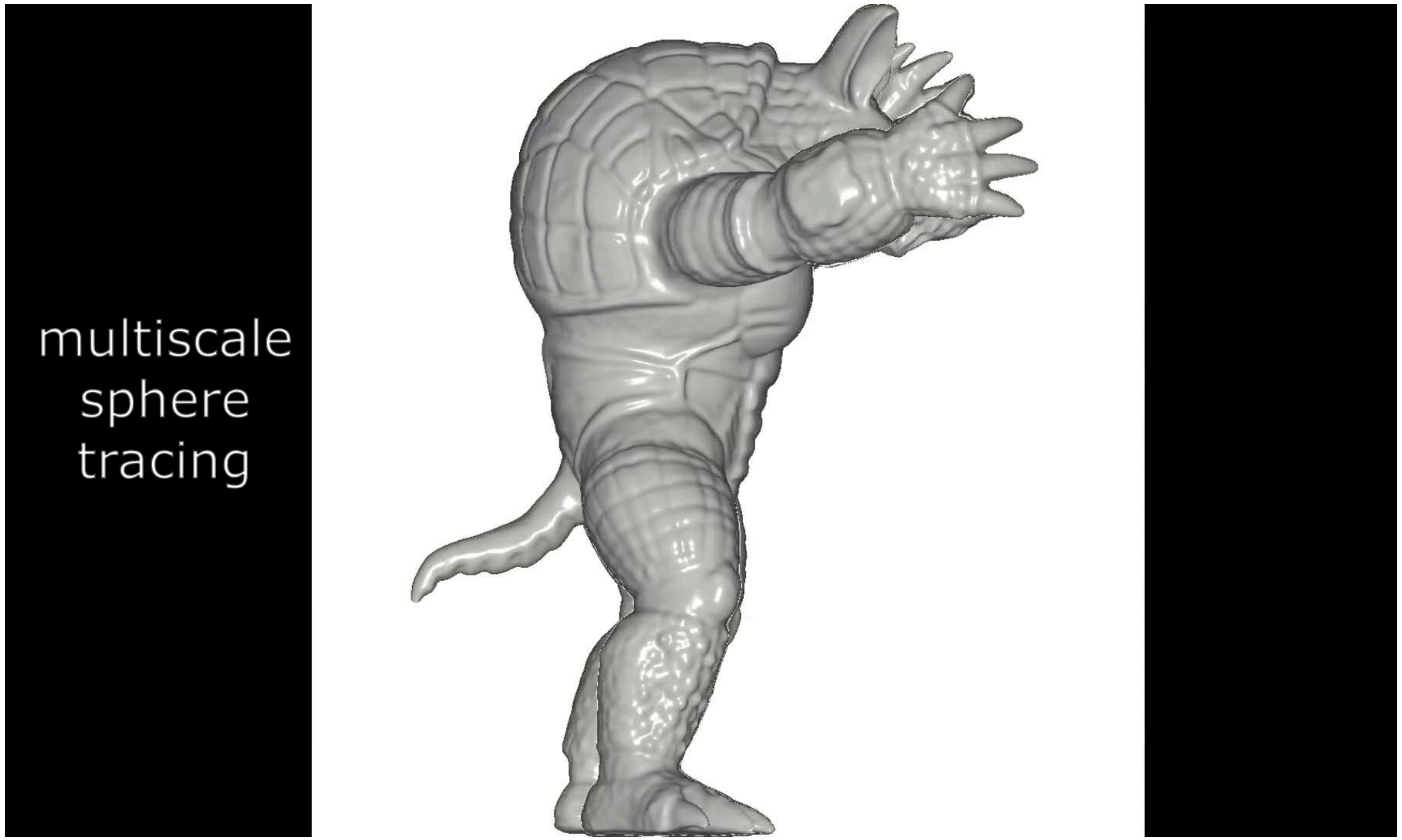


Silva et al. Neural Implicit Mapping via Nested Neighborhoods. arXiv. 2022.

Rendering

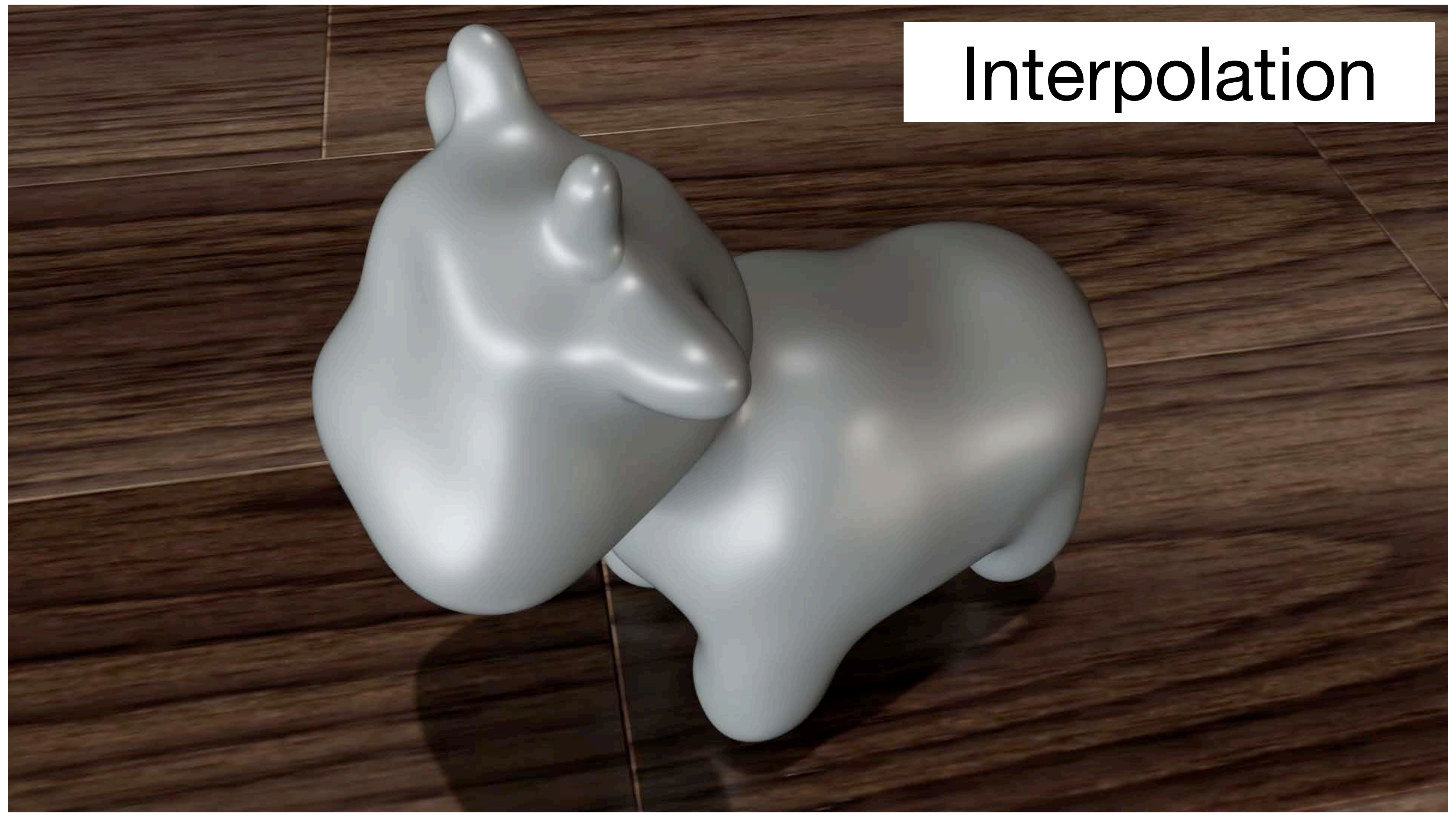
- **Problem**: Real-time rendering of (neural) level sets using sphere tracing.
- **Idea**: Use coarser networks in the early iterations of the algorithm.
- We need the existence of a nested sequence of zero-level sets neighborhood.

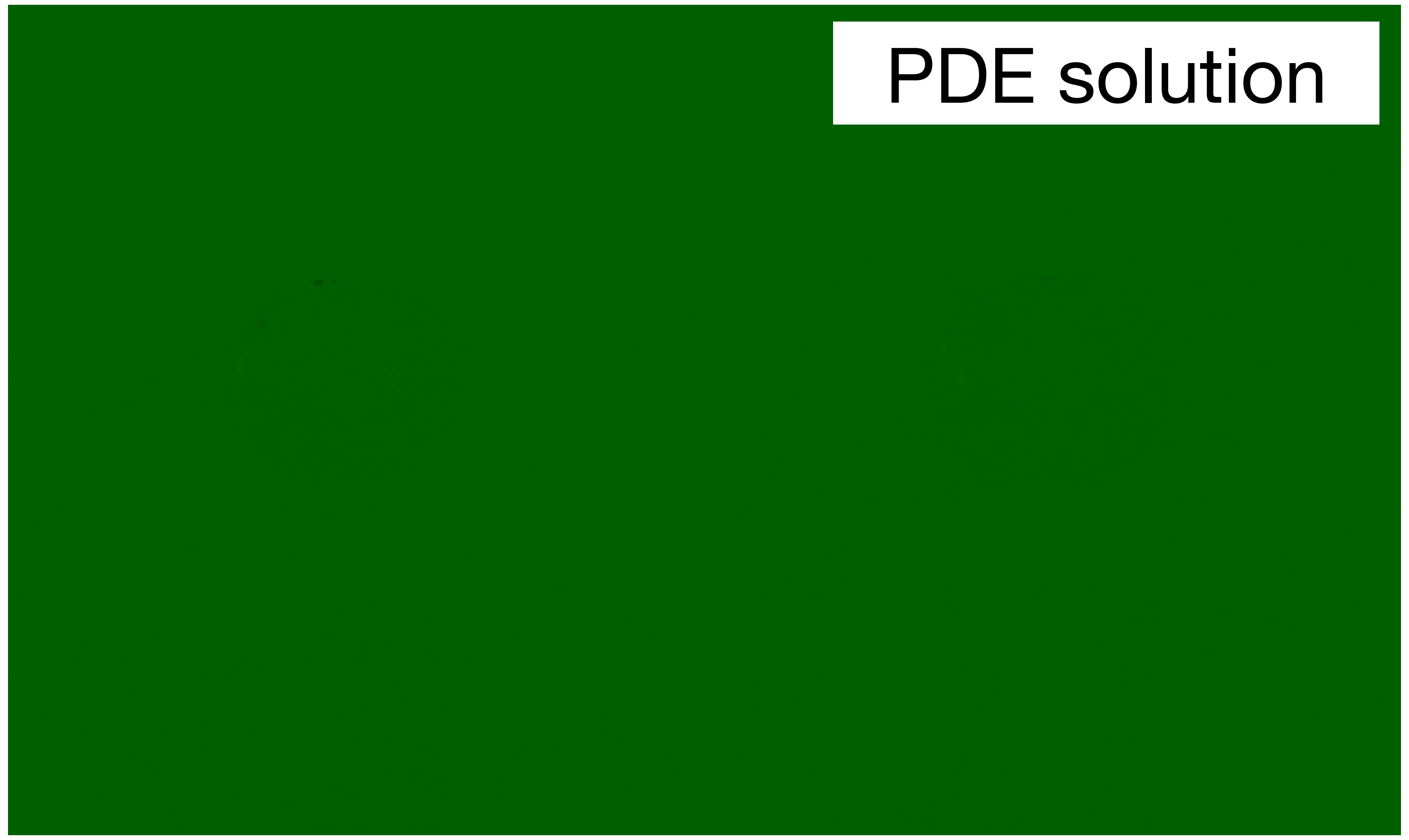


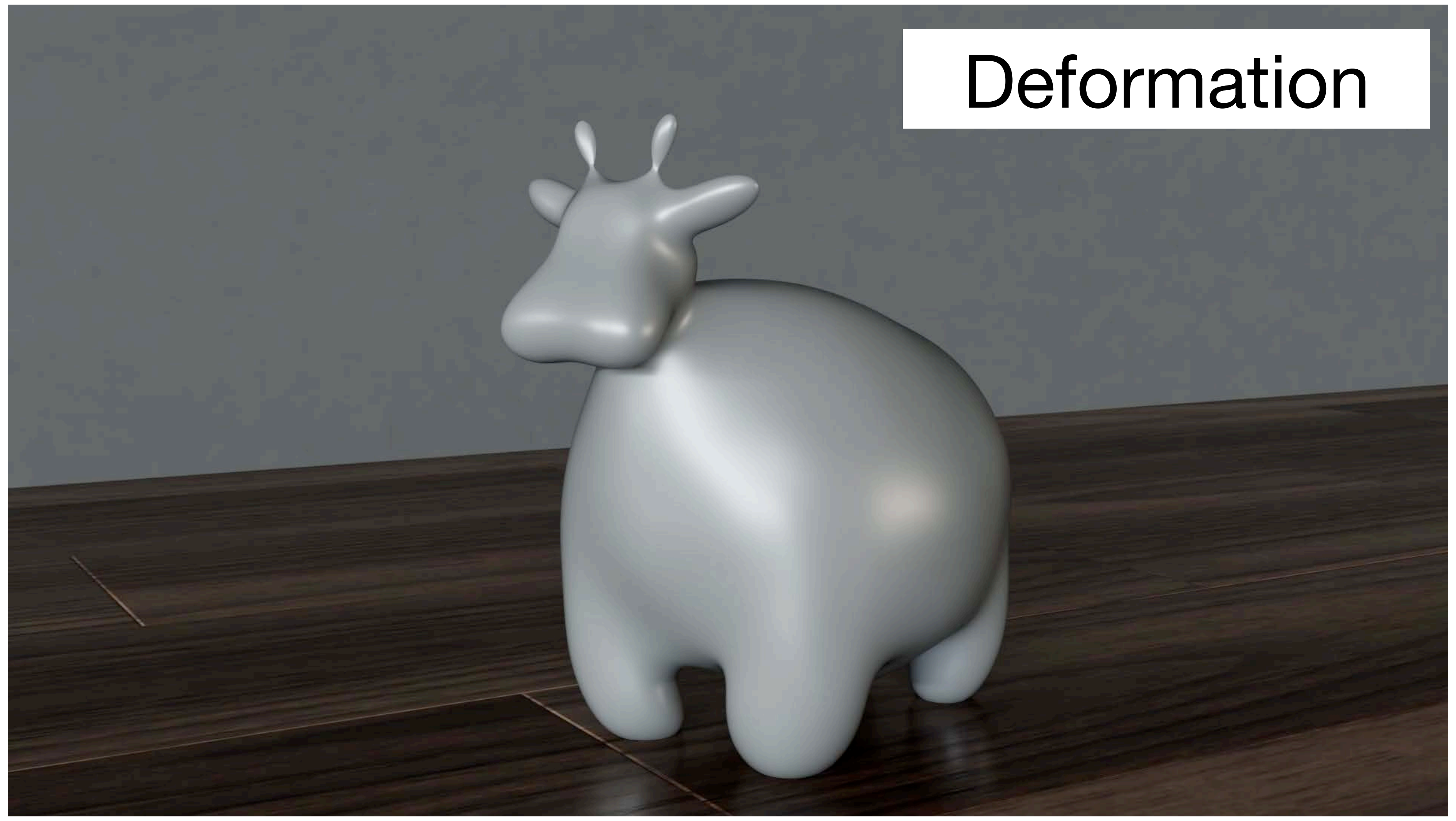


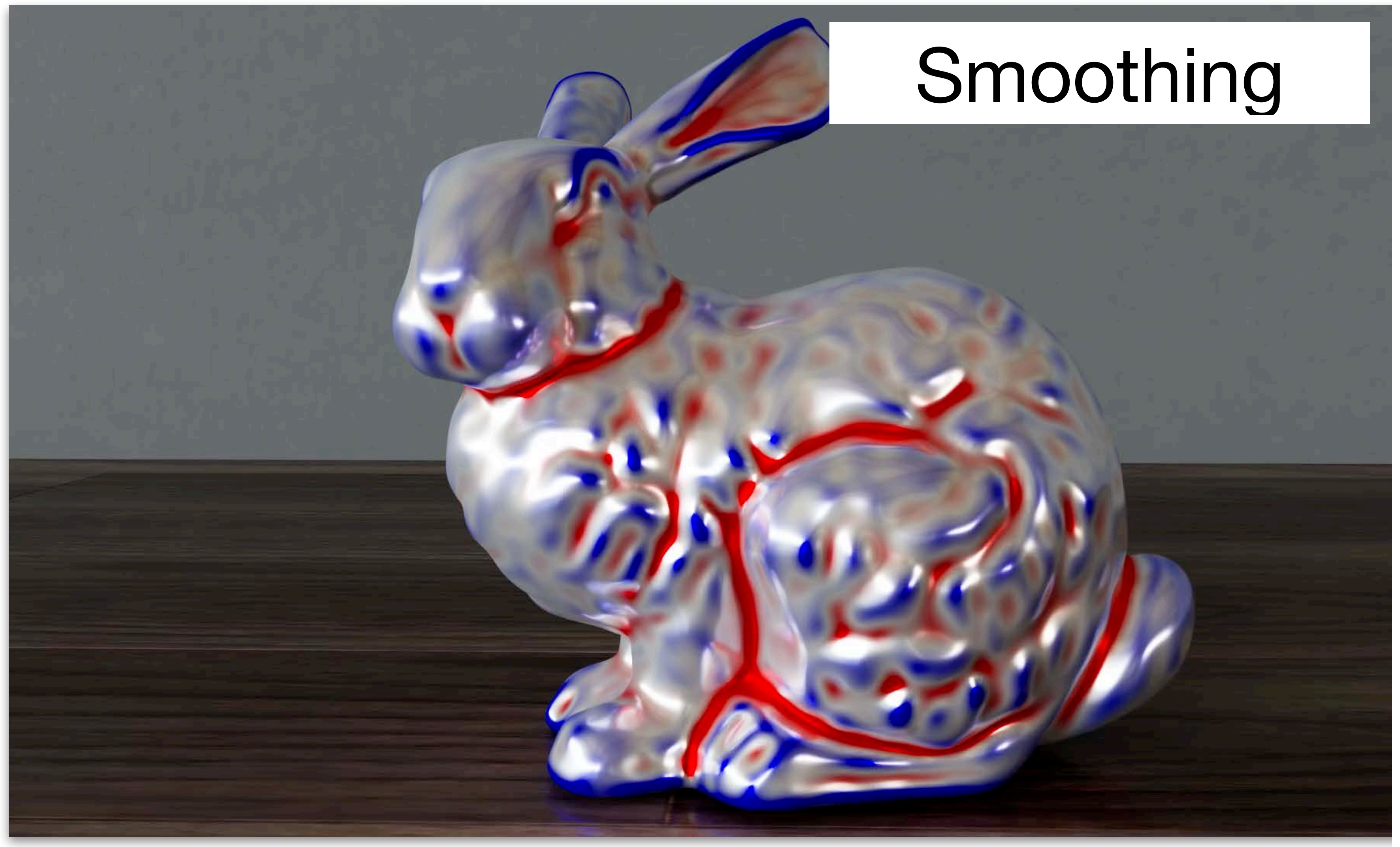
Silva et al. Neural Implicit Mapping via Nested Neighborhoods. arXiv. 2022.

Evolving the level sets of INRs









Evolving the level sets of INRs

Input data

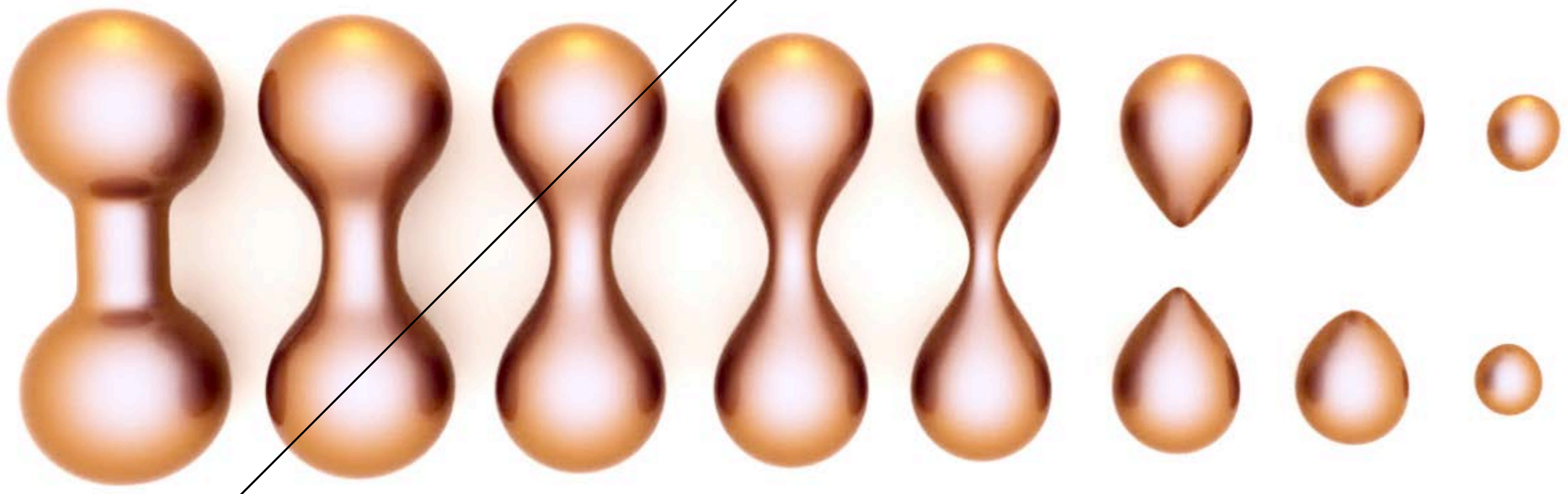
Trained INR $g : \mathbb{R}^3 \to \mathbb{R}$.

Sampling

Training

Find a minimum of $\mathscr{L}$ using *gradient descent.*

Rendering



$t = 0$

- $f_t^{-1}(0)$ is **smooth** both in space and time.

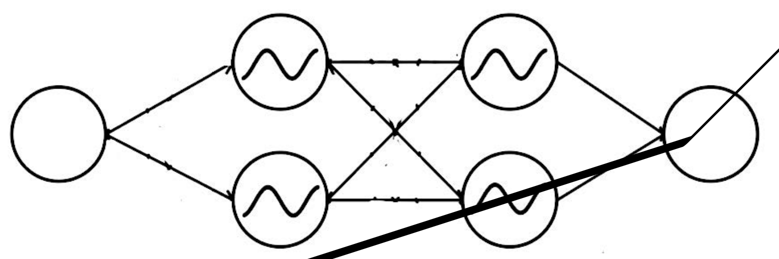
Loss function

?

Regularization

?

Smooth neural network



$f : \mathbb{R}^3 \times \mathbb{R} \to \mathbb{R}$

Novello et al. Neural Implicit Surface Evolution. ICCV. 2023.

Evolving the level sets of INRs

Input data

Trained INR $g : \mathbb{R}^3 \to \mathbb{R}$.

Sampling

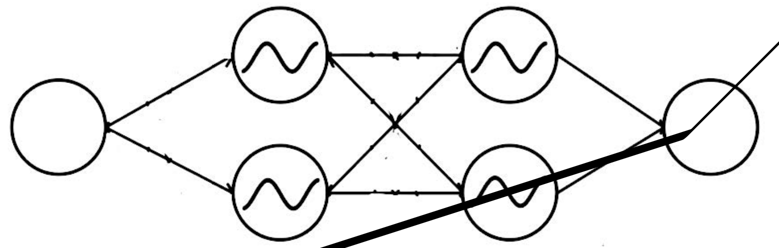
Training

Find a minimum of $\mathscr{L}$ using *gradient descent.*

Loss function

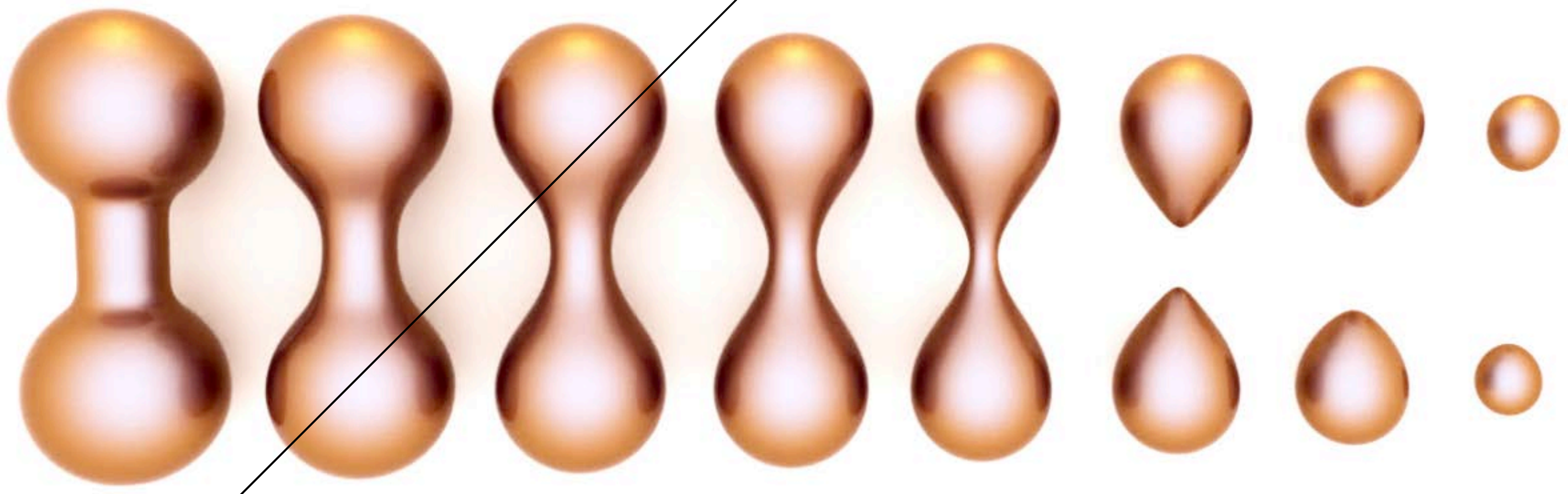
$$\mathscr{L}(f) = \underbrace{\int_{\Omega\times\{0\}} (f-g)^2 dx}_{\text{data term}} + \underbrace{\int_{\Omega\times(a,b)} \mathscr{F}^2 dxdt}_{\text{PDE term}}$$

Smooth neural network



$f : \mathbb{R}^3 \times \mathbb{R} \to \mathbb{R}$

Rendering



$t = 0$

- $f_t^{-1}(0)$ is **smooth** both in space and time.

Regularization

Differential equation

$$\begin{cases} \mathscr{F} = 0 \text{ in } \Omega \times (a,b), \\ f = g \quad \text{on } \Omega \times \{0\}. \end{cases}$$

Novello et al. Neural Implicit Surface Evolution. ICCV. 2023.

Evolving the level sets of INRs

Input data

Trained INR $g : \mathbb{R}^3 \to \mathbb{R}$.

Sampling →

Training

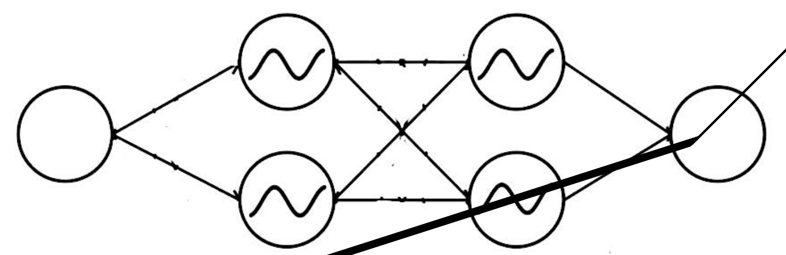
Find a minimum of $\mathscr{L}$ using *gradient descent*.

↑

Loss function

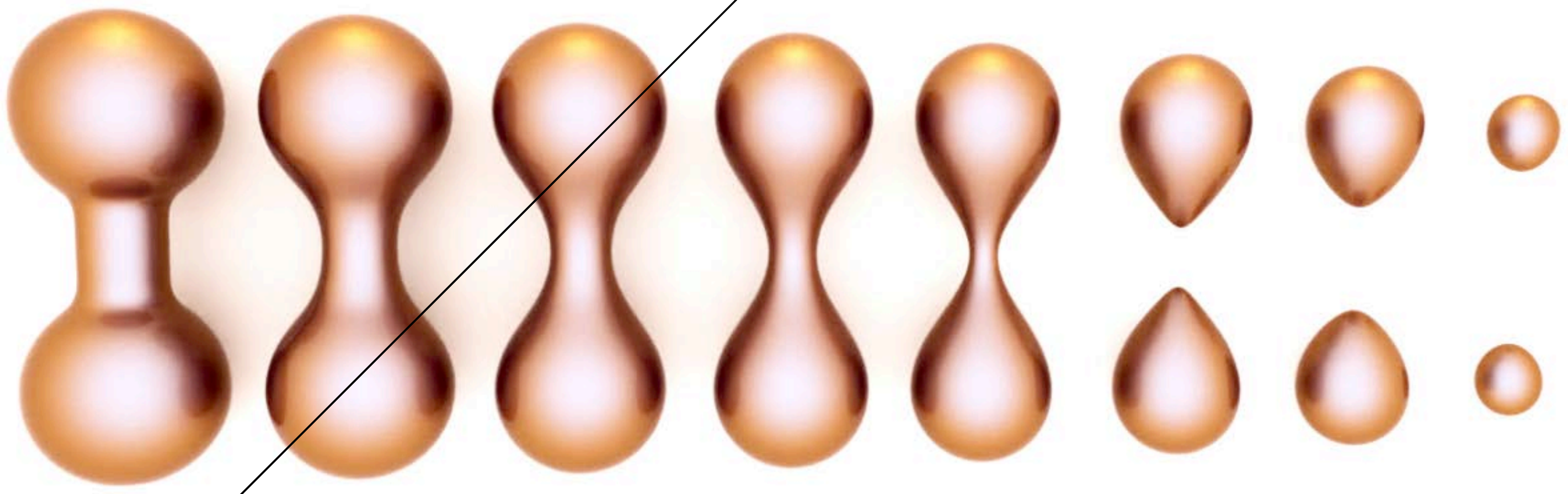
$$\mathscr{L}(f) = \underbrace{\int_{\Omega\times\{0\}} (f-g)^2 dx}_{\text{data term}} + \underbrace{\int_{\Omega\times(a,b)} \mathscr{F}^2 dxdt}_{\text{PDE term}}$$

Smooth neural network



$f : \mathbb{R}^3 \times \mathbb{R} \to \mathbb{R}$

Rendering



$t = 0$

- $f_t^{-1}(0)$ is **smooth** both in space and time.

Regularization

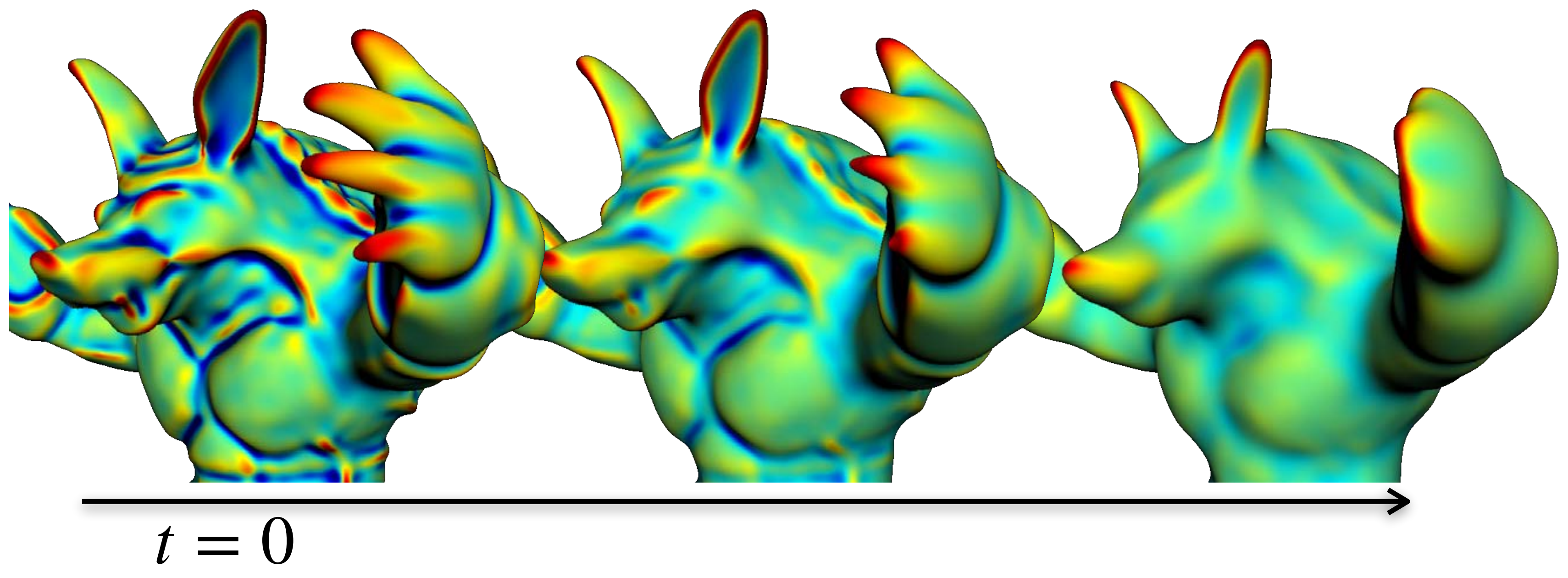
Level set equation

$$\begin{cases} \mathscr{F} = \frac{\partial f}{\partial t} + v|\nabla f| = 0 \text{ in } \Omega \times (a,b), \\ f = g \quad \text{ on } \Omega \times \{0\}. \end{cases}$$

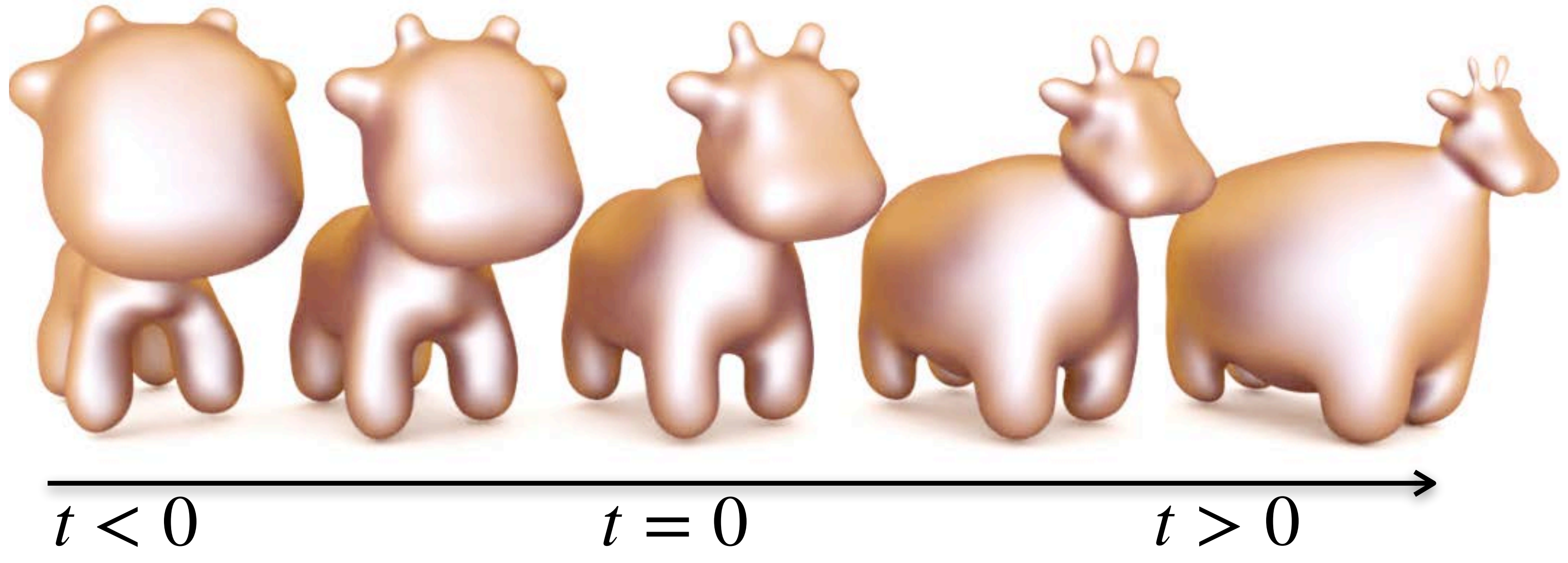
Novello et al. Neural Implicit Surface Evolution. ICCV. 2023.

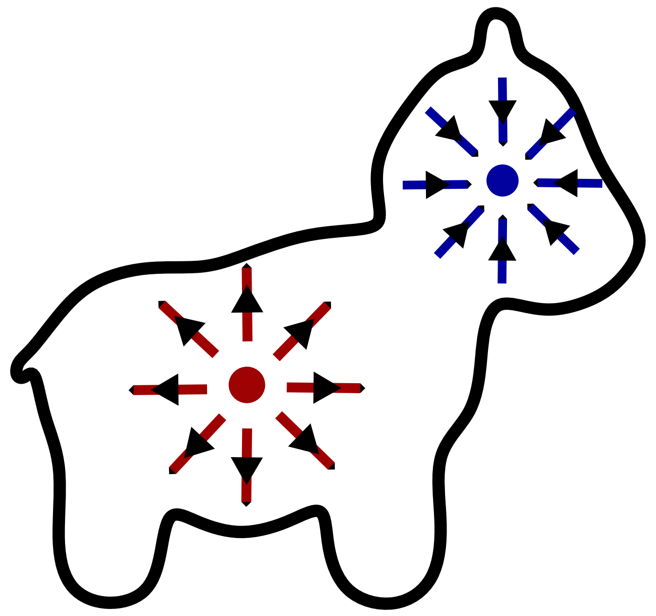
Evolving the level sets of neural networks

Mean curvature equation



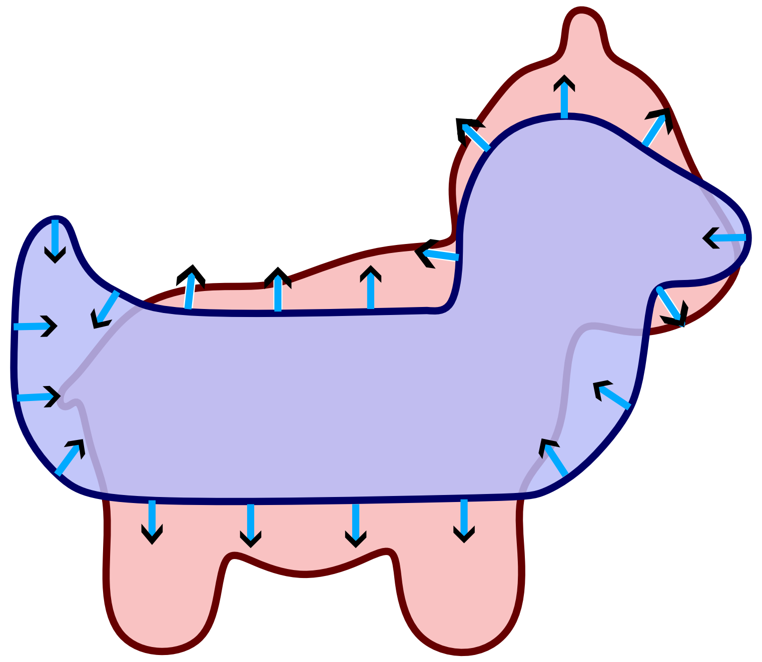
Deformation driven by vector





We blend the **sink** and **source** VFs using gaussians.

Interpolation between implicit surfaces



Future works:

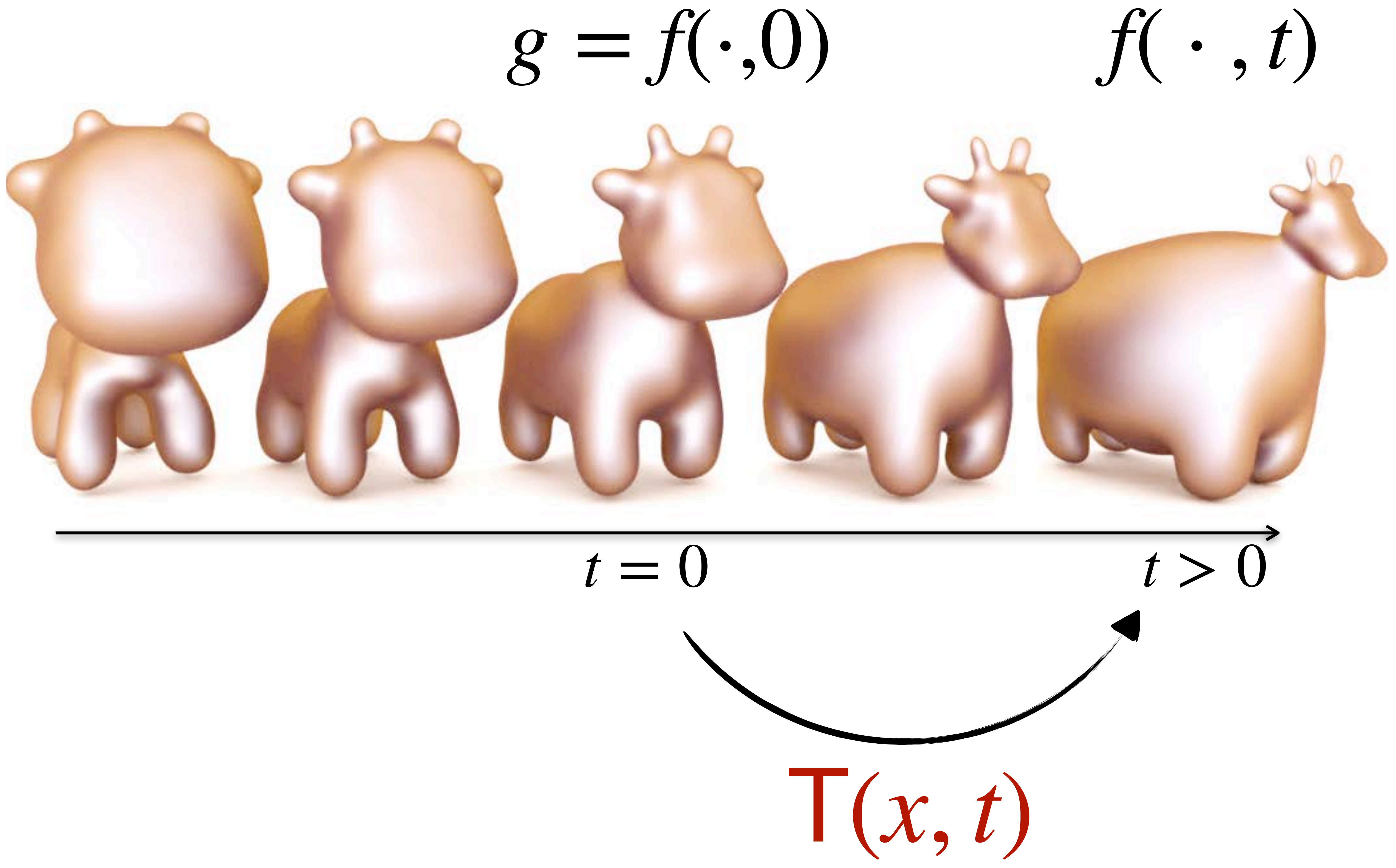
- Relate it with MR neural networks to use to accelerate rendering.

Future works:

- Disentangle the surfaces from the deformation $T : \mathbb{R}^3 \times \mathbb{R} \to \mathbb{R}^3$.

Disentangle deformation from the object

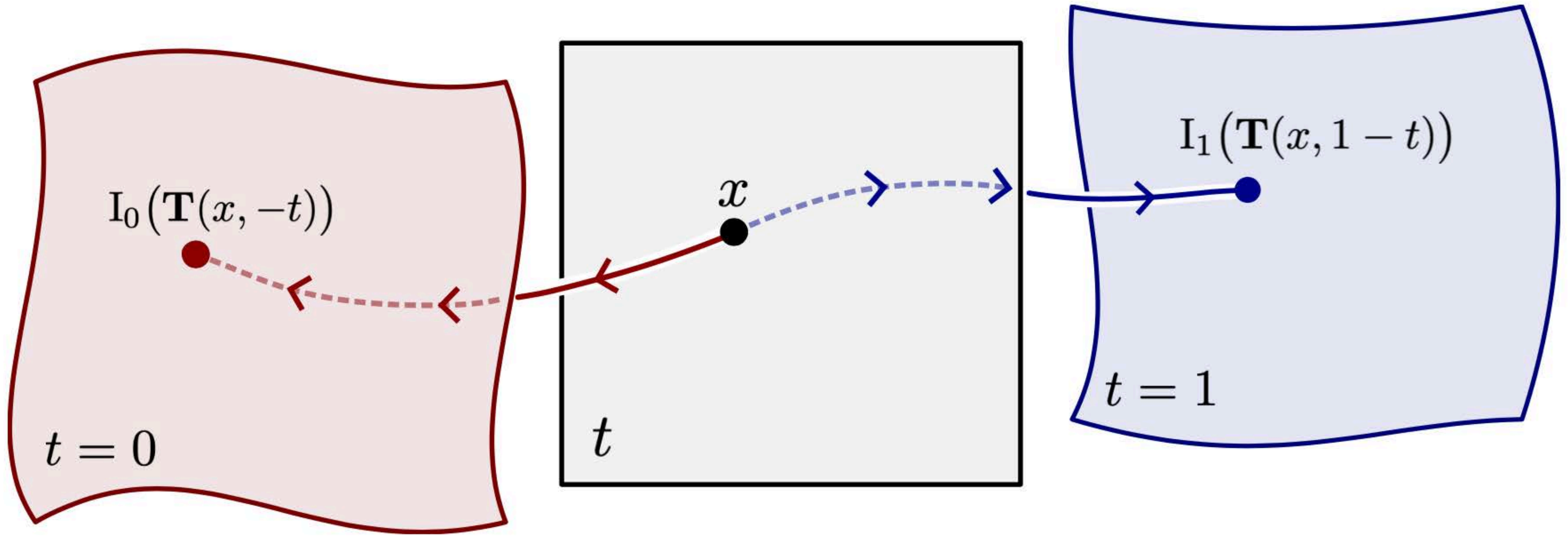
- **Problem**: the INR $f : \mathbb{R}^n \times \mathbb{R} \to \mathbb{R}$ has to learn a deformation of $g = f(\cdot,0)$ at each time t.
- **Proposal**: represent such deformations by another INR $\mathsf{T} : \mathbb{R}^n \times \mathbb{R} \to \mathbb{R}^n$.
  - $f(x,t) = g \circ \mathsf{T}^{-1}(x,t)$
  - Consider T to be a flow:
    - $\mathsf{T}(x,0) = x$, and $\mathsf{T}\big(\mathsf{T}(x,s),t\big) = \mathsf{T}(x,t+s)$.
      - $\mathsf{T}^{-1}(x,t) = \mathsf{T}(x,-t)$



Morphing of objects

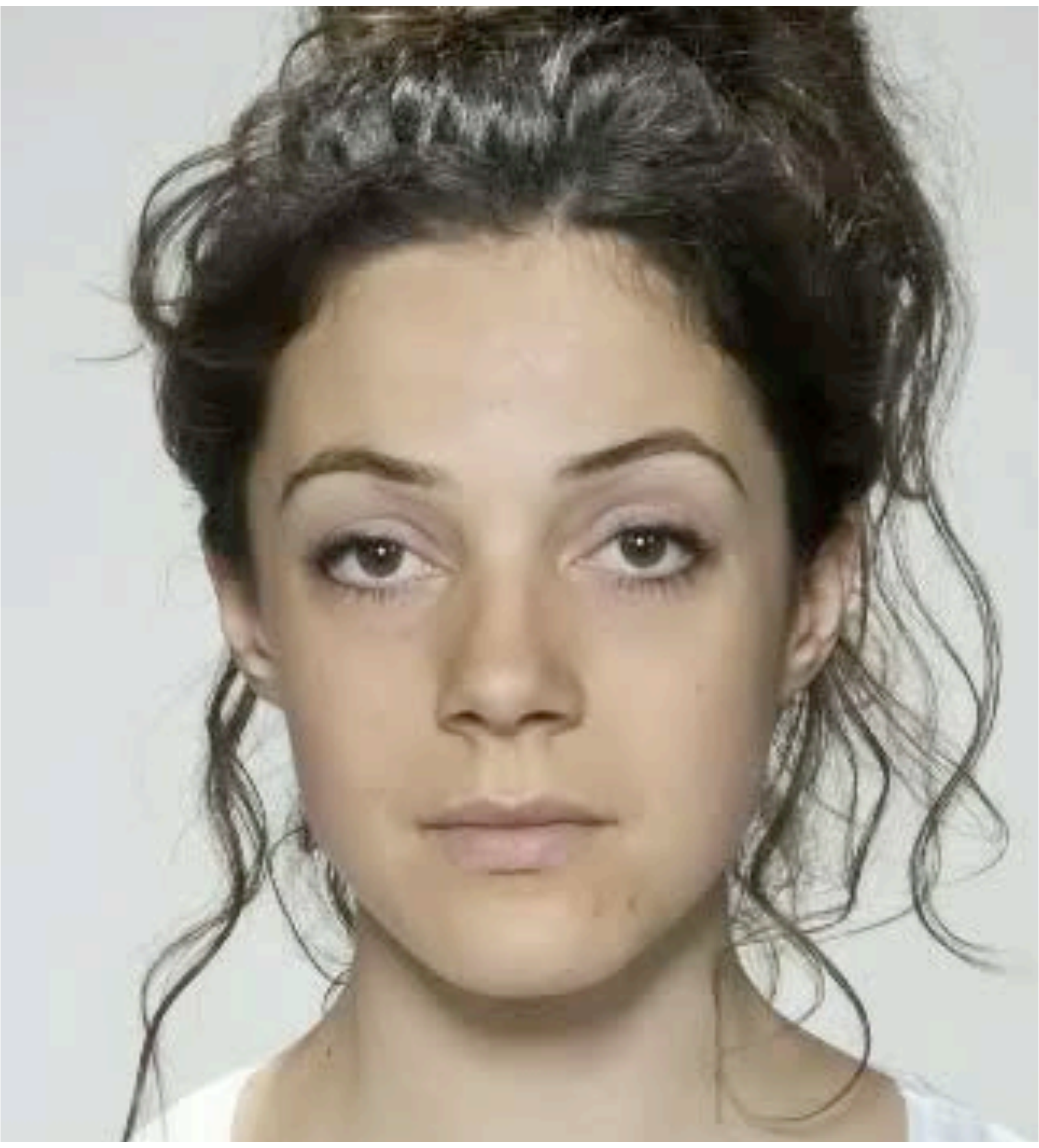
- **Problem**: morphing between two images $I_i : \mathbb{R}^2 \to \mathscr{C}$.
- Train a flow $T : \mathbb{R}^2 \times \mathbb{R} \to \mathbb{R}^2$ to align the features.

$\mathbf{I}_0 : \mathbb{R}^2 \to \mathscr{C}$



$\mathbf{I}_1 : \mathbb{R}^2 \to \mathscr{C}$

$I_0(T(x, -t))$ $I_1(T(x, 1-t))$



Morphing $f : \mathbb{R}^2 \times [0,1] \to \mathscr{C}$

Schardong et al. Neural implicit morphing of face images. CVPR. 2024.

Feature alignment along time

Schardong et al. Neural implicit morphing of face images. CVPR. 2024.

Blending using diffusion models

[Ours]

[diffAE]

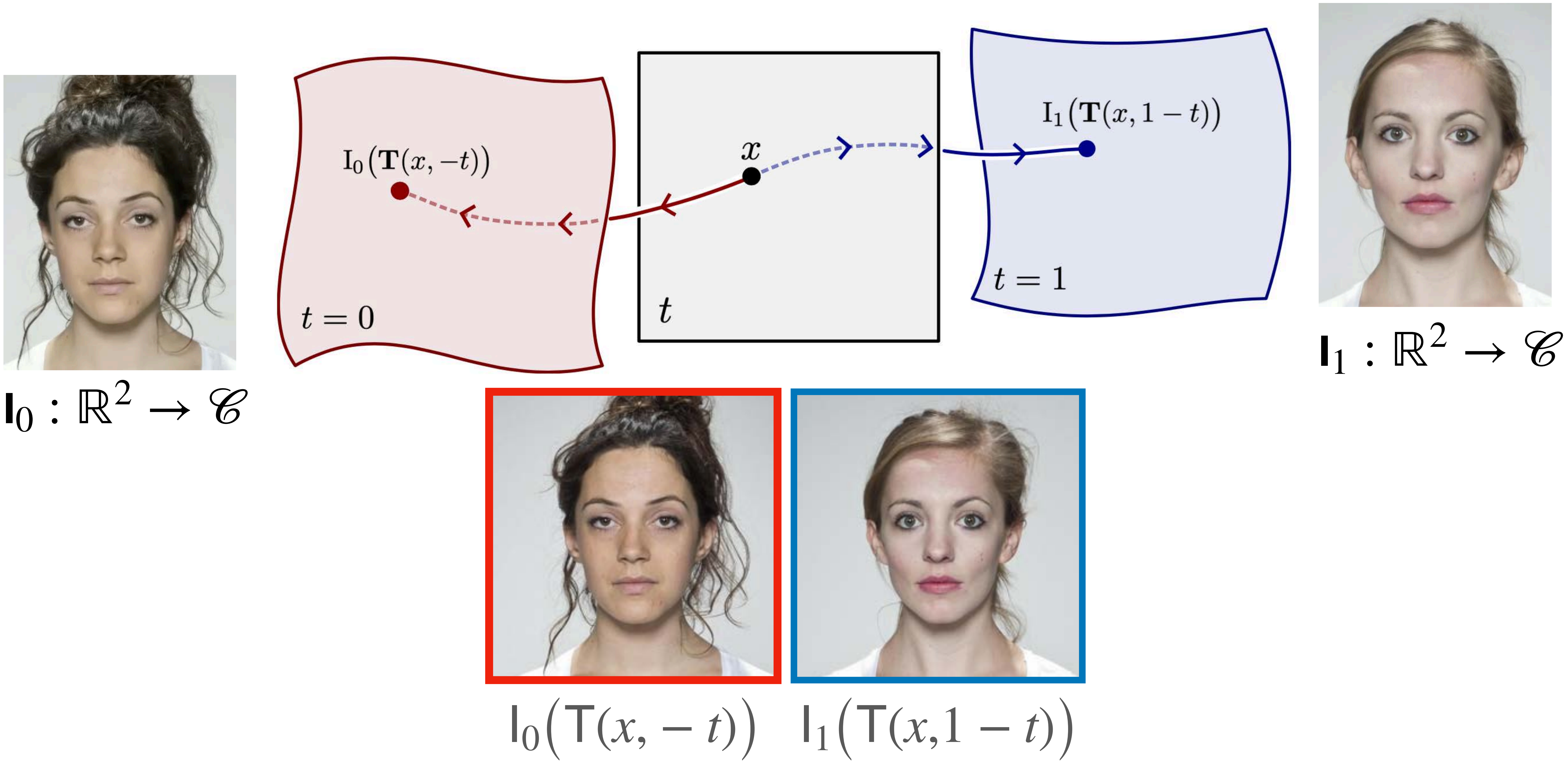
Schardong et al. Neural implicit morphing of face images. CVPR. 2024.

Faces of different ethnicities and genders

Schardong et al. Neural implicit morphing of face images. CVPR. 2024.

Morphing of objects

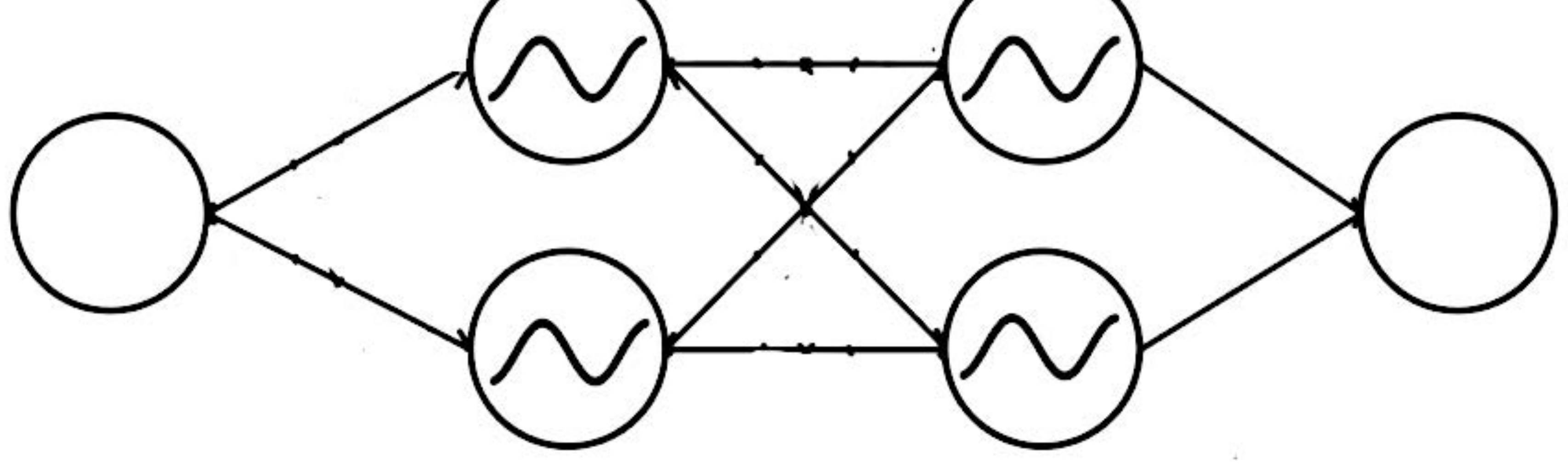
- **Problem**: morphing between two images $\mathrm{I}_i : \mathbb{R}^2 \to \mathscr{C}$.
- Train a flow $\mathrm{T} : \mathbb{R}^2 \times \mathbb{R} \to \mathbb{R}^2$ to align the features.



Future works:

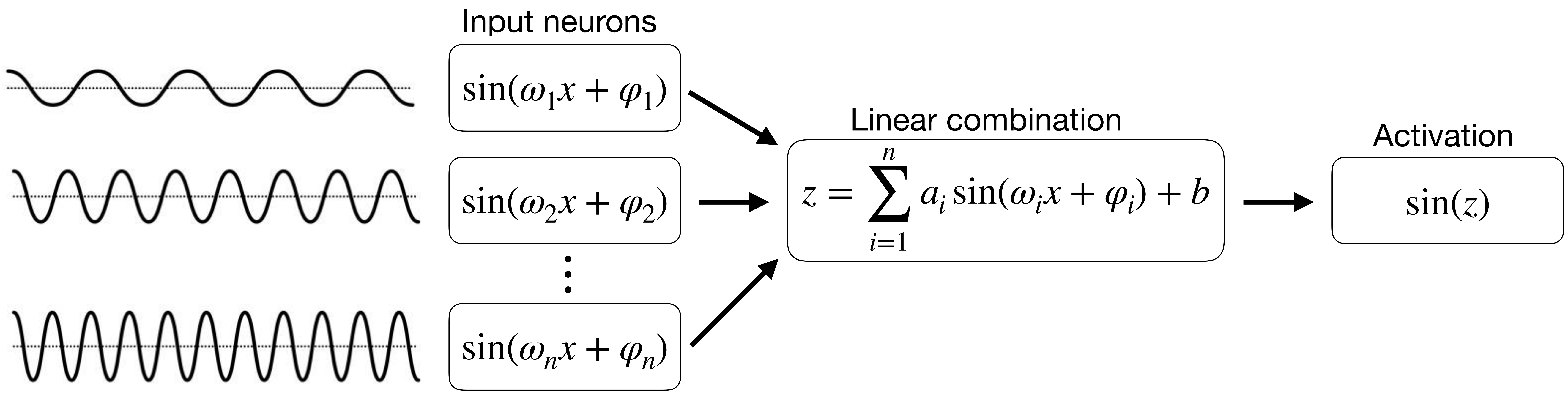
- Morphing between surfaces.

Sinusoidal INRs



Sinusoidal neuron

- $$h(x) = \sin\left(\sum_{i=1}^{n} a_i \sin(\omega_i x + \varphi_i) + b\right)$$



Multiresolution Neural Networks for Imaging

Input

64×64 **Level 3** | 32×32 **Level 2** | 16×16 **Level 1**

Data in multiresolution

Training

Fit each **level** m to $g_1 + \cdots + g_m$

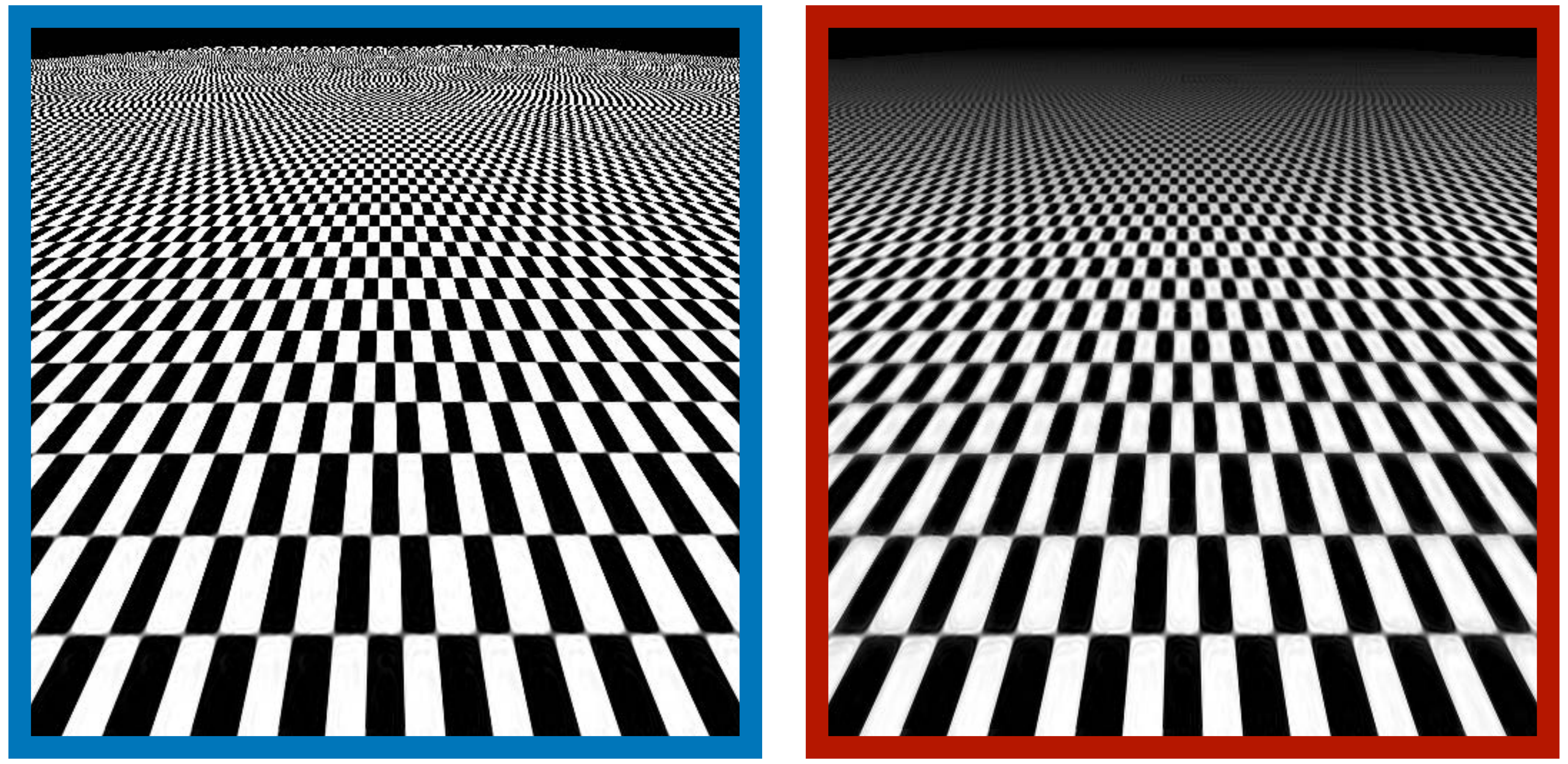
$$f = g_1 + g_2 + g_3$$

Sum of sinusoidal nets

$$g_i : \mathbb{R}^2 \to \mathbb{R}$$

Output

$g_1 + g_2 + g_3$ | $g_1 + g_2$ | g_1



Anti-aliasing

Paz et al. Multiresolution neural networks for imaging. SIBGRAPI. 2022.

Paz et al. MR-Net: Multiresolution sinusoidal neural networks. Computers & Graphics. 2023.

Multiresolution Neural Networks for Imaging

Input

64×64 | 32×32 | 16×16

Level 3 **Level 2** **Level 1**

Data in multiresolution

Training

Fit each **level** m to $g_1 + \cdots + g_m$

$f = g_1 + g_2 + g_3$

Sum of sinusoidal nets
$g_i : \mathbb{R}^2 \to \mathbb{R}$

Output

$g_1 + g_2 + g_3$ | $g_1 + g_2$ | g_1

Work in progress:

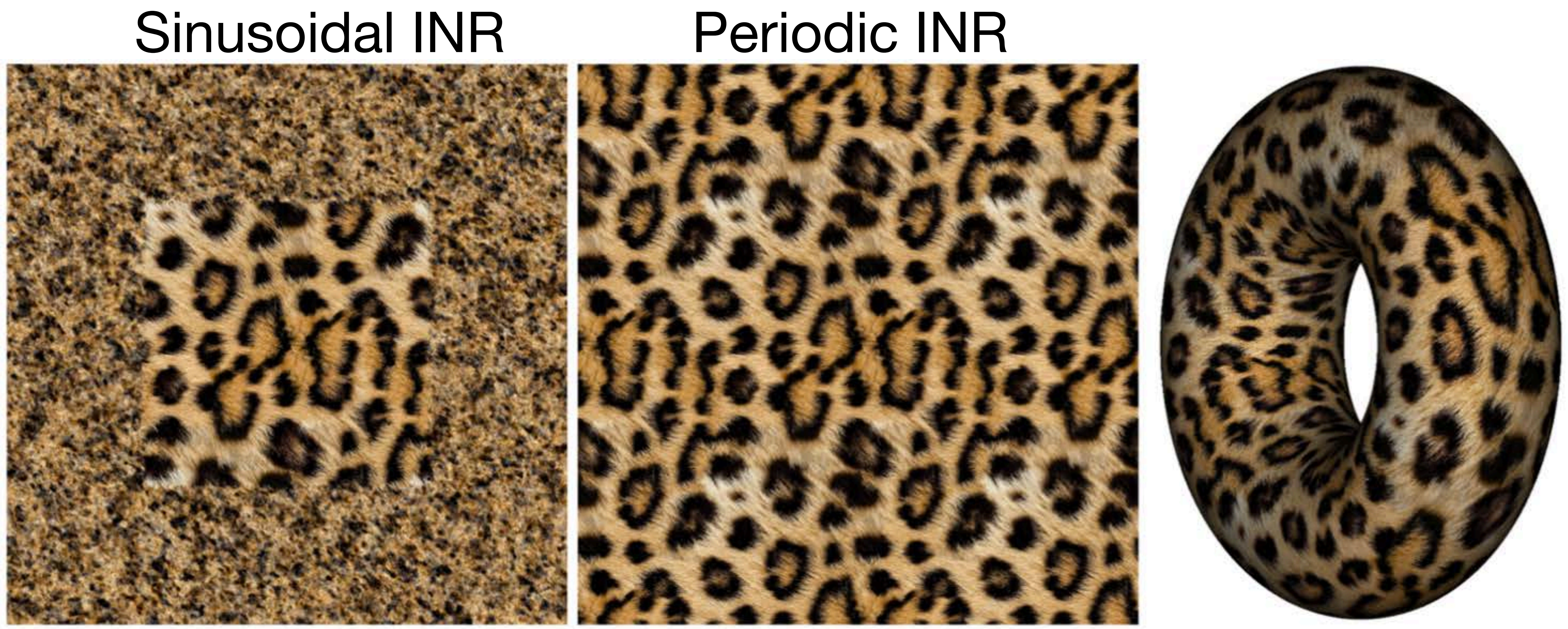
- Curves and surfaces in multiresolution (using the mean curvature equation).

Paz et al. Multiresolution neural networks for imaging. SIBGRAPI. 2022.
Paz et al. MR-Net: Multiresolution sinusoidal neural networks. Computers & Graphics. 2023.

Periodic networks (Work in progress)

- **Sinusoidal neuron** $h(x) = \sin\left(\sum_{i=1}^{n} a_i \sin(\omega_i x + \varphi_i) + b\right) = \sum_{\mathbf{k}\in\mathbb{Z}^n} \alpha_{\mathbf{k}}(a)\sin\Big(\langle \mathbf{k}, \omega x + \varphi\rangle + b\Big).$

- If the input neurons are periodic with period P, the neuron h is periodic with period P.
  - Texture representation.



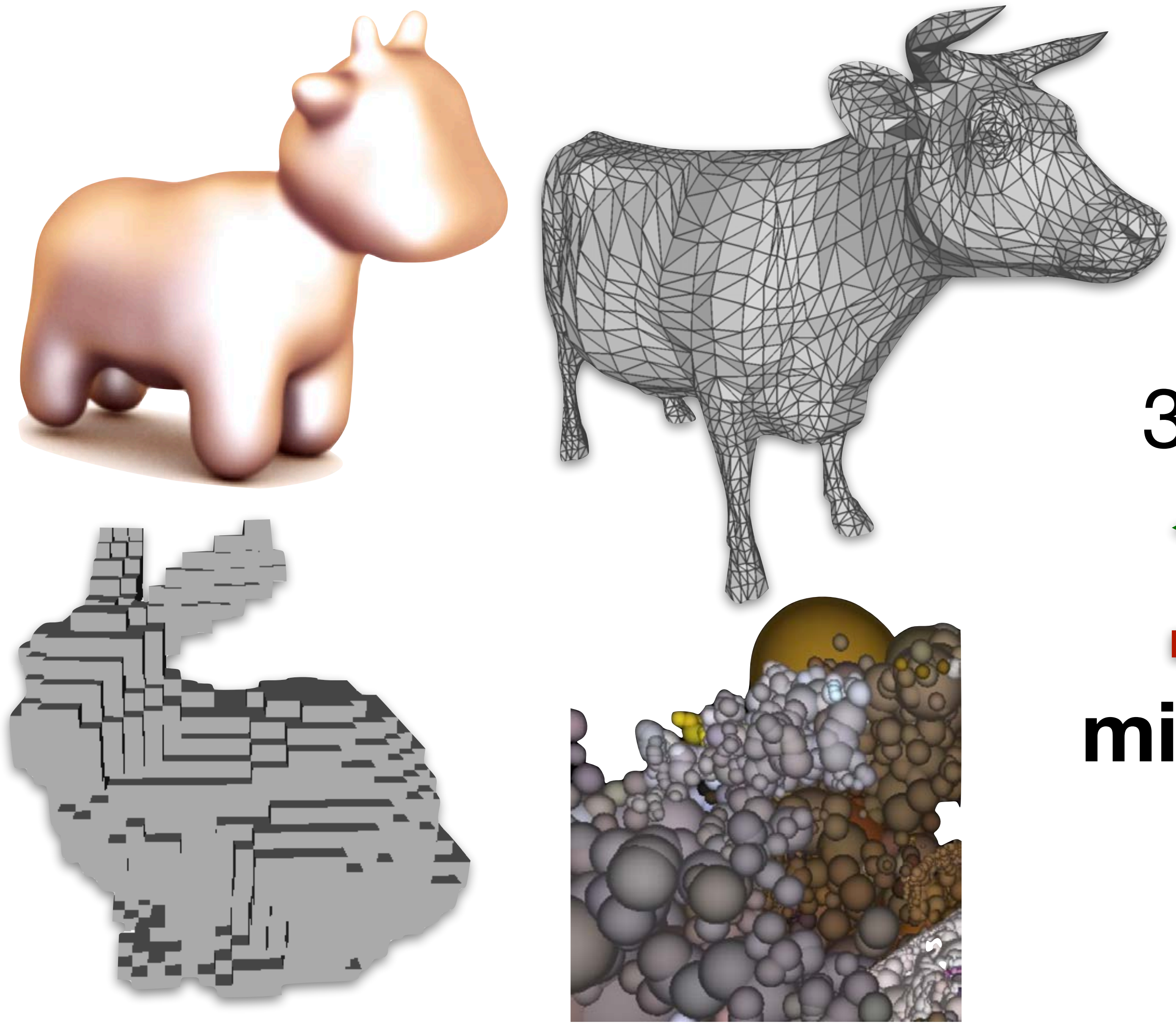
Future works:

- Represent panoramic images.
- Closed curves.
- Surfaces having the topology $\mathbb{S}^1 \times \mathbb{S}^1$.

Paz et al. Implicit Neural Representation of Tileable Material Textures. arXiv. 2024
Novello. Understanding Sinusoidal Neural Networks. arXiv, 2023.

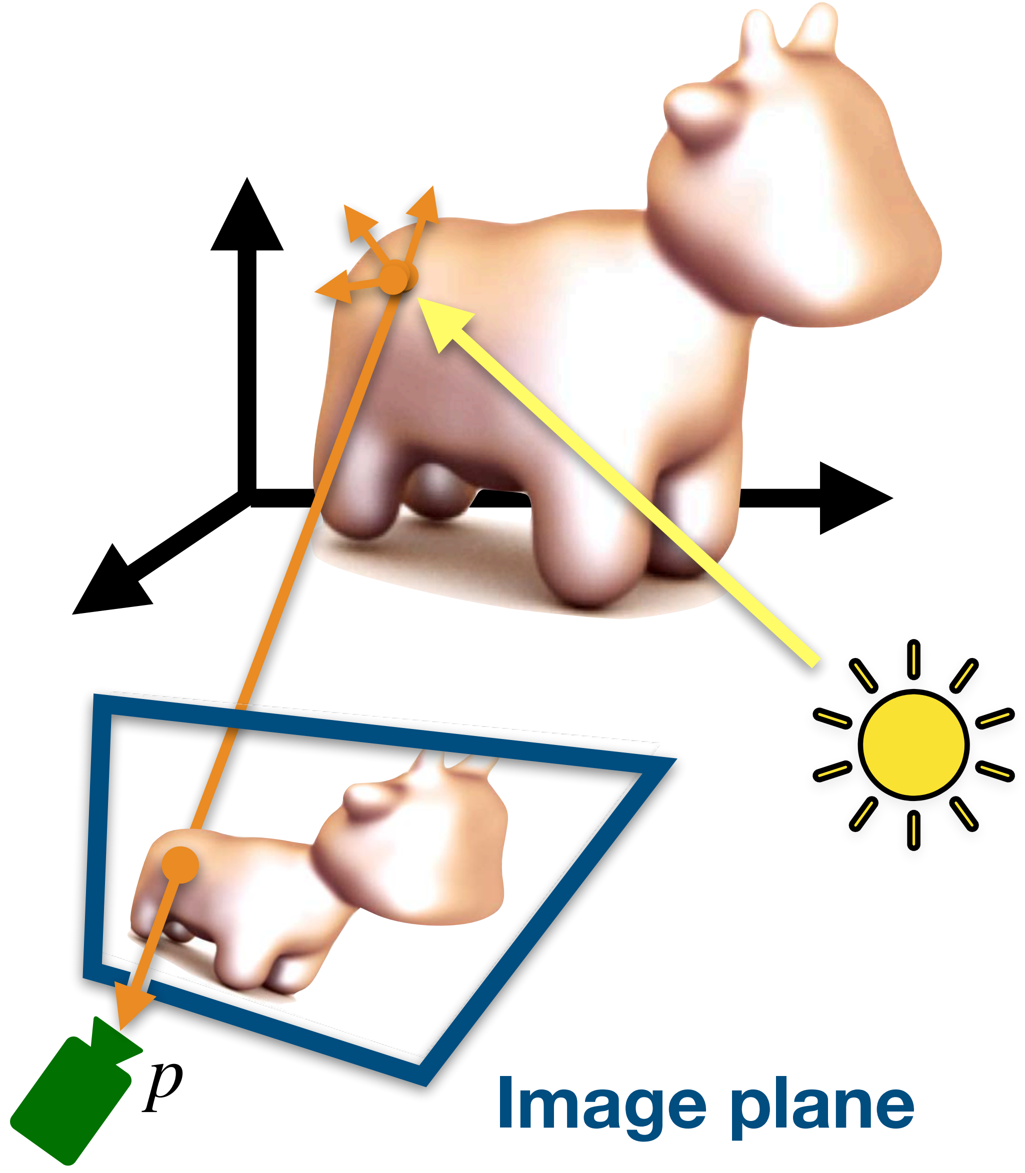
Back to the graphics pipeline

- **Diff scene representation.**
  - Mesh, volume, implicits…
  - Domain in $\mathbb{R}^3$
- **Diff rendering.**
  - Rasterization, ray tracing, volume ray casting.
- **Rendered images.**



3D rec.

mipplicits



3D rec.

mipplicits

$\mathbf{I} : \mathbb{R}^2 \to \mathscr{C}$

[i3D, i4D, taming**]**

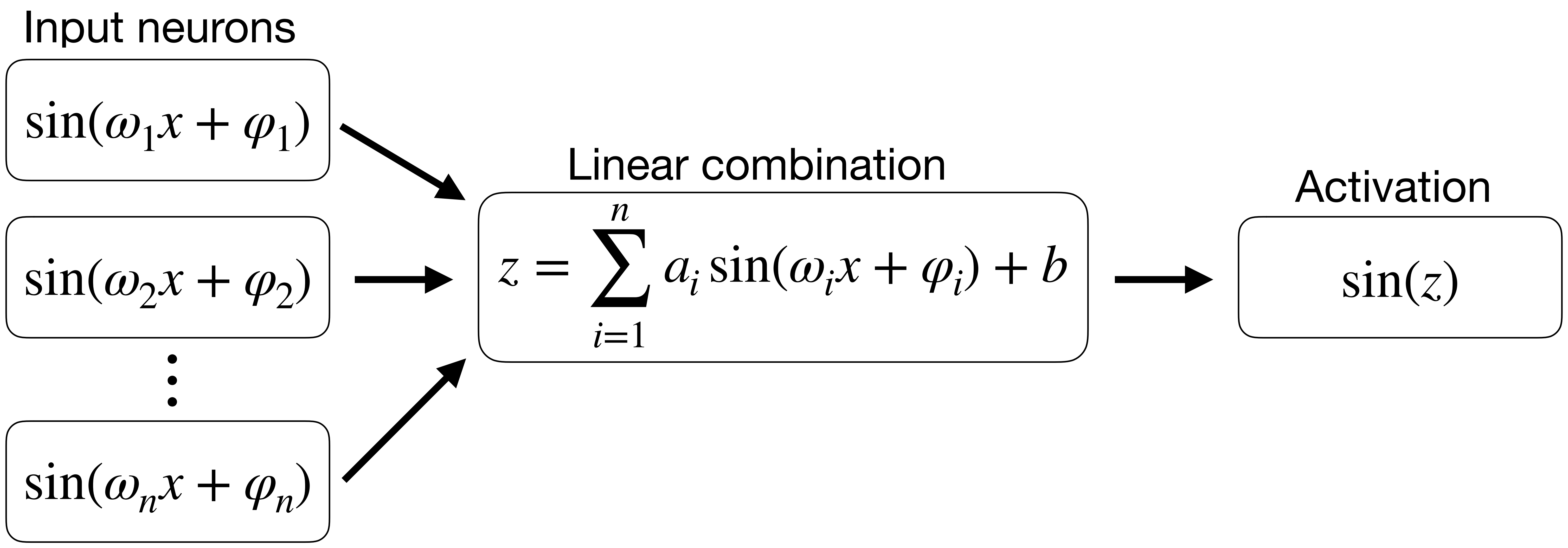
[MRnet, morph, texture, taming**]**

Obrigado!

Understanding Sinusoidal Neural Networks

A trigonometric identity…

- **Sinusoidal neuron** $h(x) = \sin\left(\sum_{i=1}^{n} a_i \sin(\omega_i x + \varphi_i) + b\right)$



- We can prove that $h(x) = \sum_{\mathbf{k}\in\mathbb{Z}^n} \alpha_{\mathbf{k}}(a)\sin\Big(\langle \mathbf{k}, \omega x + \varphi\rangle + b\Big)$ with $\alpha_{\mathbf{k}}(a) = \prod_{i=1}^{n} J_{k_i}(a_i)$
- $J_{k_i}(a_i) = \int_0^{\pi} \cos\big(k_i t - a_i \sin(t)\big)dt$ are the Bessel functions of the first

Novello. Understanding Sinusoidal Neural Networks. arXiv, 2023.

Some consequences…

- **Sinusoidal neuron** $h(x) = \sin\left(\sum_{i=1}^{n} a_i \sin(\omega_i x + \varphi_i) + b\right) = \sum_{\mathsf{k}\in\mathbb{Z}^n} \alpha_{\mathsf{k}}(a)\sin\Big(\langle \mathsf{k}, \omega x + \varphi\rangle + b\Big)$

- The sinusoidal neural is producing a large number of new frequencies $\langle \mathsf{k}, \omega\rangle$;

  - We prove that amplitudes $\alpha_{\mathsf{k}}(a)$ are bounded by $\prod_{i=1}^{n} \frac{\left(\frac{|a_i|}{2}\right)^{|k_i|}}{|k_i|!}$;

  - Controlling the frequency band of the network during training (Diana Aldana's Thesis)