T03: BVM2025 Hands-on Tutorial on Implicit Neural Representations (INRs) in Medical Imaging

Mattias, Fenja, Ziad and Christoph | University of Lübeck

Sunday 09 March 2025, 14:00 am - 17:20 pm (CET)

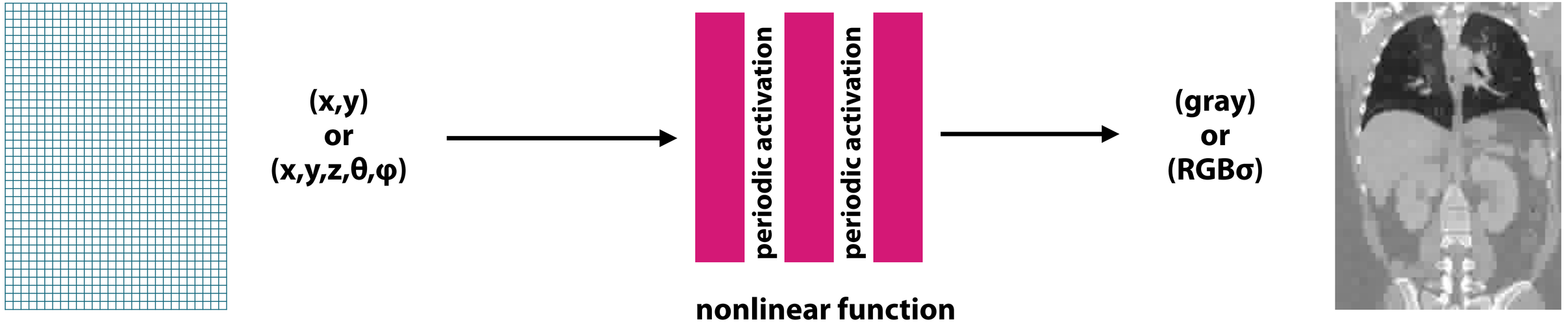
https://ziadhemidi.github.io/INR4BVM25.github.io/

Schedule for today

Time	**Topic**	Contents
14:00 - 14:10	**Welcome & Overview**	Introduction of speakers and organisational tutorial information
14:10 - 14:30	**Introduction to INR**	Basic conceptual and mathematical foundations on INRs and associated activation functions
14:30 - 16:30	**Hands-On (incl. individual breaks)**	Assisted programming of INR solutions for reconstruction, denoising and image registration in Jupyter notebooks
16:30 - 17:10	**Advanced Topic Mini-Lectures (4x10min)**	10 Minute Mini-Lecutures on challenging MRI reconstruction, rendering INRs (NERF), implicit neural image registration and multi-instance INR generalization
17:10 - 17:30	**Evaluation**	Discussion / Q&A

Key idea: encode image as nonlinear function (MLP)

often also called coordinate MLP



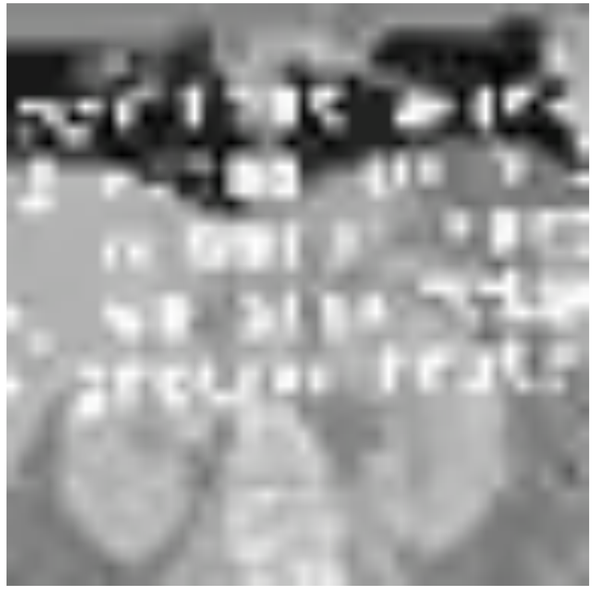
- **input** to neural field is a set of **x,y coordinates** (+ potential Fourier encoding)
- **output are gray/colour values** or additionally depth/alpha-transparency
- **multi-layer perceptron** is used to **learn functional mapping** from xy coordinates to gray-value

"training" on single input

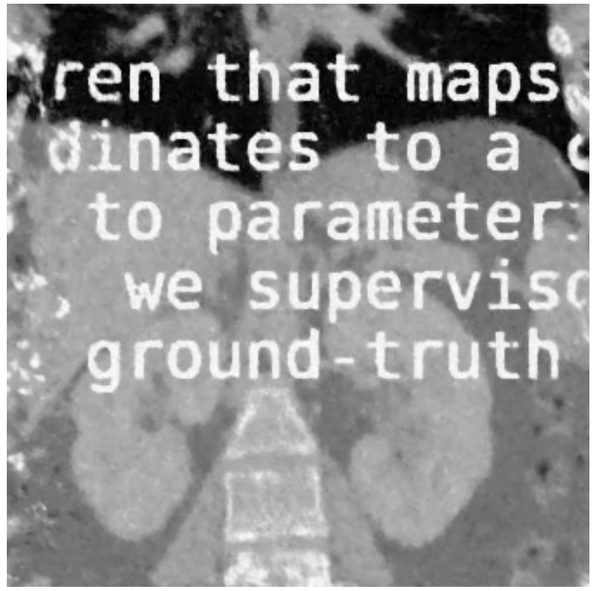
can be used **out-of-the-box optimisation** for superresolution / denoising / restoration

periodic activation is important

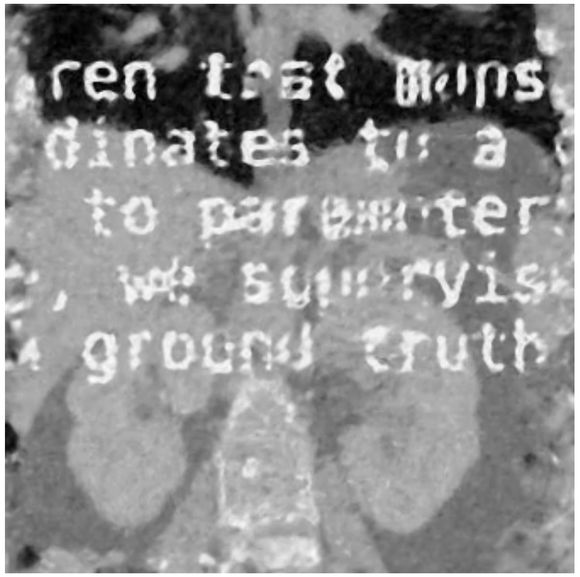
Motivation/Teaser I: multi-view superresolution



input sequence is low resolution (84x84 pixels)
"camera position" of 64 frames is known

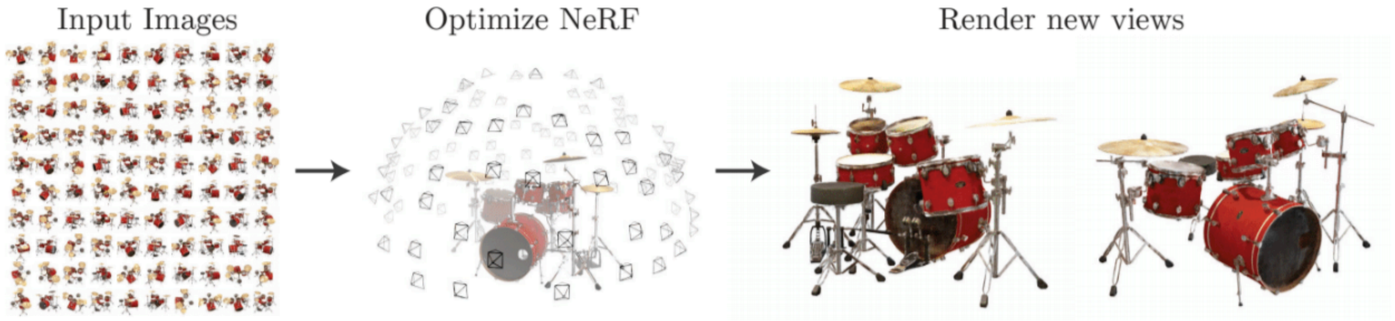


SIREN coordinate-MLP
reconstructs 8x super-resolved image
with dimensions 334x334



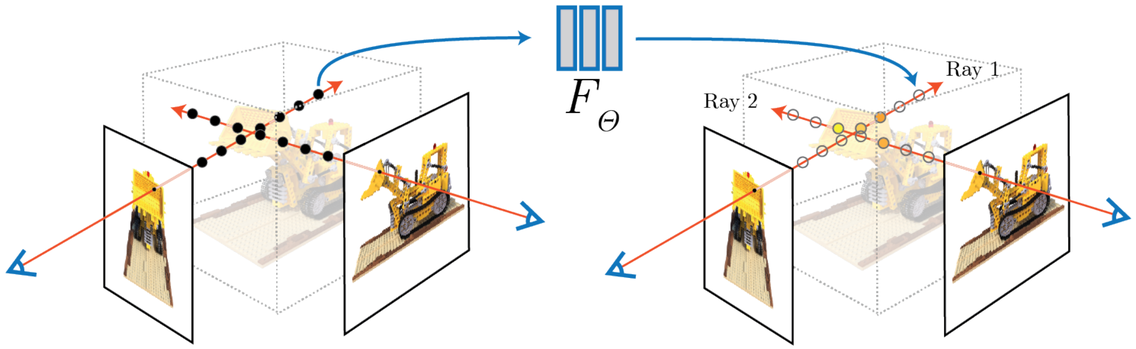
even first 8 frames of low-resolution sequence are sufficient to make most of the text readable

Motivation/Teaser II: Neural Radiance Fields



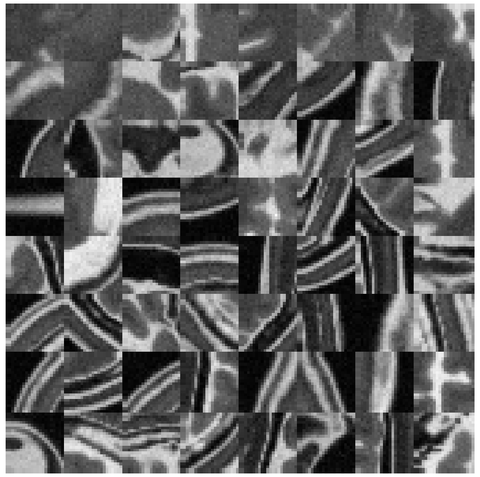
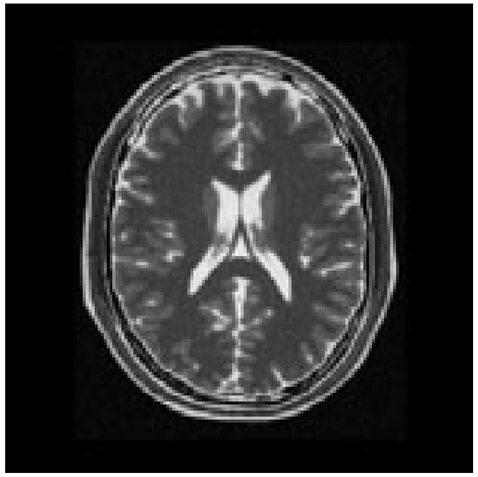
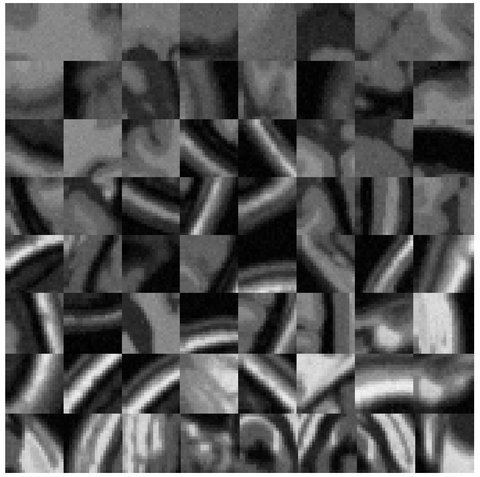
A NeRF stores a volumetric scene representation as the weights of an MLP, trained on many images with known pose.

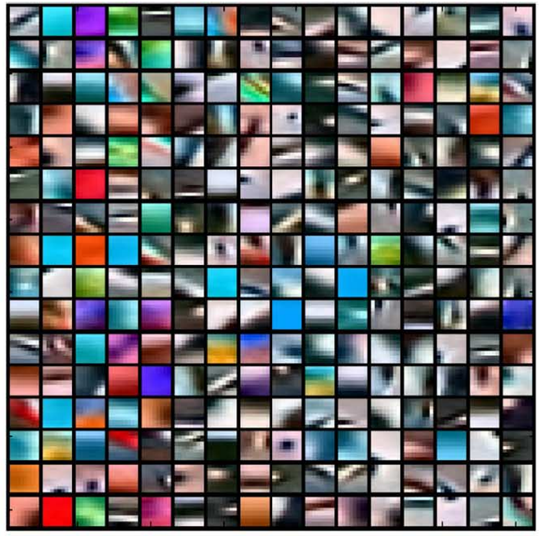
New views are rendered by integrating the density and color at regular intervals along each viewing ray.



excellent depth estimation
"for free"

Representative elements of images



Yi, Soatta: “Multimodal Registration via Spatial-Context Mutual Information“ IPMI 2011
Mairal et al.: “Sparse Representation for Color Image Restoration“ IEEE TIP 2008

Periodic Image/Coordinate Representations

Discrete cosine transformation (DCT)

Reminder: Fourier transform

Represents (complex) image by frequency/phase spectrum

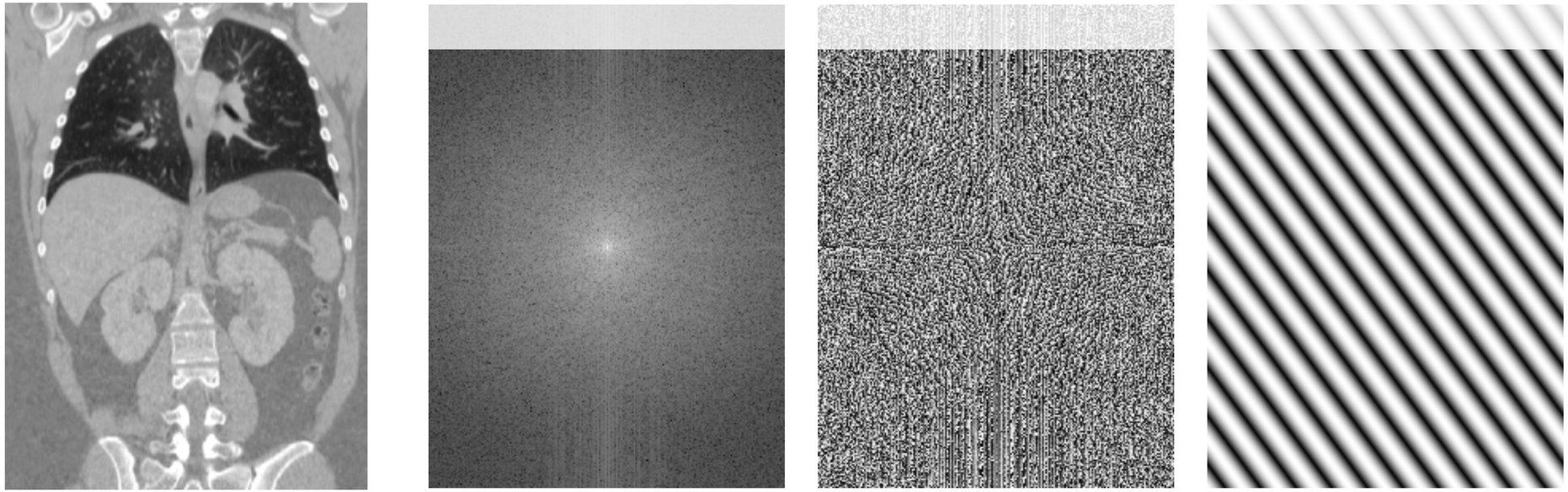


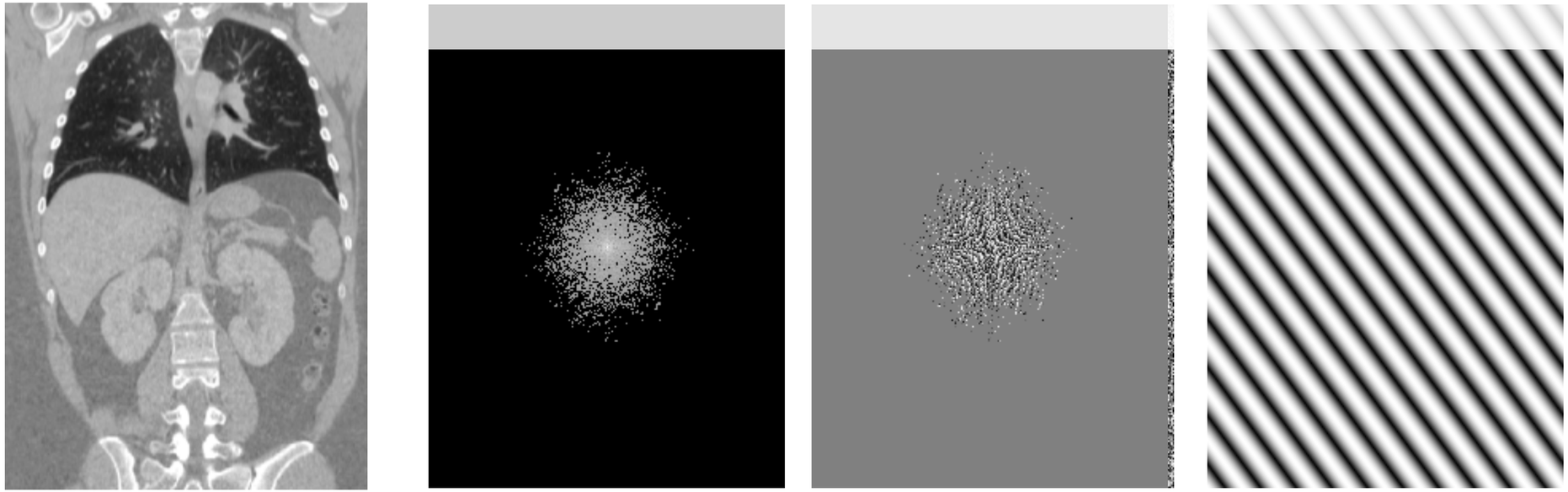
Image is created as a superposition of 2D sine waves

Reconstruction of the original image possible without loss

Discrete cosine transformation (DCT)

Reminder: Fourier transform

Represents (complex) image by frequency/phase spectrum



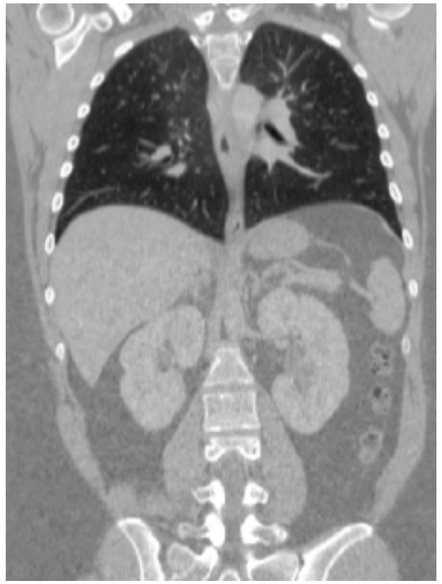
Lossy compression in Fourier space

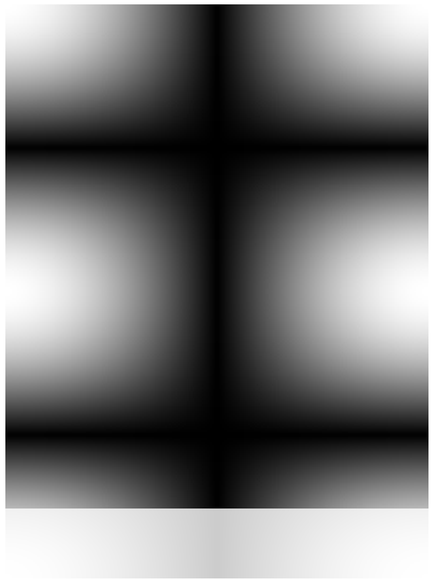
Discard (=0) of the 95% smallest coefficients in terms of magnitude

Discrete Cosine Transformation (DCT)

DCT does not contain an imaginary part

Forward transformation 1D: $$X_k = \sum_{n=0}^{N-1} x_n \cos\left[\frac{\pi}{N}\left(n+\frac{1}{2}\right)k\right] \qquad k = 0, \ldots, N-1.$$



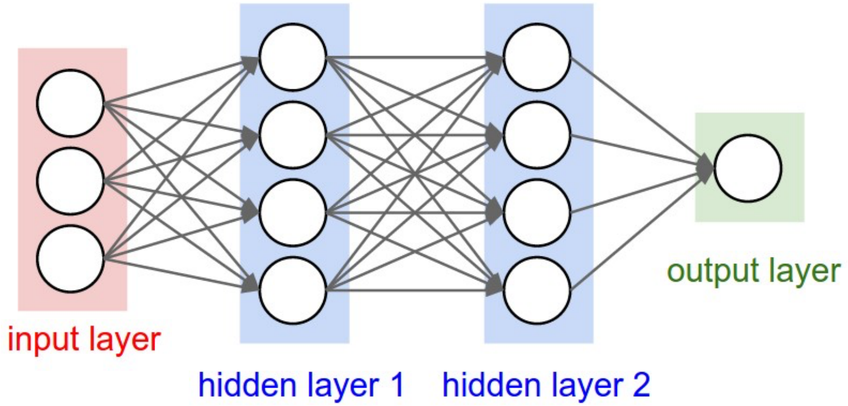



Transformation with known dimensions can be represented as a twofold matrix product

Representing images as learned functions

Representing images as **learned** functions

- **what if we could use non-linear functions?**

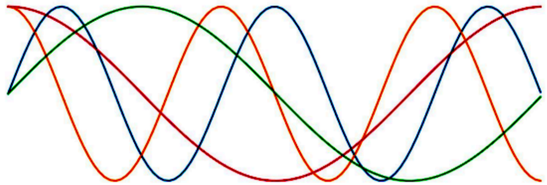


why did no one use this before 2020? Problem → **MLPs** with ReLU or tanh/sigmoid as activations **are surprising bad at learning periodic functions**

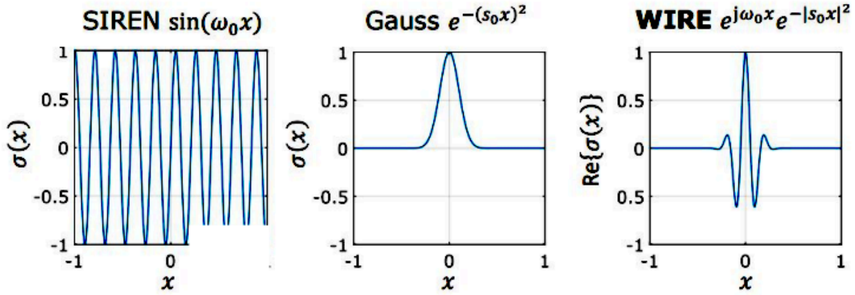
- **solution I:** use Fourier features (similar to cosine transform)
- **solution II**: use sinusoidal or exponential activation

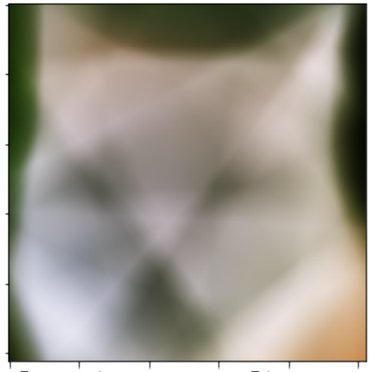
it is **important to mitigate low-frequency bias of ReLU MLP**

solution I: Coordinate Encoding / Projection



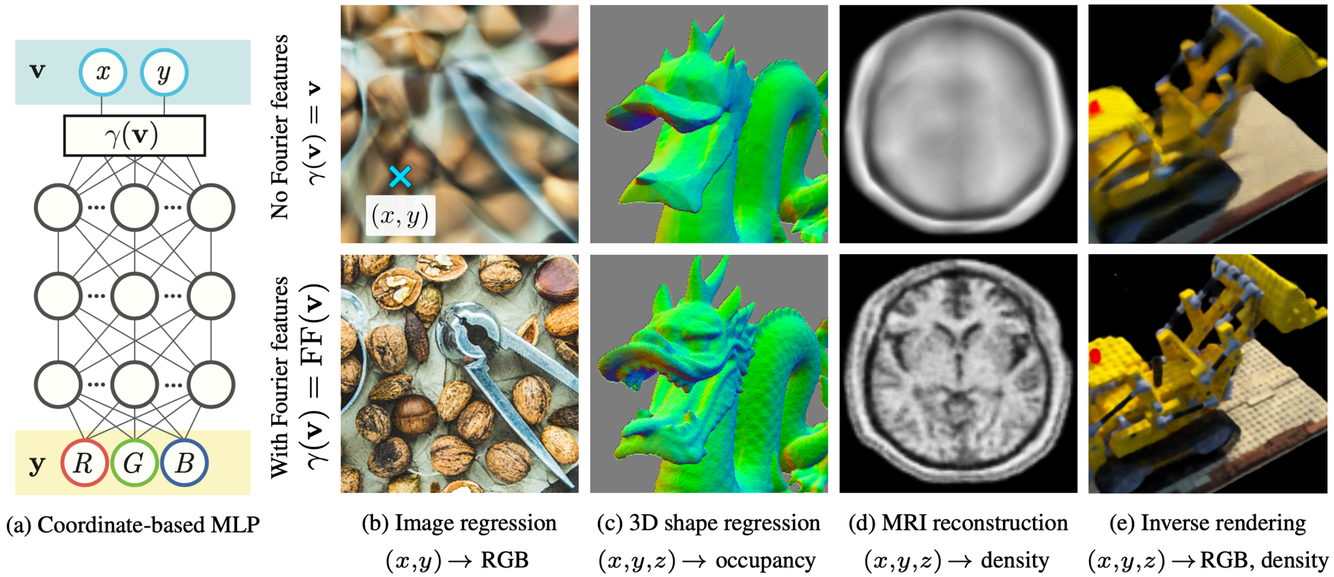
solution II: Non-ReLU Activation Functions





basic coordinate ReLU-MLP

Representing images as functions of Fourier features



(a) Coordinate-based MLP

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

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

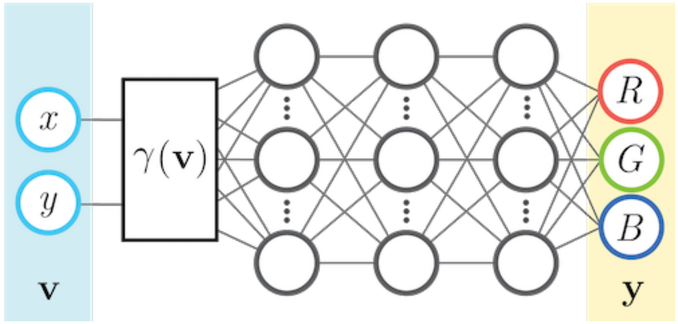
(d) MRI reconstruction
$(x,y,z) \rightarrow$ density

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

Matthew Tancik et al.: **Fourier Features Let Networks Learn High Frequency Functions in Low Dimensional Domains**
NeurIPS 2020 (spotlight)

Fourier feature mapping

- as mentioned in teasers: **the idea is to represent the mapping from xy-coordinates to RGB-values as multi-layer perceptron (MLP)**



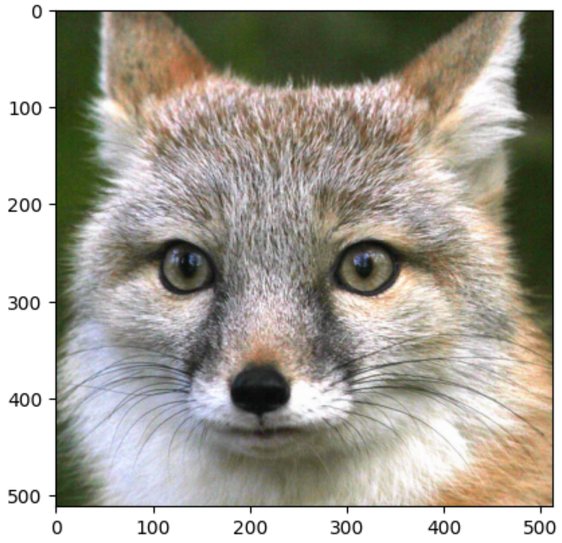
→ mitigates spectral / low-frequency bias

(Gaussian) Fourier feature mapping → transforms input coordinates

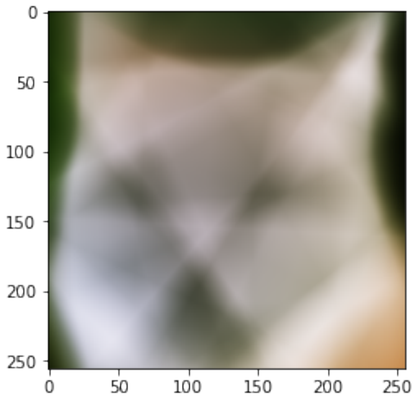
$$\gamma(\mathbf{v}) = \left[\cos(2\pi\mathbf{B}\mathbf{v}), \sin(2\pi\mathbf{B}\mathbf{v})\right]^{\mathrm{T}}$$

each entry in $\mathbf{B} \in \mathbb{R}^{m \times d}$ is sampled from $\mathcal{N}(0, \sigma^2)$ **d is usually 32-64**

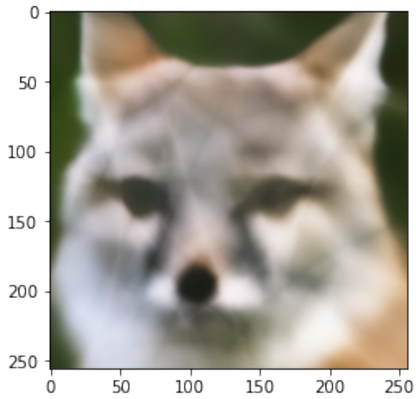
Representing images as functions of Fourier features



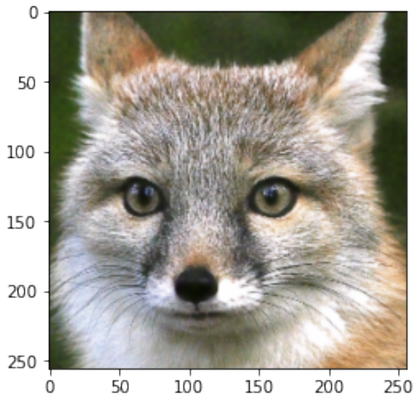
input RGB



basic coordinate ReLU-MLP



best effort coordinate ReLU-MLP
(pre-trained with periodic embedding)



(Gaussian) Fourier feature mapping

remaining issue, the dimensions and range (std.dev.) of the basis matrix B needs to be carefully chosen

$$\gamma(\mathbf{v}) = \left[\cos(2\pi\mathbf{B}\mathbf{v}), \sin(2\pi\mathbf{B}\mathbf{v})\right]^{\mathrm{T}}$$

each entry in $\mathbf{B} \in \mathbb{R}^{m \times d}$ is sampled from $\mathcal{N}(0, \sigma^2)$

https://github.com/tancik/fourier-feature-networks

Evolution of iterations for different input features

exp-Fourier #net2 263’683

#hash 70'770 #net 206’595

→ hash-grid encoding not covered today due to time constraints (feel free to ask)
Thomas Müller et al.: **Instant Neural Graphics Primitives with a Multiresolution Hash Encoding**
ACM Transactions on Graphics 2022 https://nvlabs.github.io/instant-ngp

Representing images as functions with periodic activations

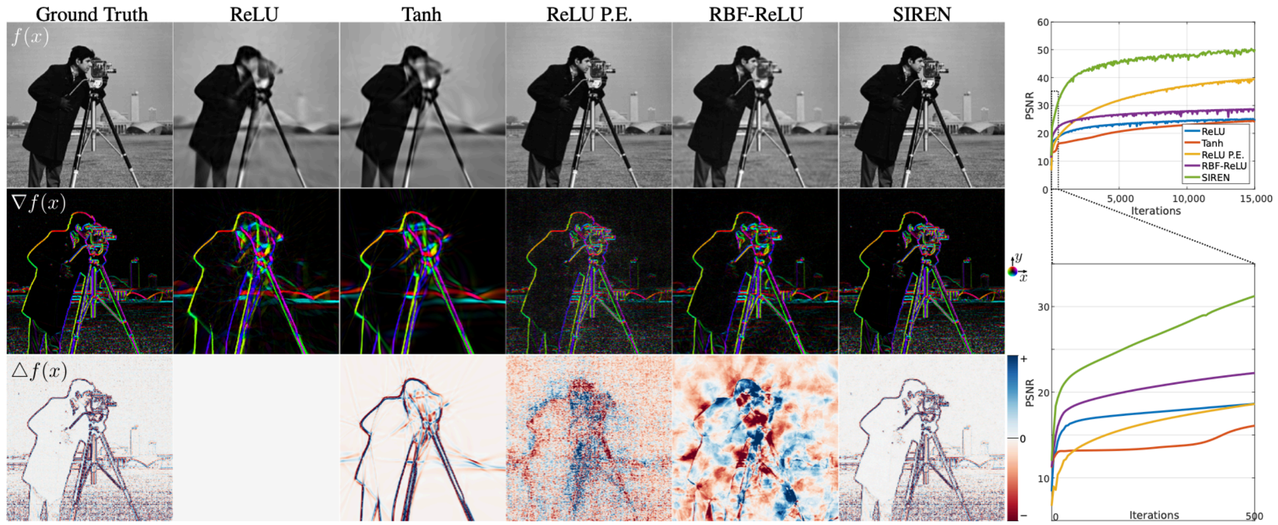


Figure 1: Comparison of different neural network architectures fitting the implicit representation of an image (ground truth: top left). The representation is only supervised on the target image but we also show first- and second-order derivatives of the function fit in rows 2 and 3, respectively.

Vincent Sitzmann*, Julien N. P. Martel* et al.: "**Implicit Neural Representations with Periodic Activation Functions**"
NeurIPS 2020

https://vsitzmann.github.io/siren/

→ introduces sinusoidal activations
networks are usually called SIRENs

Theoretical formulation

We are interested in a class of functions Φ that satisfy equations of the form

$$F\left(\mathbf{x}, \Phi, \nabla_{\mathbf{x}}\Phi, \nabla^2_{\mathbf{x}}\Phi, \ldots\right) = 0, \quad \Phi : \mathbf{x} \mapsto \Phi(\mathbf{x}). \tag{1}$$

This implicit problem formulation takes as input the spatial or spatio-temporal coordinates $\mathbf{x} \in \mathbb{R}^m$ and, optionally, derivatives of Φ with respect to these coordinates. Our goal is then to learn a neural network that parameterizes Φ to map $\mathbf{x}$ to some quantity of interest while satisfying the constraint presented in Equation (1). Thus, Φ is implicitly defined by the relation defined by F and we refer to neural networks that parameterize such implicitly defined functions as *implicit neural representations*.

Our goal is to solve problems of the form presented in Equation (1). We cast this as a feasibility problem, where a function Φ is sought that satisfies a set of M constraints $\{\mathcal{C}_m(\mathbf{a}(\mathbf{x}), \Phi(\mathbf{x}), \nabla\Phi(\mathbf{x}), ...)\}_{m=1}^{M}$, each of which relate the function Φ and/or its derivatives to quantities $\mathbf{a}(\mathbf{x})$:

$$\text{find } \Phi(\mathbf{x}) \quad \text{subject to } \mathcal{C}_m\left(\mathbf{a}(\mathbf{x}), \Phi(\mathbf{x}), \nabla\Phi(\mathbf{x}), ...\right) = 0, \ \forall \mathbf{x} \in \Omega_m, \ m = 1, \ldots, M \tag{2}$$

This problem can be cast in a loss function that penalizes deviations from each of the constraints on their domain Ω_m:

$$\mathcal{L} = \int_{\Omega} \sum_{m=1}^{M} \mathbf{1}_{\Omega_m}(\mathbf{x}) \, \|\mathcal{C}_m(\mathbf{a}(\mathbf{x}), \Phi(\mathbf{x}), \nabla\Phi(\mathbf{x}), ...)\| d\mathbf{x}, \tag{3}$$

with the indicator function $\mathbf{1}_{\Omega_m}(\mathbf{x}) = 1$ when $\mathbf{x} \in \Omega_m$ and 0 when $\mathbf{x} \notin \Omega_m$. In practice, the loss function is enforced by sampling Ω. A dataset $\mathcal{D} = \{(\mathbf{x}_i, \mathbf{a}_i(\mathbf{x}))\}_i$ is a set of tuples of coordinates $\mathbf{x}_i \in \Omega$ along with samples from the quantities $\mathbf{a}(\mathbf{x}_i)$ that appear in the constraints. Thus, the loss in Equation (3) is enforced on coordinates $\mathbf{x}_i$ sampled from the dataset, yielding the loss $\tilde{\mathcal{L}} = \sum_{i \in \mathcal{D}} \sum_{m=1}^{M} \|\mathcal{C}_m(a(\mathbf{x}_i), \Phi(\mathbf{x}_i), \nabla\Phi(\mathbf{x}_i), ...)\|$. In practice, the dataset $\mathcal{D}$ is sampled dynamically at training time, approximating $\mathcal{L}$ better as the number of samples grows, as in Monte Carlo integration.

We parameterize functions Φ_θ as fully connected neural networks with parameters θ, and solve the resulting optimization problem using gradient descent.

continuous implicit neural representation using periodic activation functions **fit complicated signals**, such as natural images and 3D shapes, **and their derivatives robustly**.

We propose SIREN, a simple neural network architecture for implicit neural representations that uses the sine as a periodic activation function:

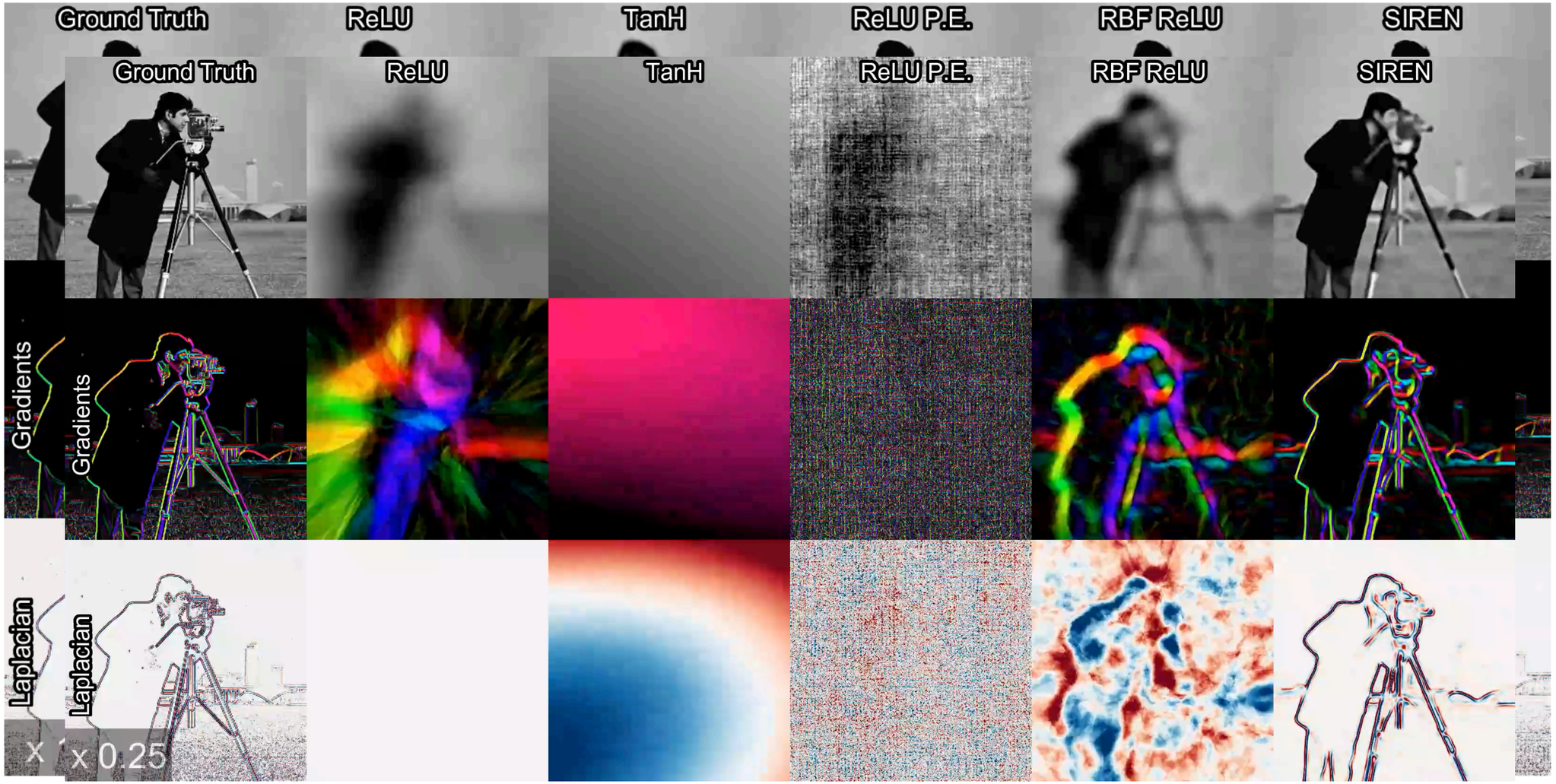
$$\Phi(\mathbf{x}) = \mathbf{W}_n \left(\phi_{n-1} \circ \phi_{n-2} \circ \ldots \circ \phi_0\right)(\mathbf{x}) + \mathbf{b}_n, \quad \mathbf{x}_i \mapsto \phi_i(\mathbf{x}_i) = \sin\left(\mathbf{W}_i \mathbf{x}_i + \mathbf{b}_i\right). \tag{4}$$

Here, $\phi_i : \mathbb{R}^{M_i} \mapsto \mathbb{R}^{N_i}$ is the i^{th} layer of the network. It consists of the affine transform defined by the weight matrix $\mathbf{W}_i \in \mathbb{R}^{N_i \times M_i}$ and the biases $\mathbf{b}_i \in \mathbb{R}^{N_i}$ applied on the input $\mathbf{x}_i \in \mathbb{R}^{M_i}$, followed by the sine nonlinearity applied to each component of the resulting vector.

Interestingly, any derivative of a SIREN *is itself a* SIREN, as the derivative of the sine is a cosine, i.e., a phase-shifted sine (see supplemental). Therefore, the derivatives of a SIREN inherit the properties of SIRENs, enabling us to supervise any derivative of SIREN with "complicated" signals. In our experiments, we demonstrate that when a SIREN is supervised using a constraint $\mathcal{C}_m$ involving the derivatives of ϕ, the function ϕ remains well behaved, which is crucial in solving many problems, including boundary value problems (BVPs).

→ some caveats: initialisation can be tricky, function has to be fitted at test-time

Implicit Neural Representations with Periodic Activation Functions

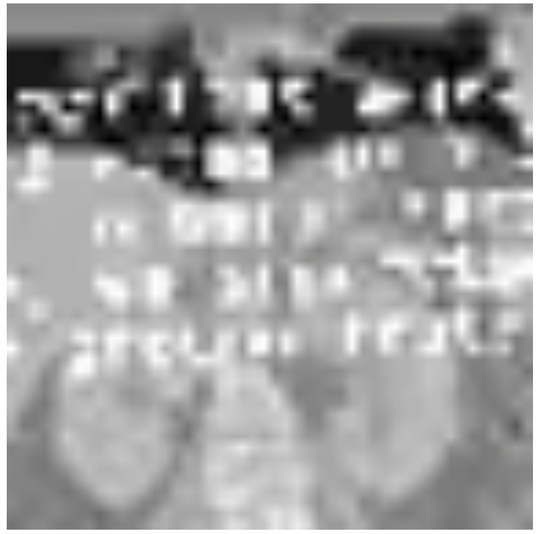


Representing images as functions with periodic activations

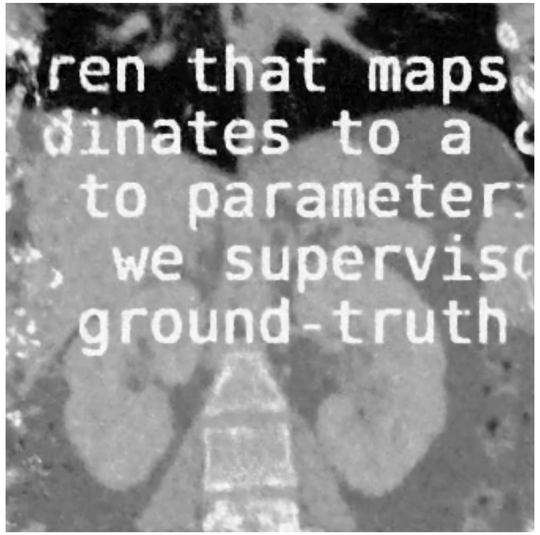
typical forward pass for MLP, first layer has to have sinusoidal (or exponential) activation, the other ones could be ReLUs

```
def forward(self,x):
    scale = self.scale
    x = torch.sin(scale*self.net[0](x))
    for i in range(1,len(self.net)-1):
        #x = torch.exp(-scale*self.net[i](x).pow(2))
        #x = F.relu(self.net[i](x))
        x = torch.sin(scale*self.net[i](x))
    return self.net[-1](x)
```

→ SIREN paper discusses **multiple advanced applications**, including inpainting, solving wave equations, and representing shapes as signed distance function



input sequence is low resolution (84x84 pixels)
"camera position" of 64 frames is known



SIREN coordinate-MLP **reconstructs 8x super-resolved image** with dimensions 334x334

same example from beginning

Hands-on Tutorial

ziadhemidi.github.io/INR4BVM25.github.io