COORDINATE-BASED NEURAL REPRESENTATIONS

Jan Schlüter

38th Vienna Deep Learning Meetup

2021-02-17

JOHANNES KEPLER UNIVERSITY LINZ

Institute of Computational Perception

GENERAL IDEA

- Define a continuous function mapping 1D/2D/3D coordinates to values
  $f(x,y;\boldsymbol{\theta}) = (r, g, b) \qquad x, y, r, g, b \in [0,1]$
- Fit this function to a discrete 1D/2D/3D signal
- Profit!

https://yinboc.github.io/liif/

CONTRAST: VAE DECODER OR GAN GENERATOR

- Define a continuous function mapping some latent space to a discrete signal

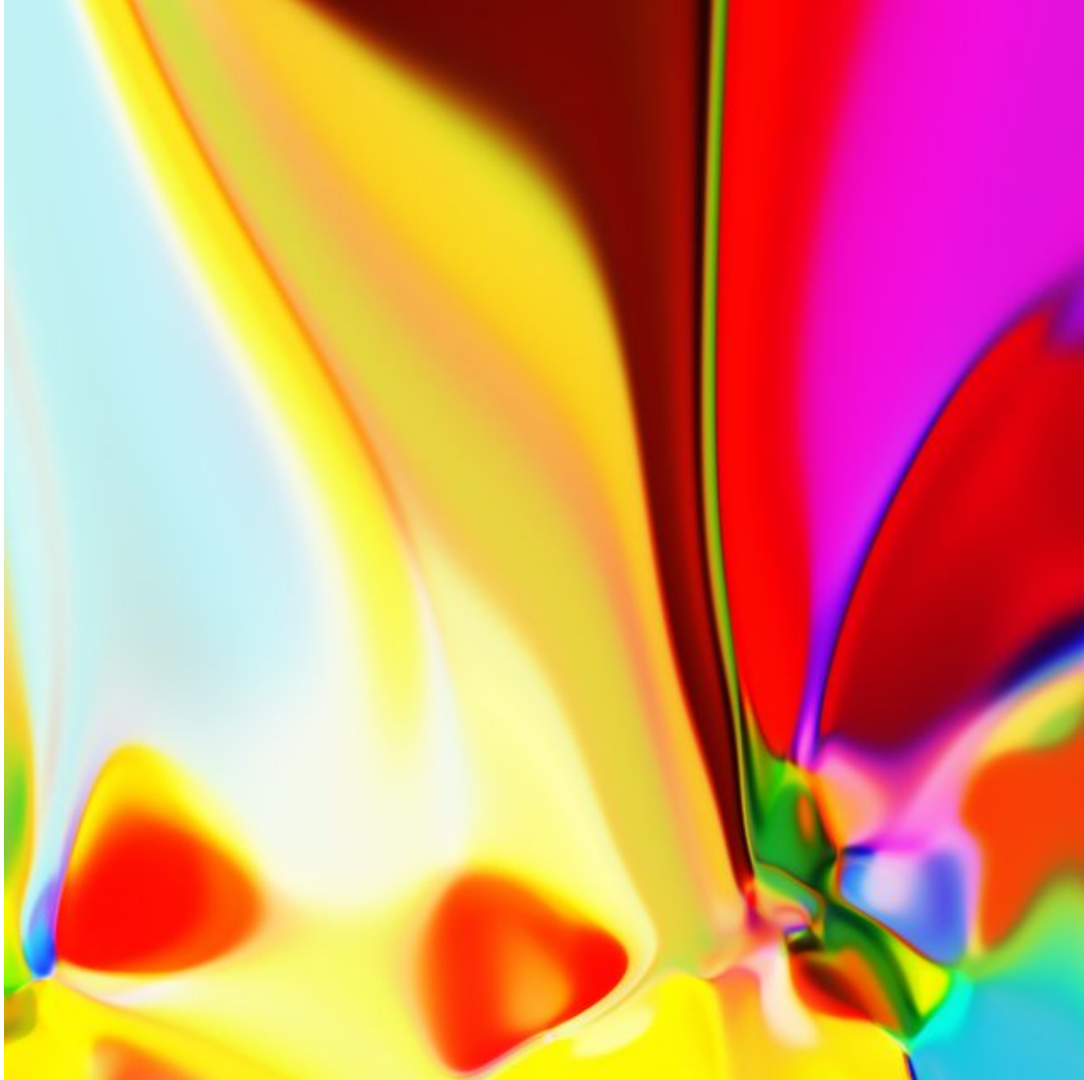
  $f(\mathbf{z}; \boldsymbol{\theta}) = \mathbf{X} \qquad \mathbf{z} \in \mathbb{R}^n, \mathbf{X} \in \mathbb{R}^{h \times w}$
- Fit this function to a set of discrete signals
- profit!

https://arxiv.org/abs/1707.05776

JOHANNES KEPLER UNIVERSITY LINZ

RANDOM WEIGHTS

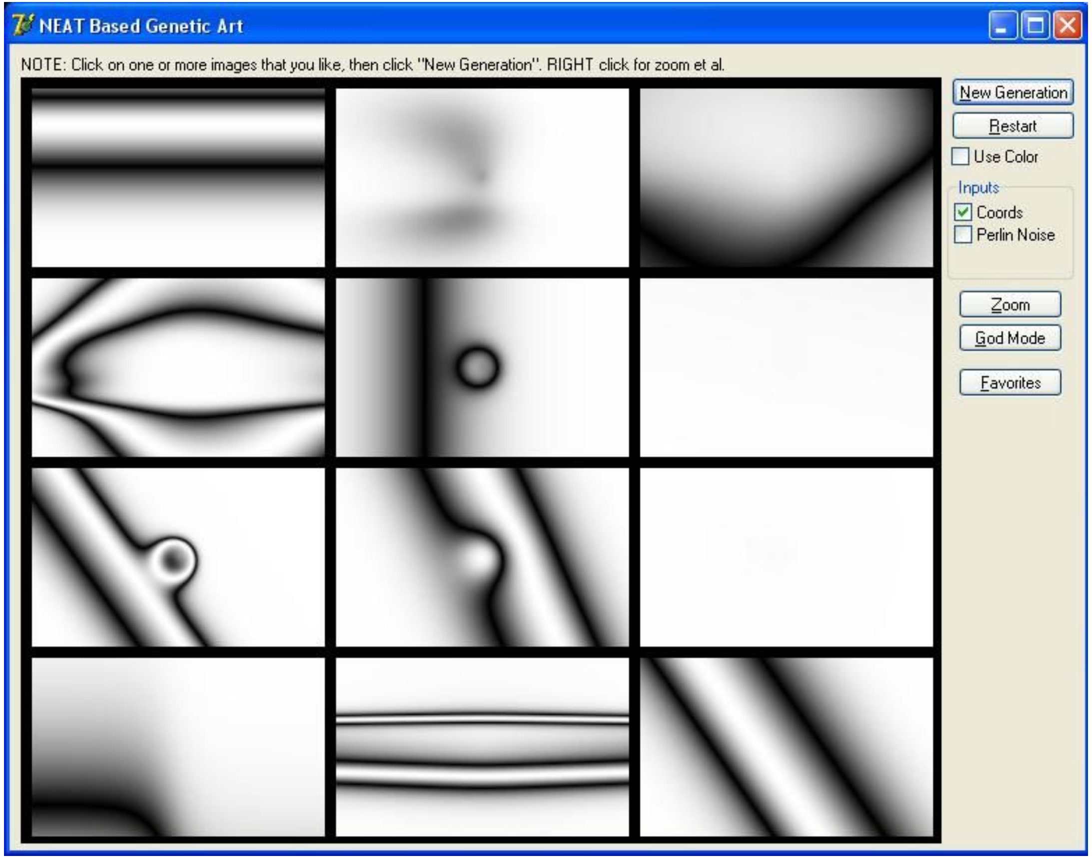
- For artistic purposes:
  Define a neural network for
  $f(x,y; \boldsymbol{\theta}) = (r, g, b)$
  and set $\boldsymbol{\theta}$ randomly,
  render image.



JKU
JOHANNES KEPLER
UNIVERSITY LINZ

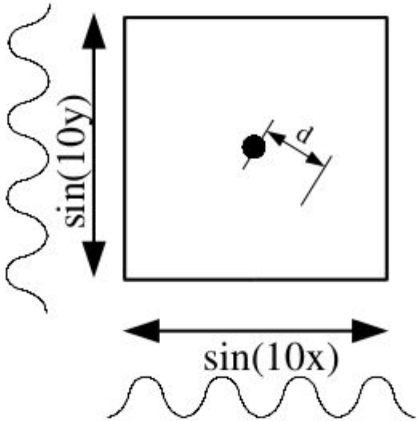
EVOLUTION

- For artistic purposes:
  Define a neural network for
  $f(x,y; \boldsymbol{\theta}) = (r, g, b)$
  and evolve $\boldsymbol{\theta}$ with user feedback.
- This was the original idea, from a 2007 paper:
  http://eplex.cs.ucf.edu/papers/stanley_gpem07.pdf
  Complex Pattern Producing Networks (CPPNs)

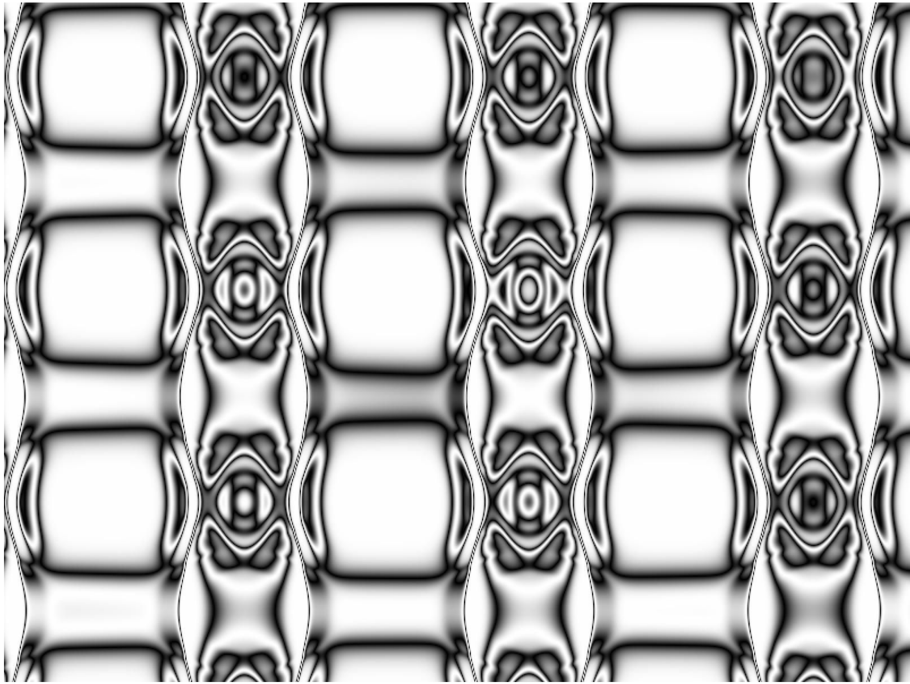


JOHANNES KEPLER UNIVERSITY LINZ

PERIODIC PATTERNS



http://eplex.cs.ucf.edu/papers/stanley_gpem07.pdf



FITTING AN EXISTING IMAGE

- Define a continuous function mapping 1D/2D/3D coordinates to values

  $$f(x,y;\boldsymbol{\theta}) = (r, g, b) \qquad x, y, r, g, b \in [0,1]$$

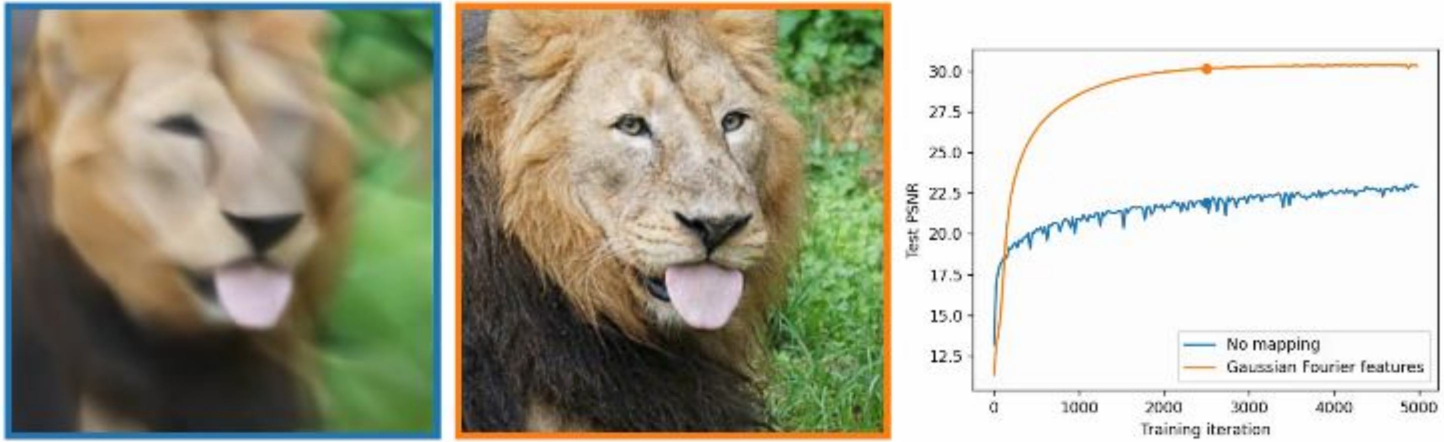
- Fit this function to a discrete 1D/2D/3D signal
- Live demo (Kaparthy, 2014): https://cs.stanford.edu/people/karpathy/convnetjs/demo/image_regression.html

FITTING A CPPN TO VISUALIZE A CNN'S FEATURES

- https://distill.pub/2018/differentiable-parameterizations/#section-xy2rgb

FOURIER FEATURES

- https://bmild.github.io/fourfeat/img/lion_none_gauss_v1.mp4



- Encode coordinates **v** in a high-dimensional feature space (with random **B**):

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

https://bmild.github.io/fourfeat/

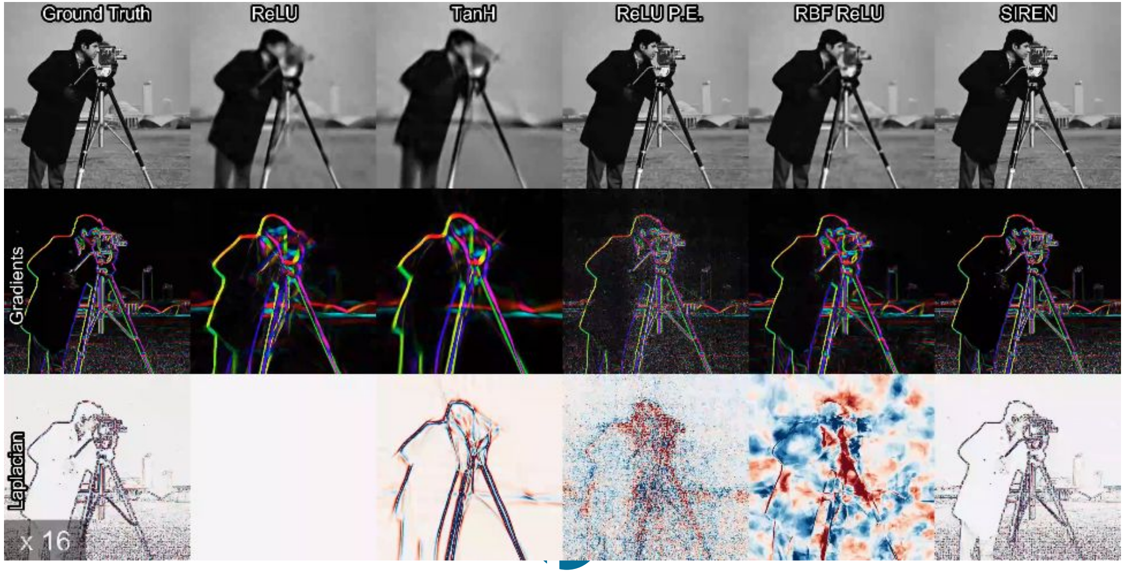
JOHANNES KEPLER UNIVERSITY LINZ

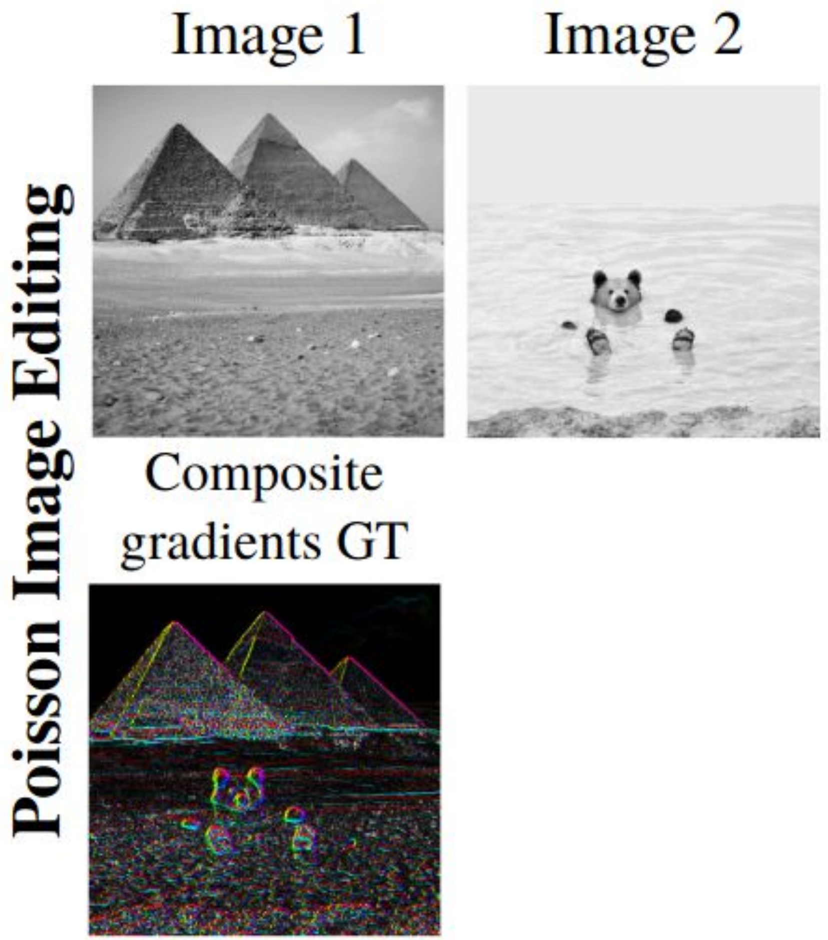
PERIODIC NONLINEARITIES

- https://vsitzmann.github.io/siren/
- Use sin() as the nonlinearity in every layer
- Outperforms positional encoding

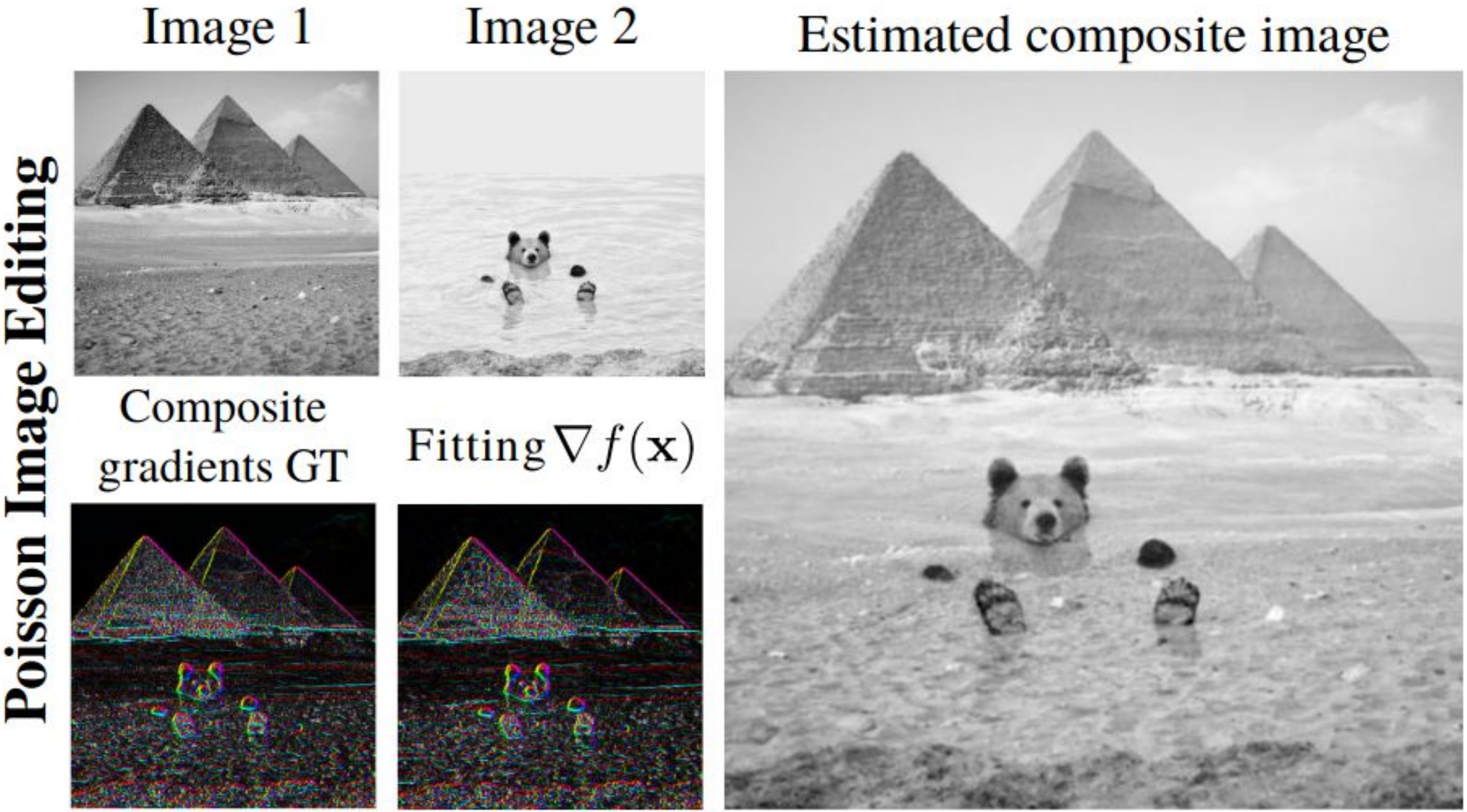
PERIODIC NONLINEARITIES

- https://vsitzmann.github.io/siren/
- Use sin() as the nonlinearity in every layer
- Outperforms positional encoding, can also model image derivatives





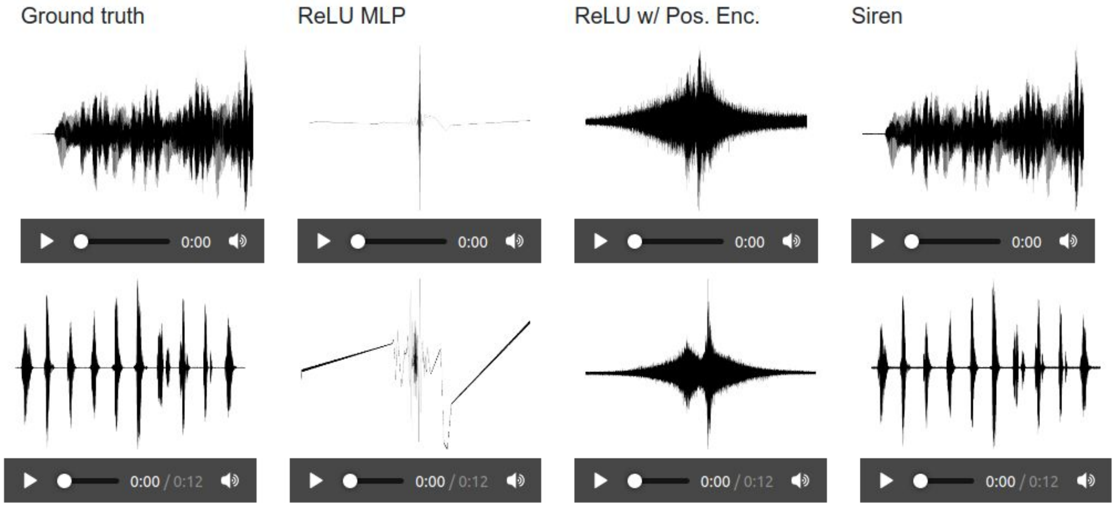
SIREN, https://arxiv.org/abs/2006.09661



SIREN, https://arxiv.org/abs/2006.09661

PERIODIC NONLINEARITIES

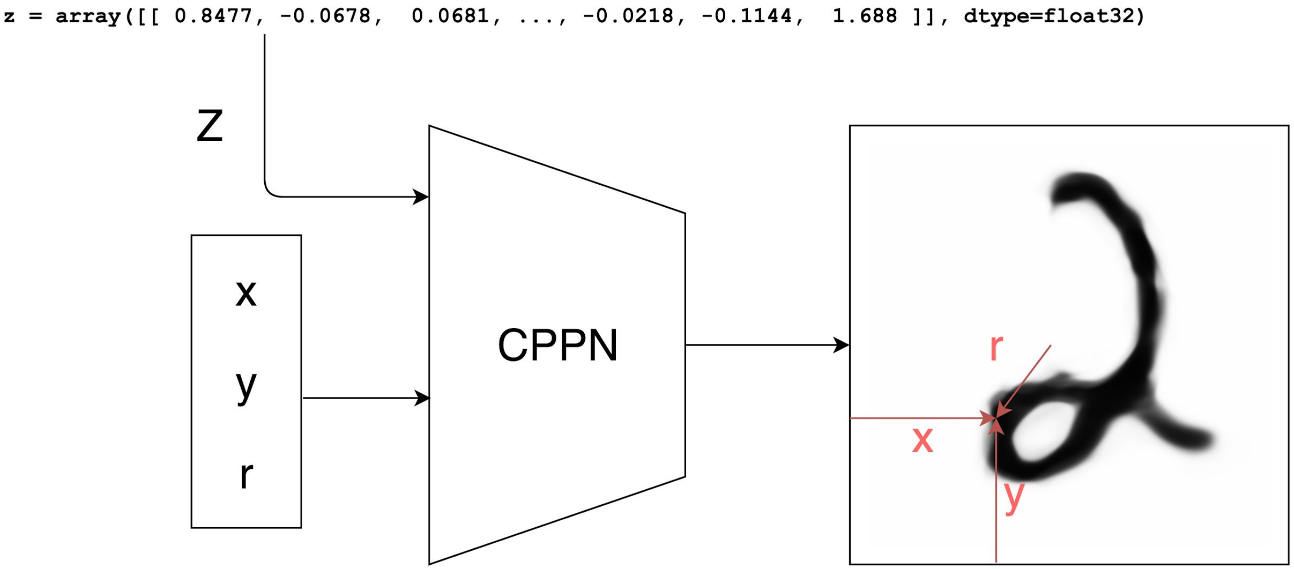
- https://vsitzmann.github.io/siren/
- Use sin() as the nonlinearity in every layer
- Can also model audio signals

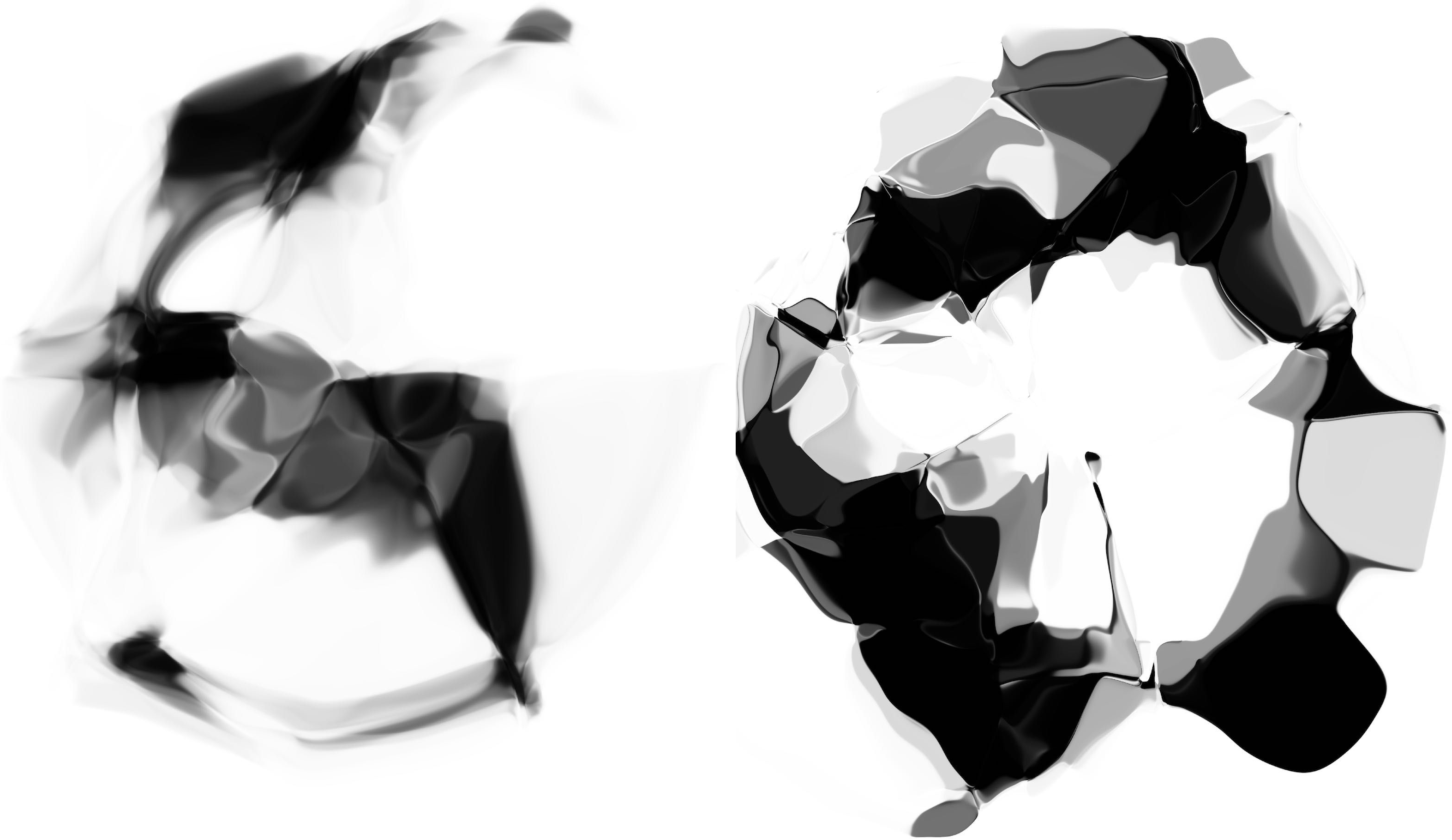


JOHANNES KEPLER UNIVERSITY LINZ

GENERATIVE ADVERSARIAL CPPNS

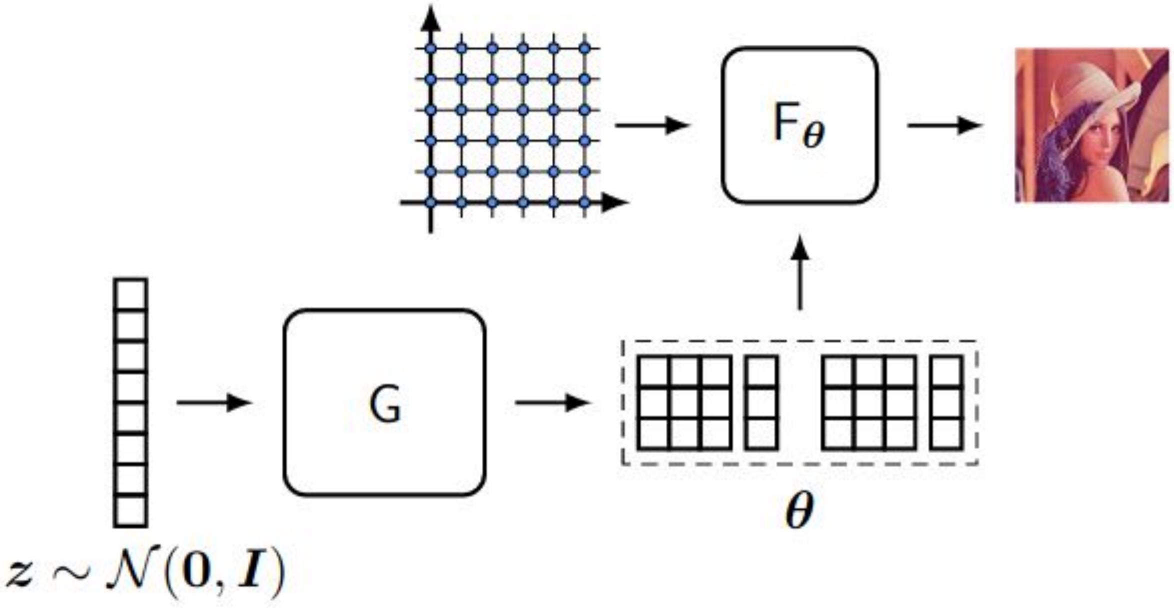
- https://blog.otoro.net/2016/04/01/generating-large-images-from-latent-vectors/





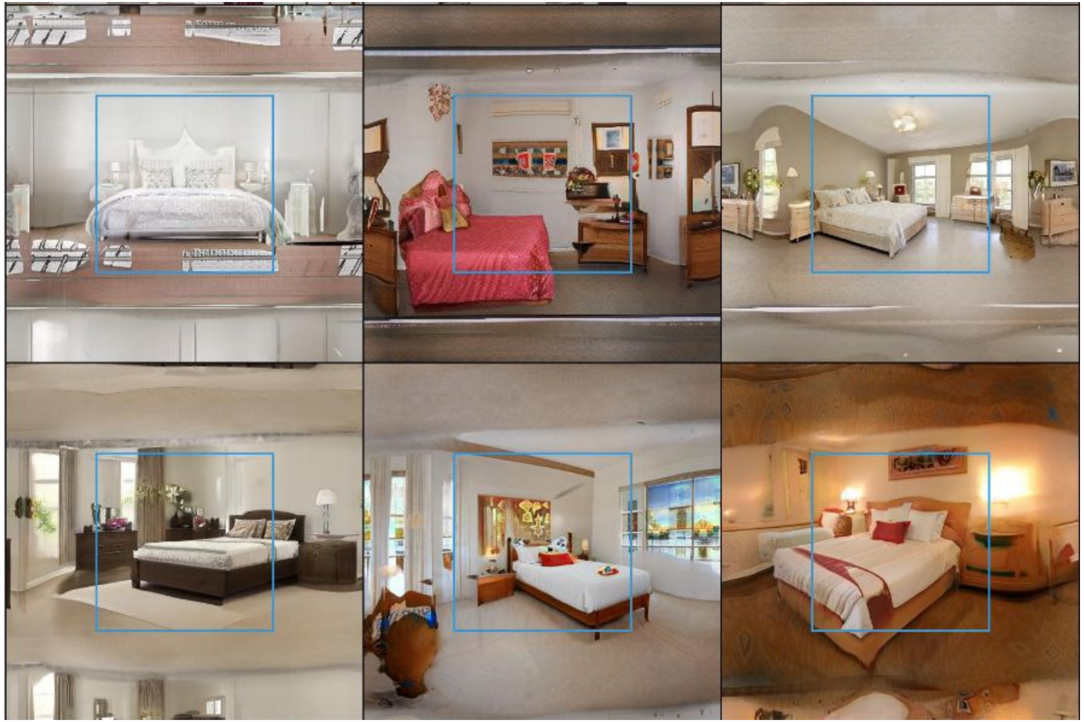
GENERATIVE ADVERSARIAL CPPNS

- https://arxiv.org/abs/2011.12026



GENERATIVE ADVERSARIAL CPPNS

- https://arxiv.org/abs/2011.12026



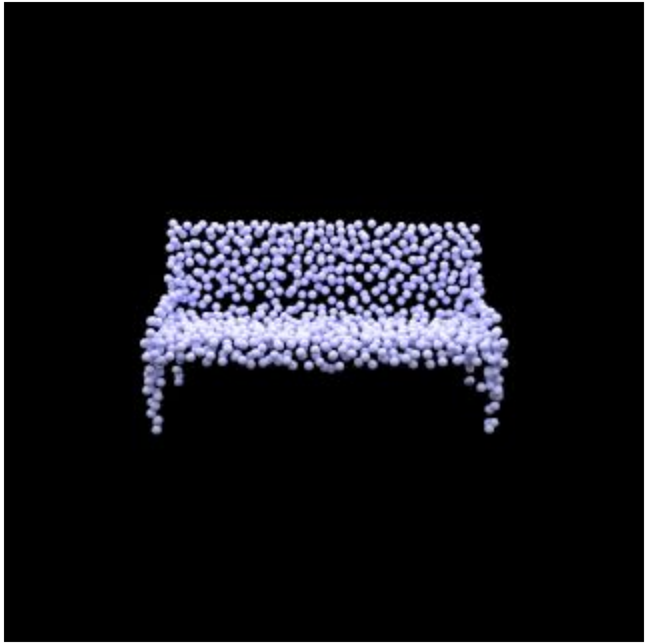
JOHANNES KEPLER UNIVERSITY LINZ

REPRESENTING 3D SHAPES

- https://autonomousvision.github.io/occupancy-networks/
- Voxels: memory-expensive or blocky

REPRESENTING 3D SHAPES

- https://autonomousvision.github.io/occupancy-networks/
- Point clouds: Lack connectivity information

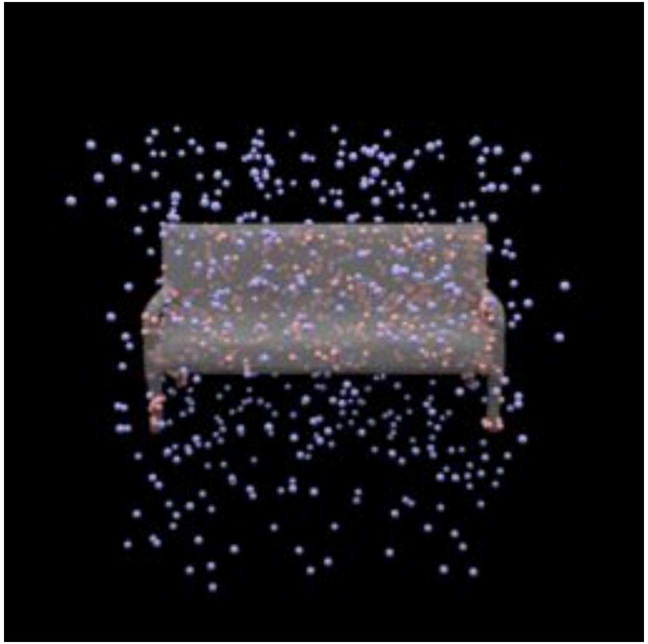


REPRESENTING 3D SHAPES

- https://autonomousvision.github.io/occupancy-networks/
- Meshes: Easy to get wrong

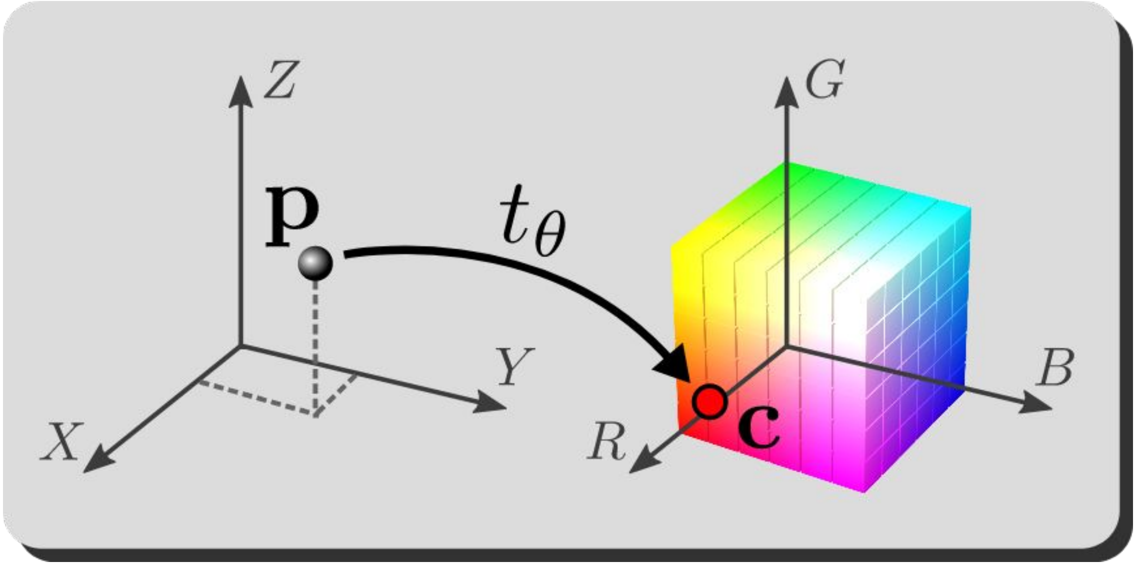
REPRESENTING 3D SHAPES

- https://autonomousvision.github.io/occupancy-networks/
- Define mapping from x,y,z to [0,1] (= is this position occupied or empty)



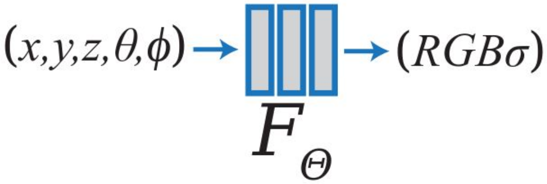
REPRESENTING 3D TEXTURE

- https://autonomousvision.github.io/texture-fields/
- Define mapping from x,y,z to r,g,b



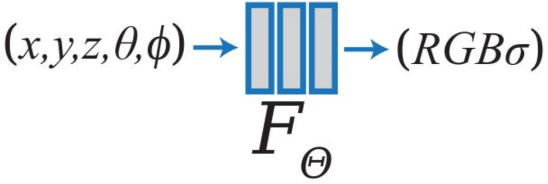
NERF: NEURAL RADIANCE FIELDS

- https://www.matthewtancik.com/nerf
- Define mapping from x,y,z (location) and θ,φ (view direction) to r,g,b (color) and σ (density)

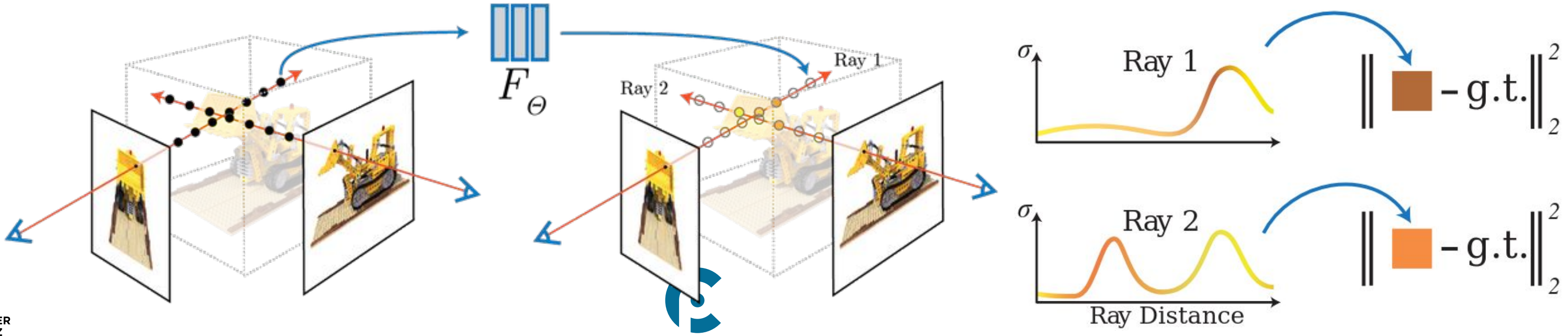


NERF: NEURAL RADIANCE FIELDS

- https://www.matthewtancik.com/nerf
- Define mapping from x,y,z (location) and θ,φ (view direction) to r,g,b (color) and σ (density)



- Optimize it using multiple 2D views of an object



NERF: NEURAL RADIANCE FIELDS

- https://www.matthewtancik.com/nerf
- Can then synthesize novel views of an object or scene

- https://storage.googleapis.com/nerf_data/website_renders/viewdirs_website_bww.mp4

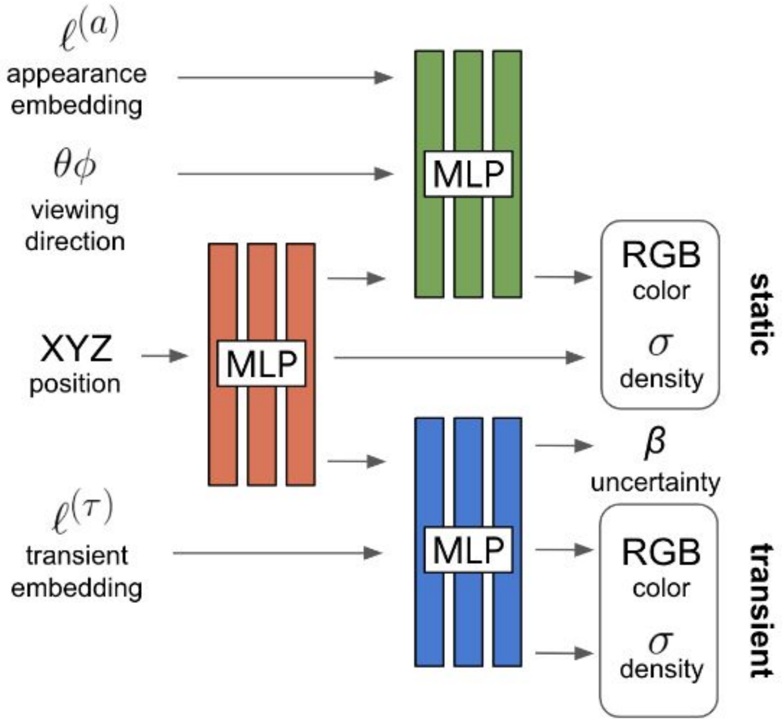
JOHANNES KEPLER
UNIVERSITY LINZ

NERF IN THE WILD

- https://nerf-w.github.io/
- NERF has view-dependent appearance, but what if there are other factors that change appearance?

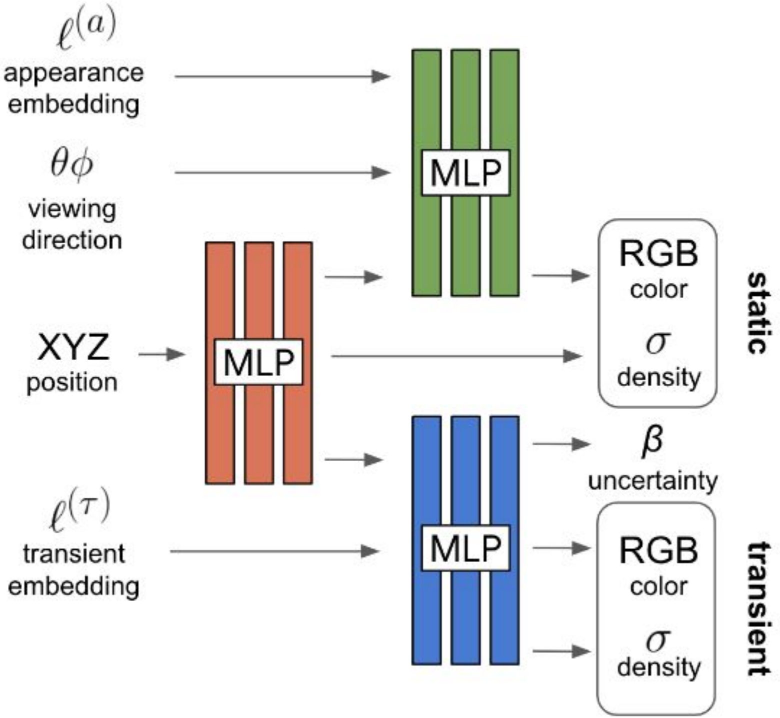
NERF IN THE WILD

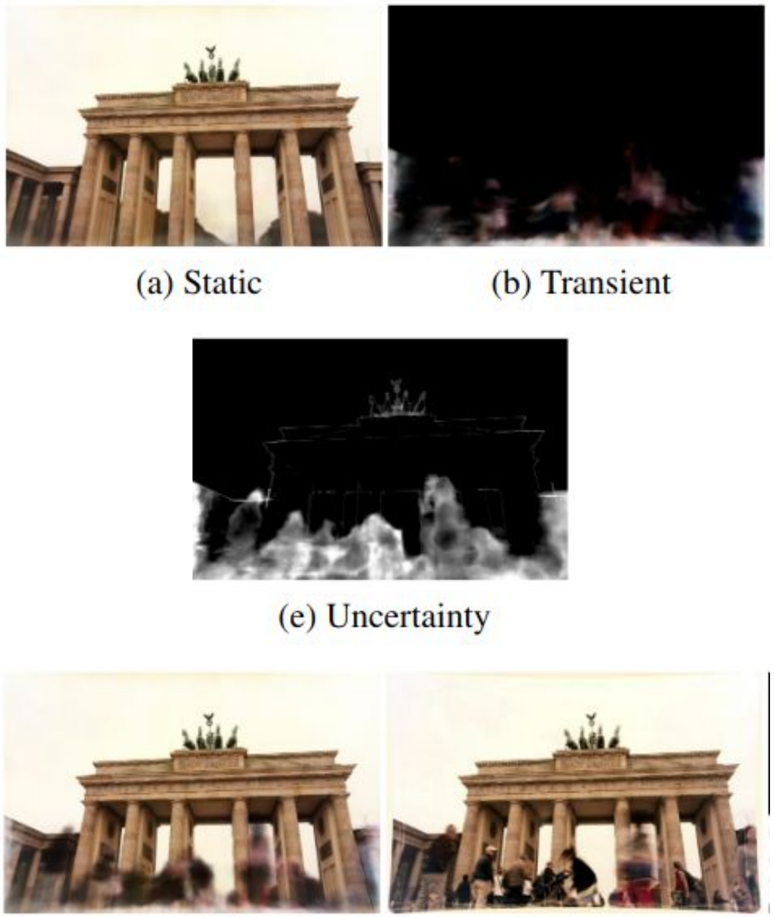
- https://nerf-w.github.io/
- NERF has view-dependent appearance, but what if there are other factors that change appearance?
- NeRF-W adds an appearance embedding



NERF IN THE WILD

- https://nerf-w.github.io/





(a) Static (b) Transient

(e) Uncertainty

(c) Composite (d) Image

NERF IN THE WILD

- https://nerf-w.github.io/
- NERF has view-dependent appearance, but what if there are other factors that change appearance?
- ⇒ NeRF-W adds an appearance embedding
- and can be trained on unconstrained image collections!
- https://youtu.be/mRAKVQj5LRA?t=47

(a) Photos

(b) Renderings

JOHANNES KEPLER UNIVERSITY LINZ

MORE SOURCES

- https://github.com/vsitzmann/awesome-implicit-representations
- https://github.com/yenchenlin/awesome-NeRF