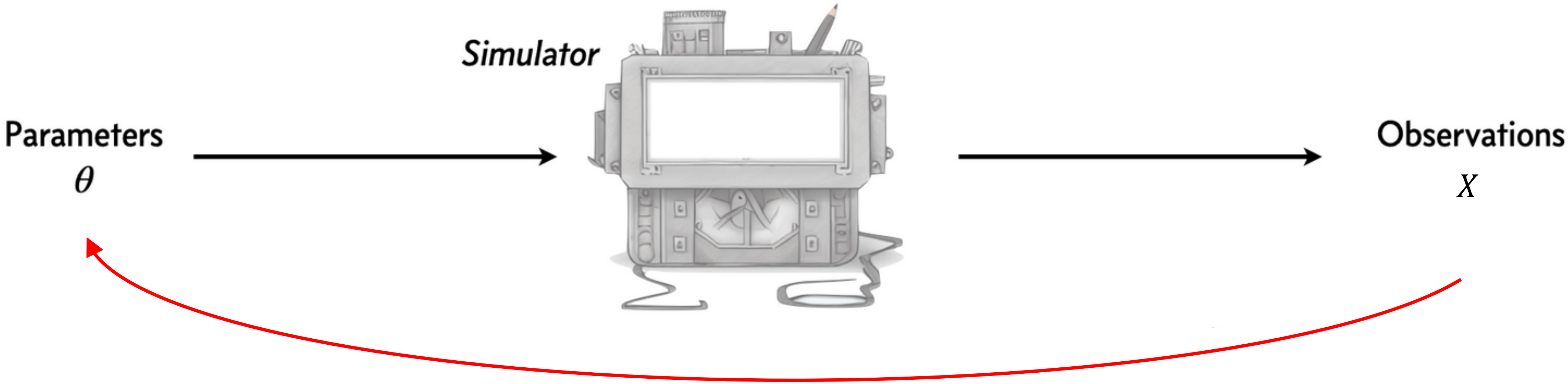
Neural Fields

Michael Kagan
SLAC

July 21, 2023

Making use of Simulators

So far, we looked at *Simulation-Based Inference*:



Making use of Simulators

So far, we looked at *Simulation-Based Inference*:

From Lukas Heinrich: learned about *Differentiable Programming*

$$f(\mathbf{x}) : \mathbb{R}^n \to \mathbb{R}$$

↓ **automatic differentiation**

$$\nabla f(\mathbf{x}) = \left(\frac{\partial f}{\partial x_1}, \ldots, \frac{\partial f}{\partial x_n}\right)$$

```
f(x) {…};
```

↓

```
df(x) {…};
```

Making use of Simulators

So far, we looked at *Simulation-Based Inference*:

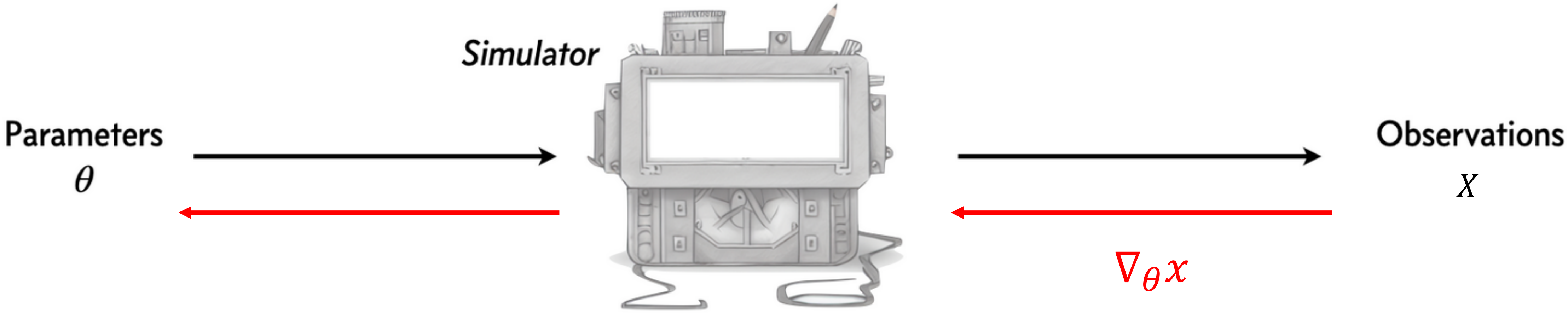
From Lukas Heinrich: learned about *Differentiable Programming*

What can we do with a differentiable simulator?

$$\frac{d}{d\theta}\mathbb{E}[L(x)] = \mathbb{E}\left[\frac{dL}{dx}\frac{dx}{d\theta}\right] = \mathbb{E}\left[\frac{dL}{dx}\frac{d}{d\theta}SIM(\theta)\right]$$

Where $x = SIM(\theta)$

Optimization through simulator

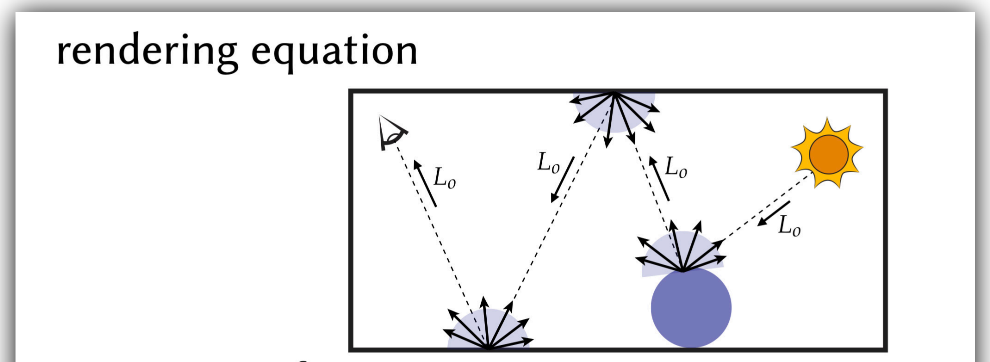


What can we do with a Differentiable Simulator?

Rendering and Inverse Rendering

Rendering:

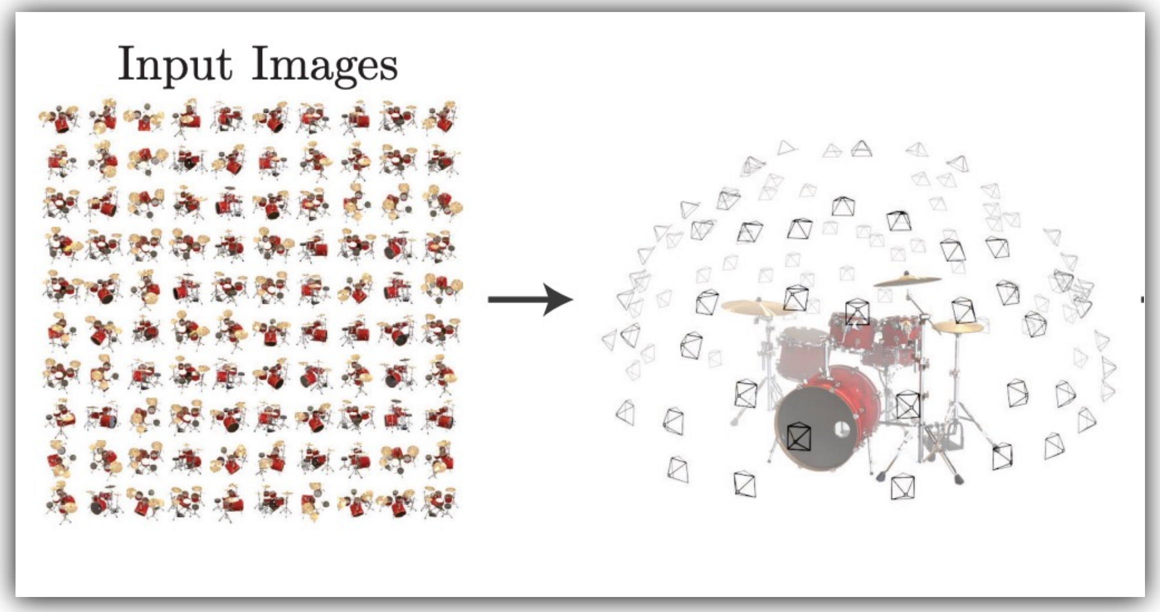
From 3D model scene, simulate image on camera at given position and angle



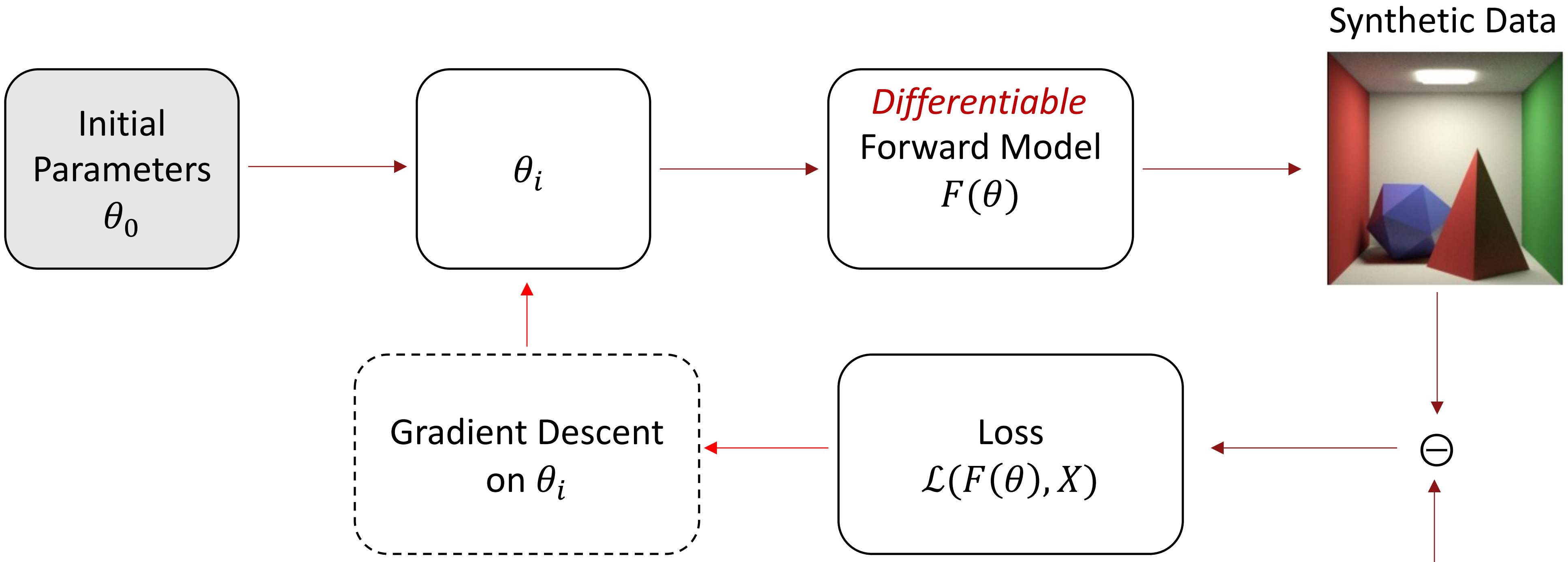
$$L(x, \vec{\omega}_o) = \int_{\mathcal{H}^2} f_r(x, \vec{\omega}_i, \vec{\omega}_o) L(x, -\vec{\omega}_i) \cos\theta \, d\omega_i$$

Inverse Rendering:

From multiple 2D images, reconstruct 3D model of scene



Analysis-by-Synthesis



$$\theta^* = \arg\min_{\theta} \mathcal{L}(F(\theta), X)$$

$$\mathcal{L} = |F(\theta) - X|^2$$

Analysis-by-Synthesis

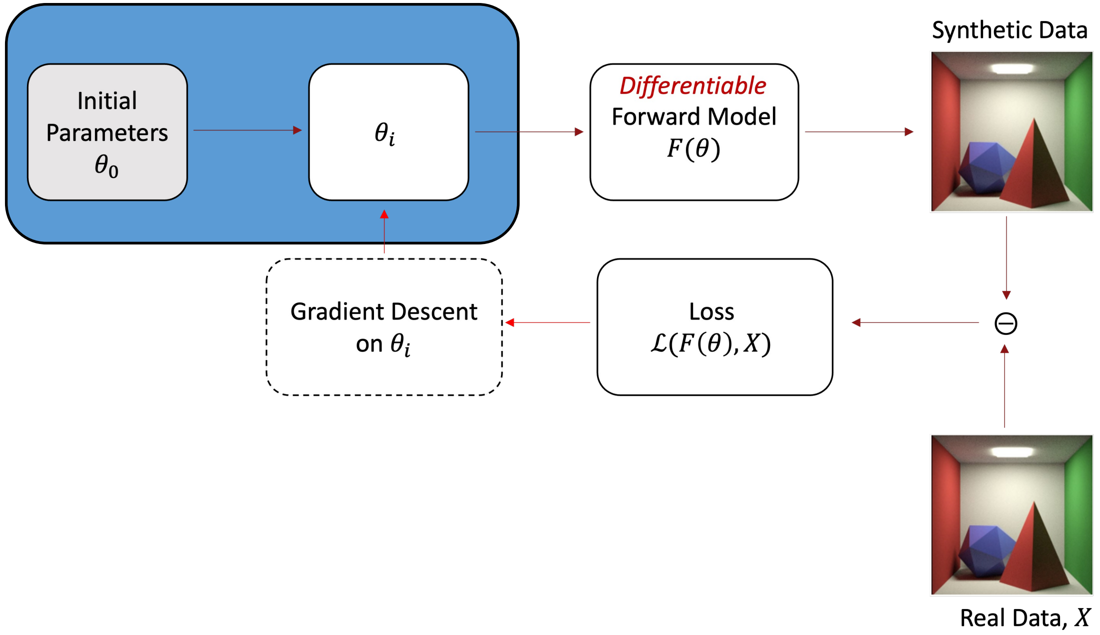
Goal:
Find parameters θ such that the simulator with these parameters, $F(\theta)$, generates synthetic data that matches the observed data

Analysis-by-Synthesis

Goal:
Find parameters θ such that the simulator with these parameters, $F(\theta)$, generates synthetic data that matches the observed data

Basic idea:

- Start with initial guess θ_0

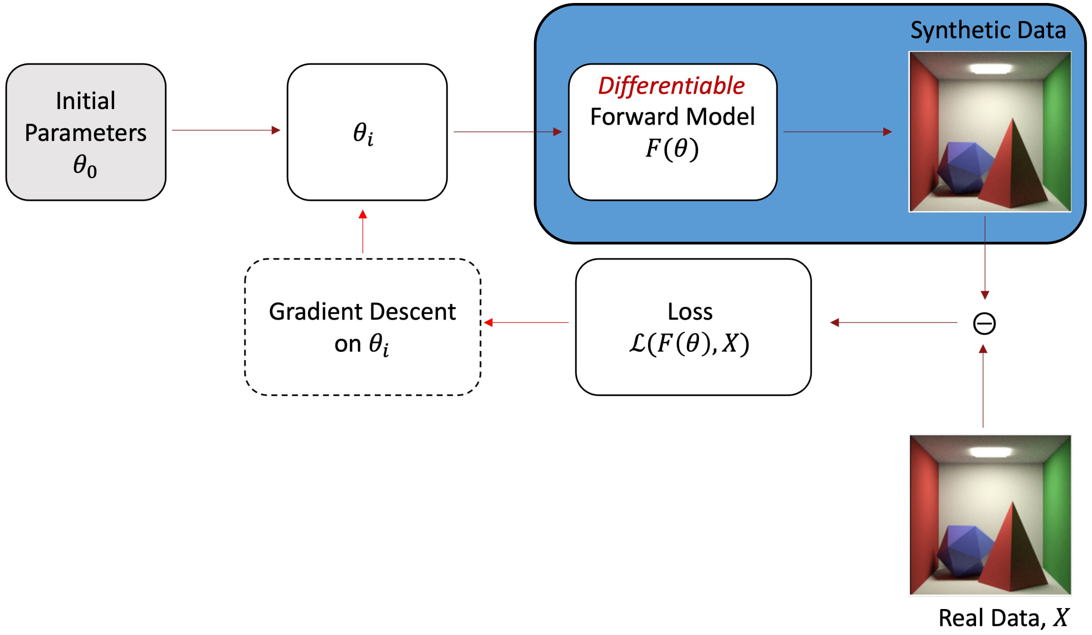


Analysis-by-Synthesis

Goal:
Find parameters θ such that the simulator with these parameters, $F(\theta)$, generates synthetic data that matches the observed data

Basic idea:

- Start with initial guess θ_0
- Given θ_i Generate synthetic data with simulator $F(\theta)$

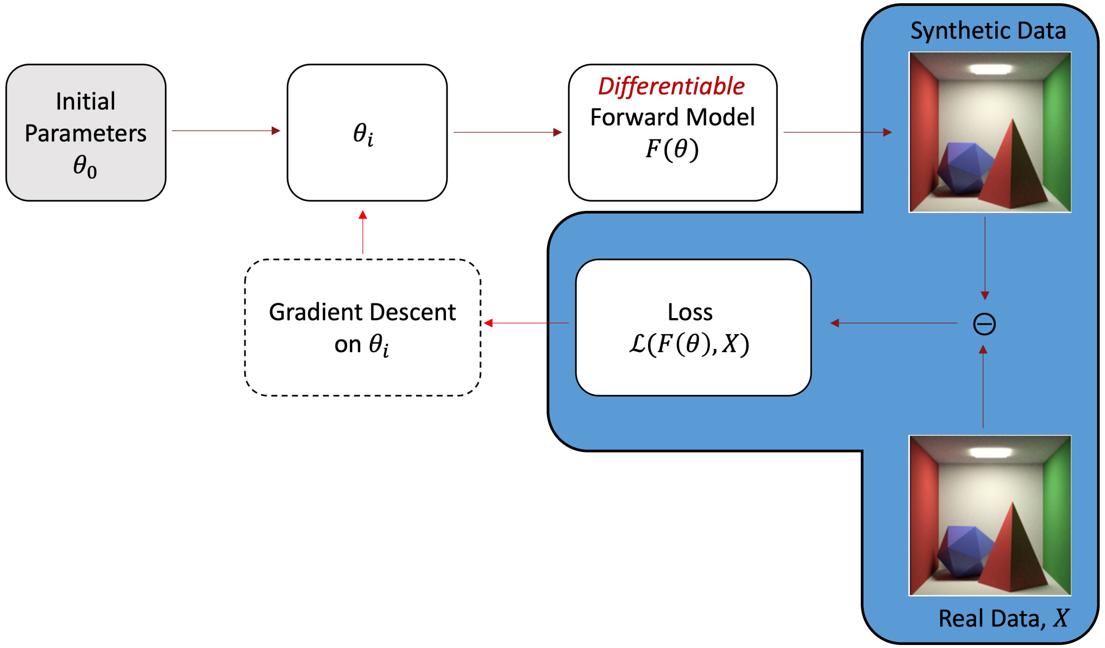


Analysis-by-Synthesis

Goal:
Find parameters θ such that the simulator with these parameters, $F(\theta)$, generates synthetic data that matches the observed data

Basic idea:

- Start with initial guess θ_0
- Given θ_i Generate synthetic data with simulator $F(\theta)$
- Loss function compares synthetic & real data

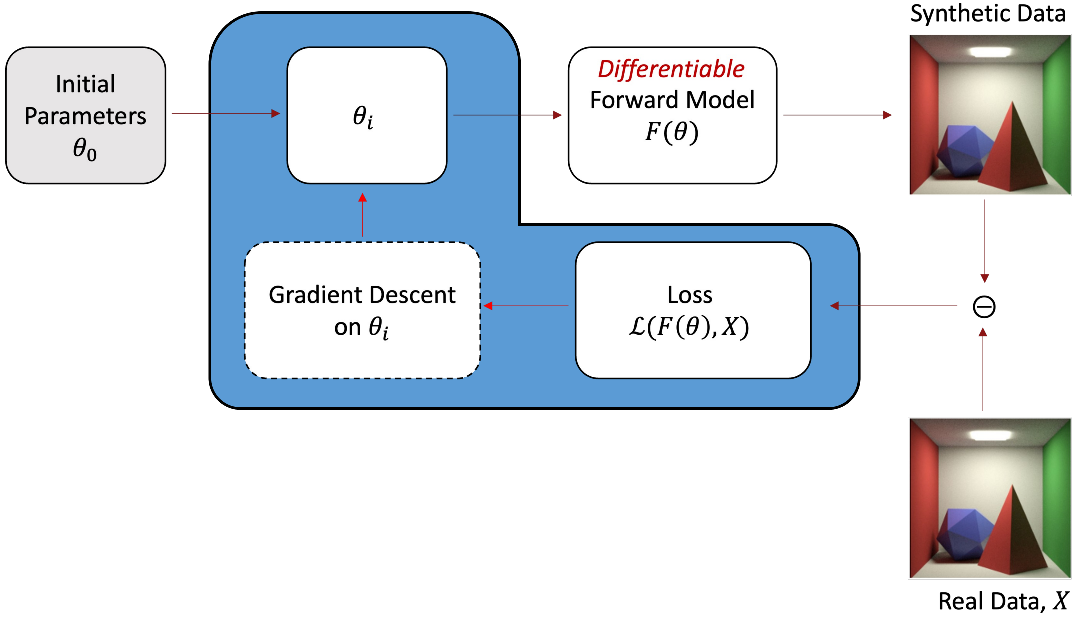


Analysis-by-Synthesis

Goal:
Find parameters θ such that the simulator with these parameters, $F(\theta)$, generates synthetic data that matches the observed data

Basic idea:

- Start with initial guess θ_0
- Given θ_i Generate synthetic data with simulator $F(\theta)$
- Loss function compares synthetic & real data
- Update θ_{i+1} to lower loss



Gradient descent with differentiable simulator: $\theta \leftarrow \theta - \eta \frac{d\mathcal{L}}{dF} \frac{dF(\theta)}{d\theta}$

Analysis-by-Synthesis

What's going on here? *Maximum Likelihood Estimation*

If we assume an error model $I = F(\theta) + \epsilon$ where $\epsilon \sim N(0,1)$

Then $$p(X|\theta) = N(F(\theta), 1)$$

And $$\mathcal{L} = -\log p(X|\theta) = |F(\theta) - X|^2$$

Analysis-by-Synthesis

What's going on here? *Maximum Likelihood Estimation*

If we assume an error model $I = F(\theta) + \epsilon$ where $\epsilon \sim N(0,1)$

Then $$p(X|\theta) = N(F(\theta), 1)$$

And $$\mathcal{L} = -\log p(X|\theta) = |F(\theta) - X|^2$$

We get a *point estimate* for the MLE $\boldsymbol{\theta}$ by solving this optimization

Analysis-by-Synthesis

What's going on here? *Maximum Likelihood Estimation*

If we assume an error model $I = F(\theta) + \epsilon$ where $\epsilon \sim N(0,1)$

Then $$p(X|\theta) = N(F(\theta), 1)$$

And $$\mathcal{L} = -\log p(X|\theta) = |F(\theta) - X|^2$$

We get a *point estimate* for the MLE θ by solving this optimization

Note: In general this is **NOT an amortized process**,

for each observation X, have to solve a different optimization problem

Combining partial measurements

Input Images

Measurements from different view points

Goal

Combine to form 3D model

So Why Analysis-by-Synthesis?

Reconstructing spatio-temporal signals from a set of observations

$$\mathcal{L} = \sum_i \left| F_i\big(S(\boldsymbol{x}, \theta)\big) - X_i \right|^2$$

$i \in$ observations

Ill-posed inverse problems: often not enough X's to fully constrain system

Often signal $S(\cdot)$ is represented with voxels, meshes, etc... with parameters θ

$F_i \equiv$ simulation of i^{th} observation... e.g. simulate camera i at specific position

So Why Analysis-by-Synthesis?

Reconstructing spatio-temporal signals from a set of observations

$$\mathcal{L} = \sum_i \left| F_i\left(NN_\theta(\boldsymbol{x})\right) - X_i \right|^2$$

$i \in$ observations

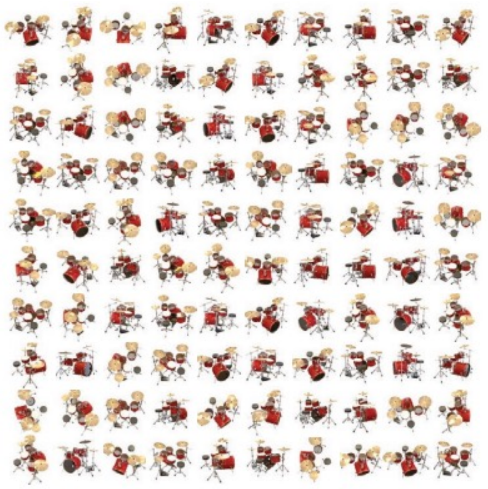
Ill-posed inverse problems: often not enough X's to fully constrain system

Analysis-by-Synthesis approach has recently been highly successful for reconstructing a signal parameterized by a neural network,

→ often called a *Neural Field*

Combining partial measurements

Input Images



Measurements from
different view points

Goal

Combine to form 3D model

Neural Network Model
of 3D system

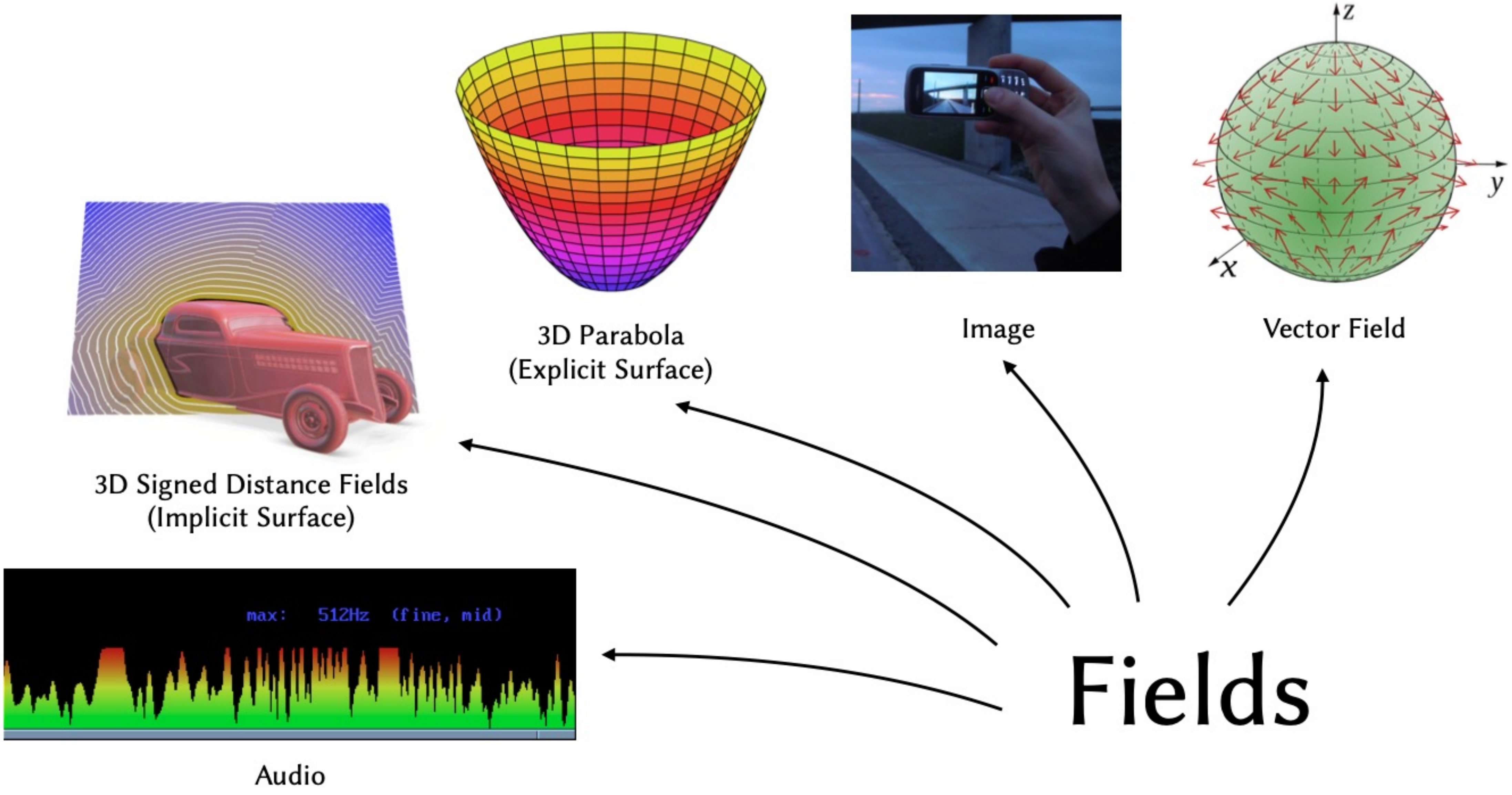
Defining Neural Field

Definition 1: A *field* is a function which assigns scalar / vector / tensor values for all spatial and / or temporal coordinates*.

Definition 2: A *neural field* is a field that is parameterized fully or partially by a neural network.

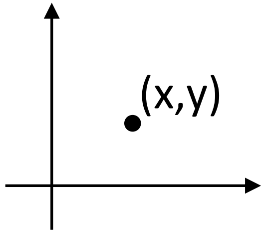
*Sometimes over other spaces, like frequency space

Examples of Neural Fields

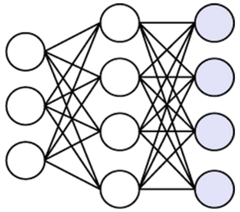


Slide credit: T. Takikawa

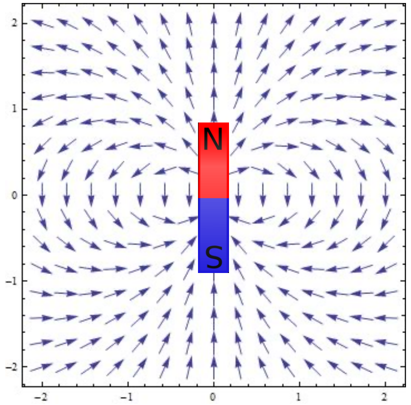
What are Neural Fields?



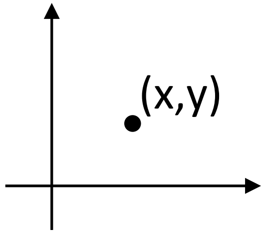
$\Phi: \mathbb{R}^2 \rightarrow \mathbb{R}^2$



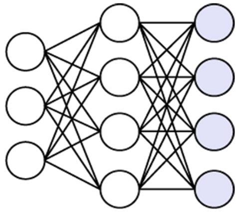
Neural Network
(Φ)



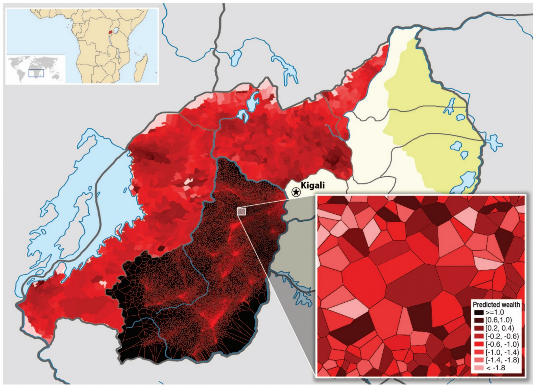
Magnetic Field



$\Phi: \mathbb{R}^2 \rightarrow \mathbb{R}^n$

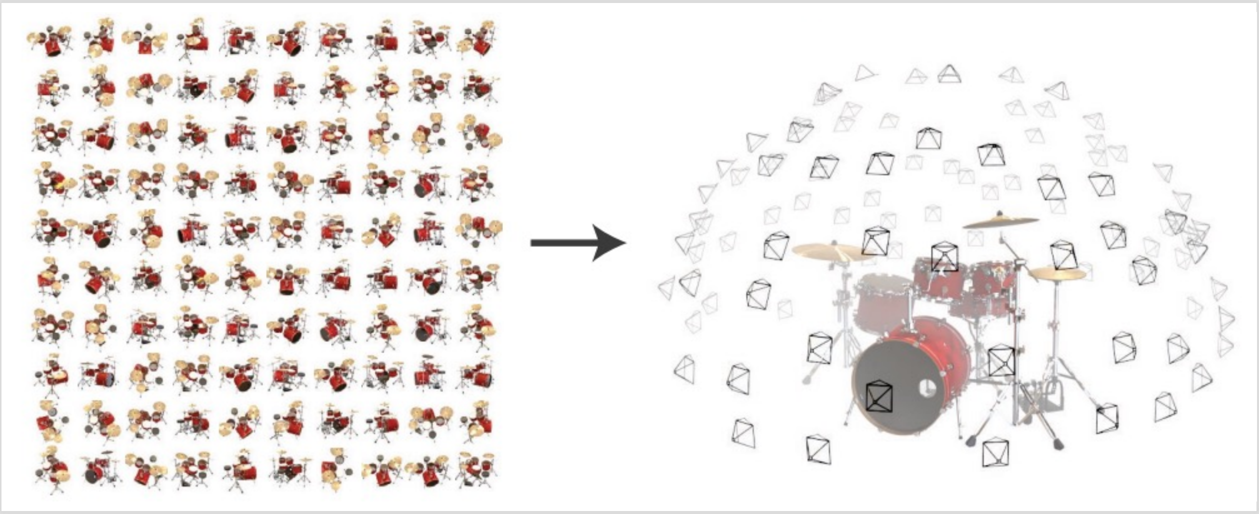


Neural Network
(Φ)



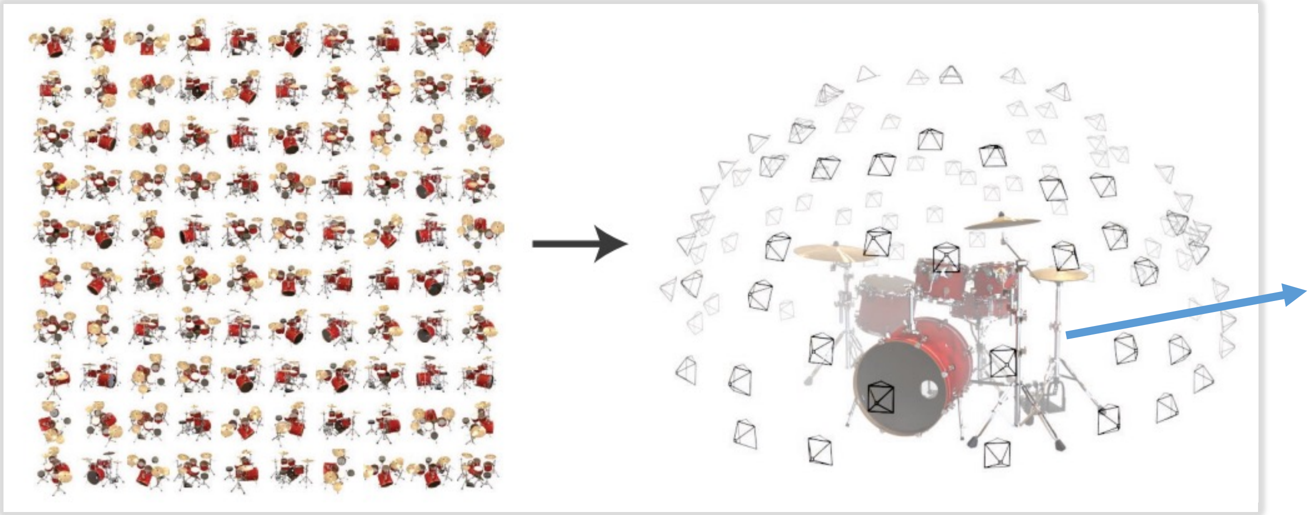
Geospatial Data
[Blumenstock et al. 2015]

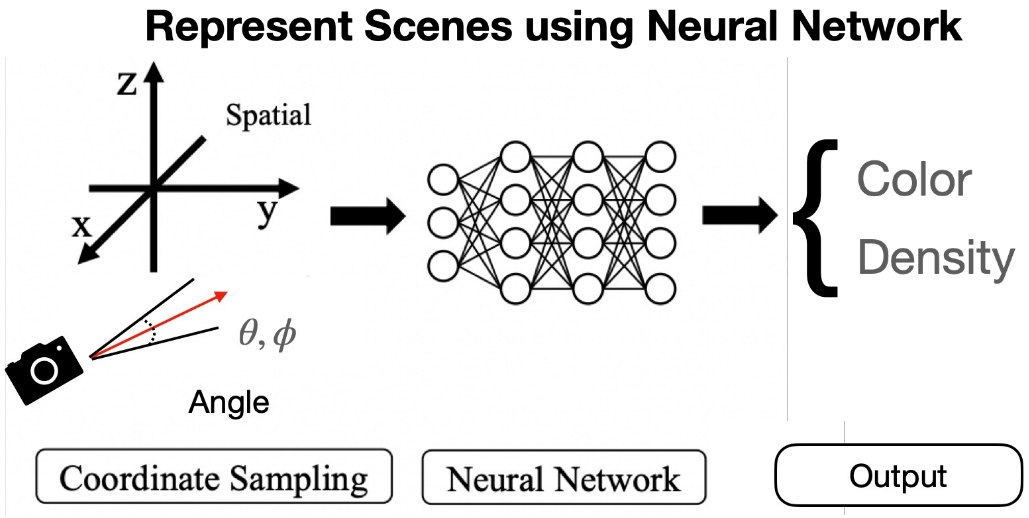
Neural Radiance Fields (NeRF)



2003.08934

Neural Radiance Fields (NeRF)



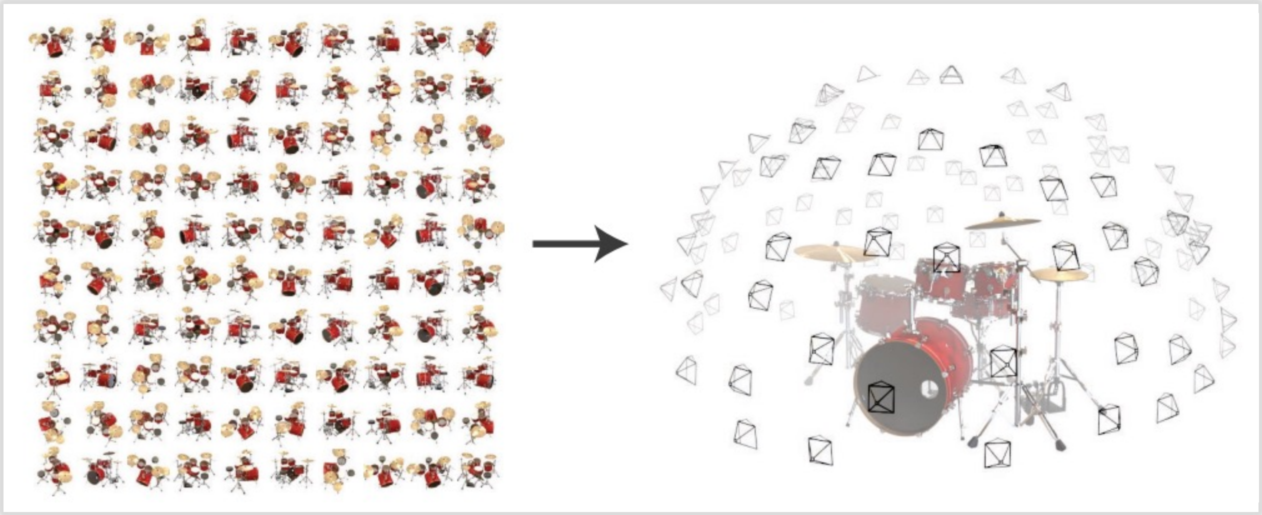


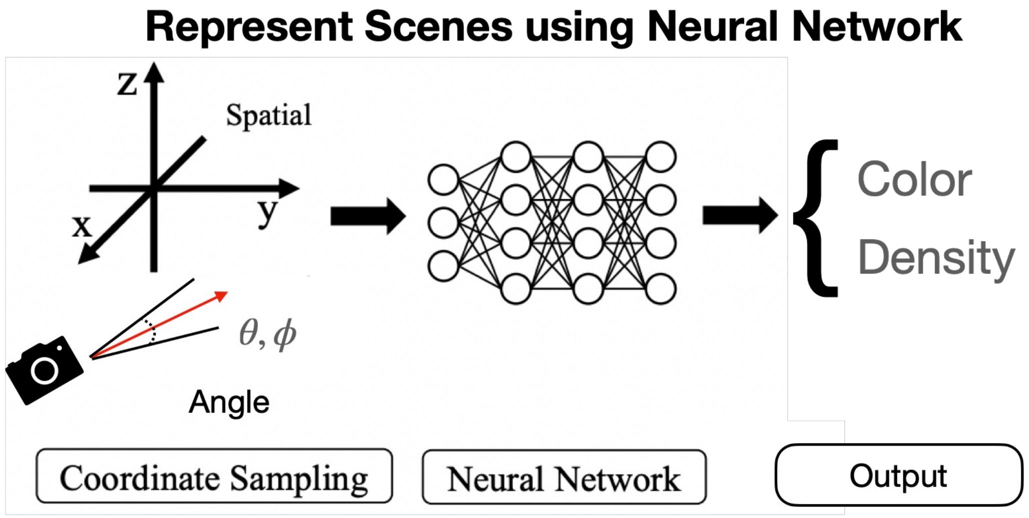
Model 3D Object density and color using neural network

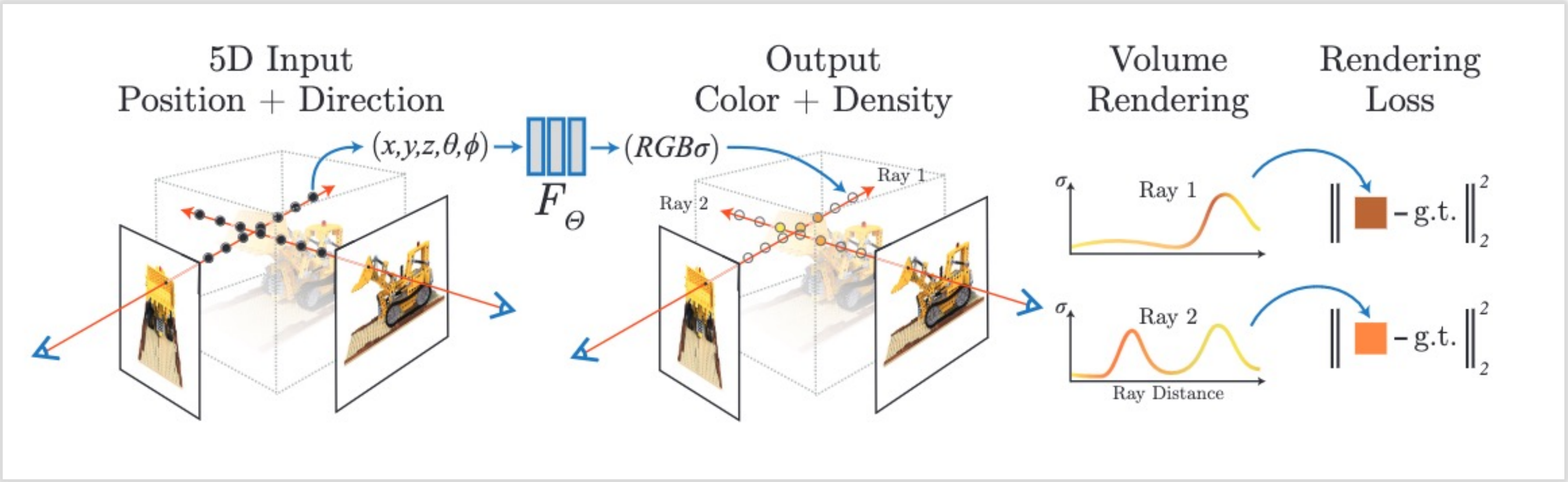
Input: position → neural network → *Output:* density & color

2003.08934

Neural Radiance Fields (NeRF)





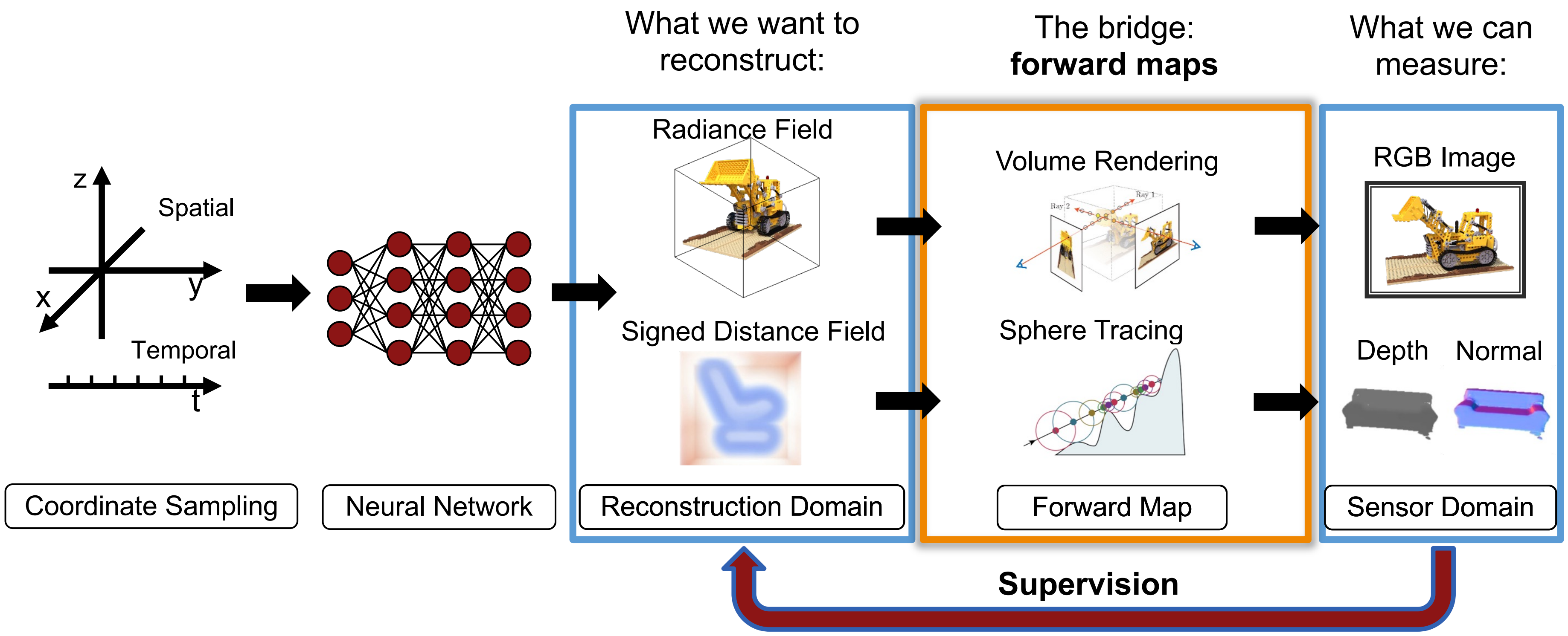


[2003.08934](https://arxiv.org/abs/2003.08934)

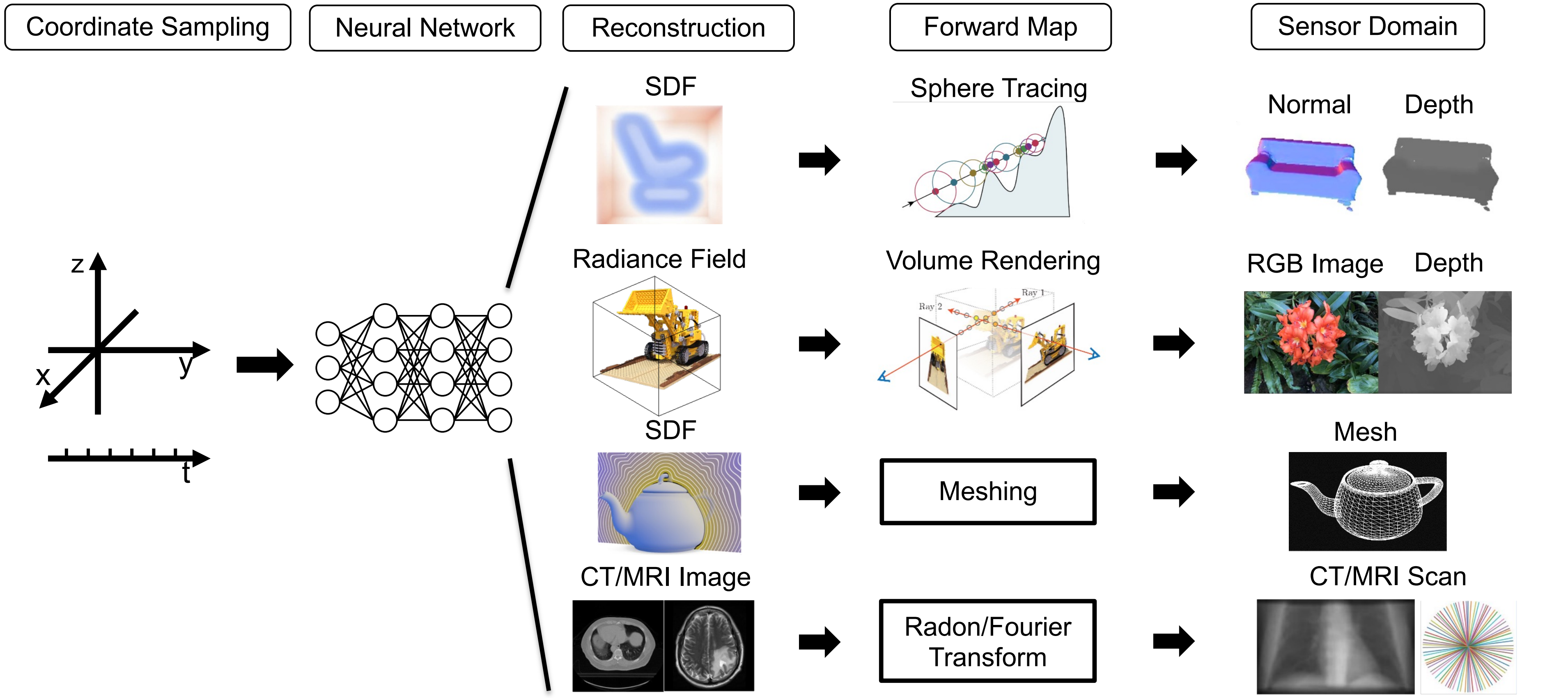
Neural Radiance Fields (NeRF)

2003.08934

Neural Field General Framework

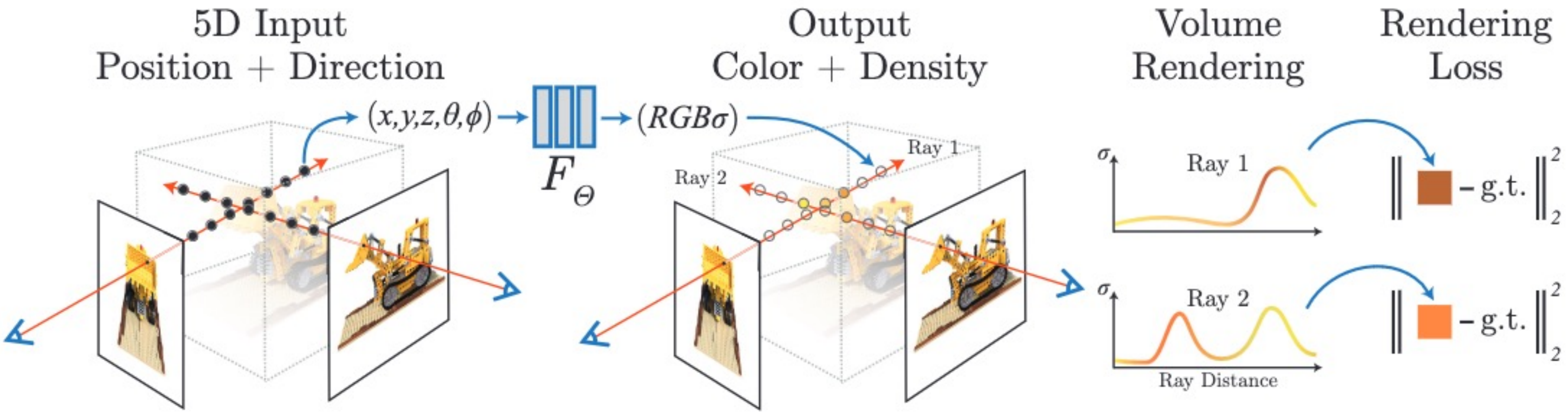


Many Applications



Figures adapted from:
Mildenhall et al. 2020 (NeRF)
Shen et al. 2021 (NeRP)

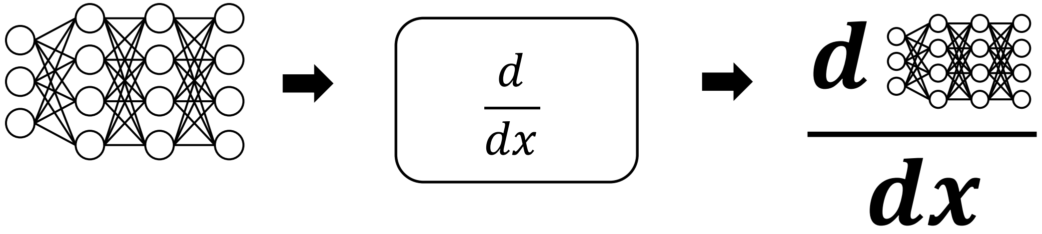
Forward Map: Differentiable Rendering



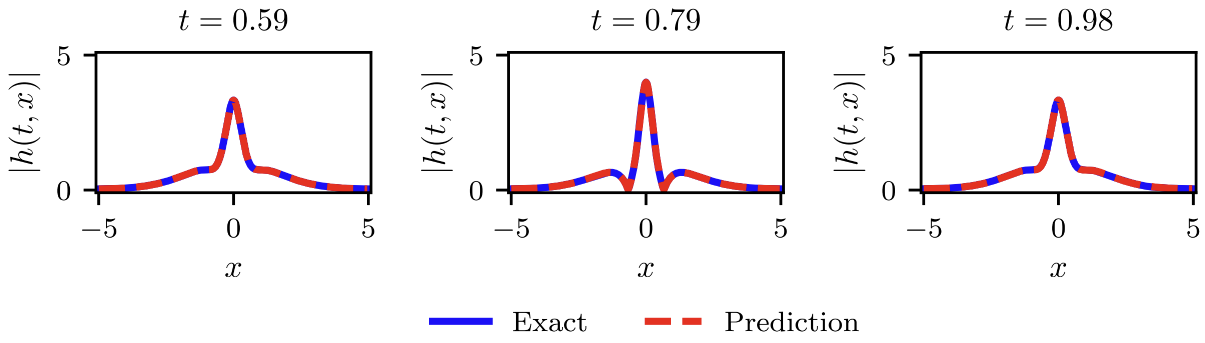
$$C(\mathbf{r}) = \int_{t_n}^{t_f} T(t)\sigma(\mathbf{r}(t))\mathbf{c}(\mathbf{r}(t), \mathbf{d})dt\,,\ \text{where } T(t) = \exp\left(-\int_{t_n}^{t} \sigma(\mathbf{r}(s))ds\right)$$

2003.08934

Neural Fields + Differential Equations: Physics-Informed Neural Networks (PINN)

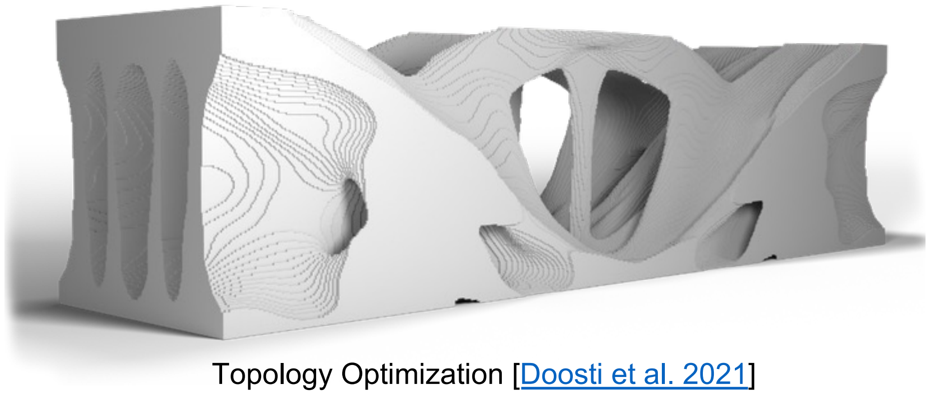


Schrödinger's Equation $$-\frac{\hbar^2}{2m}\nabla^2\psi + V\psi = E\psi$$

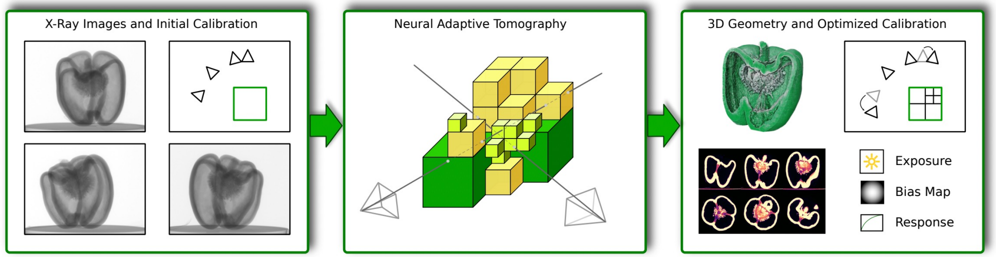


[Raissi et al. 2019 (PINN)]

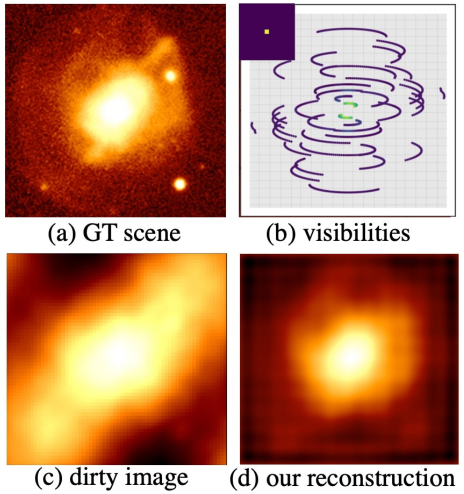
Neural Fields in Science and Engineering



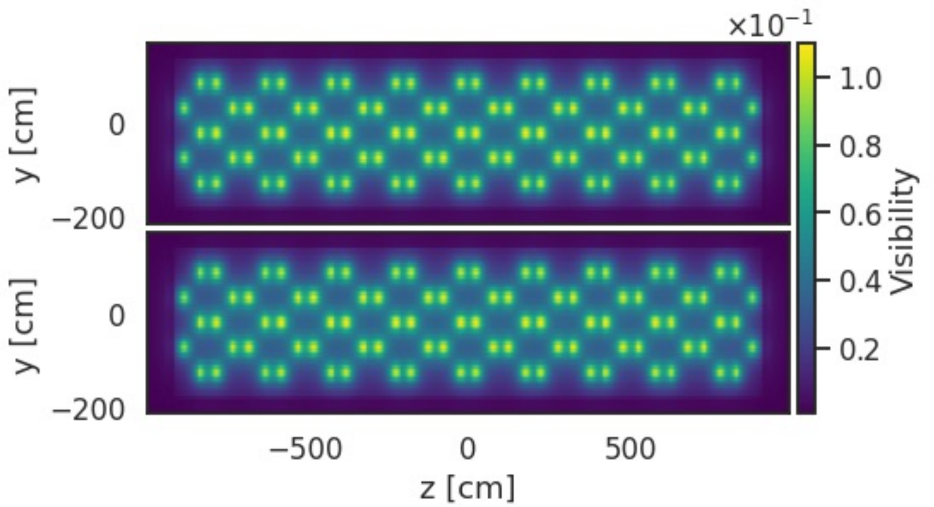
Topology Optimization [Doosti et al. 2021]



Tomographic Reconstruction [Ruckert et al. 2022]



Astronomical Interferometry [Wu et al. 2021]



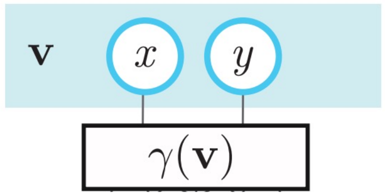
Neutrino Detectors 2211.01505

Model Structures

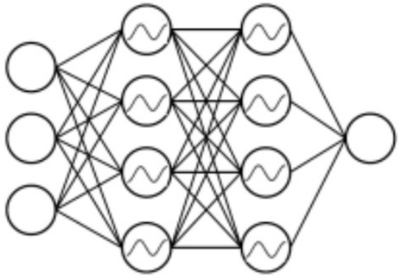
Many of the neural networks in are relatively simple MLPs, well placed within the synthesis pipeline.

A few novel(-ish) features, especially for modeling spatio-temporal data

Input Encoding



Activation Functions



Spectral Bias

On the Spectral Bias of Neural Networks

Nasim Rahaman [*1 2] Aristide Baratin [*1] Devansh Arpit [1] Felix Draxler [2] Min Lin [1] Fred A. Hamprecht [2]
Yoshua Bengio [1] Aaron Courville [1]

Abstract

Neural networks are known to be a class of highly expressive functions able to fit even random input-output mappings with 100% accuracy. In this work we present properties of neural networks that complement this aspect of expressivity. By using tools from Fourier analysis, we highlight a learning bias of deep networks towards low frequency functions – i.e. functions that vary globally without local fluctuations – which manifests itself as a frequency-dependent learning speed. Intuitively, this property is in line with the observation that over-parameterized networks prioritize learning simple patterns that generalize across data samples. We also investigate the role of the shape of the data manifold by presenting empirical and theoretical evidence that, somewhat counter-intuitively, learning higher frequencies gets *easier* with increasing manifold complexity.

1. Introduction

The remarkable success of deep neural networks at generalizing to natural data is at odds with the traditional notions of model complexity and their empirically demonstrated ability to fit arbitrary random data to perfect accuracy (Zhang et al., 2017a; Arpit et al., 2017). This has prompted recent investigations of possible implicit regularization mechanisms inherent in the learning process which induce a bias towards low complexity solutions (Neyshabur et al., 2014; Soudry et al., 2017; Poggio et al., 2018; Neyshabur et al., 2017).

In this work, we take a slightly shifted view on implicit regularization by suggesting that low-complexity functions are *learned faster* during training by gradient descent. We expose this bias by taking a closer look at neural networks through the lens of Fourier analysis. While they can approximate arbitrary functions, we find that these networks prioritize learning the low frequency modes, a phenomenon we call the *spectral bias*. This bias manifests itself not just in the process of learning, but also in the parameterization of the model itself: in fact, we show that the lower frequency components of trained networks are more robust to random parameter perturbations. Finally, we also expose and analyze the rather intricate interplay between the spectral bias and the geometry of the data manifold by showing that high frequencies get easier to learn when the data lies on a lower-dimensional manifold of complex shape embedded in the input space of the model. We focus the discussion on networks with rectified linear unit (ReLU) activations, whose continuous piece-wise linear structure enables an analytic treatment.

*Equal contribution [1]Mila, Quebec, Canada [2]Image Analysis and Learning Lab, Ruprecht-Karls-Universität Heidelberg, Germany. Correspondence to: Nasim Rahaman <nasim.rahaman@live.com>, Aristide Baratin <aristide.baratin@umontreal.ca>, Devansh Arpit <devansharpit@gmail.com>.

Proceedings of the 36th International Conference on Machine Learning, Long Beach, California, PMLR 97, 2019. Copyright 2019 by the author(s).

Contributions[1]

1. We exploit the continuous piecewise-linear structure of ReLU networks to evaluate its Fourier spectrum (Section 2).
2. We find empirical evidence of a *spectral bias*: i.e. lower frequencies are learned first. We also show that lower frequencies are more robust to random perturbations of the network parameters (Section 3).
3. We study the role of the shape of the data manifold: we show how complex manifold shapes can facilitate the learning of higher frequencies and develop a theoretical understanding of this behavior (Section 4).

2. Fourier analysis of ReLU networks

2.1. Preliminaries

Throughout the paper we call 'ReLU network' a scalar function $f : \mathbb{R}^d \mapsto \mathbb{R}$ defined by a neural network with L hidden layers of widths $d_1, \cdots d_L$ and a single output neuron:

$$f(\mathbf{x}) = \left(T^{(L+1)} \circ \sigma \circ T^{(L)} \circ \cdots \circ \sigma \circ T^{(1)}\right)(\mathbf{x}) \quad (1)$$

[1]Code: https://github.com/nasimrahaman/SpectralBias

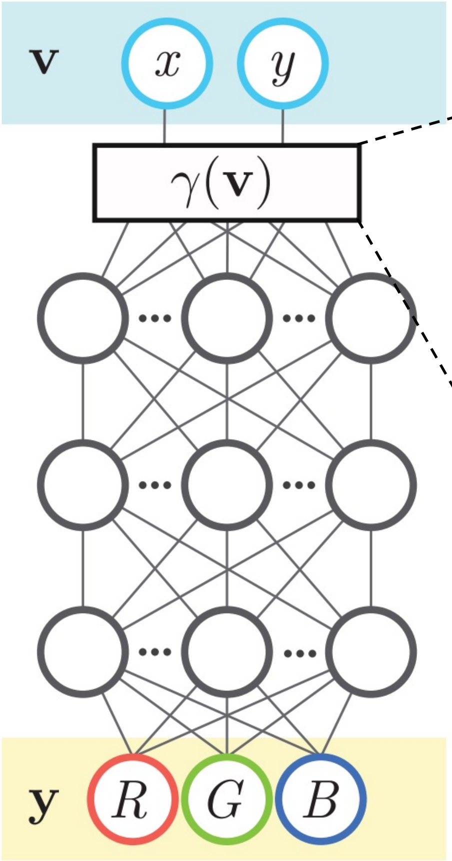
"Neural networks are biased to fit lower frequency signals"

[Baatz et al. 2021]

That's a problem if we want to learn fine details!

Positional Encoding

Map input value x to a vector of sine & cosine values at different frequencies



Positional Encodings

$$\gamma(\mathbf{x}) = [\gamma_1(\mathbf{x}), \gamma_2(\mathbf{x}), \ldots, \gamma_m(\mathbf{x})]$$

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

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

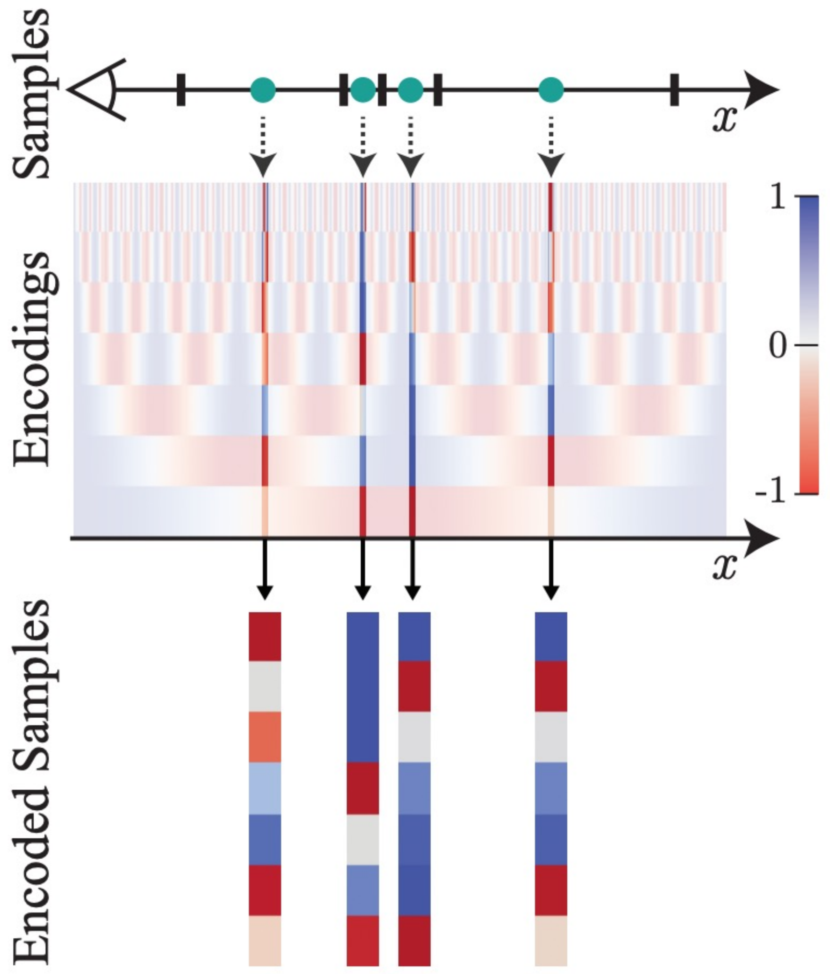
Originally introduced in the context of transformers

1706.03762

Positional Encoding

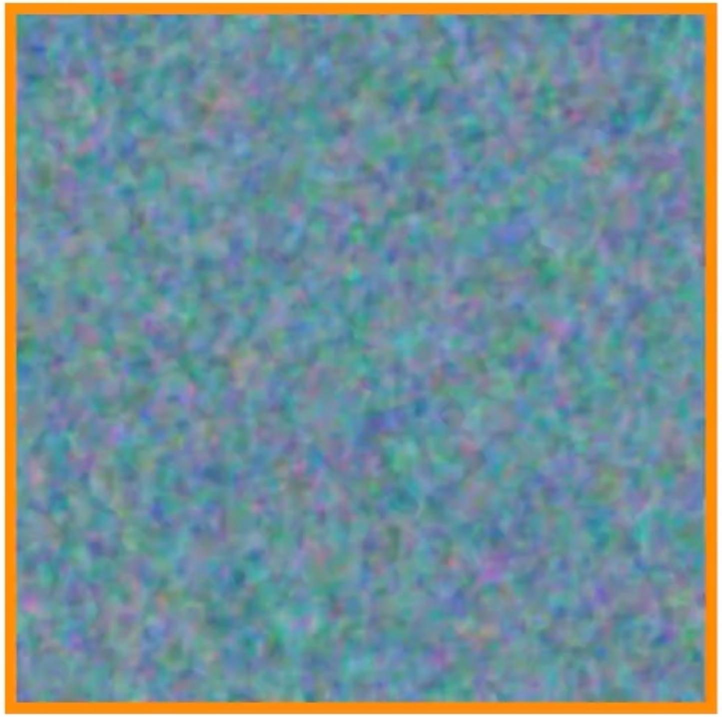
Map input value x to a vector of sine & cosine values at different frequencies

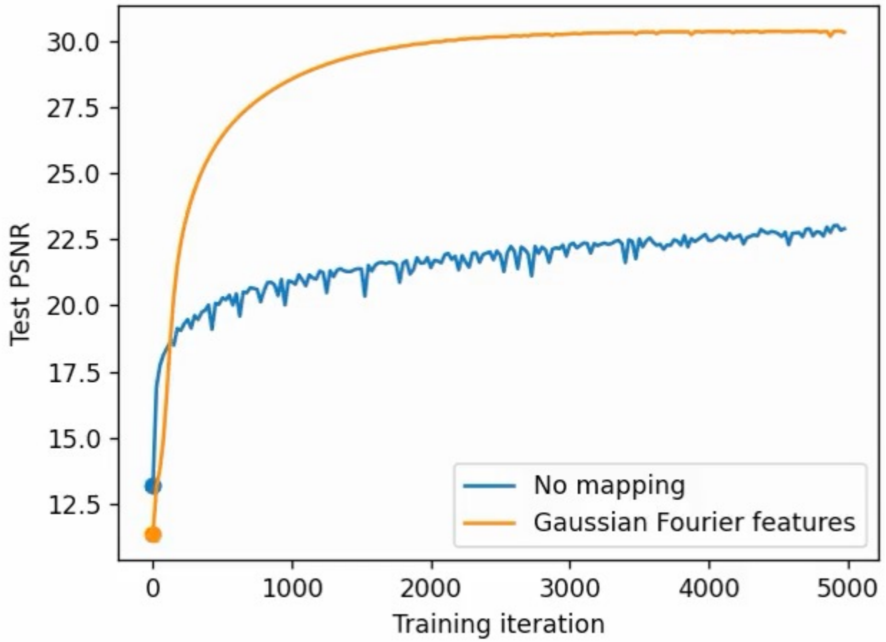
$$\gamma(\boldsymbol{x}) = [\sin(\boldsymbol{x}), \cos(\boldsymbol{x}), \ldots, \sin(2^{L-1}\boldsymbol{x}), \cos(2^{L-1}\boldsymbol{x})]^T$$



2103.13415

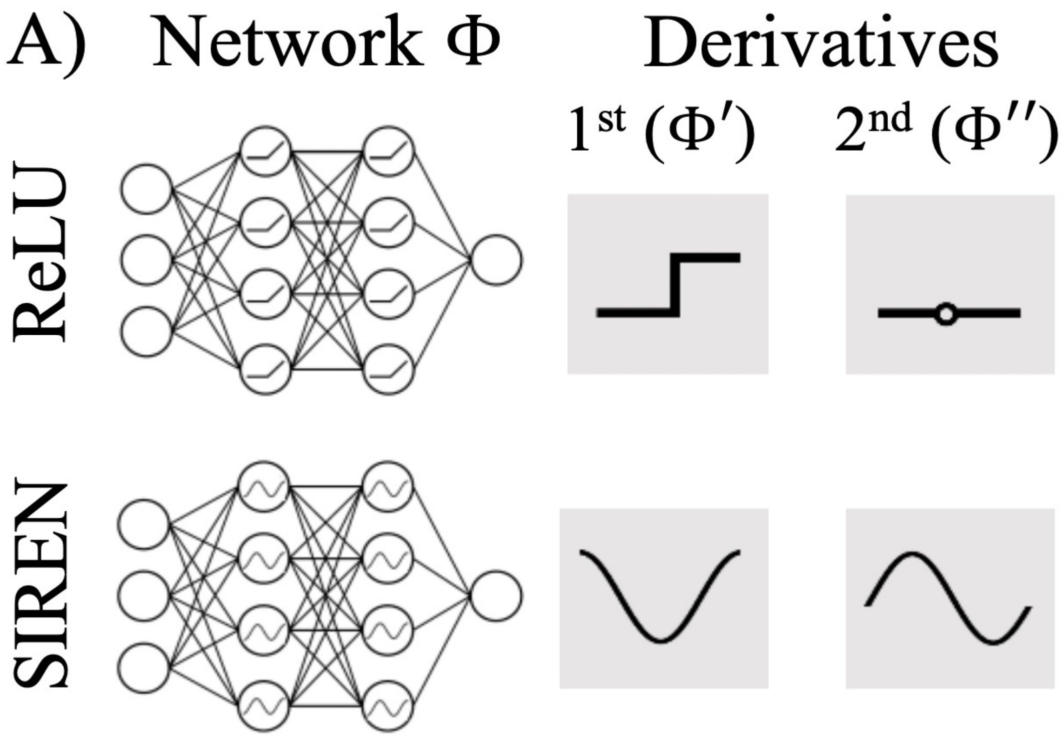
Positional Encoding In Action



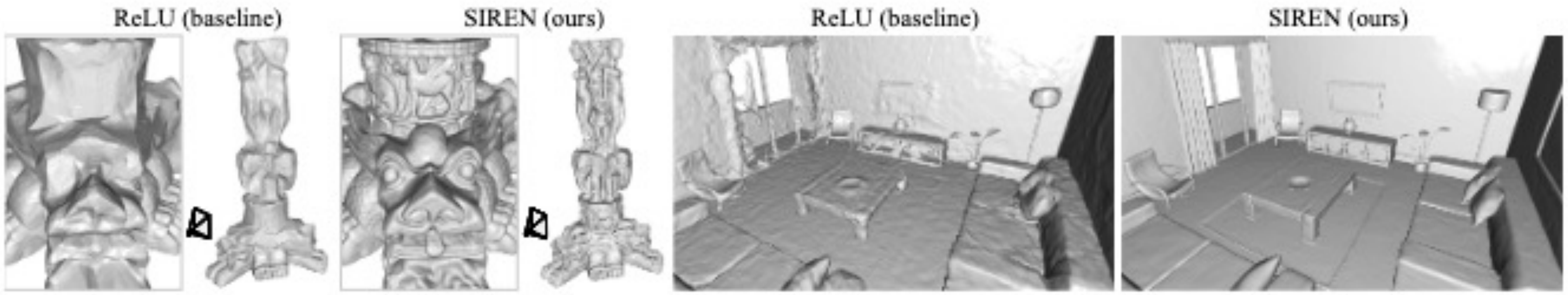


2006.10739

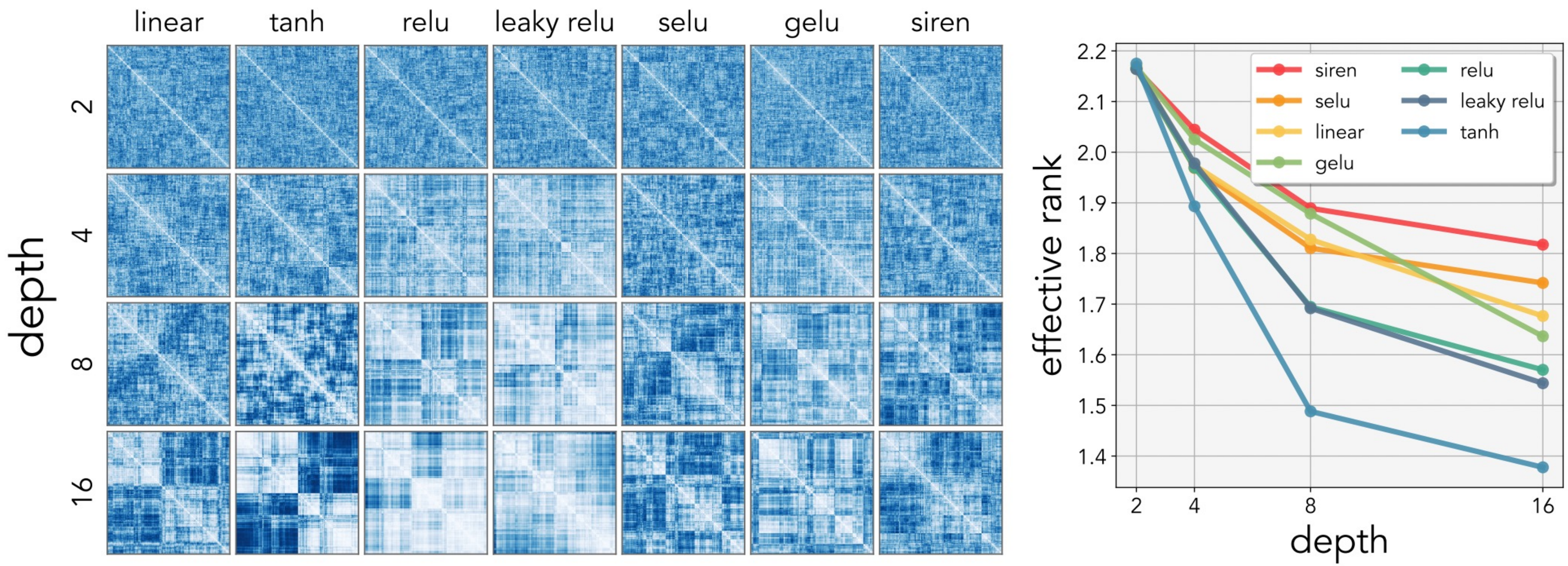
Activation Functions



Activation Functions



2006.09661



2103.10427

A few points on why neural fields?

Compact:

Can efficiently summarize signals without dense grids

Combining measurements lets us surpass resolution of any one measurement

Regularization:

Explicit regularization: in the form of priors: $\min_{\theta} \left| F\left(NN_{\theta}(x)\right) - I \right|^2 + \Omega(NN(\theta))$

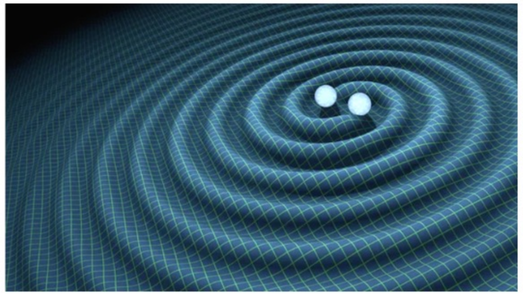
- E.g. total variation, constraining model smoothness $\nabla_x NN_{\theta}(x)$

Implicit regularization: NN imposes implicit priors in the form of architecture choice which determines the set of functions we can fit to data and how we interpolate

Application for MAGIS-100 Experiment

Atomic Sensors

Gravitational waves cause a small modulation in the distance between objects



$$L\,(1 + h\sin(wt))$$

strain — frequency

In MAGIS, atoms play two roles

Inertial References

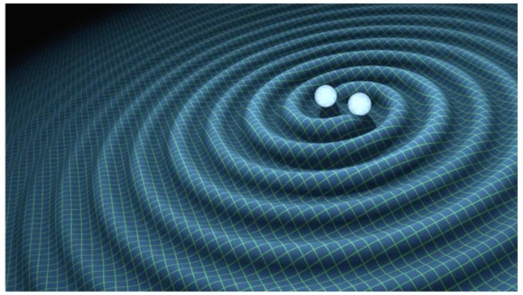
- Freely falling objects, separated by a large baseline
- Must be insensitive to perturbations from non-gravitational forces

Clock

- Monitor separation b/w inertial frames
- Measure time for light to cross baseline

Atomic Sensors

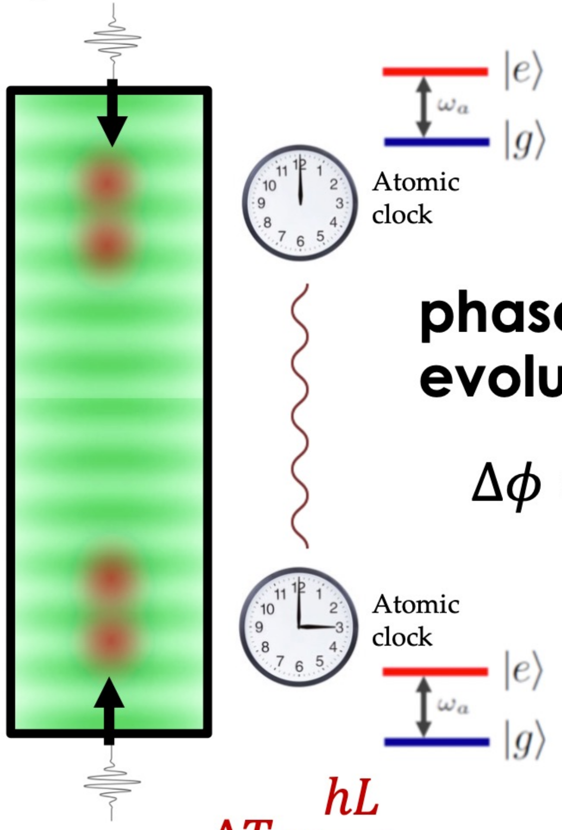
Gravitational waves cause a small modulation in the distance between objects



$$L\,(1 + h\sin(wt))$$

strain frequency

Quantum

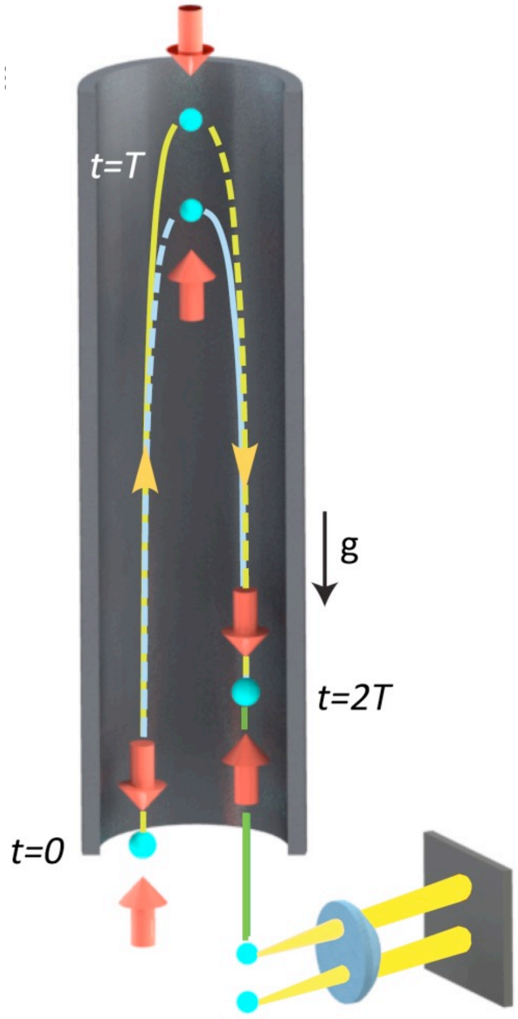


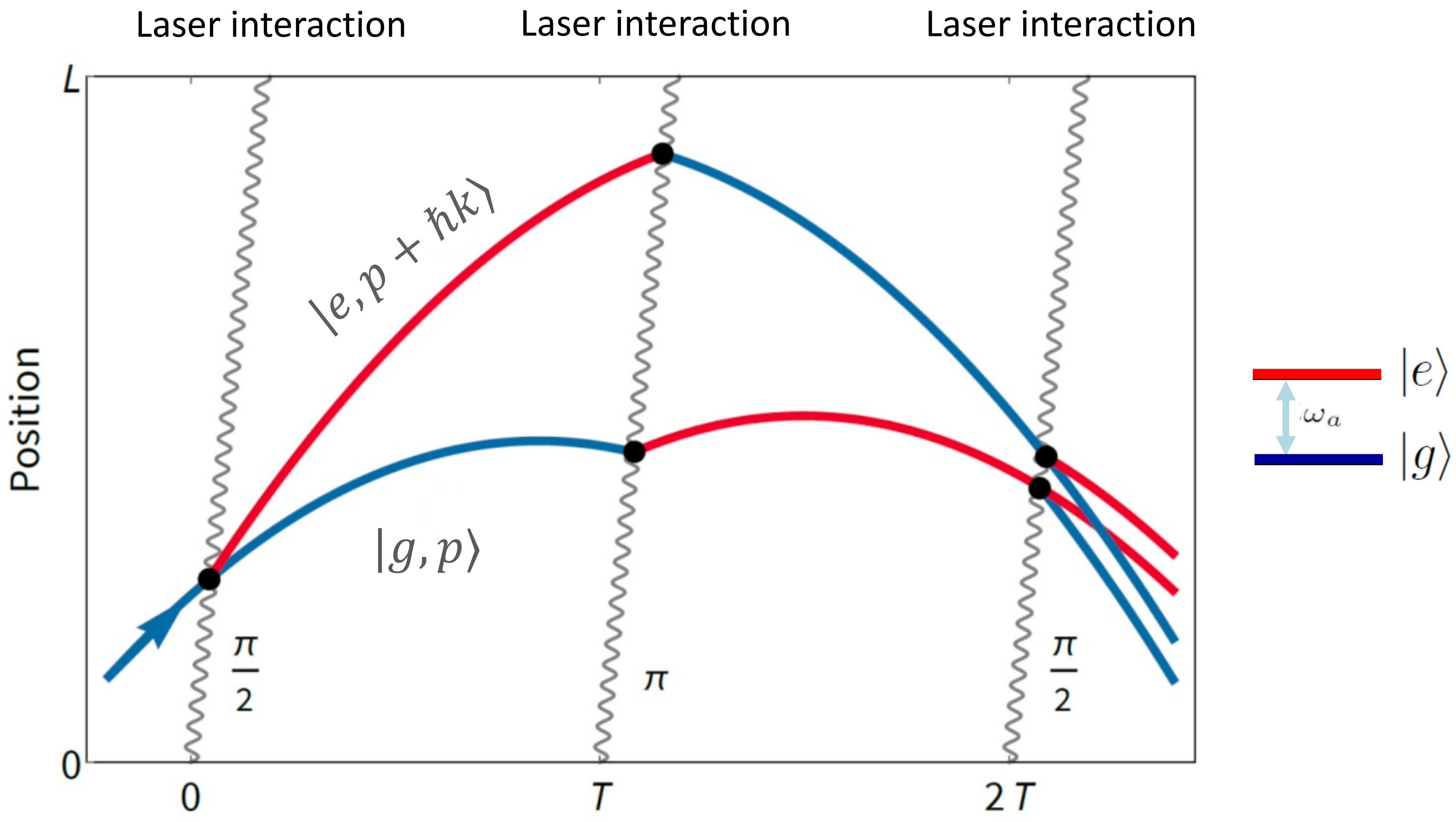
phase evolution:

$$\Delta\phi = w_A\left(\frac{2L}{c}\right)$$

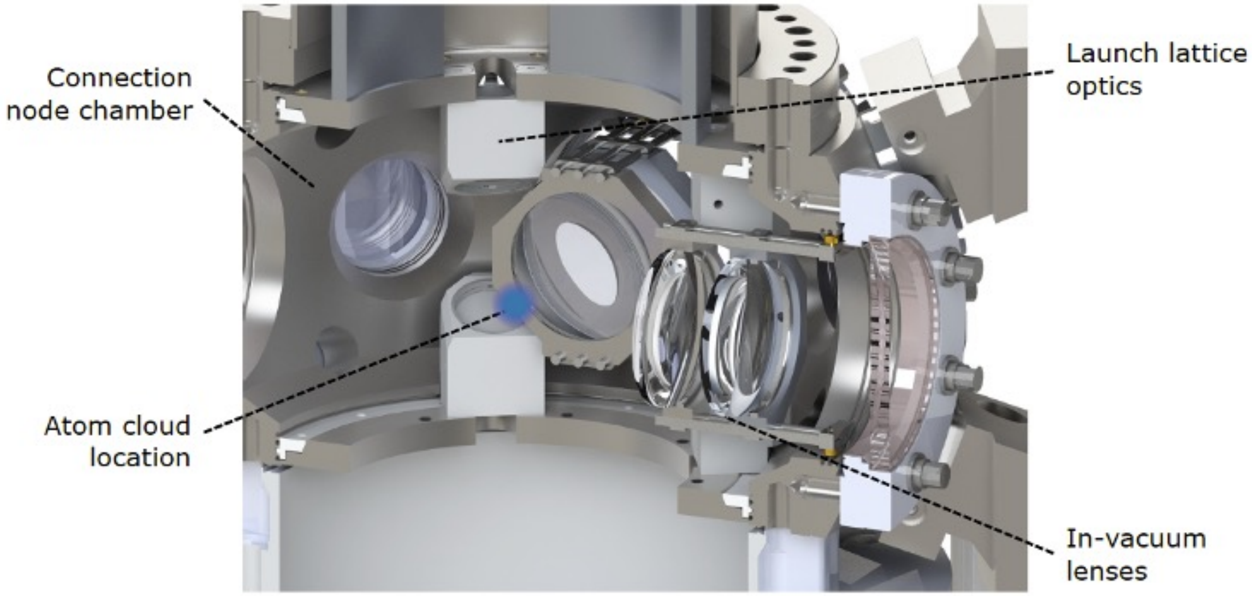
$$\Delta T \sim \frac{hL}{c}$$

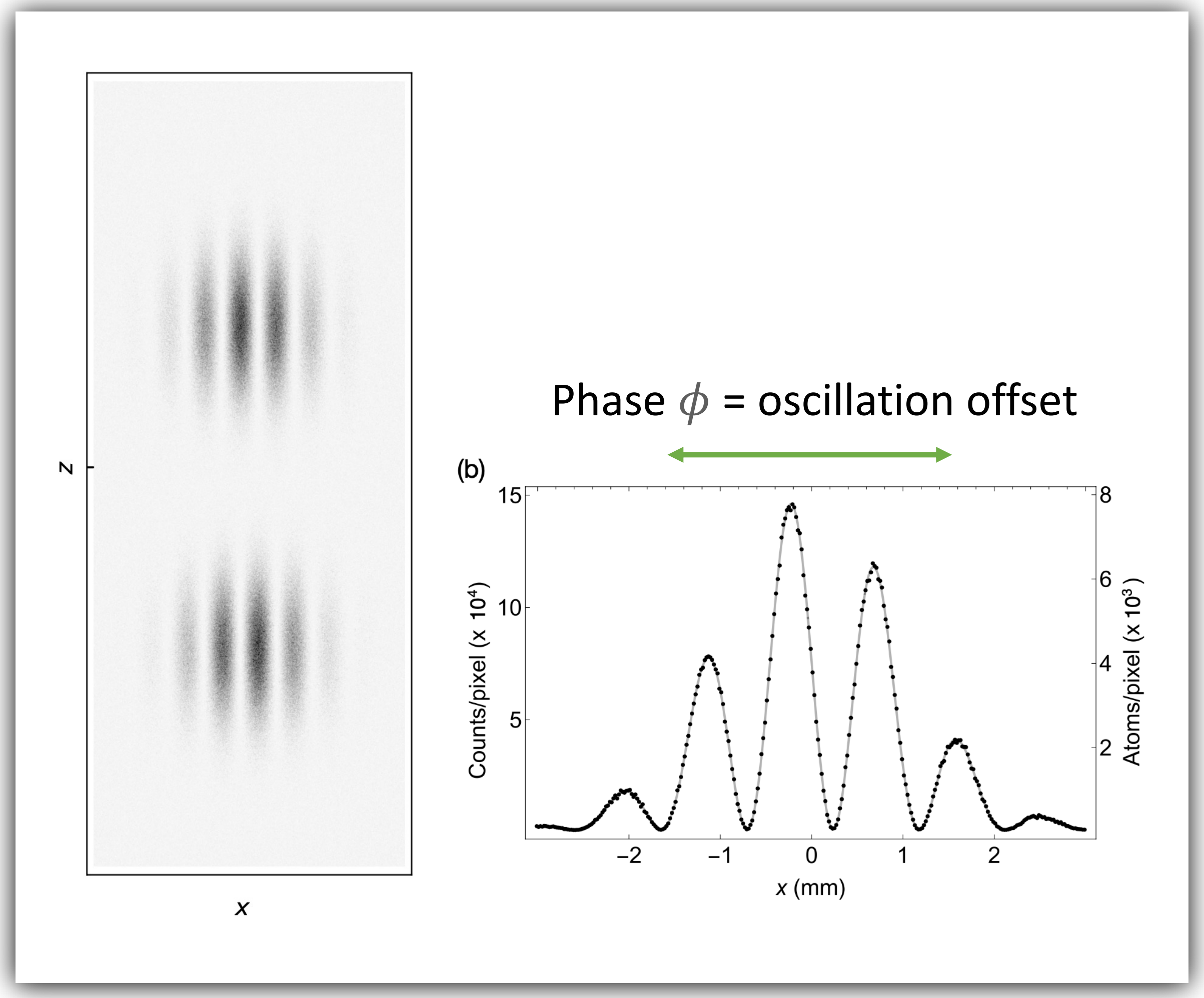
Single Atom Interferometer





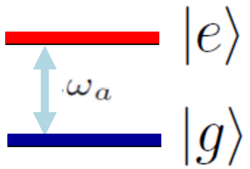
Imagine the Cloud

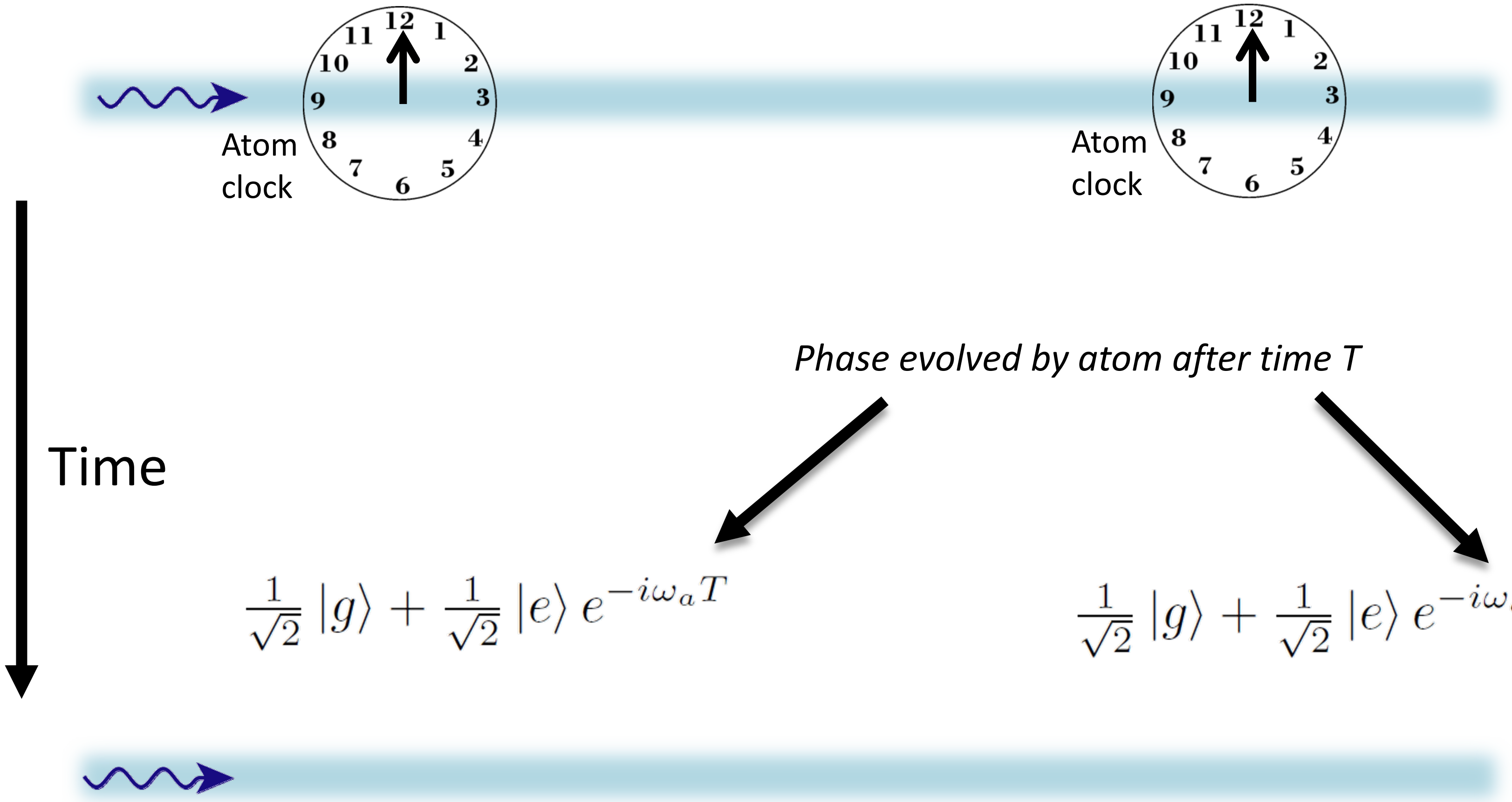




arXiv:2104.02835

Simple Example: Two Atomic Clocks



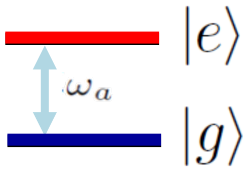


Separated (vertically) Baseline Distance L

$$\frac{1}{\sqrt{2}}|g\rangle + \frac{1}{\sqrt{2}}|e\rangle\, e^{-i\omega_a T}$$

$$\frac{1}{\sqrt{2}}|g\rangle + \frac{1}{\sqrt{2}}|e\rangle\, e^{-i\omega_a T}$$

Simple Example: Two Atomic Clocks

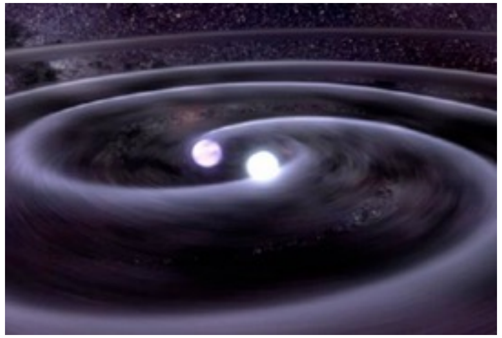


$\frac{1}{\sqrt{2}}|g\rangle + \frac{1}{\sqrt{2}}|e\rangle$

$\frac{1}{\sqrt{2}}|g\rangle + \frac{1}{\sqrt{2}}|e\rangle$

Separated (vertically)
Baseline Distance L

Time

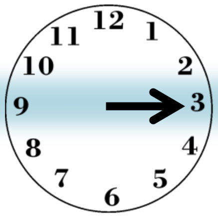


GW changes light travel time

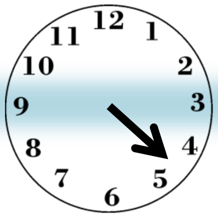
$\Delta T \sim hL/c$

$\frac{1}{\sqrt{2}}|g\rangle + \frac{1}{\sqrt{2}}|e\rangle\, e^{-i\omega_a T}$

$\frac{1}{\sqrt{2}}|g\rangle + \frac{1}{\sqrt{2}}|e\rangle\, e^{-i\omega_a (T+\Delta T)}$

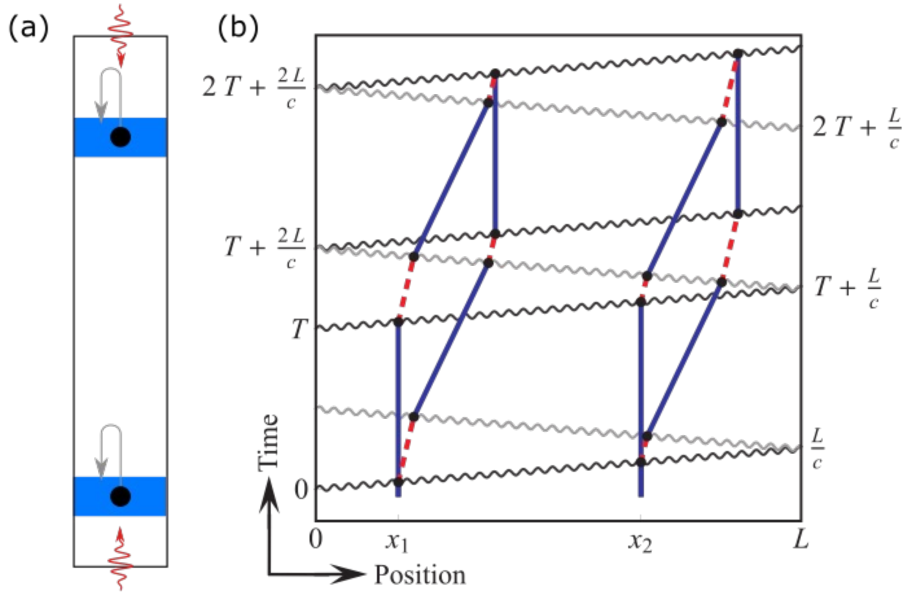


Atom clock



Atom clock

Clock Gradiometer



$|e\rangle$

$|g\rangle$

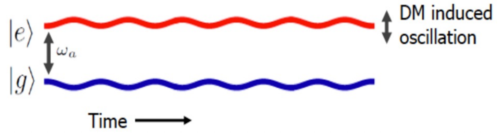
Excited state phase evolution difference:

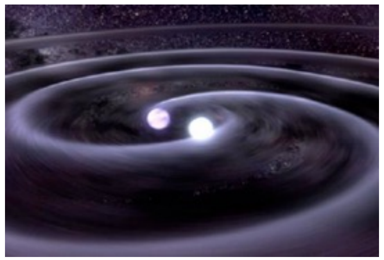
$$\Delta\phi \sim \omega_A\,(2L/c)$$

Two ways for phase to vary:

$\delta\omega_A$	*Dark matter*
$\delta L = hL$	*Gravitational wave*

Ultra-light DM coupling causes time-varying atomic energy levels





$L\,(1 + h\sin(\omega t))$

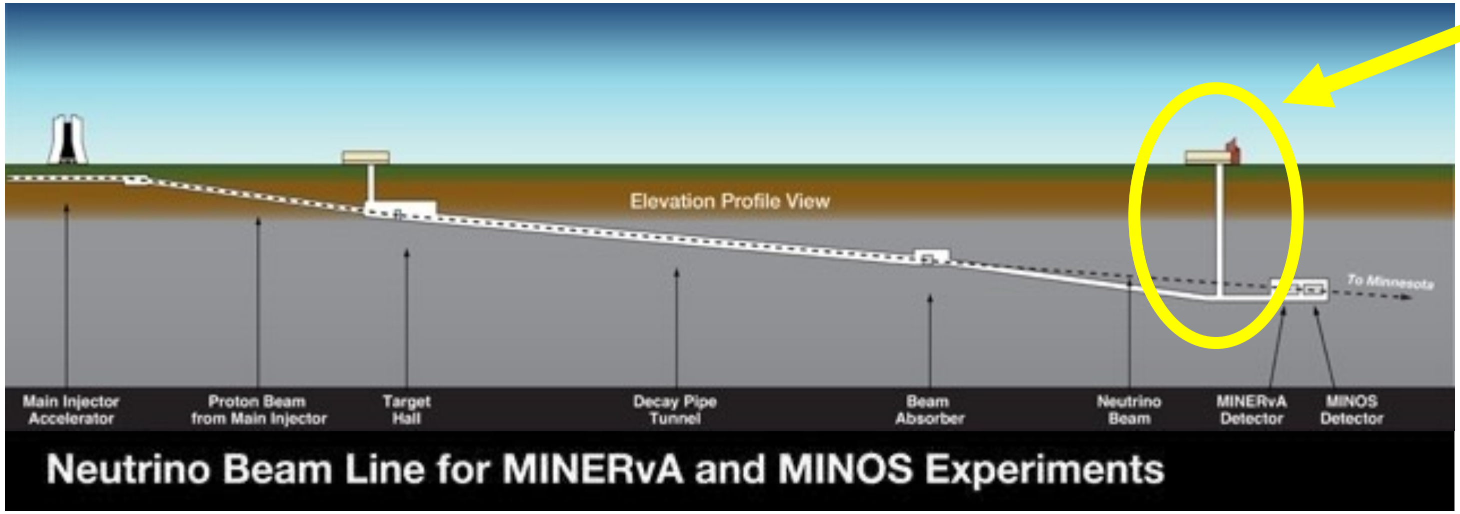
Each interferometer measures the change over time T

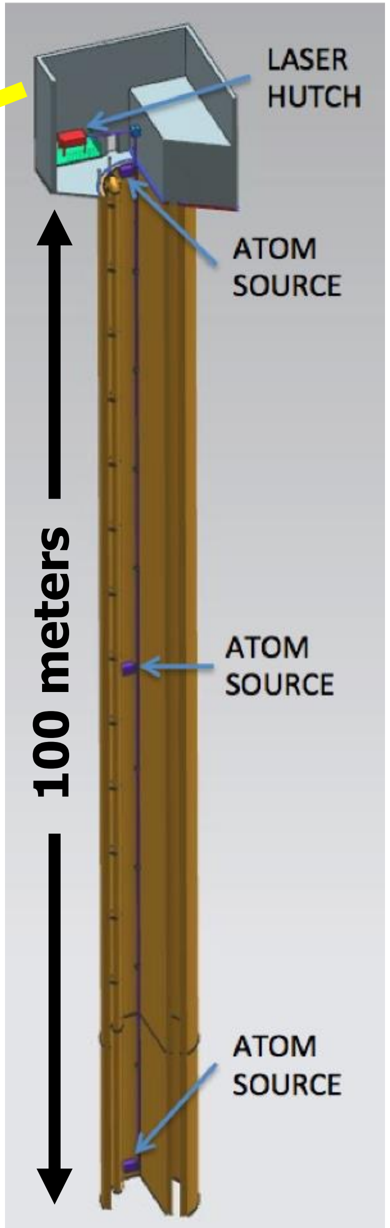
Laser noise is common-mode suppressed in the gradiometer

Graham et al., PRL **110**, 171102 (2013).
Arvanitaki et al., PRD **97**, 075020 (2018).

MAGIS-100

Matter wave **A**tomic **G**radiometer **I**nterferometric **S**ensor



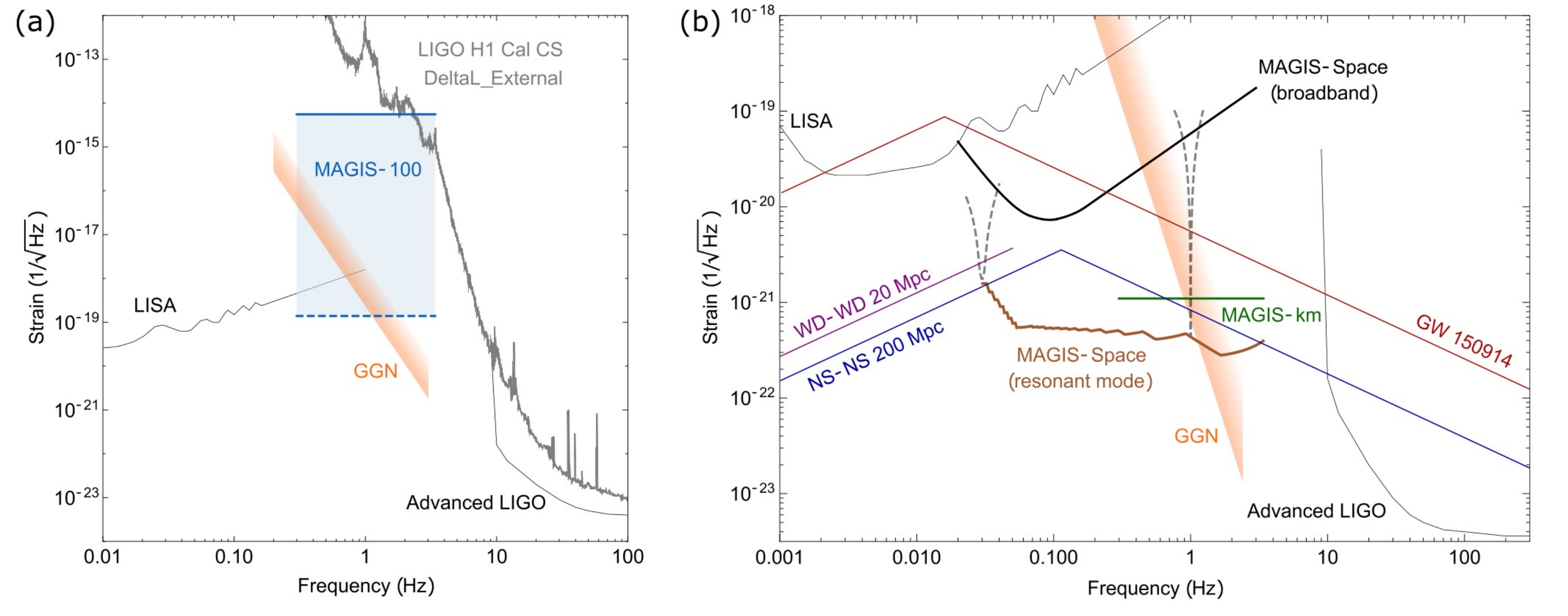


- 100-meter baseline atom interferometry in existing shaft at Fermilab
- Intermediate step to full-scale (km) detector for gravitational waves
- Clock atom sources (Sr) at three positions to realize a gradiometer
- Probes for ultralight scalar dark matter beyond current limits (Hz range)
- Extreme quantum superposition states: >meter separation, up to 9 s duration

STANFORD
JOHNS HOPKINS UNIVERSITY
Northwestern University
Fermilab
SLAC
UNIVERSITY OF CAMBRIDGE
Northern Illinois University
UNIVERSITY OF OXFORD
UNIVERSITY OF LIVERPOOL

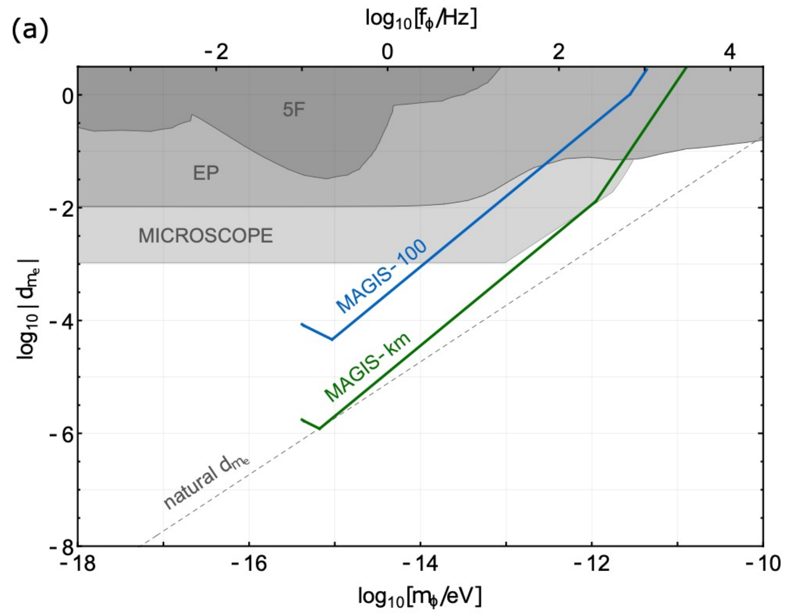
Expected Sensitivity

Future MAGIS experiments



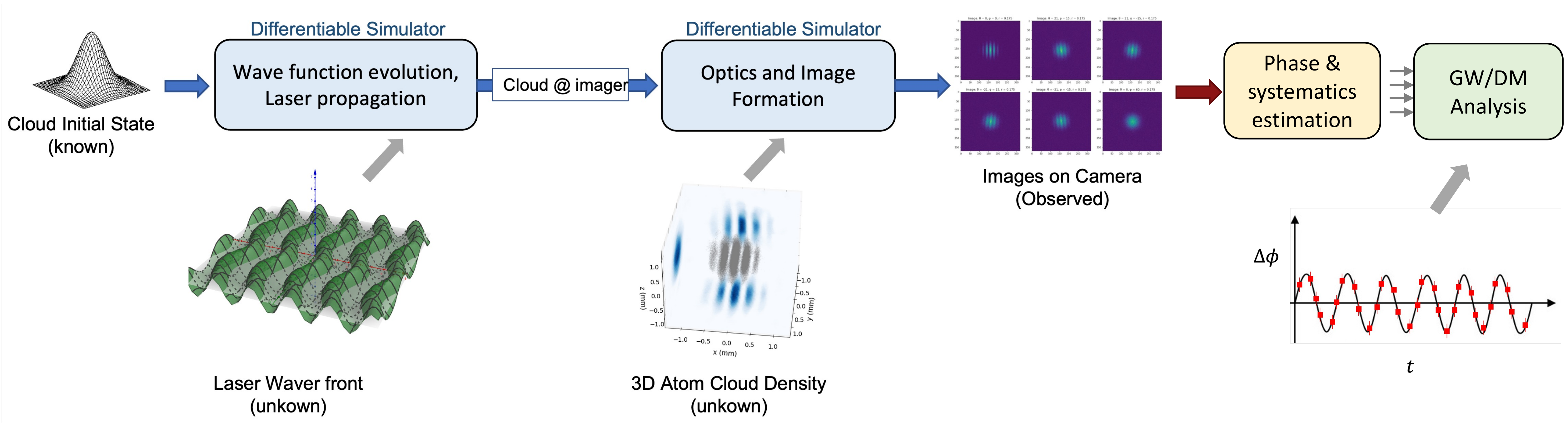
Gravitational Waves

Dark Matter

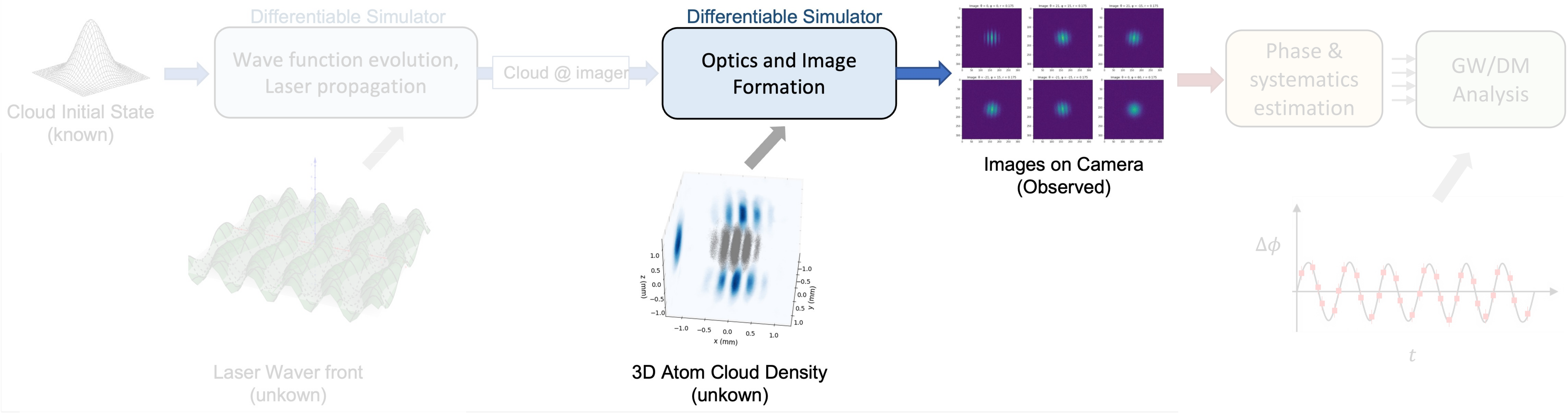


arXiv:2104.02835

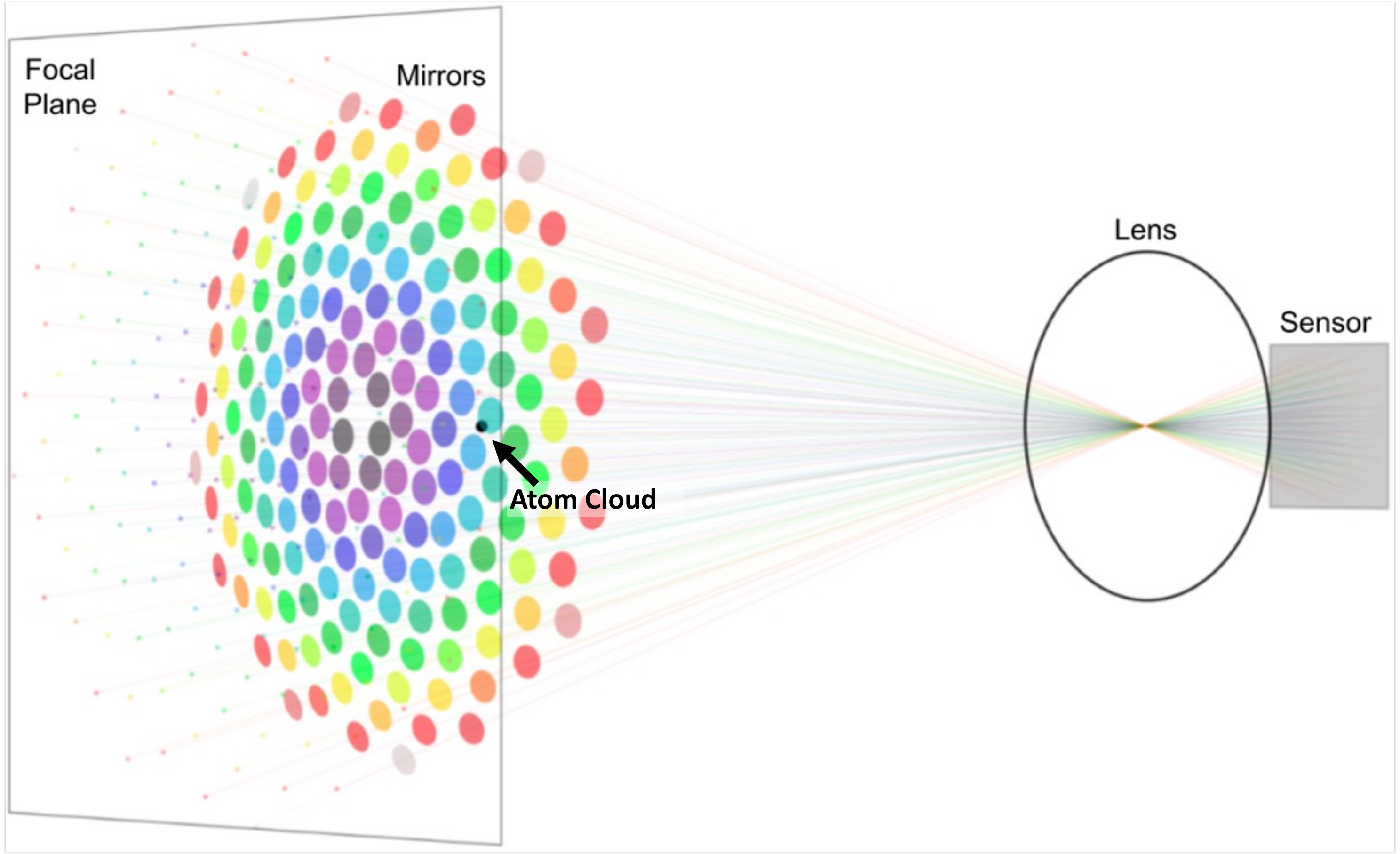
Simulation and Analysis Pipeline

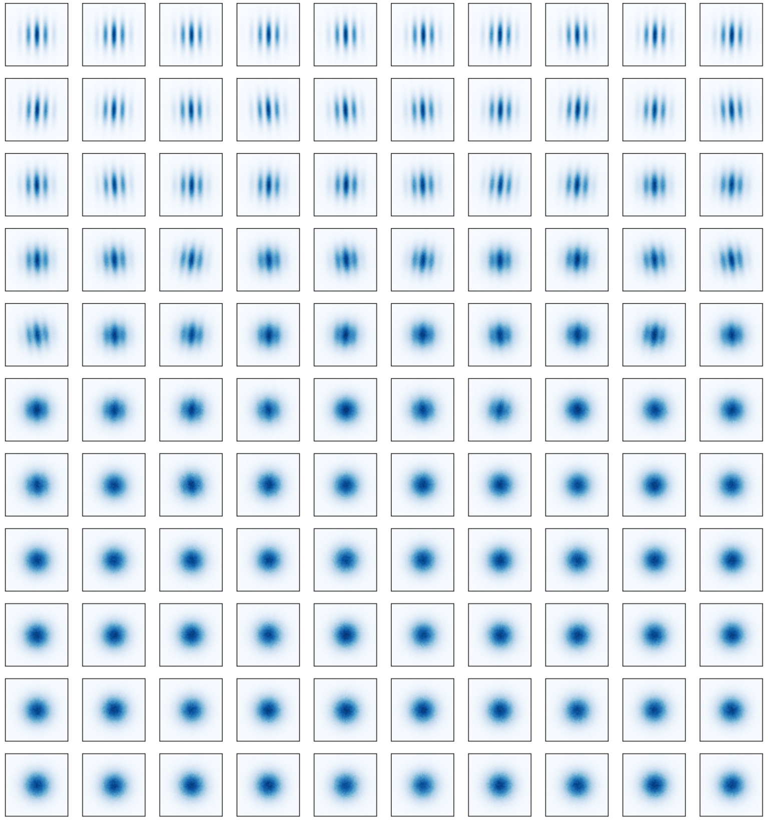


Simulation and Analysis Pipeline



New Single-Shot Multi-View Imaging



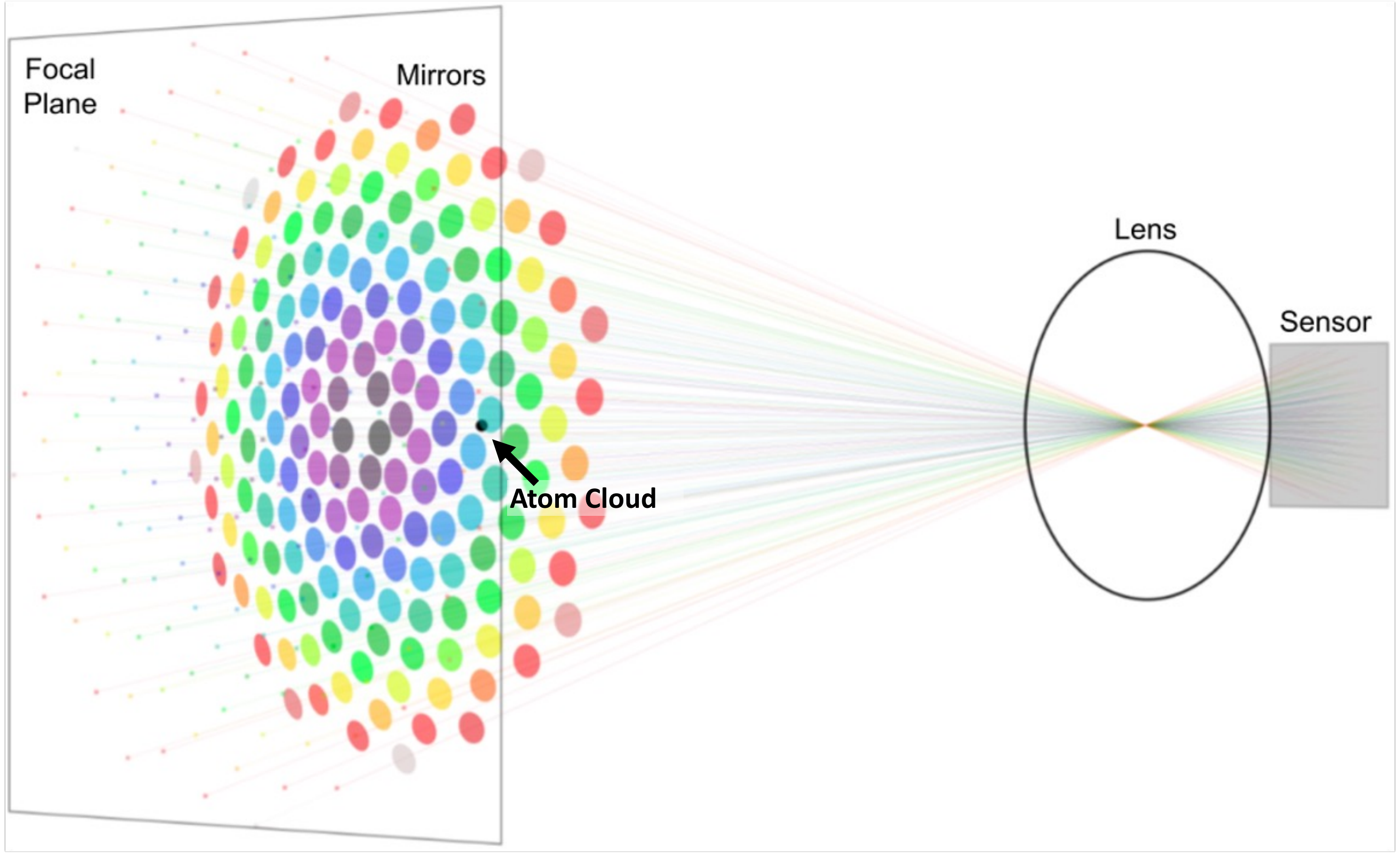


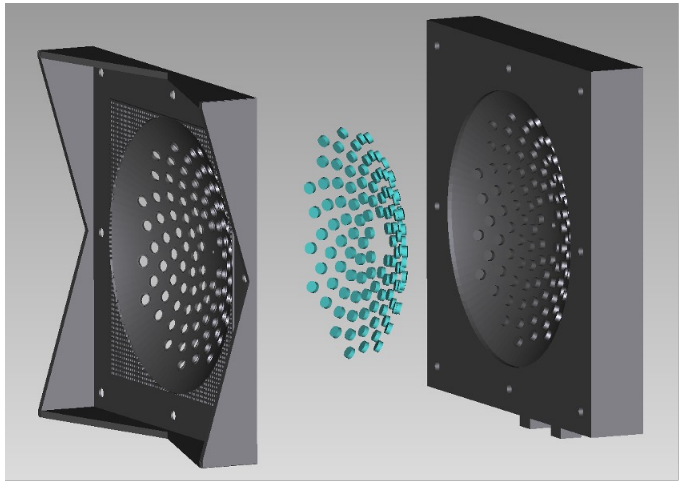
Design Goals

- Increased light detection → improve phase estimation
- Multi-view imaging for 3D reconstruction → improve systematics estimation

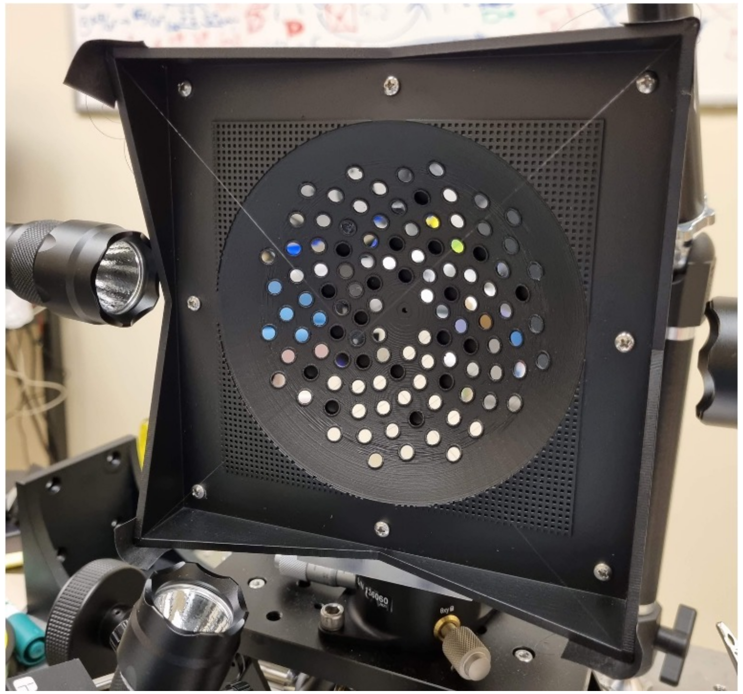
arXiv:2205.11480

New Single-Shot Multi-View Imaging

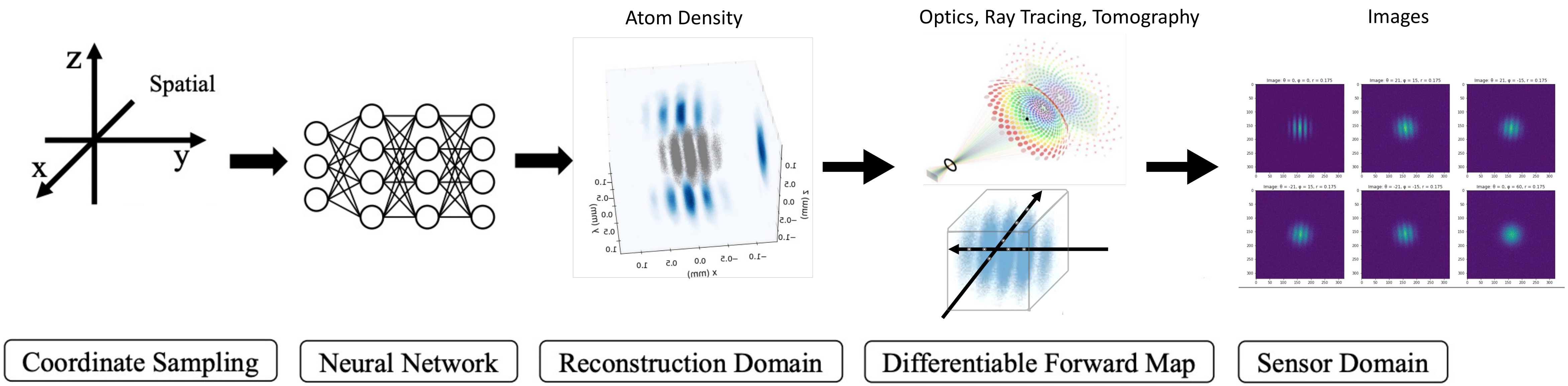




We built it!



Neural Fields for Atom Cloud Reconstruction

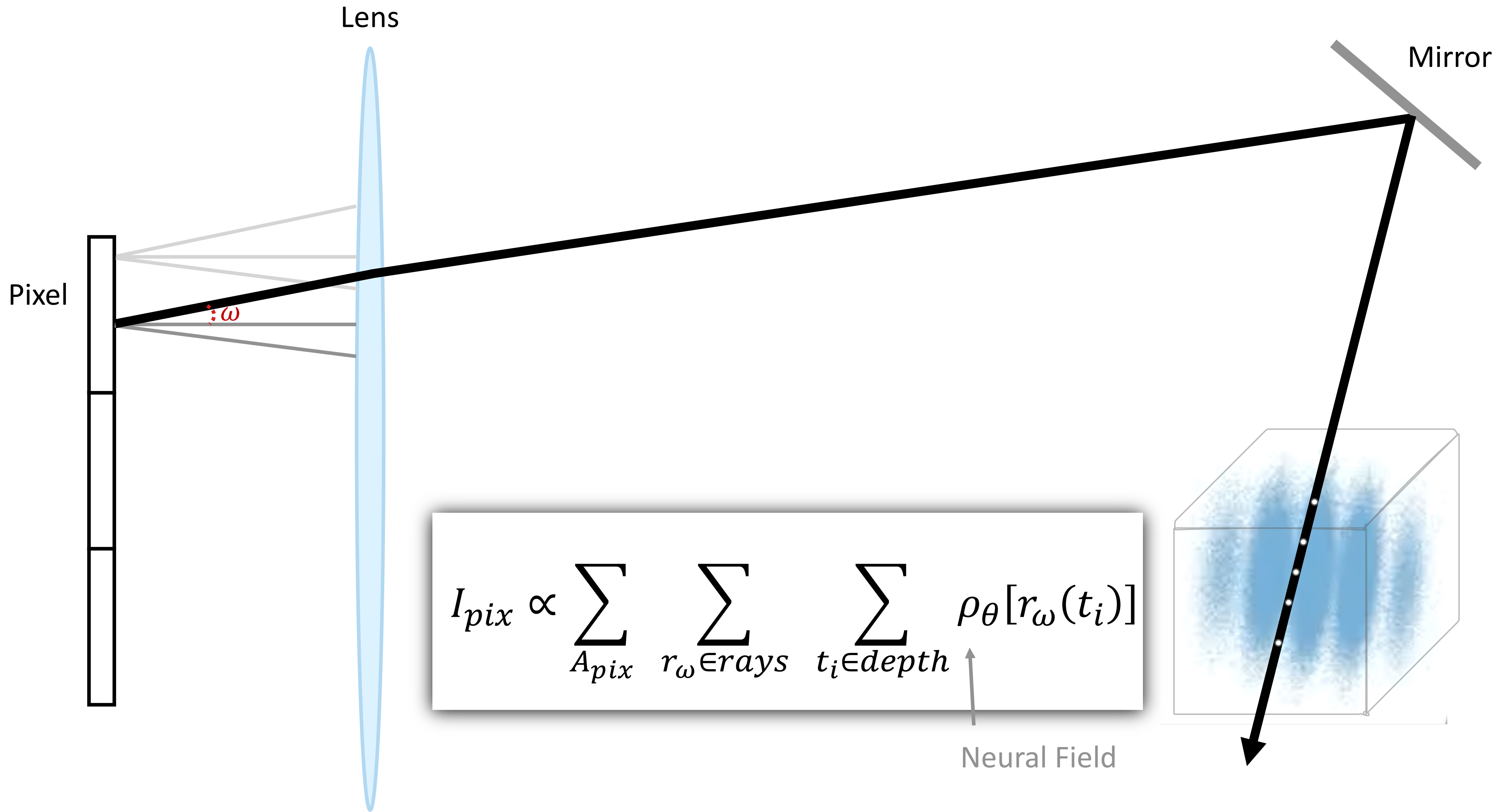


Neural field to model atom cloud density

Forward model: differentiable ray tracing and optics → tomographic imaging

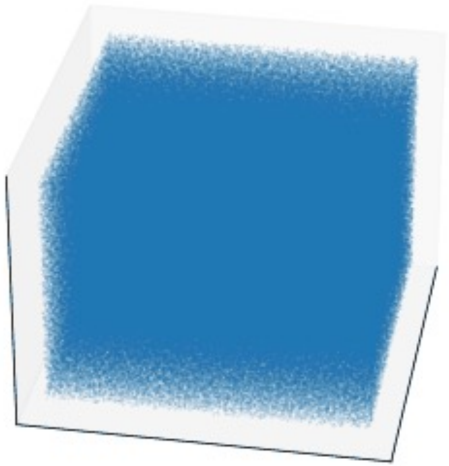
Code: [gradoptics]

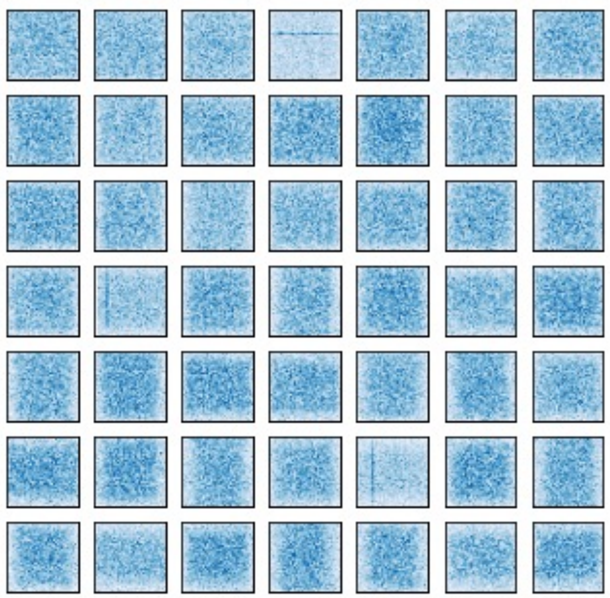
Computing Pixel Intensities



$$I_{pix} \propto \sum_{A_{pix}} \sum_{r_\omega \in rays} \sum_{t_i \in depth} \rho_\theta[r_\omega(t_i)]$$

3D Cloud Reconstruction in Simulation

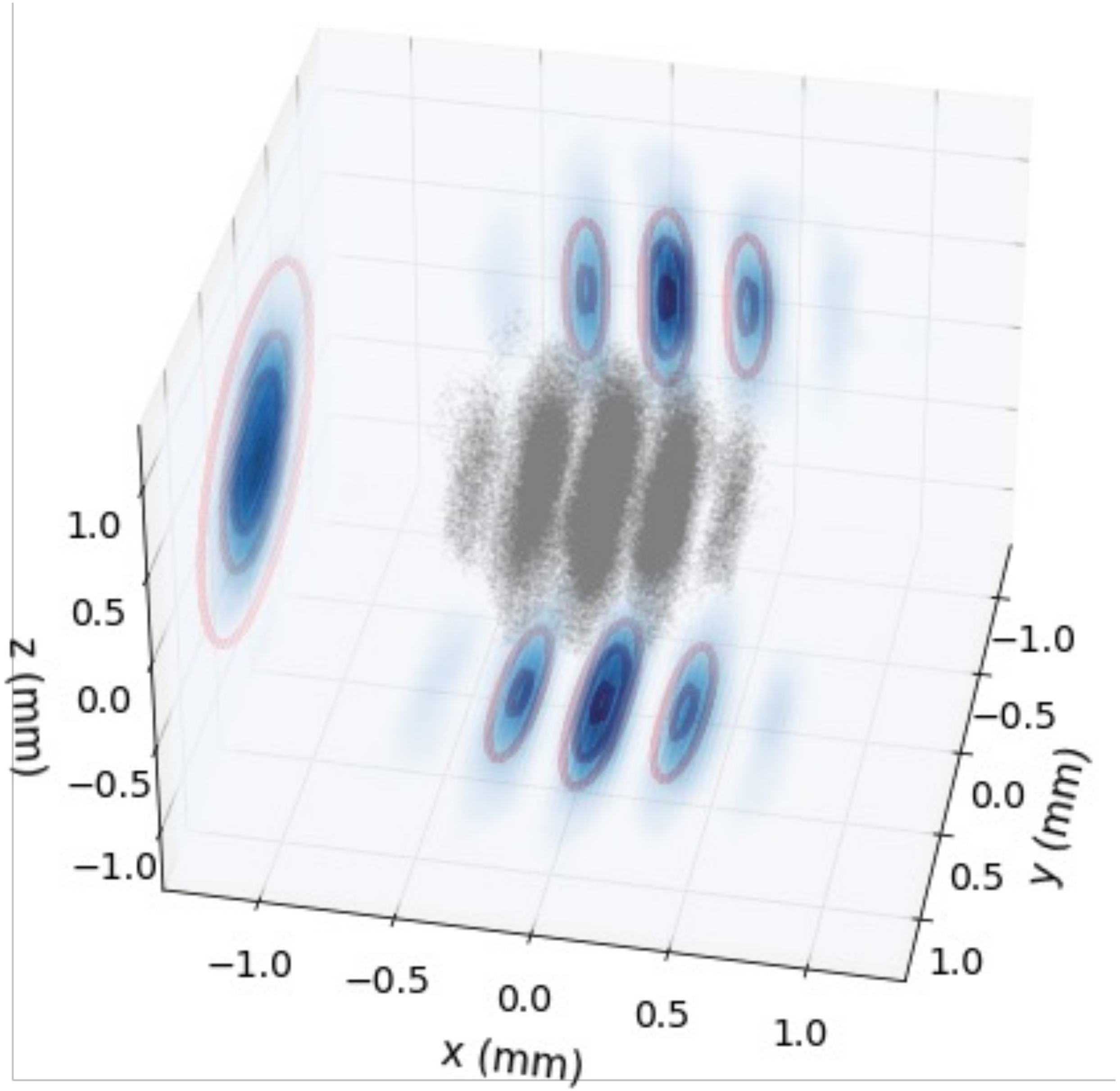




Model

Target: Measured Data

3D Cloud Reconstruction in Simulation



- Learned 3D cloud
- Learned 2D marginal
- Target 2D marginal

Resolution ~ 60μm
Computed comparison to ground truth density with Fourier Shell Correlation

arXiv:2205.11480

Reconstructing Real Objects

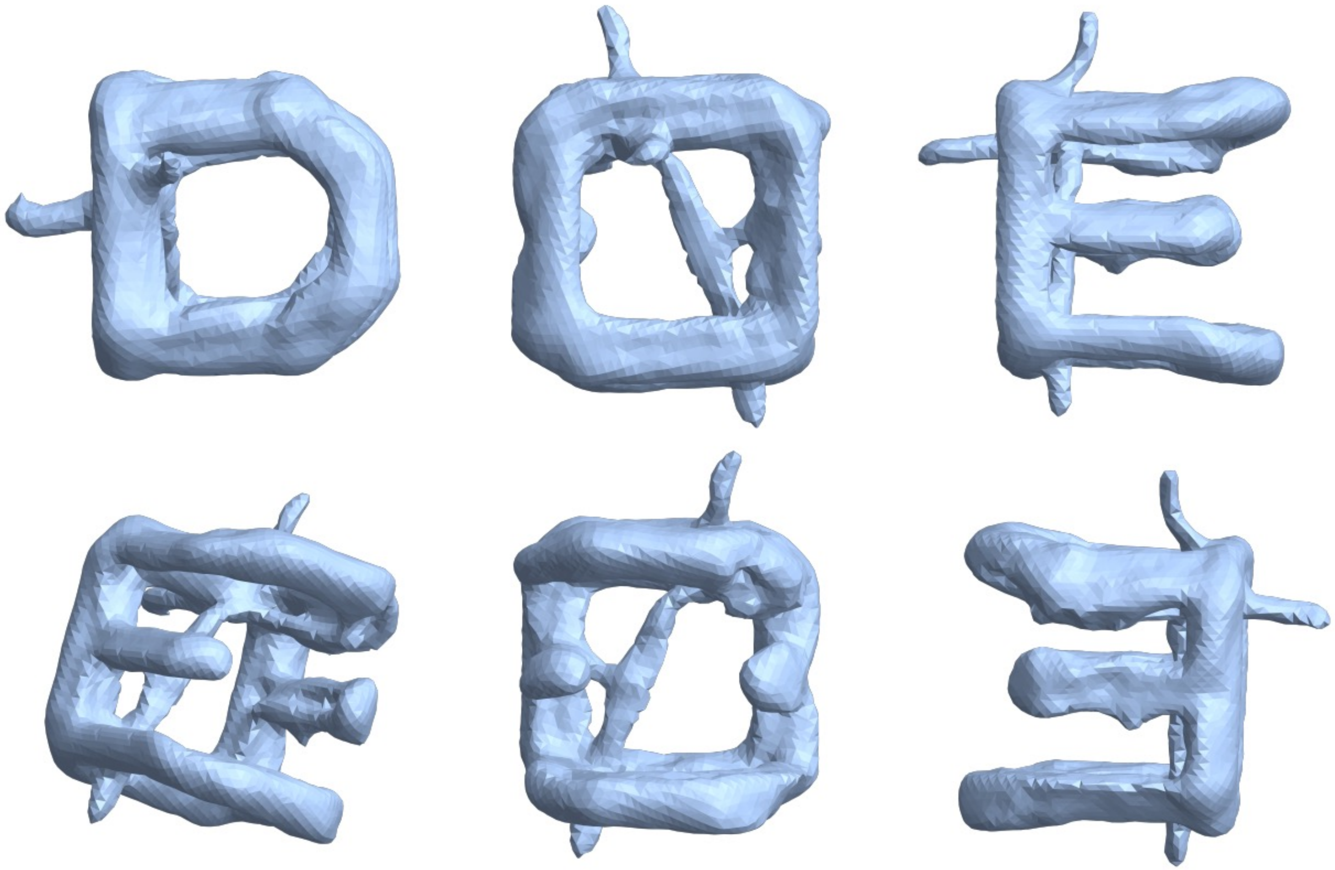
Real Views

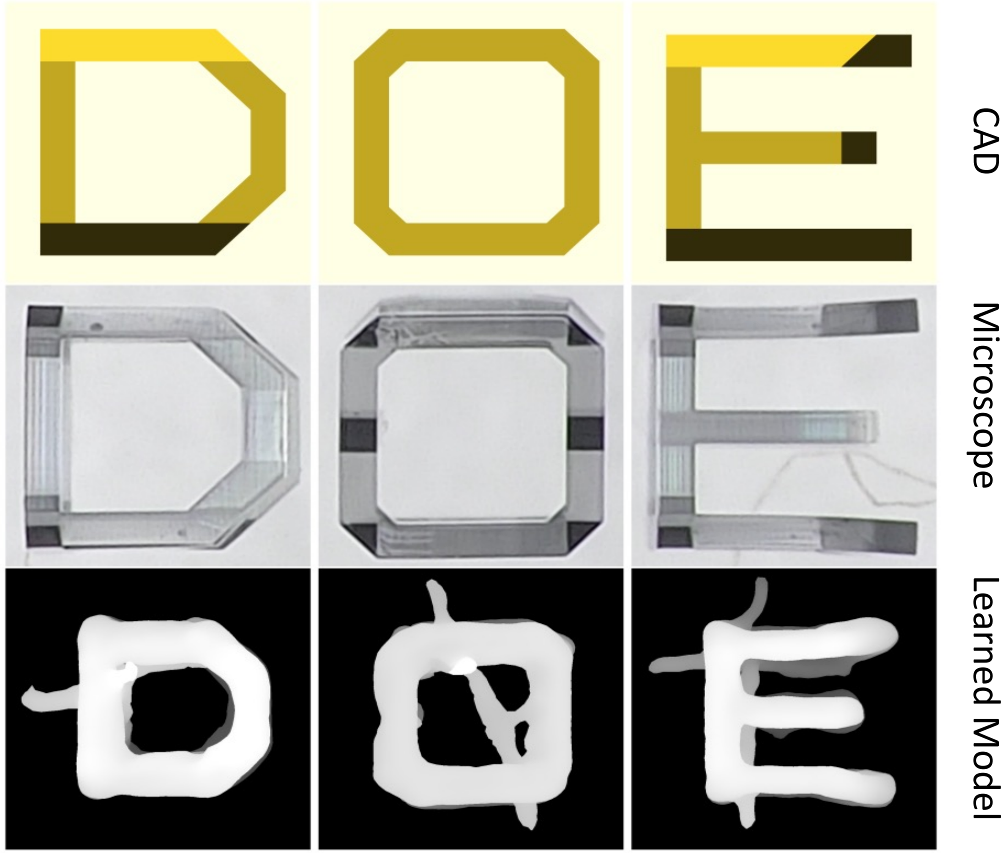
Generated Views

arXiv:2205.11480

3D Reconstruction Comparisons

Mesh of Reconstructed Surface





Resolution ~ 70μm
Computed using split-halves
Fourier Shell Correlation

arXiv:2205.11480

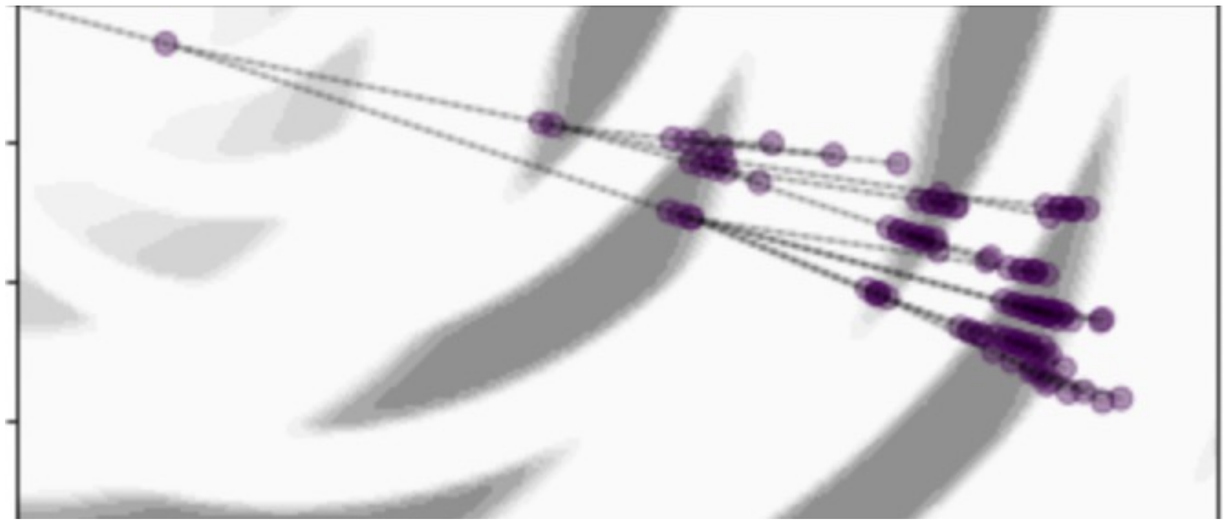
Application for Design Optimization

Detector Design Optimization

As we saw in L. Heinrich's talk:

Goal is to optimize the parameters ϕ of a detector models to minimize a design function $f(\cdot)$ when evaluated on samples of data x

$$\mathbb{E}_{p_{\phi}(x)}[f(x)] = \int dx\, f(x) p_{\phi}(x)$$

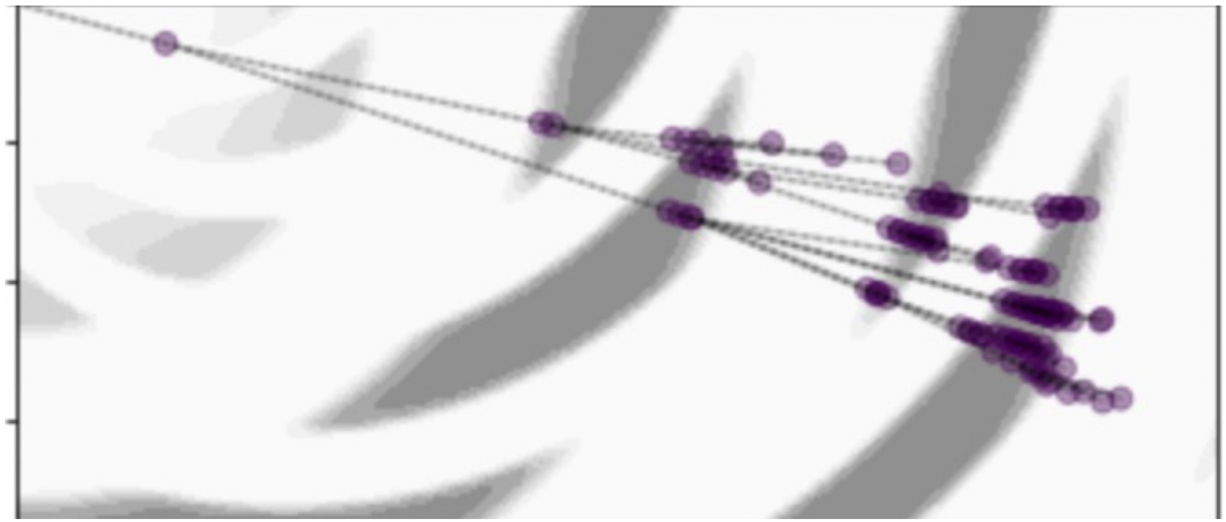


Detector Design Optimization

As we saw in L. Heinrich's talk:

Goal is to optimize the parameters ϕ of a detector models to minimize a design function $f(\cdot)$ when evaluated on samples of data x

$$\mathbb{E}_{p_\phi(x)}[f(x)] = \int dx\, f(x) p(\text{interact at } x | \lambda = \text{Detector}_\phi(x))$$

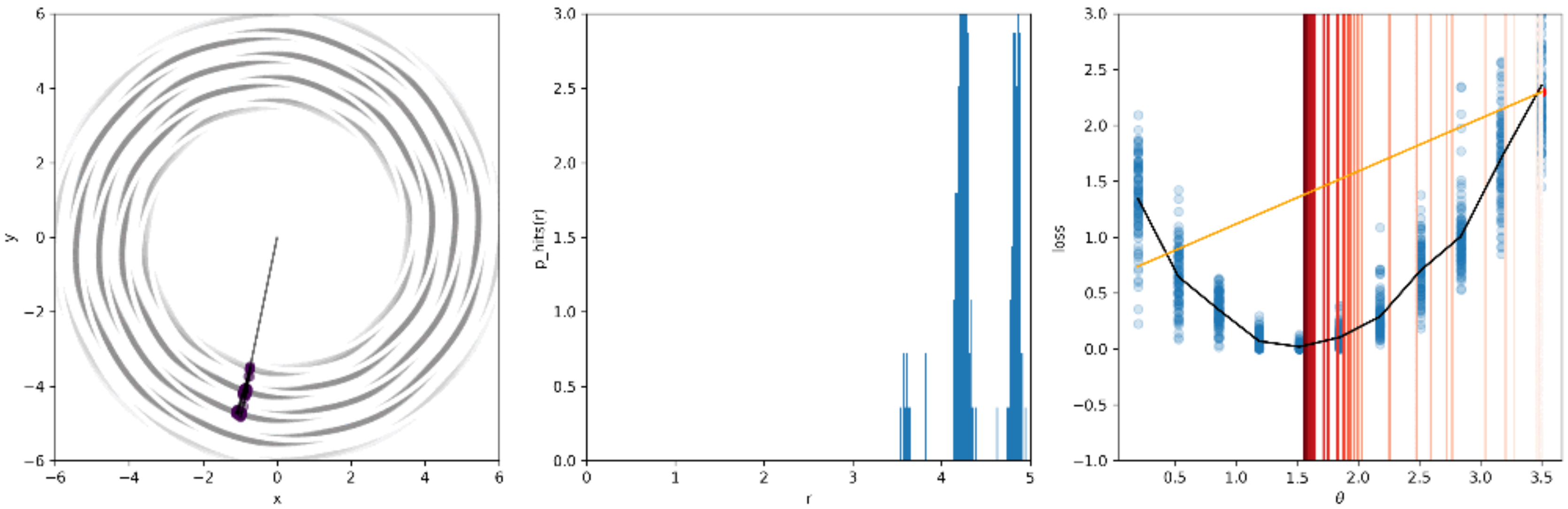


Detector Design Optimization

As we saw in L. Heinrich's talk:

Goal is to optimize the parameters ϕ of a detector models to minimize a design function $f(\cdot)$ when evaluated on samples of data x

$$\mathbb{E}_{p_\phi(x)}[f(x)] = \int dx\, f(x) p(\text{interact at } x | \lambda = \text{Detector}_\phi(x))$$



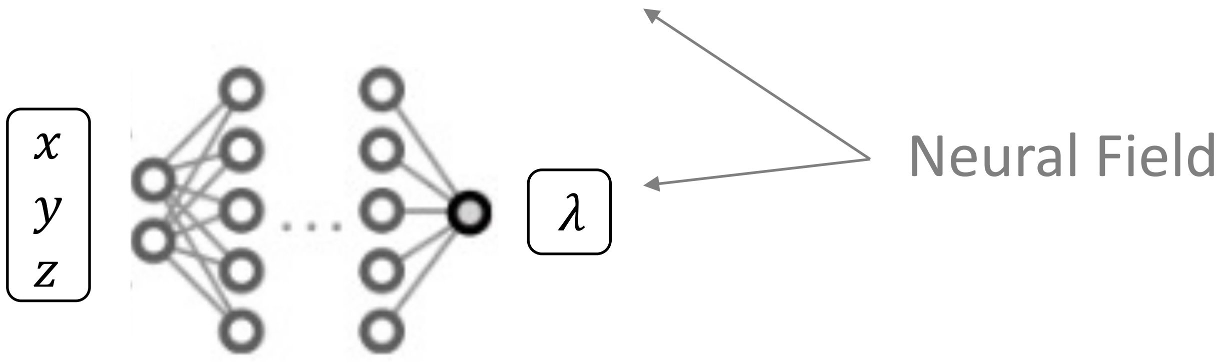
Neural field as a detector

Detector in this example was a very simply toy with 1 parameter

- Predetermined detector structure with varying radius, i.e. $\phi = radius$

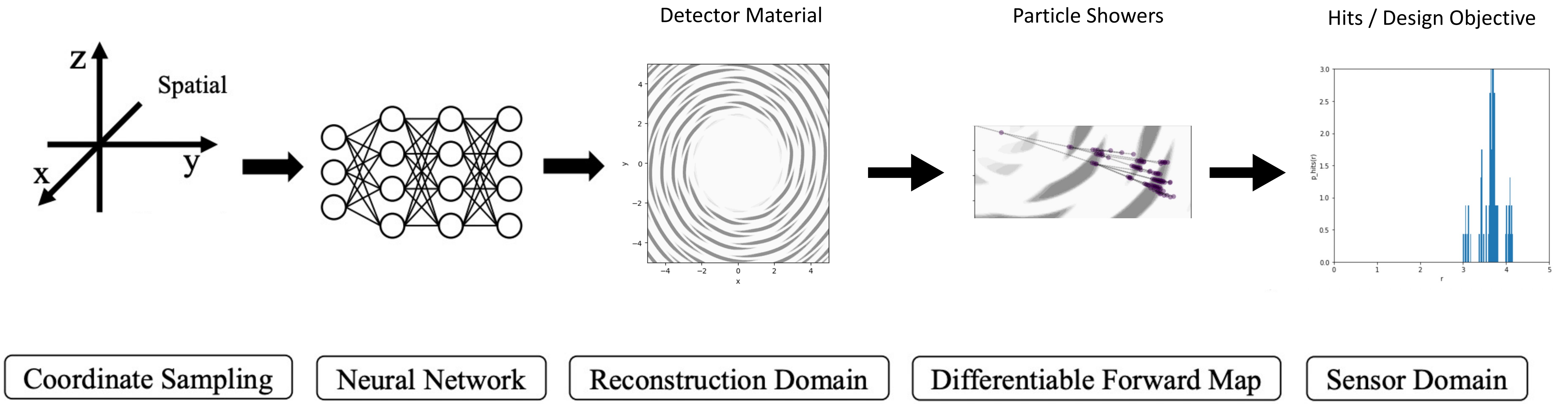
Instead, we could try to optimize a detector freeform from scratch

$$p(\text{interact at } x | \lambda = \text{NN}_\phi(x))$$



20k parameters

Neural Fields for Detector Design

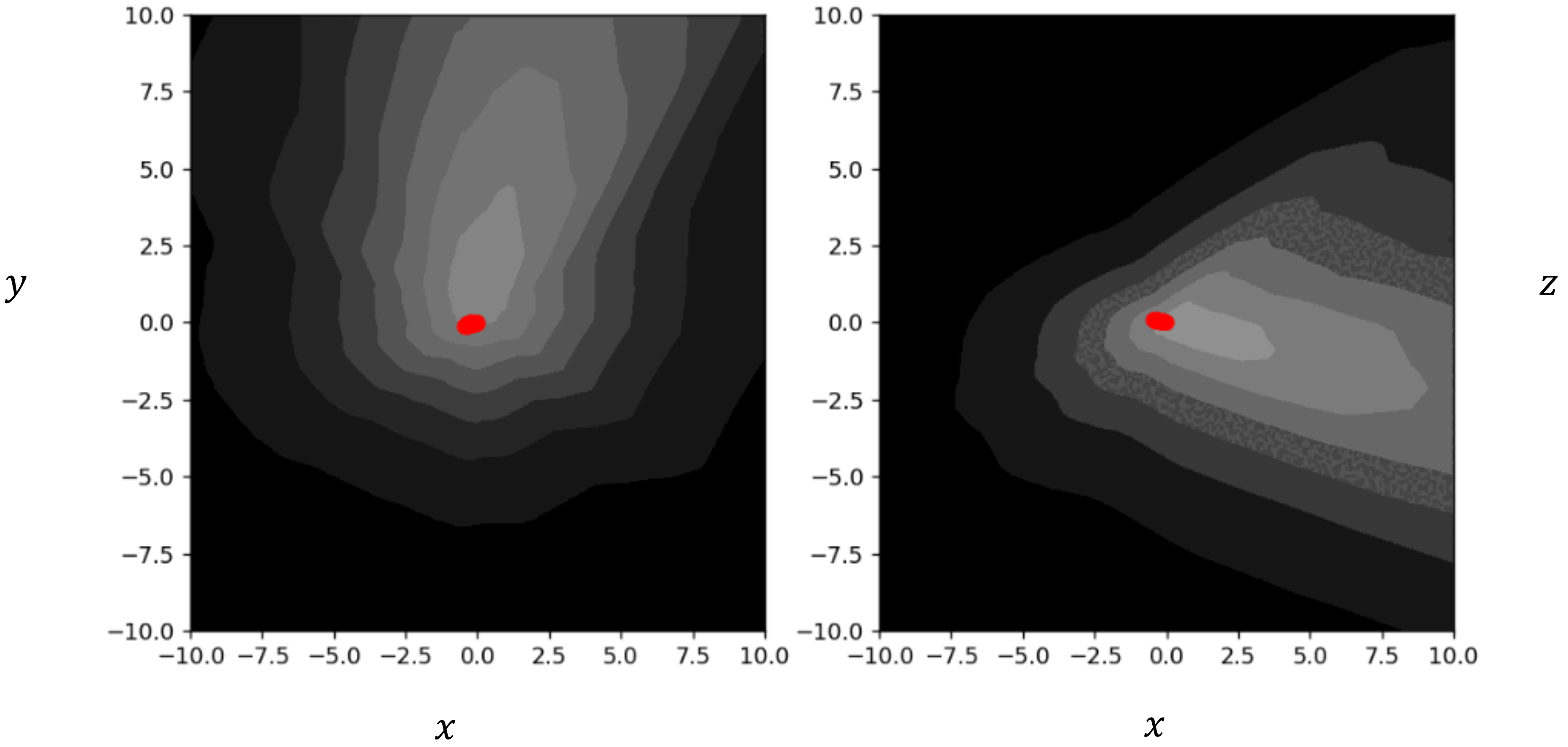


Neural field to model material distribution

Forward model: particle showers

Code: [gradoptics]

Detector Design Optimization with Neural Field



Rediscovers “classic” layout:

Learns to contain shower with no material in middle, material outside

A Few Comments Before Wrapping Up

Ongoing work on Neural Fields

Generalization: how to amortize, get beyond per-instance optimization

Uncertainty Quantification: how to get uncertainty on fitted signal for science applications?

Inference: How to extra information, e.g. physical parameter estimates, from a fit neural field?

Generative modeling: How can we generate neural fields instead of generating data in measurement space?

Many open questions, but more progress on some topics (e.g. generalization, generative modeling) which get a lot of focus in graphics community

Physics + AI

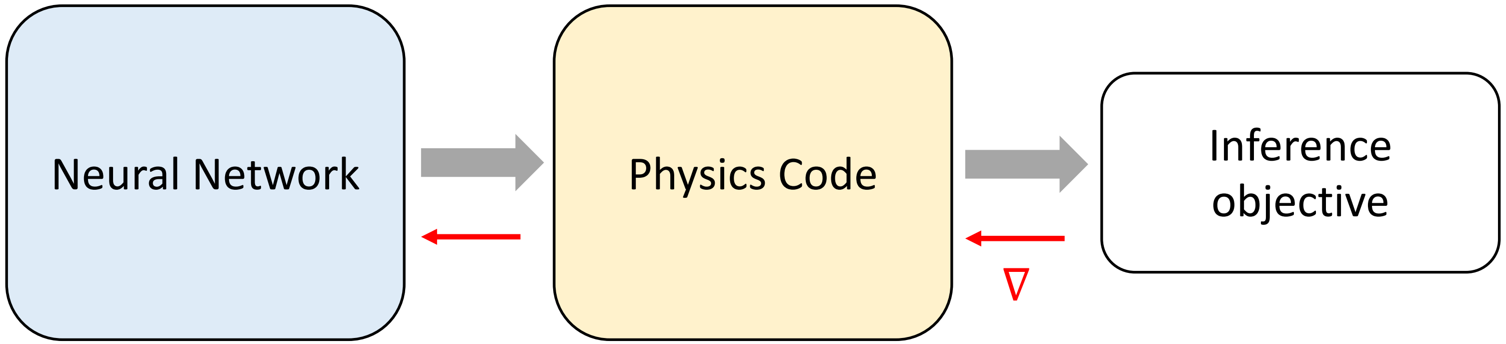
Neural fields are an interesting example of *hybrid Physics-AI systems*

Combines physics knowledge (differentiable simulator) with neural networks to model complex signals

Physics guides learning & ensures we can make physical plausible predictions

Way to add physics inductive bias, in the form of physics code, in NNs

- Where adding symmetries to architectures is like adding structure information, now we can add dynamical information in the form of physics code



Wrapping Up

Summary

Neural Fields combine neural networks and differentiable simulators to enable powerful gradient-based optimization of signals

Originally developed in computer graphics, generalizing the concept enable exciting applications in science, from reconstruction to system design

Can be seen as a development of novel *Hybrid Physics+AI Systems* by directly integrating physics code into ML models

Still much to be done! Generalization, Inference on neural fields, generative modeling, uncertainty quantification, …

Backup

Benefits of Neural Fields

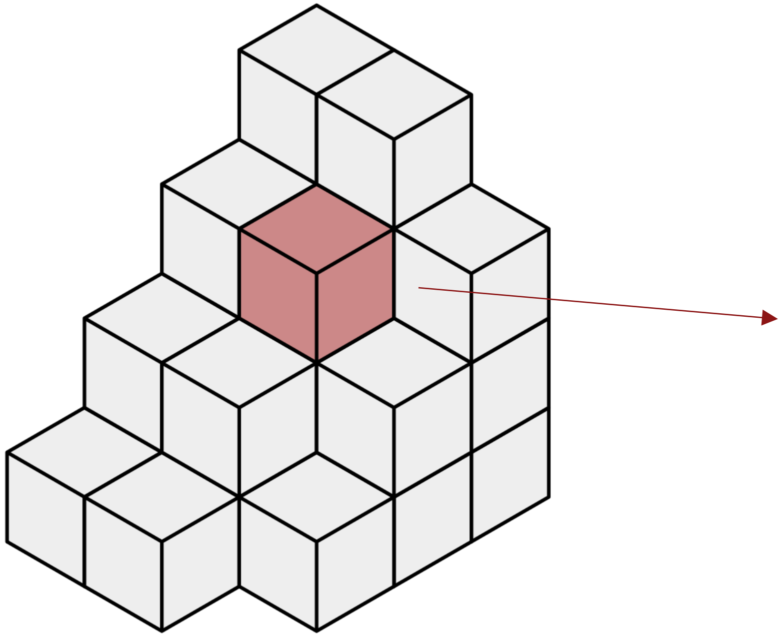
Compact:

1024 x 1024 x 3 = 3MB

Equivalent size neural network has 750k floating point weights

Why Neural Fields?

Compact:



Even more challenging in 3D

Color, density, distance, …

Expensive in storage!

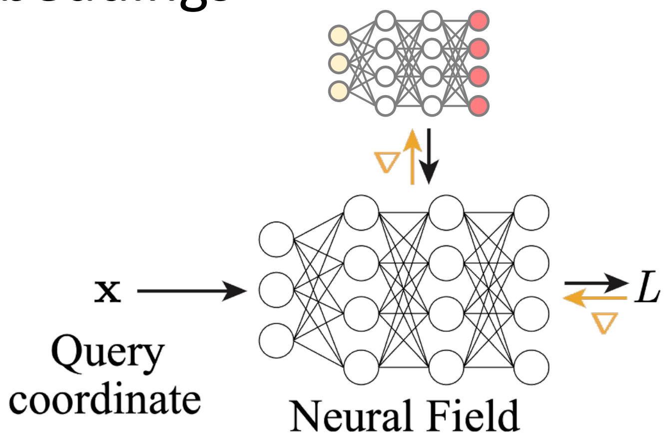
Challenges with Neural Fields

Generalization:

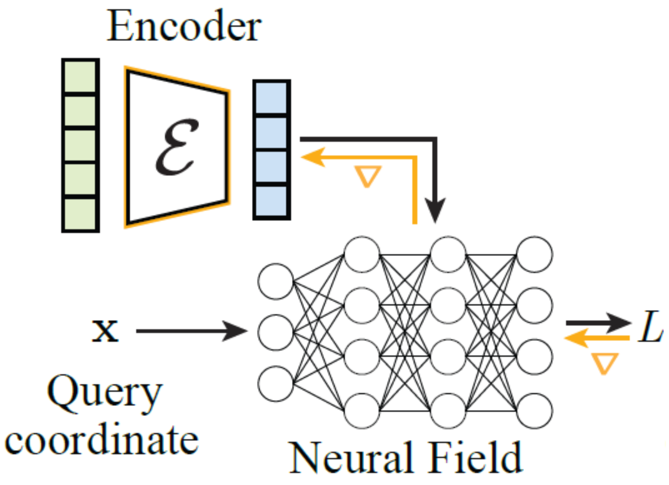
- Basic setup requires per-instance optimization
- Lots of work the share knowledge across reconstructions

Per-instance Optimization

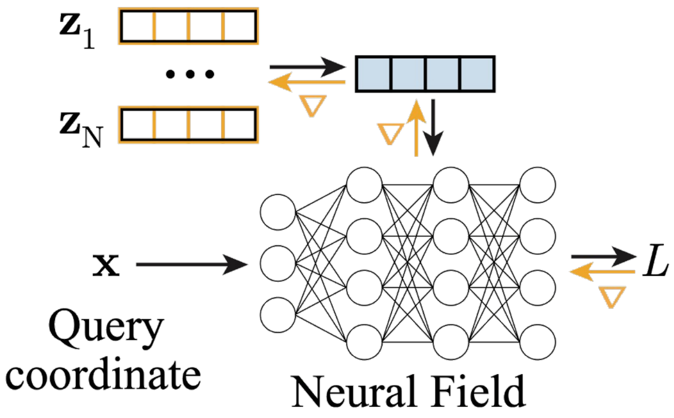
Pre-trained or optimized embeddings



Feature encoding



Auto-decoding



MAGIS-100

Detector Technology: Atom Interferometer

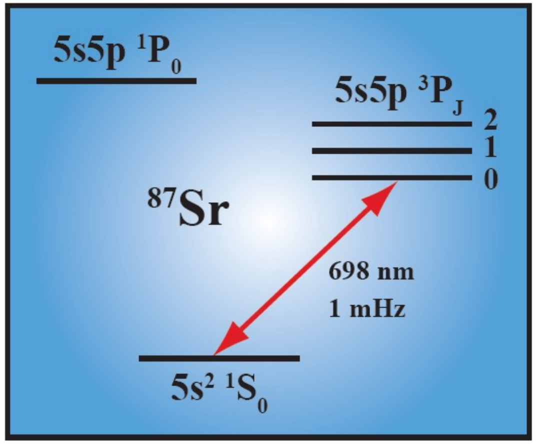
Best clocks in the world now lose <1 second in 10^{18} seconds

MAGIS-100 is based on same physics as Sr optical lattice clock

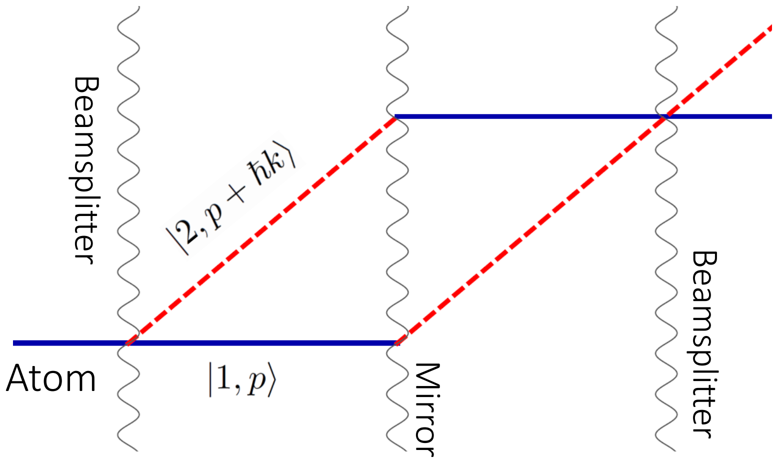
Atom interferometry provides a pristine inertial reference

Compare two (or more) atom ensembles separated by a large baseline

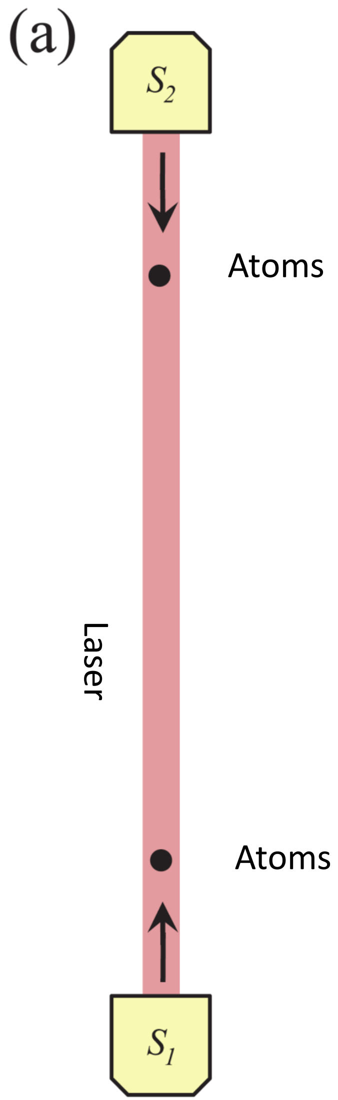
Differential measurement suppresses many sources of common noise and systematic errors



Atomic clock transition



Atom interferometer

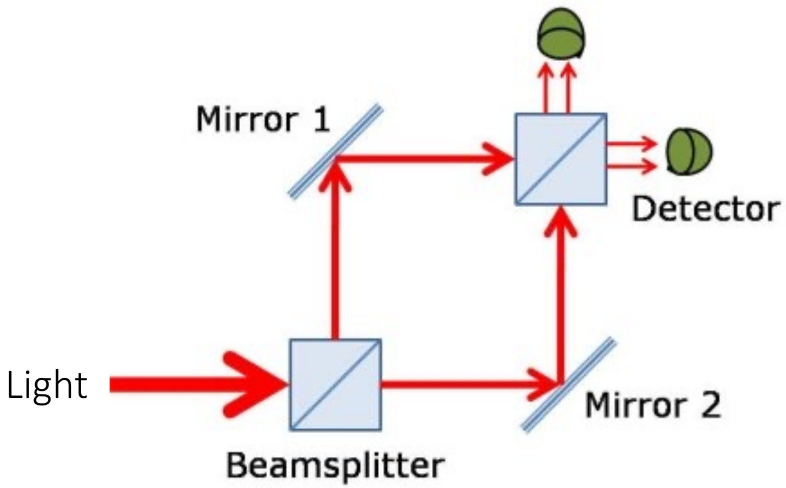


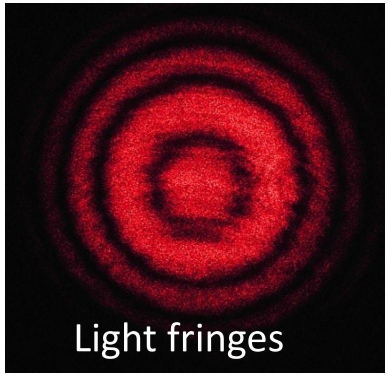
Gradiometer

Slide credit:
J. Hogan

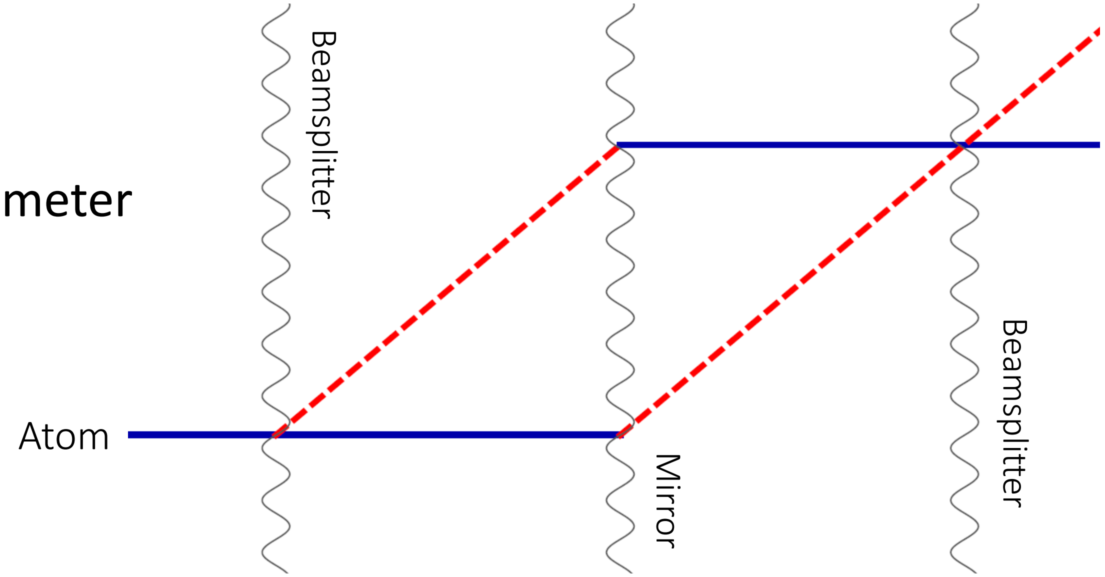
Atom Interference

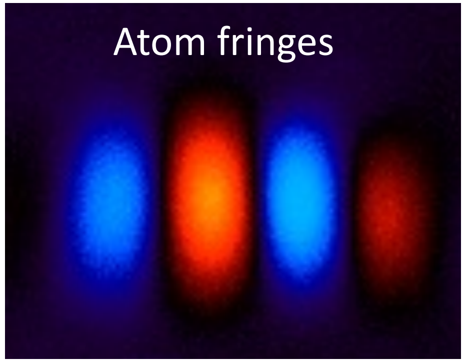
Light interferometer





Atom interferometer





http://scienceblogs.com/principles/2013/10/22/quantum-erasure/
http://www.cobolt.se/interferometry.html

Slide credit: J. Hogan

Light pulse atom interferometry

$$|\text{atom}\rangle = |p\rangle$$

π/2 pulse “beamsplitter”

$$\frac{1}{\sqrt{2}}(|p\rangle + e^{i\Delta\phi}|p+\hbar k\rangle)$$

π pulse “mirror”

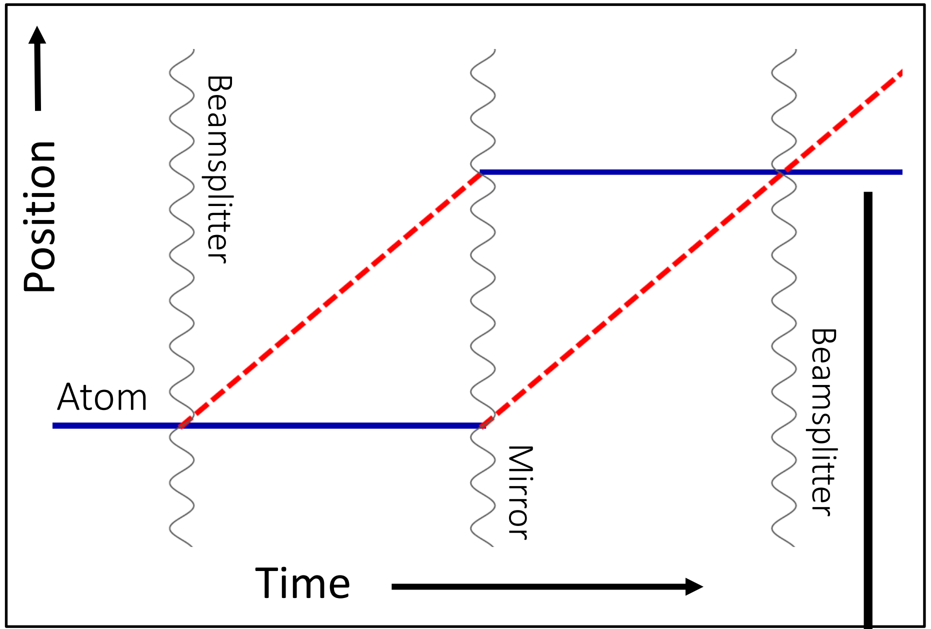
$$\frac{1}{\sqrt{2}}(|p+\hbar k\rangle + e^{i\Delta\phi}|p\rangle)$$

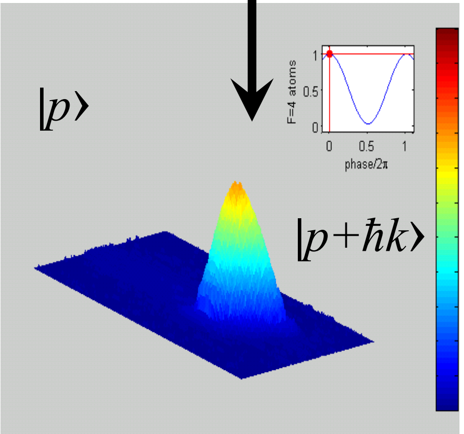
π/2 pulse “beamsplitter”

$$\frac{1}{2}((1+e^{i\Delta\phi})|p\rangle + ((1-e^{i\Delta\phi}))|p+\hbar k\rangle)$$

$$\text{Probability in State } |p\rangle = \cos^2\left(\frac{\Delta\phi}{2}\right)$$

$$\text{Probability in State } |p+\hbar k\rangle = \sin^2\left(\frac{\Delta\phi}{2}\right)$$





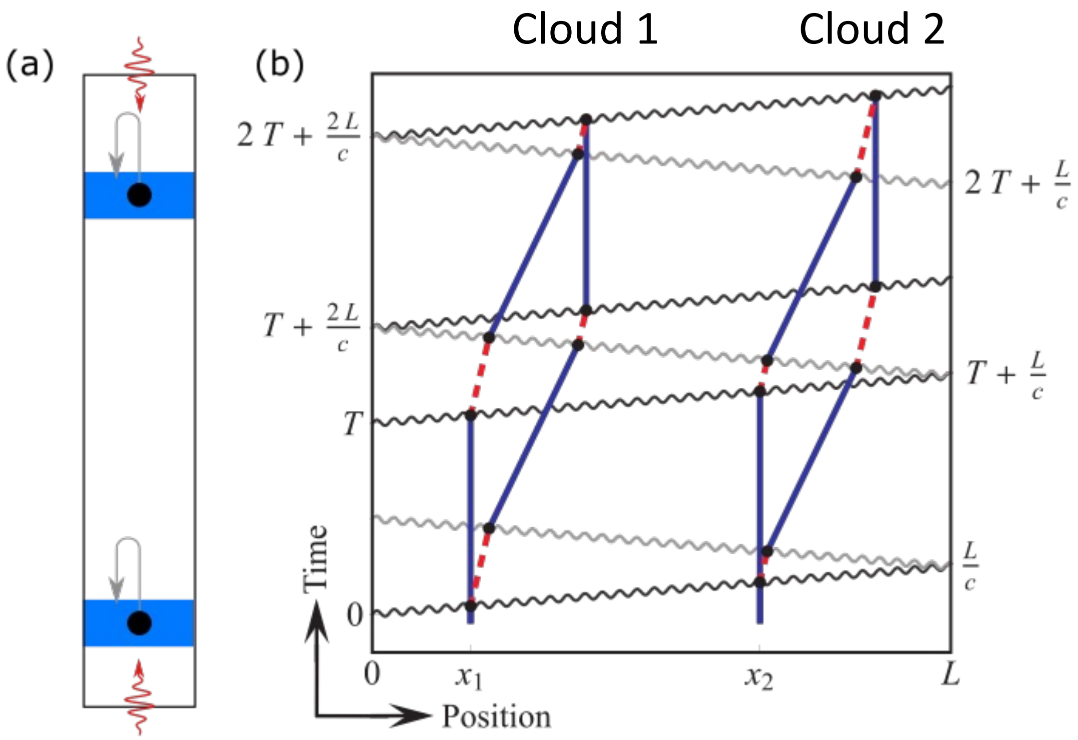
Clock Gradiometer

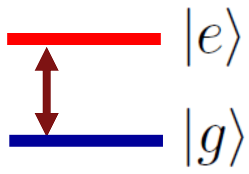
Clock:
measure light travel time

Accelerometer:
atoms excellent inertial test masses
→ follow geodesics

Gradiometer:
Differential accelerometer
→ remove laser noise w/ single baseline

Measure differential acceleration by comparing the light travel times





Graham et al., PRL **110**, 171102 (2013).
Arvanitaki et al., PRD **97**, 075020 (2018).

GW / DM Analysis Principle

Launch sequence repetition rate ~1 Hz

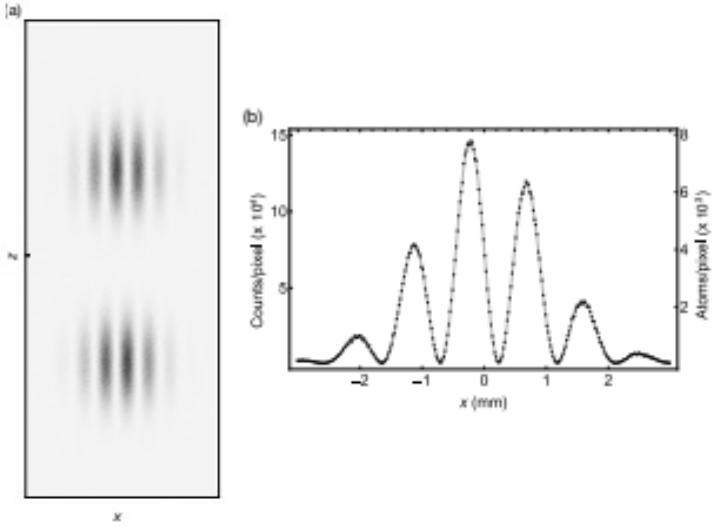
Image clouds (fluorescent imaging) at end of sequence

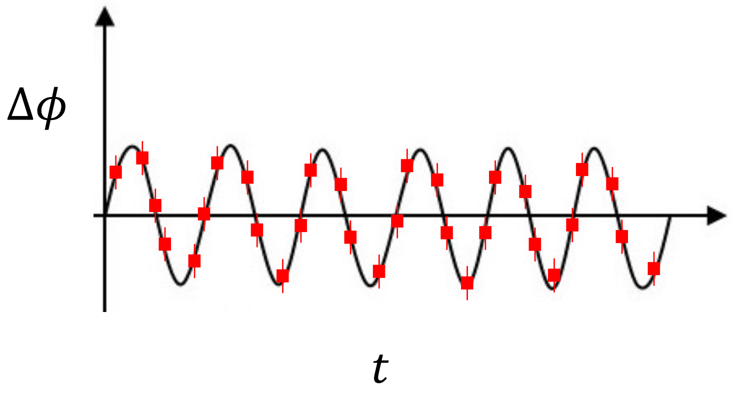
Extract phase from interference pattern per launch

Look for signals of time-varying phase difference

In this Mid-band frequency range

- GW signals can persist for months
- Can have multiple overlapping signals
- Measurements at different times of year can help signal localization





Rendering Views

Real Views

Generated Views

Phase Extraction & Uncertainties

Many systematic effects can be sources of noise in phase measurements

- Especially laser wavefront variations – i.e. laser phase varying spatially

Goal: 3D Cloud Reconstruction

Why:

- Increased light detection
- Model spatially varying systematic effects

How: Single-Shot Multi-View Imaging System

Challenge: How to reconstruct the cloud in 3D?

- Tomographic imaging with complex geometry

Table 3. Summary of key experimental parameters and target values of MAGIS-100 (initial) (see Table 2). Spectral densities are taken to be in the $\sim 0.1 - 3$ Hz frequency band of interest. Note that the cloud kinematics can either be stabilized to below the target values or measured each shot at the target uncertainty.

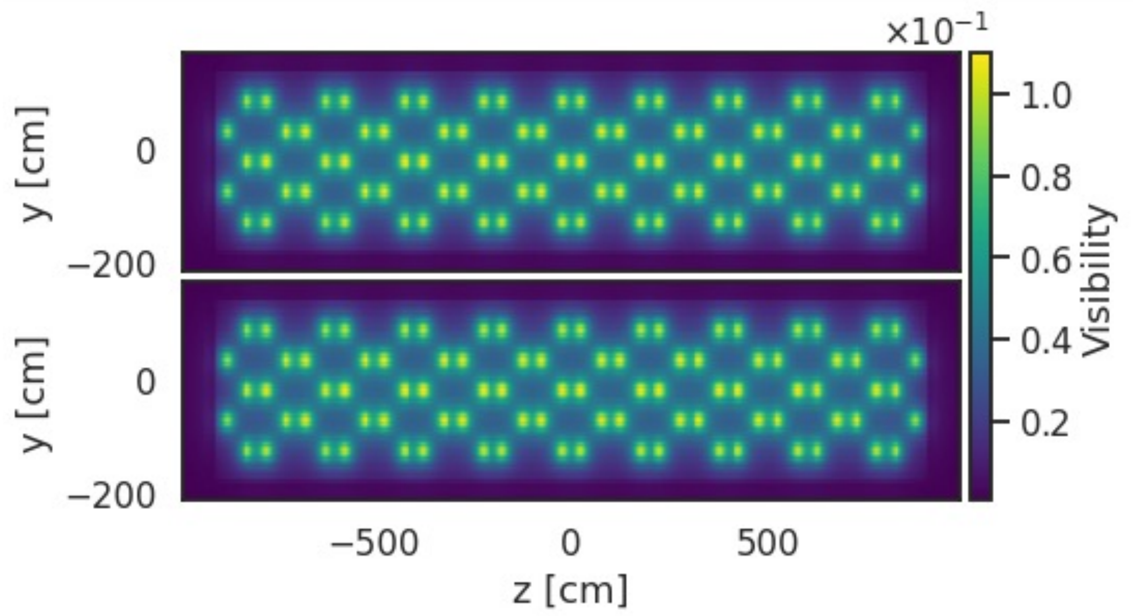
Parameter	Target Value	Primary Driving Factors
LMT atom optics	$n = 100$	Increase sensitivity to science signals
Phase resolution	10^{-3} rad/$\sqrt{\text{Hz}}$	Increase sensitivity to science signals
Frequency noise/drift	< 10 Hz	Increase pulse transfer efficiency (Section 4.3)
Per shot position uncertainty	10 μm/$\sqrt{\text{Hz}}$	Coupling to wavefront aberrations (Section 5.2)
Per shot velocity uncertainty	10 μm/s/$\sqrt{\text{Hz}}$	
Laser wavefront variation	5 mrad*	Coupling to cloud kinematic and laser pointing jitter (Section 5.2 and Section 5.4)
Laser intensity stabilization	0.1%/$\sqrt{\text{Hz}}$	AC Stark shifts (Section 5.5)
Laser pointing stability	30 nrad/$\sqrt{\text{Hz}}$	Coupling to wavefront aberrations (Section 5.4)
Magnetic field uniformity	1 mG (rms)	Clock frequency shifts

*at transverse length scales $\lesssim 3$ mm

Application for Dune

DUNE Photon Response

Expensive Simulation → Lookup table: Visibility(x) → PMT Response



Photon simulation involves modeling optical visibility

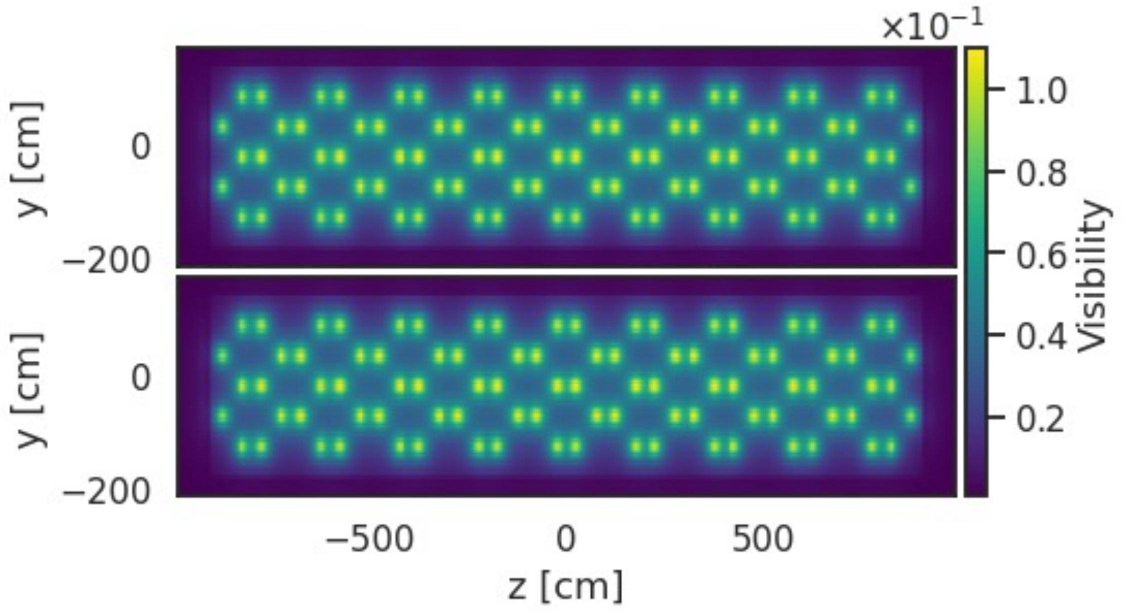
- Probability of observing a photon produced at a given point per PMT

Typically, expensive simulation done to produce grid of points (voxels)

Very large grid → large memory consumption, not easily tunable to data

DUNE Photon Response

Expensive Simulation → Lookup table: Visibility(x) → PMT Response



Lookup table

↓

Neural Network (SIREN)

Train neural network to represent lookup table

Voxels → network weights

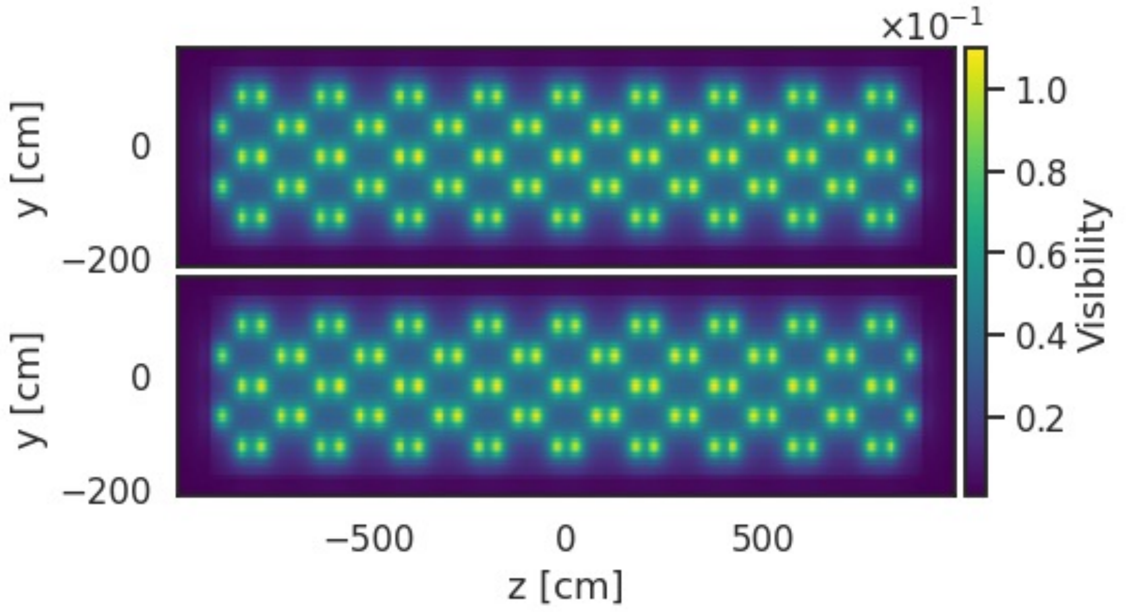
- More memory efficient

Network easily trainable

- Tune representation on measured data

DUNE Photon Response

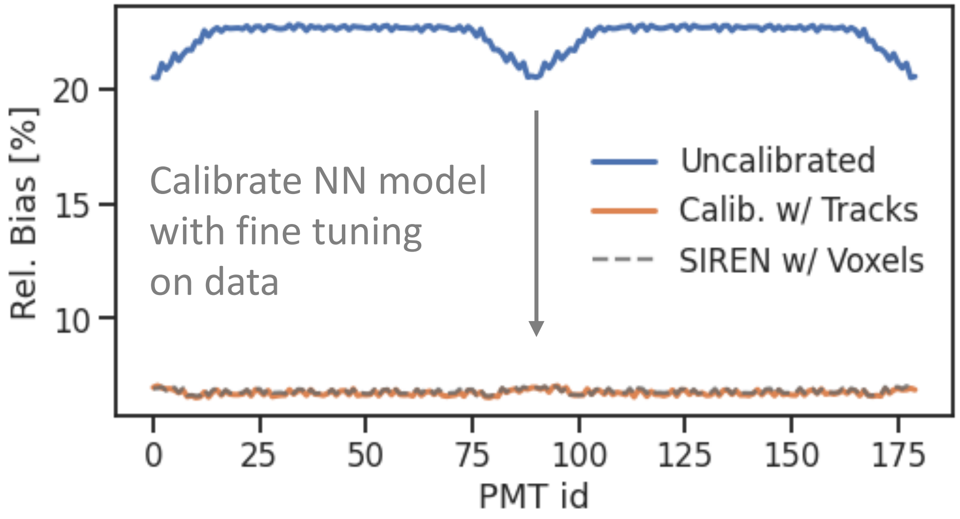
Expensive Simulation → Lookup table: Visibility(x) → PMT Response



Lookup table

↓

Neural Network (SIREN)



2211.01505