Model-Assisted Generative Adversarial Networks

Leigh Whitehead
ICL Seminar
05/06/20

UNIVERSITY OF CAMBRIDGE

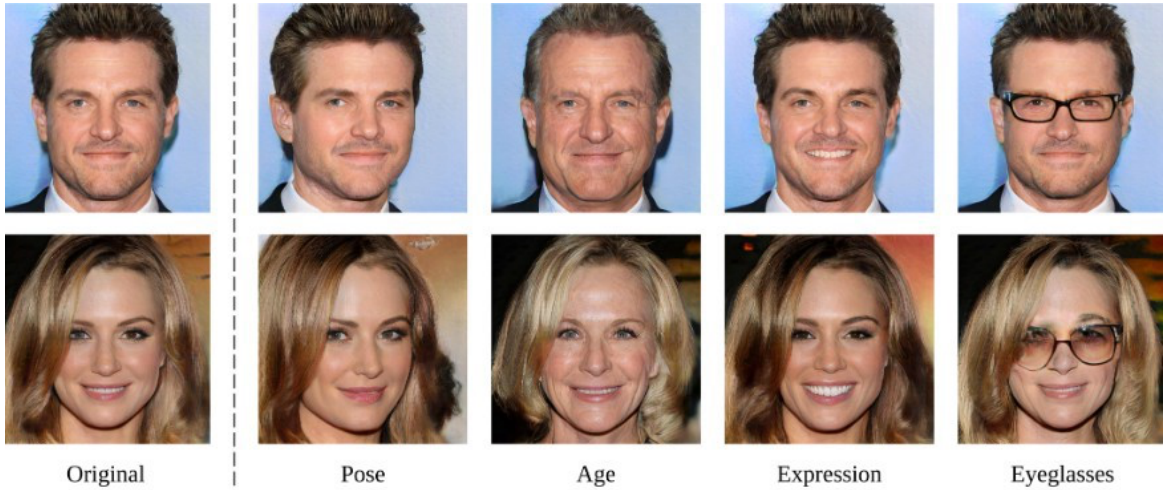
Overview

- What are Generative Adversarial Networks (GANs)?
- The Model-Assisted GAN
- Case Studies
  - Case Study I (from the paper)
  - Case Study II (from the paper)
  - Light simulation in DUNE
- Outlook and Summary

Generative Adversarial Networks

What are GANs?

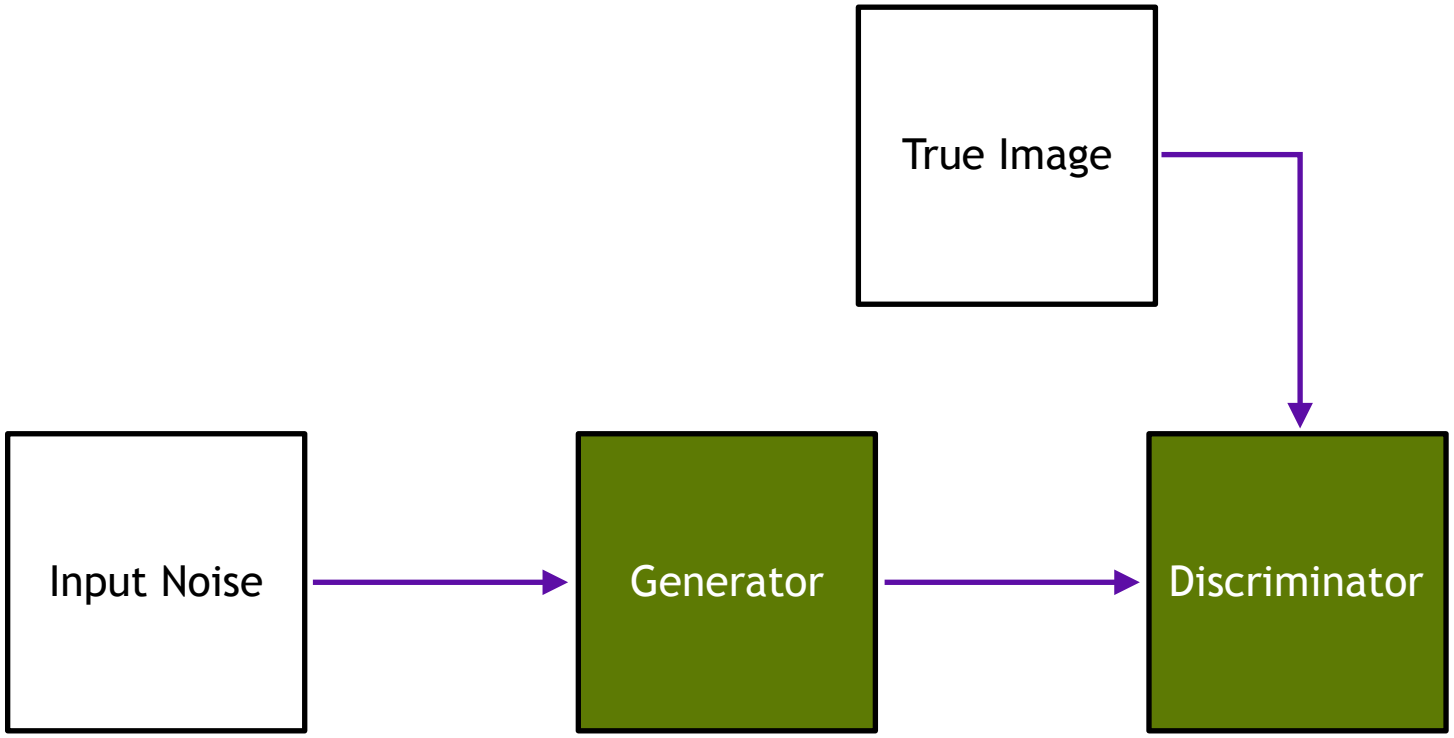
- GANs are a type of neural network composed of two different networks
  - Typically one is known as the *generator* and the other, the *discriminator*
  - Invented by Ian Goodfellow in 2014 (arXiv:1406.2661)
- They are typically used for generating images



https://medium.com/swlh/face-morphing-using-generative-adversarial-network-gan-c751bba45095

What are GANs?

- A very simple schematic of the network architecture



The generator takes an input noise vector
and produces a generated image

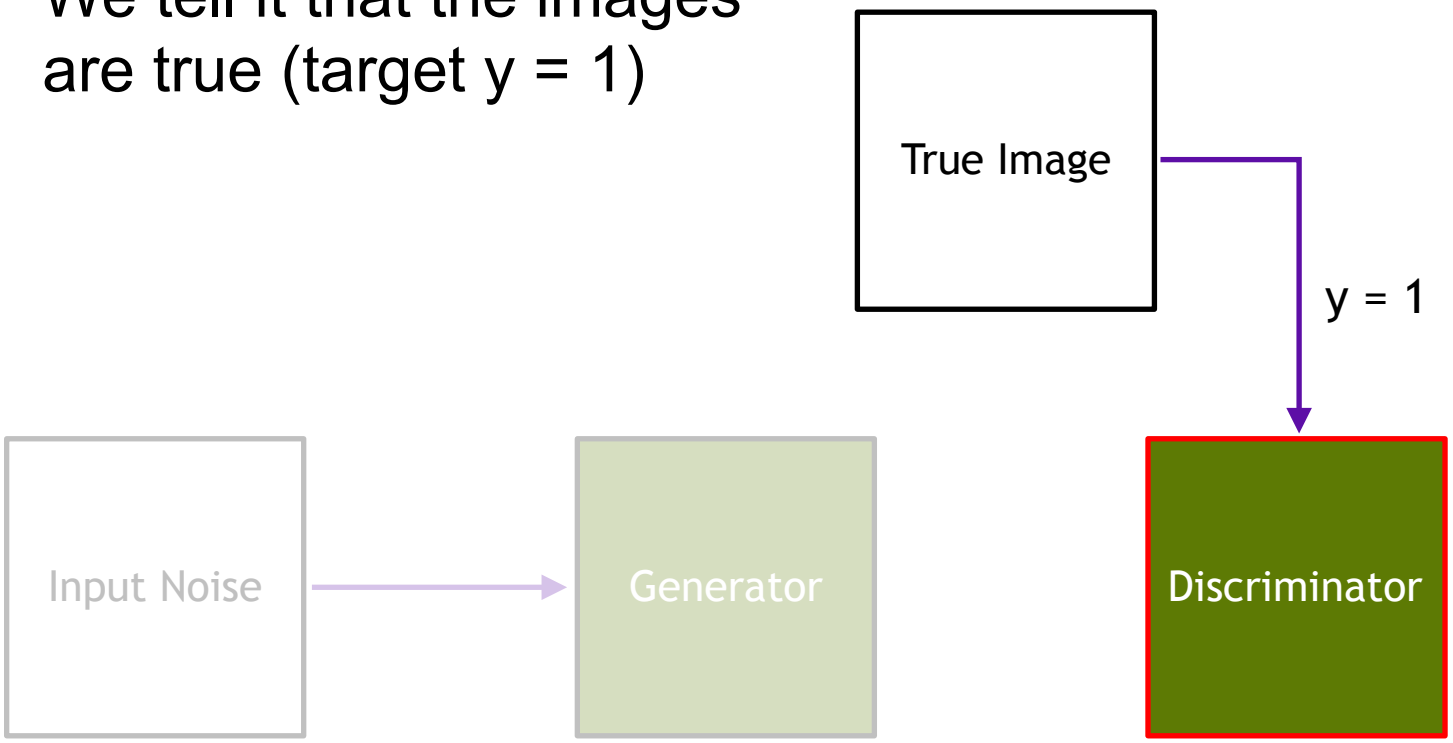
Training GANs

- The process of training GANs is a competition between the two networks
  - The *generator* learns to trick the *discriminator* into classifying its images as real
  - The *discriminator* learns to tell the difference between real and generated images
  - Mathematically speaking, it is a two player minimax game

- In each training iteration, we need to perform three steps to train these networks
  - Repeat these until an equilibrium is reached, and accurate generated images are produced

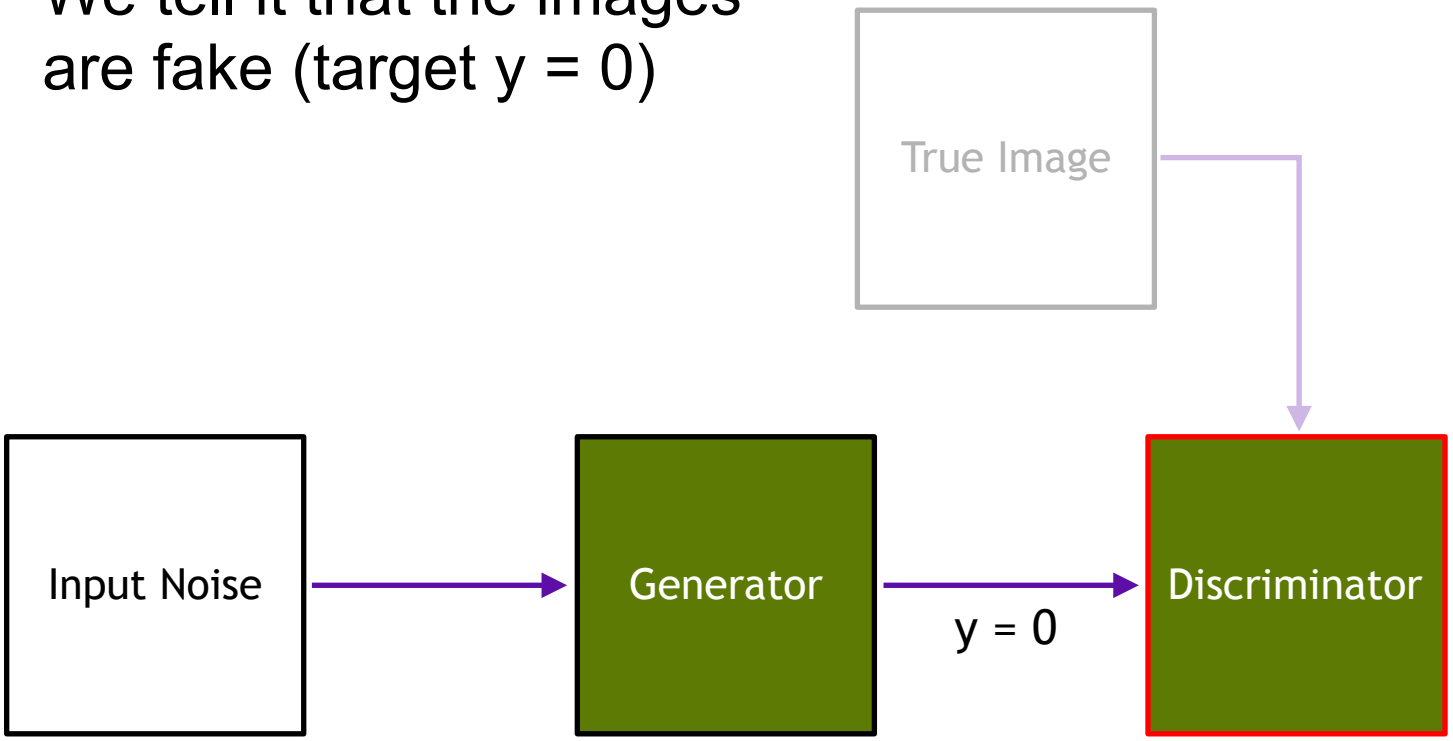
Training Step 1

- Train the *discriminator* to identify true images
  - We tell it that the images are true (target y = 1)



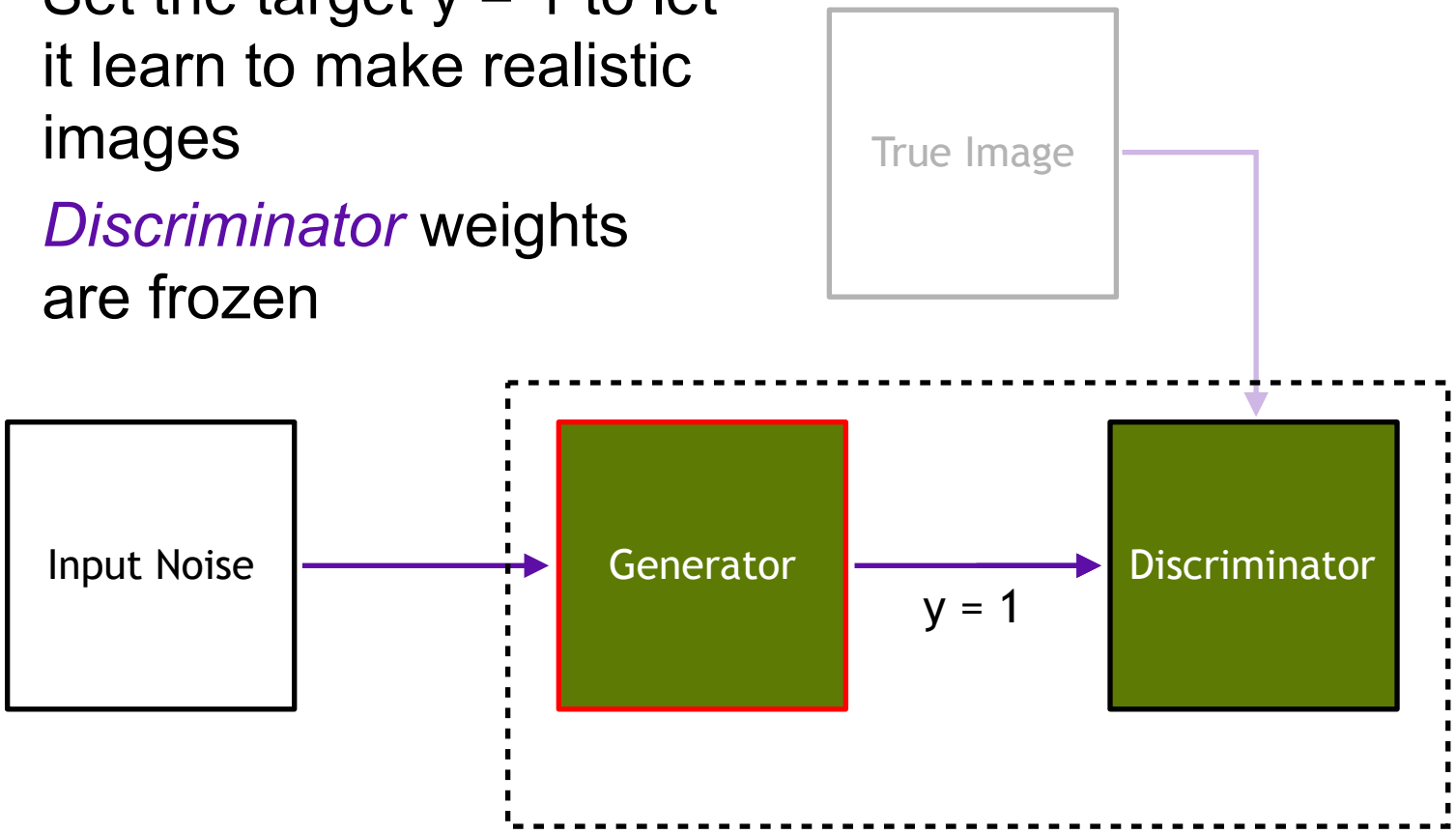
Training Step 2

- Train the *discriminator* to distinguish true and fake images
  - We tell it that the images are fake (target y = 0)



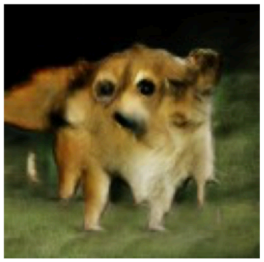
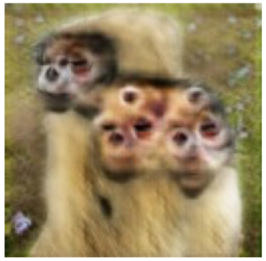
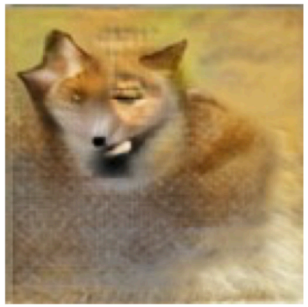
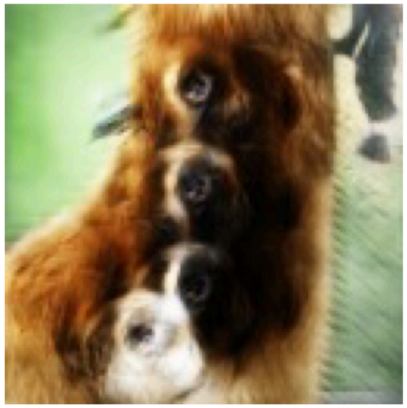
Training Step 3

- We now train the *generator* and *discriminator* as one model
  - Set the target y = 1 to let it learn to make realistic images
  - *Discriminator* weights are frozen



Fast moving field

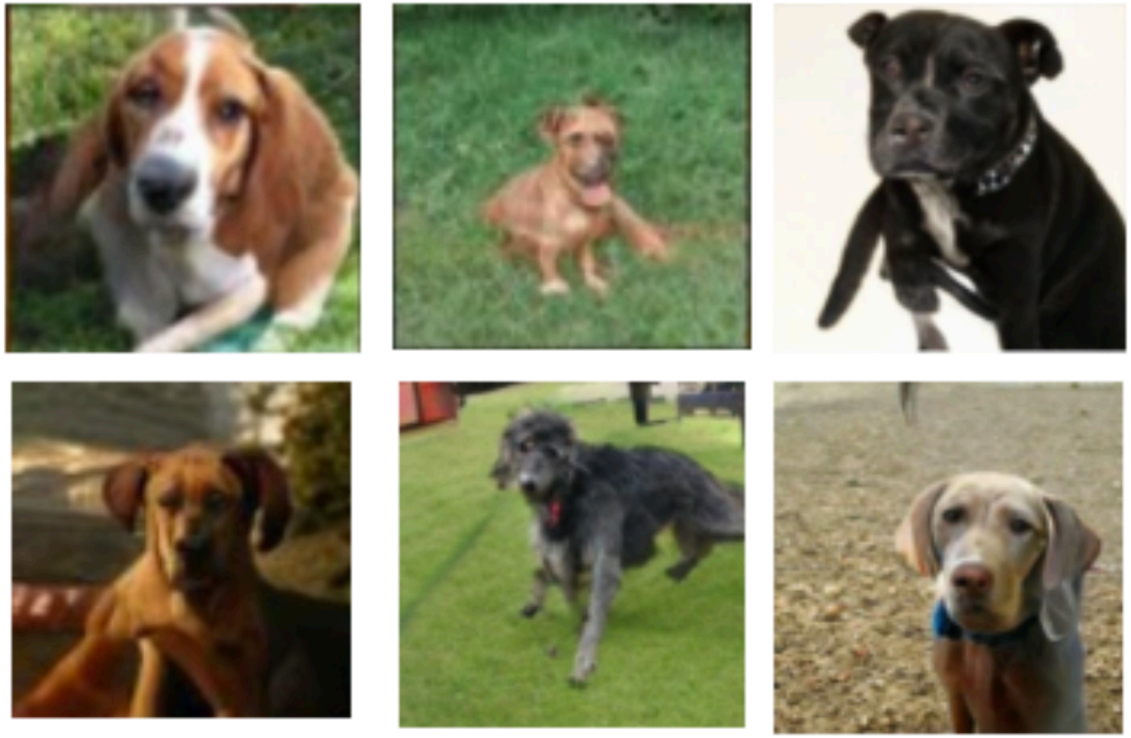
- Things have progressed very quickly
  - Back in 2016 there were a lot of horrors created
  - I think these are supposed to be dogs


Ian Goodfellow, NIPS 2016 Tutorial: Generative Adversarial Networks

Fast moving field

- Things have progressed very quickly
  - We can now see much better images



https://www.kaggle.com/c/generative-dog-images

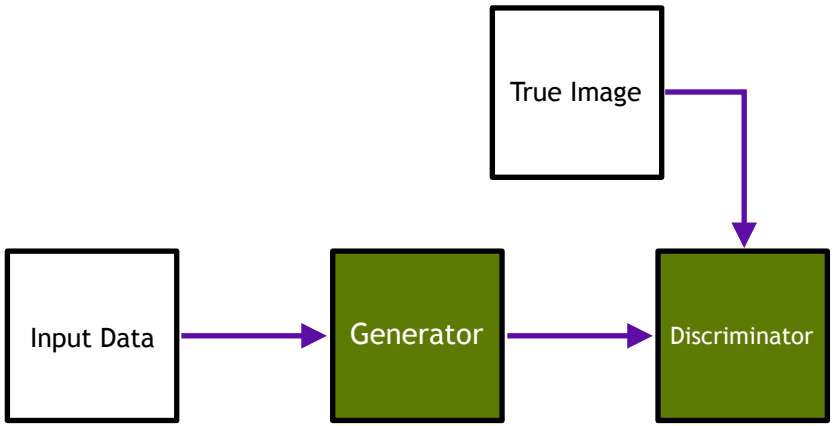
Fast moving field

- Things have progressed very quickly
  - We can now see much better images

https://twitter.com/goodfellow_ian/status/1084973596236144640

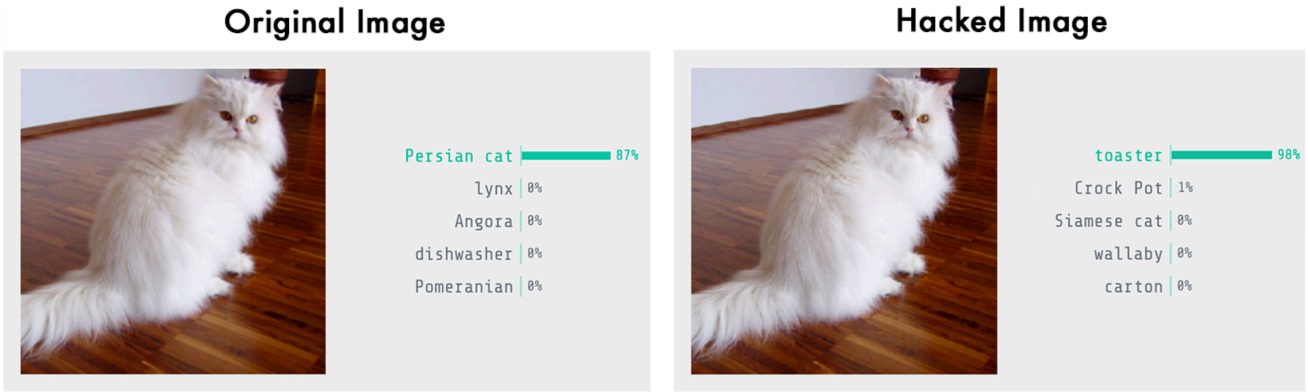
Conditional GANs

- The examples I have shown so far have had noise input to the generator
- Conditional GANs have generator outputs that are *conditional* on the input
  - The *generator* input hence has some meaning
- Conditional GANs have found some applications in high energy physics for fast simulations



Useful examples

- GANs can also be used in useful ways
- Image upscaling: increase image resolution
  - Maybe the CSI-style “zoom in, enhance” is on the way
- Robustness and security of image recognition
  - Important for self driving cars!
- Physics!
- Etc, etc, etc



https://medium.com/@ageitgey/machine-learning-is-fun-part-8-how-to-intentionally-trick-neural-networks-b55da32b7196

Adversarial Attacks

- Application of noise invisible to the eye can completely fool some image recognition neural networks

Jiajun Lu, Hussein Sibai, Evan Fabry Adversarial Examples that Fool Detectors, arXiv:1712.02494, 2017

- Physical changes can also cause incorrect classification

Kevin Eykholt, et al.,Robust Physical-World Attacks on Deep Learning Models arXiv:1707.08945, 2017

- Training image classifying networks adversarially can help to make them more robust

The Model-Assisted GAN

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

Image-Based Model Parameter Optimization Using Model-Assisted Generative Adversarial Networks

Saúl Alonso-Monsalve and Leigh H. Whitehead

***Abstract*—We propose and demonstrate the use of a model-assisted generative adversarial network (GAN) to produce fake images that accurately match the true images through the variation of the parameters of the model that describes the features of the images. The generator learns the model parameter values that produce fake images that best match the true images. Two case studies show excellent agreement between the generated best match parameters and the true parameters. The best match model parameter values can be used to retune the default simulation to minimize any bias when applying image recognition techniques to fake and true images. In the case of a real-world experiment, the true images are experimental data with unknown true model parameter values, and the fake images are produced by a simulation that takes the model parameters as input. The model-assisted GAN uses a convolutional neural network to emulate the simulation for all parameter values that, when trained, can be used as a conditional generator for fast fake-image production.**

***Index Terms*—Fast simulation, generative adversarial networks (GANs), model-assisted GAN, parameter optimization.**

images that best match the true images such that $p_{\text{generator}}(p_{\text{bm}}) =$ p_{d}[illegible]. The default simulation for an experiment will typically use default model parameter values that do not exactly match the true model parameter values. The parameters $p_{\text{bm}} \sim p_{\text{generator}}(p_{\text{bm}})$ can be extracted to update the default simulation model parameters so that the fake images will more accurately reproduce the true images. Furthermore, the difference between $p_{\text{generator}}(p_{\text{bm}})$ and the default parameters distribution gives physical insight into the understanding of the model and the model parameter values that were not correct in the default simulation. The key advantage of the model-assisted GAN is that the simulation only needs to be run once, and then, any number of model parameter optimizations can be performed with different true data sets very quickly. Another advantage is that any additional number of fake images can be produced efficiently using the emulator from the model-assisted GAN.

S. Alonso-Monsalve and L. H. Whitehead, "Image-Based Model Parameter Optimization Using Model-Assisted Generative Adversarial Networks," in *IEEE Transactions on Neural Networks and Learning Systems, 2020*

Model-Assisted GAN

- I first had the idea for the MAGAN in 2018
- Image-recognition approaches are now common in HEP, but differences between simulation and data are a concern
- Saw examples of applying a GAN to the simulated images
  - This is effectively arbitrary bin-by-bin reweighting
- I wanted to find a method to modify the simulated images in a *physically motivated* way
  - The MAGAN knows about the physics parameters that are used by the simulation

Model-Assisted GAN - Aims

- Instead of making images directly like standard GANs, create a *vector of model parameters* instead
- These parameters are the input that control the simulation
  - In reality this could be noise, energy scale, etc
- We want to train a neural network to reproduce the simulation outputs for the *whole parameter space*
- We also want to be able to *extract the model parameter values* from a defined data sample
  - Allows us to tune the simulation

Model-Assisted GAN - Details

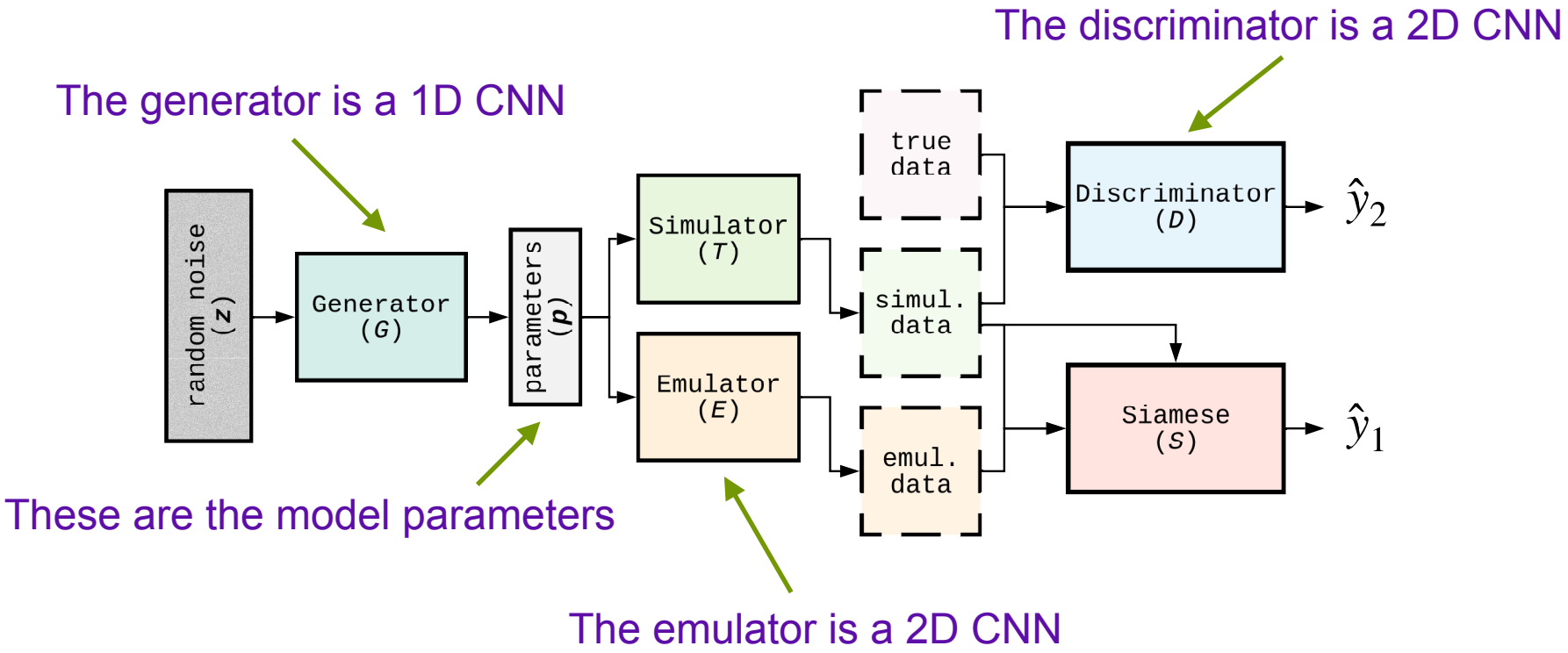
- To achieve those goals, the Model-Assisted GAN is a bit more complex than a standard GAN:

- Pre-training stage:
  - Train an *emulator* (E) to mimic the *simulation* (T) for the *same model parameters* using a *siamese network* (S)
  - This stage is similar to training a conditional GAN

- Training stage:
  - Train a *generator (G)* against a *discriminator* (D) to make a *model parameter vector* such that the *emulator* makes images to match true data

Model-Assisted GAN: Overview

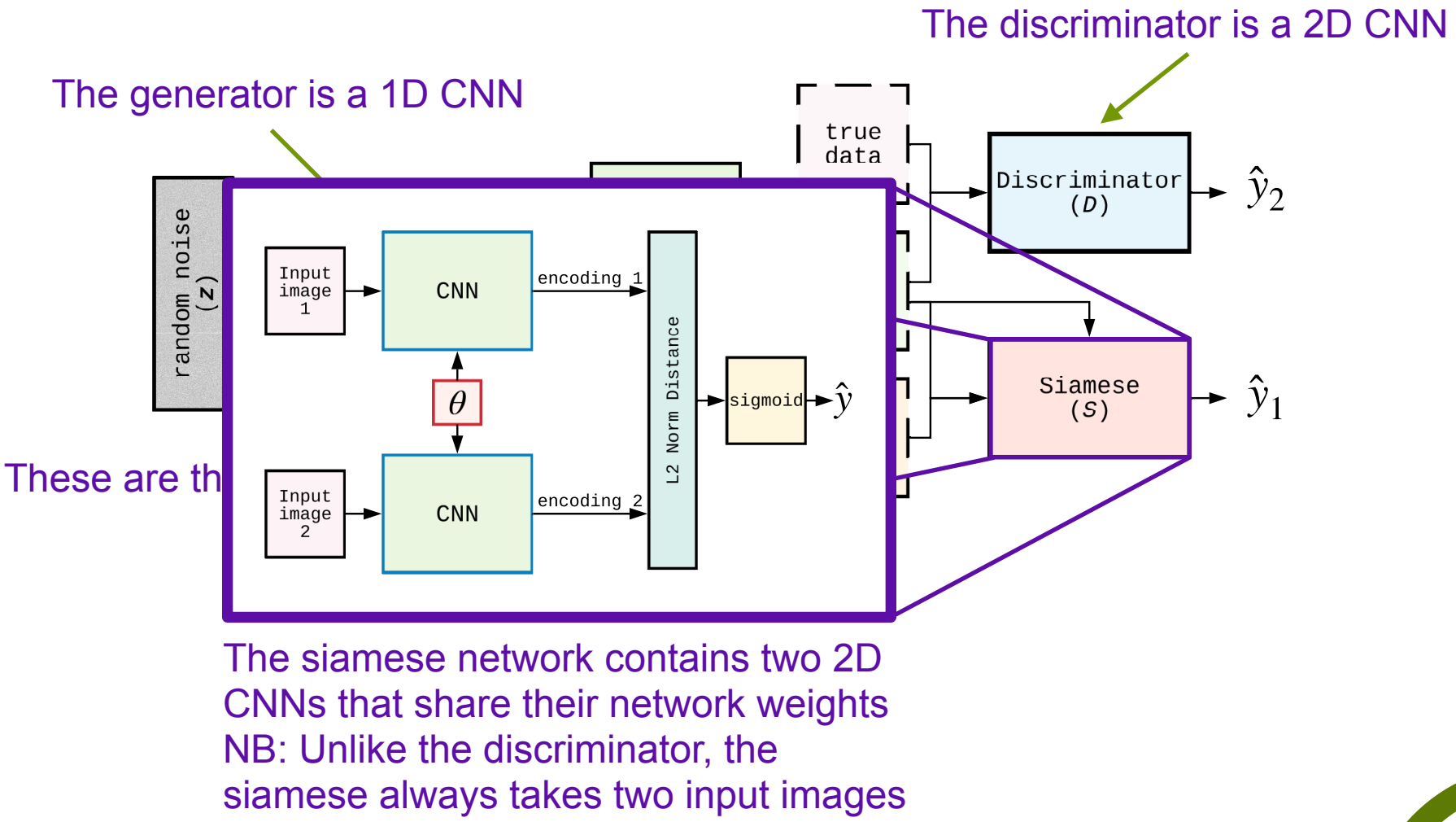
- The full architecture of the MAGAN:



- Similar to two GANs working together

Model-Assisted GAN

- The full architecture of the MAGAN:

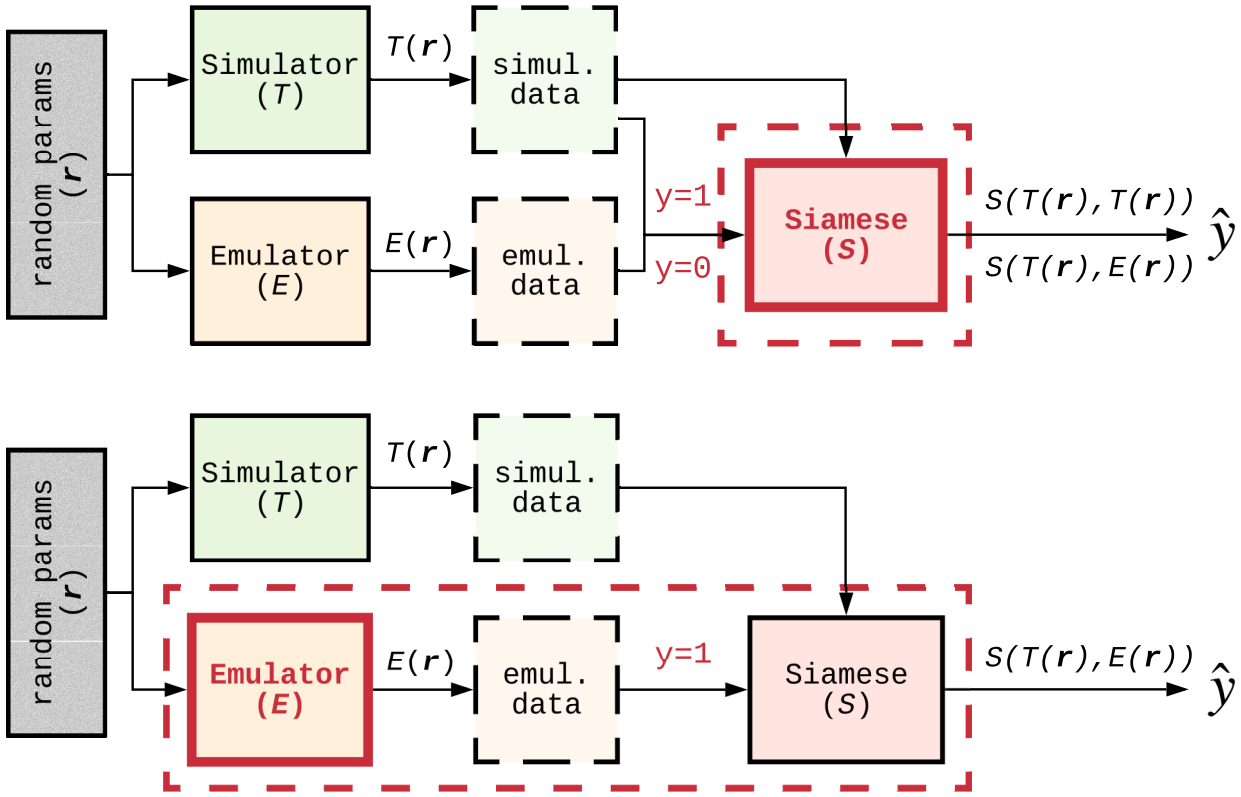


Pretraining step

- This is very similar to the standard GANs
  - The *siamese* here is doing a similar job to a discriminator

These are our physics model parameters

Choose random values *r* within some defined parameter space



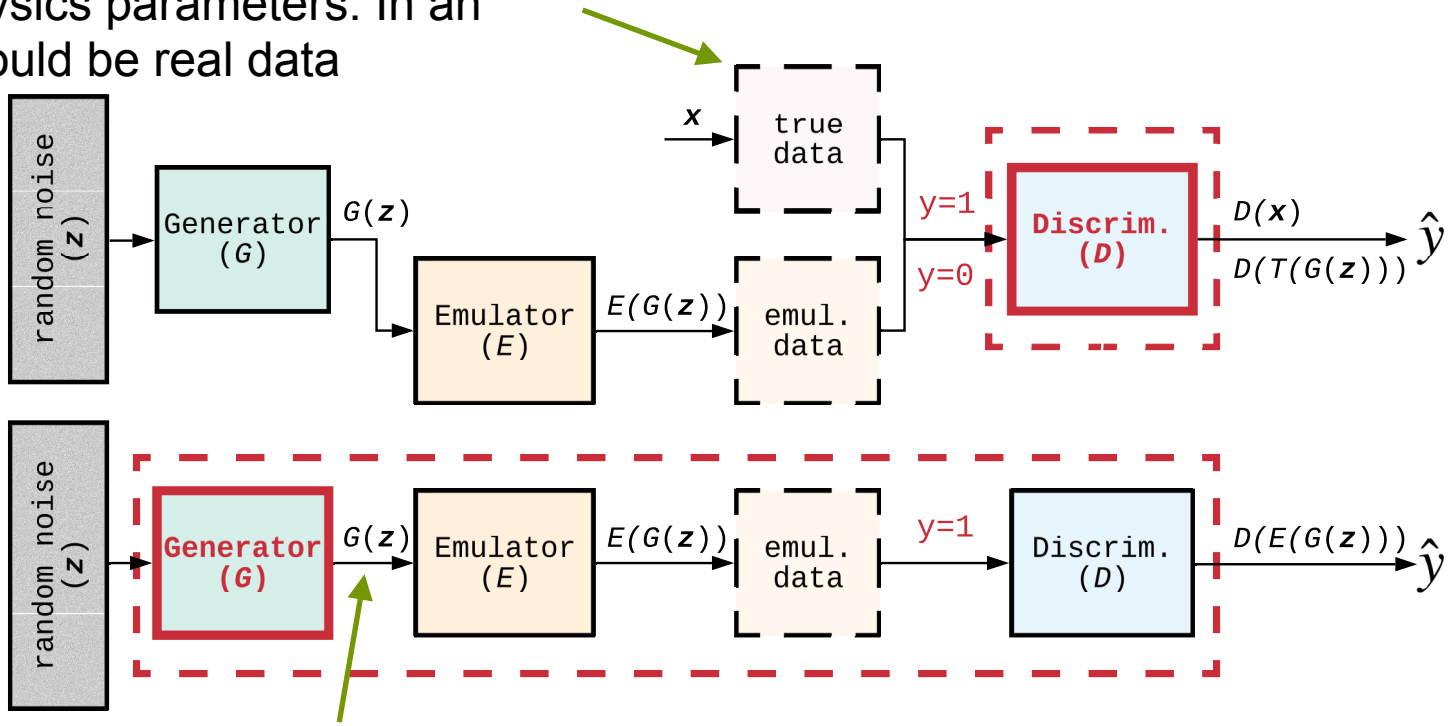
Post pretraining

- We now have an *emulator* that produces the same images as the *simulation* for all *model parameter values*
- We can use the *emulator* for fast simulation
  - It is effectively the generator of a conditional GAN
- We can now move on to the second goal of extracting the model parameters from a true data sample

Training step

- This stage allows us to extract the best matching physics parameters to the true data

Here, true data sample that we produce using some choice of physics parameters. In an experiment, this would be real data



G(z) is our generated vector of physics parameters

Image-Based Model Parameter Optimization Using Model-Assisted Generative Adversarial Networks

Saúl Alonso-Monsalve and Leigh H. Whitehead

Abstract— We propose and demonstrate the use of a model-assisted generative adversarial network (GAN) to produce fake images that accurately match true images through the variation of the parameters of the model that describes the features of the images. The generator learns the model parameter values that produce fake images that best match the true images. Two case studies show excellent agreement between the generated best match parameters and the true parameters. The best match model parameter values can be used to retune the default simulation to minimize any bias when applying image recognition techniques to fake and true images. In the case of a real-world experiment, the true images are experimental data with unknown true model parameter values, and the fake images are produced by a simulation that takes the model parameters as input. The model-assisted GAN uses a convolutional neural network to emulate the simulation for all parameter values that, when trained, can be used as a conditional generator for fast fake-image production.

Index Terms— Fast simulation, generative adversarial networks (GANs), model-assisted GAN, parameter optimization.

Case Study I

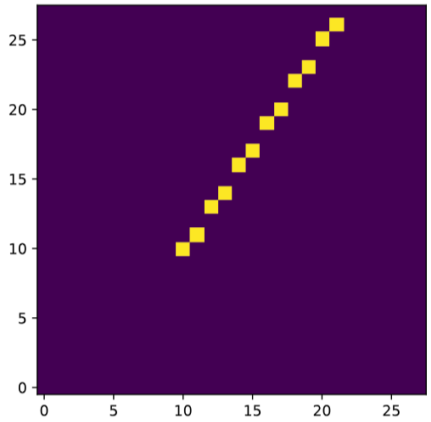
images that best match the true images such that $p_{\text{generator}}(\boldsymbol{p}_{\text{bm}}) = p_{\text{data}}(\boldsymbol{p})$. The default simulation for an experiment will typically have default model parameter values that do not exactly match the true model parameter values. The parameters $\boldsymbol{p}_{\text{bm}} \sim p_{\text{generator}}(\boldsymbol{p}_{\text{bm}})$ can be extracted to update the default simulation model parameters so that the fake images will more accurately reproduce the true images. Furthermore, the difference between $p_{\text{generator}}(\boldsymbol{p}_{\text{bm}})$ and the default parameters distribution gives physical insight into the understanding of the model and the model parameter values that were not correct in the default simulation. The key advantage of the model-assisted GAN is that the simulation only needs to be run once, and then, any number of model parameter optimizations can be performed with different true data sets very quickly. Another advantage is that any additional number of fake images can be produced efficiently using the emulator from the model-assisted GAN.

S. Alonso-Monsalve and L. H. Whitehead, "Image-Based Model Parameter Optimization Using Model-Assisted Generative Adversarial Networks," in *IEEE Transactions on Neural Networks and Learning Systems, 2020*

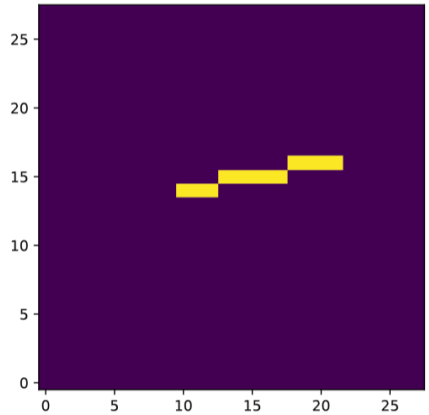
Case Study I - Outline

- Start simple: image containing a single line

$$y = mx + c$$

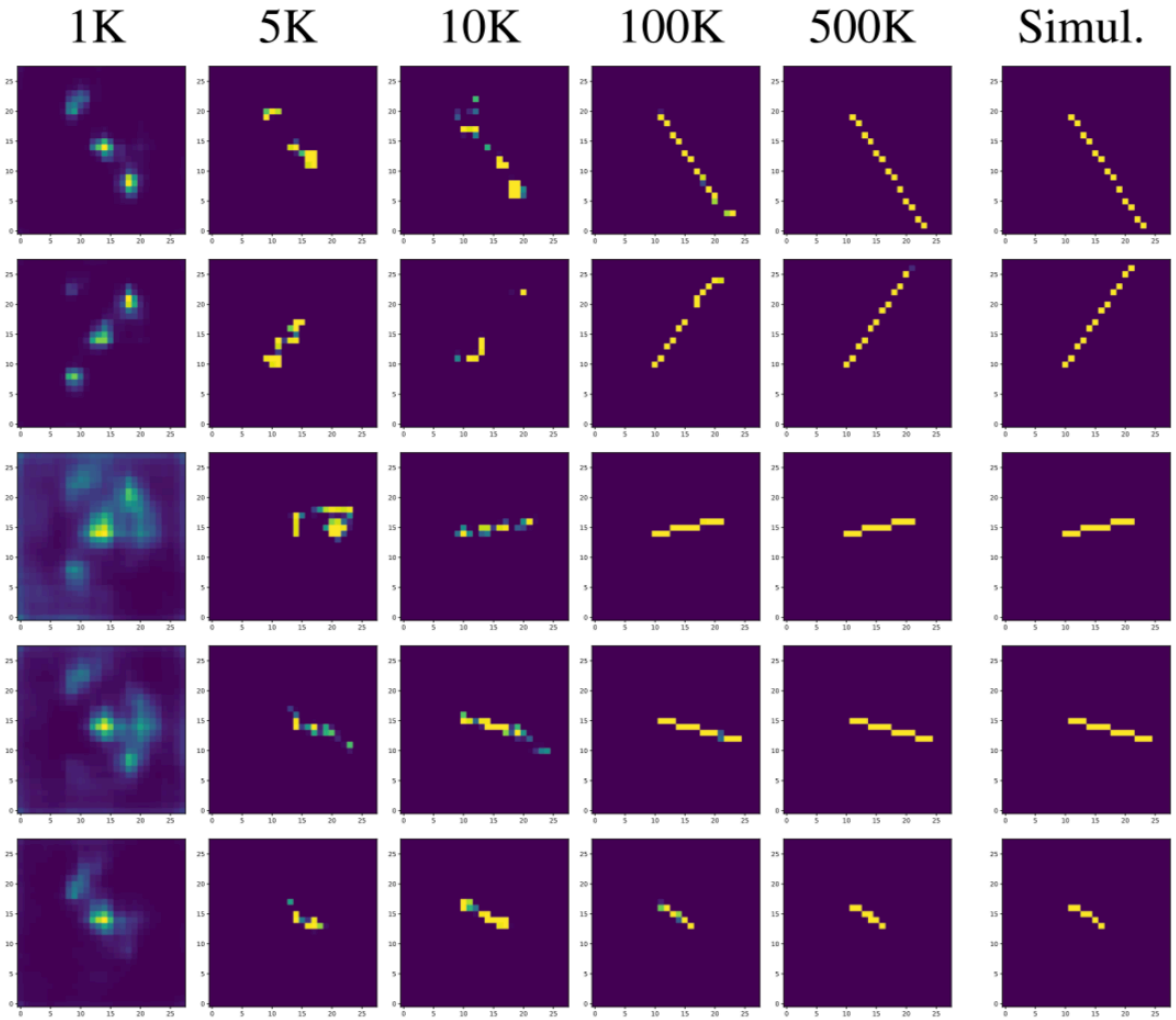


- Model parameters are hence:
  - Gradient *m*
  - Offset *c*
  - Start position in x: x_0
  - Length in x: x_{steps}



- Parameter space limited to values that produce lines that start and stop in the 28 x 28 pixel images

Case Study I - Pretraining



- Nice agreement after 500k steps!
- At this stage we now have a well-trained emulator

Case Study I - Training

- Firstly, we need to define a *true data* sample
  - Choose values with some smearing to create a range of different images

Parameter	m	x_0	c	x_{steps}
True μ	1.5	10.0	0.5	9.0
True σ	0.3	0.5	0.1	0.5

- Now, can we use the MAGAN to extract these values from the true data-set using the trained generator?
  - Training converged after 30k iterations
  - Run the trained generator to produce the mean and sigma of each parameter…

Case Study I - Results

- Firstly, we need to define a *true data* sample
  - Choose values with some smearing to create a range of different images

Parameter	m	x_0	c	x_{steps}
True μ	1.5	10.0	0.5	9.0
True σ	0.3	0.5	0.1	0.5
Best match μ	1.501	9.988	0.512	9.003
Best match σ	0.266	0.418	0.129	0.432

- Yes, we can!
  - Very good agreement between the extracted values and the true data sample

Image-Based Model Parameter Optimization Using Model-Assisted Generative Adversarial Networks

Saúl Alonso-Monsalve and Leigh H. Whitehead

Case Study II

Abstract— We propose and demonstrate the use of a model-assisted generative adversarial network (GAN) to produce fake images that accurately match true images through the variation of the parameters of the model that describes the features of the images. The generator learns the model parameter values that produce fake images that best match the true images. Two case studies show excellent agreement between the generated best match parameters and the true parameters. The best match model parameter values can be used to retune the default simulation to minimize any bias when applying image recognition techniques to fake and true images. In the case of a real-world experiment, the true images are experimental data with unknown true model parameter values, and the fake images are produced by a simulation that takes the model parameters as input. The model-assisted GAN uses a convolutional neural network to emulate the simulation for all parameter values that, when trained, can be used as a conditional generator for fast fake-image production.

Index Terms— Fast simulation, generative adversarial networks (GANs), model-assisted GAN, parameter optimization.

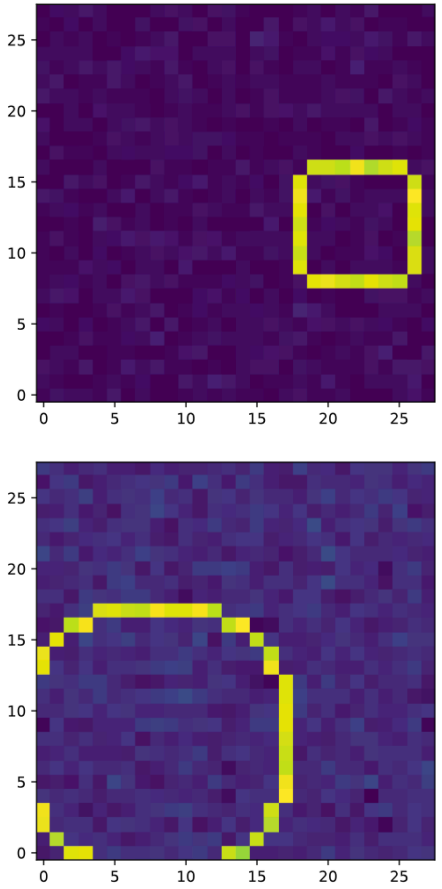
images that best match the true images such that $p_{\text{generator}}(\boldsymbol{p}_{\text{bm}}) = \delta(\boldsymbol{p}_{\text{true}})$. The default simulation for an experiment will typically have input model parameter values that do not exactly match the true model parameter values. The parameters $\boldsymbol{p}_{\text{bm}} \sim p_{\text{generator}}(\boldsymbol{p}_{\text{bm}})$ can be extracted to update the default simulation model parameters so that the fake images will more accurately reproduce the true images. Furthermore, the difference between $p_{\text{generator}}(\boldsymbol{p}_{\text{bm}})$ and the default parameters distribution gives physical insight into the understanding of the model and the model parameter values that were not correct in the default simulation. The key advantage of the model-assisted GAN is that the simulation only needs to be run once, and then, any number of model parameter optimizations can be performed with different true data sets very quickly. Another advantage is that any additional number of fake images can be produced efficiently using the emulator from the model-assisted GAN.

S. Alonso-Monsalve and L. H. Whitehead, "Image-Based Model Parameter Optimization Using Model-Assisted Generative Adversarial Networks," in *IEEE Transactions on Neural Networks and Learning Systems, 2020*

Case Study II - Outline

- More complex example with additional intensity variation and noise

$$x^2 + y^2 = r^2$$

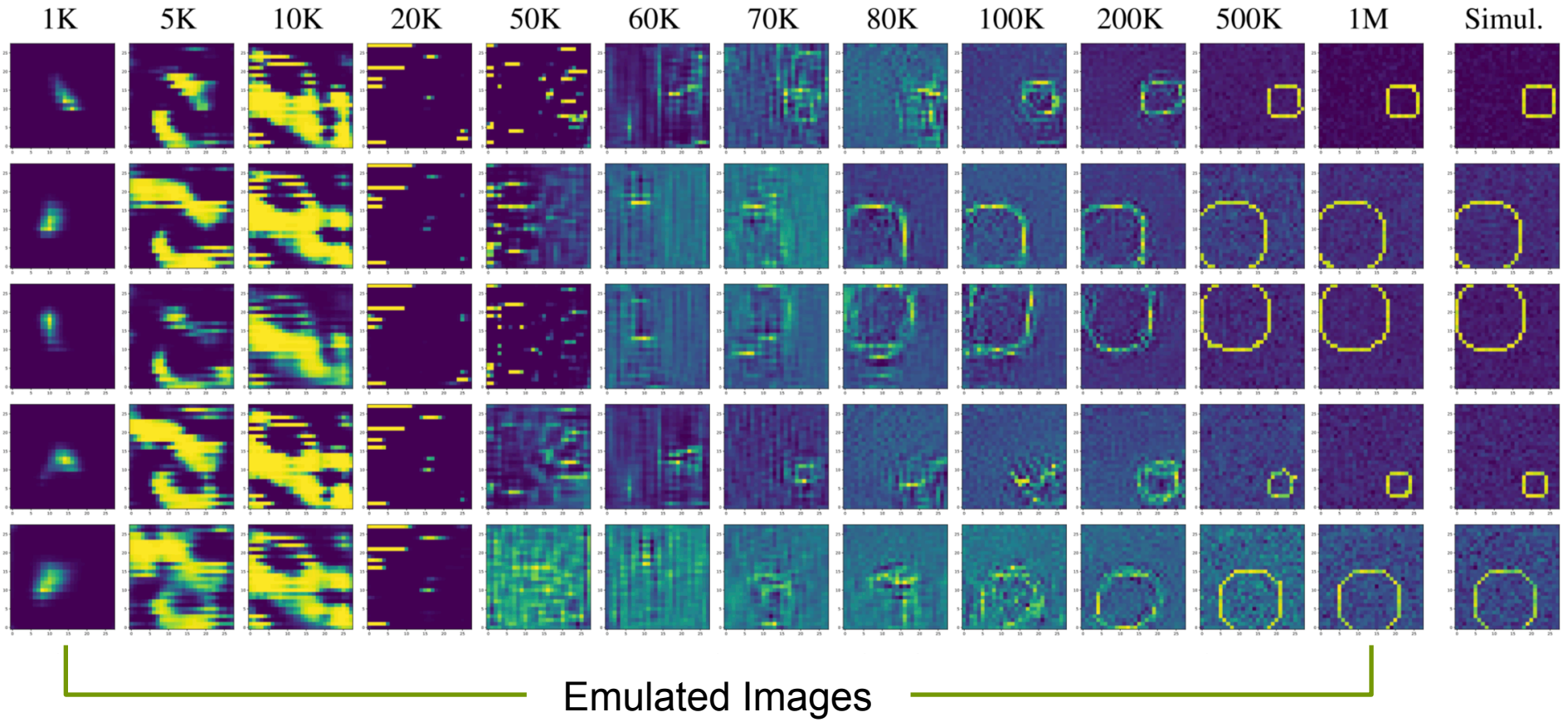


- Model parameters are hence:
  - Geometric: x_0, y_0, r (centre and radius)
  - Brightness b
  - Random noise intensity n

- Parameter space limited to values that produce circles with centres inside the 28 x 28 pixel images

Case Study II - Pretraining

- We ran the pre-training for 1 million iterations

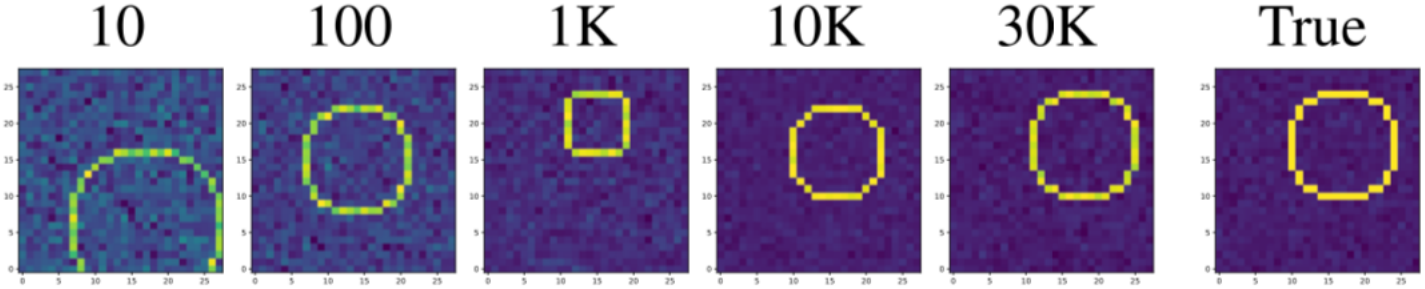


Case Study II - Training

- As before, we define a *true data* set with a mean and sigma for each parameter

Parameter	x_0	y_0	r	n	b
True μ	18	18	8	0.15	0.9
True σ	0.8	0.8	0.5	0.05	0.05

- Train the generator for 30k iterations
  - Below shows images produced from the generator parameters



Case Study II - Results

- As before, we define a *true data* set with a mean and sigma for each parameter

Parameter	x_0	y_0	r	n	b
True μ	18	18	8	0.15	0.9
True σ	0.8	0.8	0.5	0.05	0.05
Best match μ	18.082	18.035	7.950	0.152	0.892
Best match σ	0.833	0.795	0.411	0.072	0.083

- We see nice agreement again on this tougher challenge

Light simulation in DUNE

Introduction

- Light simulations can be memory and CPU intensive
  - Ray-tracing in run-time simulation not feasible for DUNE
  - We currently use a look-up library
- For some (x,y,z) voxel we get the amount of light observed by each photon detector
- This case study only uses the first step of the MAGAN
  - Produces an *emulator* for fast simulation
- The model parameters are just (x,y,z)

Input definitions

- Divide the detector volume into 100 x 100 x 300 voxels
  - Geometric extent of the inputs:

Dimension	Minimum (cm)	Maximum (cm)
X	-379.662	379.662
Y	-658.09	699.59
Z	-302.946	1443.5

- Thus voxel size = (7.6cm x 13.6cm x 5.8cm)

- These three million voxels form the training sample

Detector Geometry

- The detector geometry* used contains 120 photon detectors embedded in the detector readout planes
- There are 12 readout planes in the centre of the detector
  - Two planes high, six planes long
- Each plane contains 10 photon detectors

* this is a smaller test geometry used by the collaboration, not the full far detector geometry

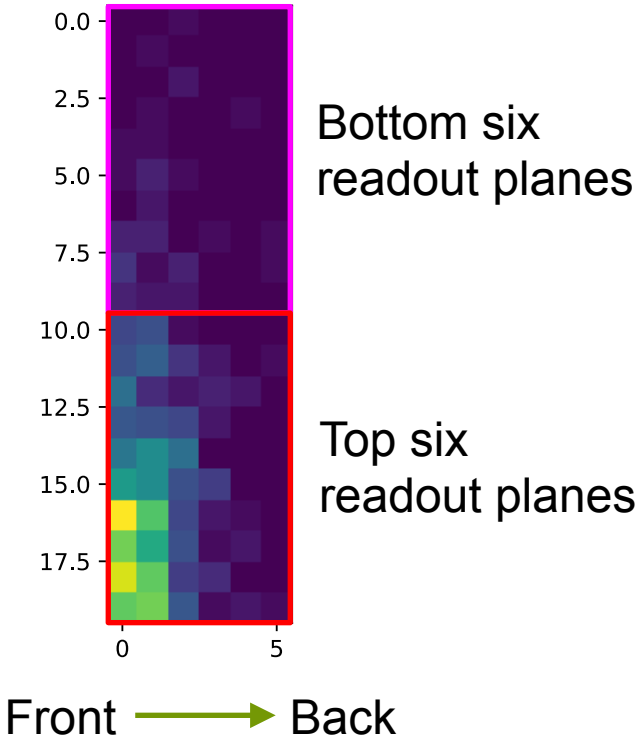
Detector Geometry

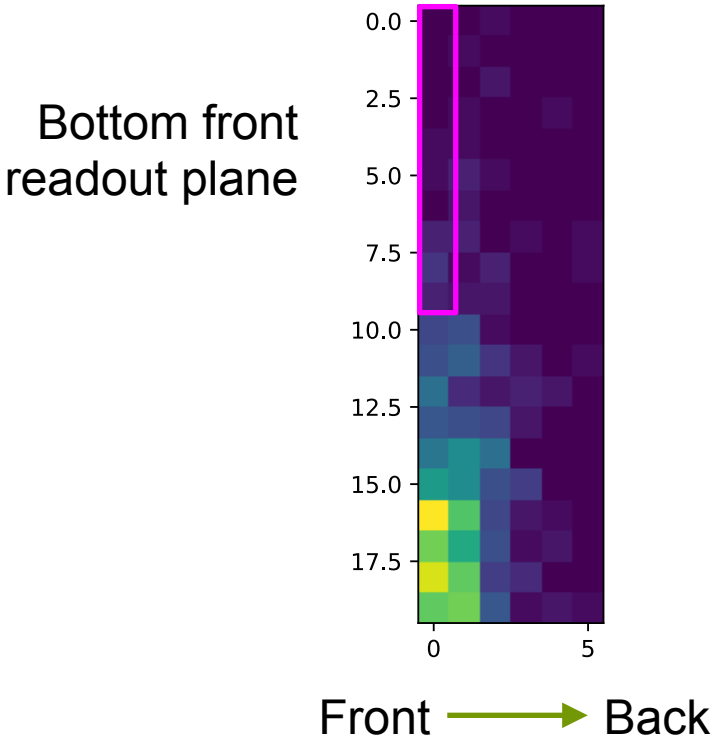
- The detector geometry* used contains 120 photon detectors embedded in the detector readout planes
- There are 12 readout planes in the centre of the detector
  - Two planes high, six planes long
- Each plane contains 10 photon detectors

* this is a smaller test geometry used by the collaboration, not the full far detector geometry

Image Format

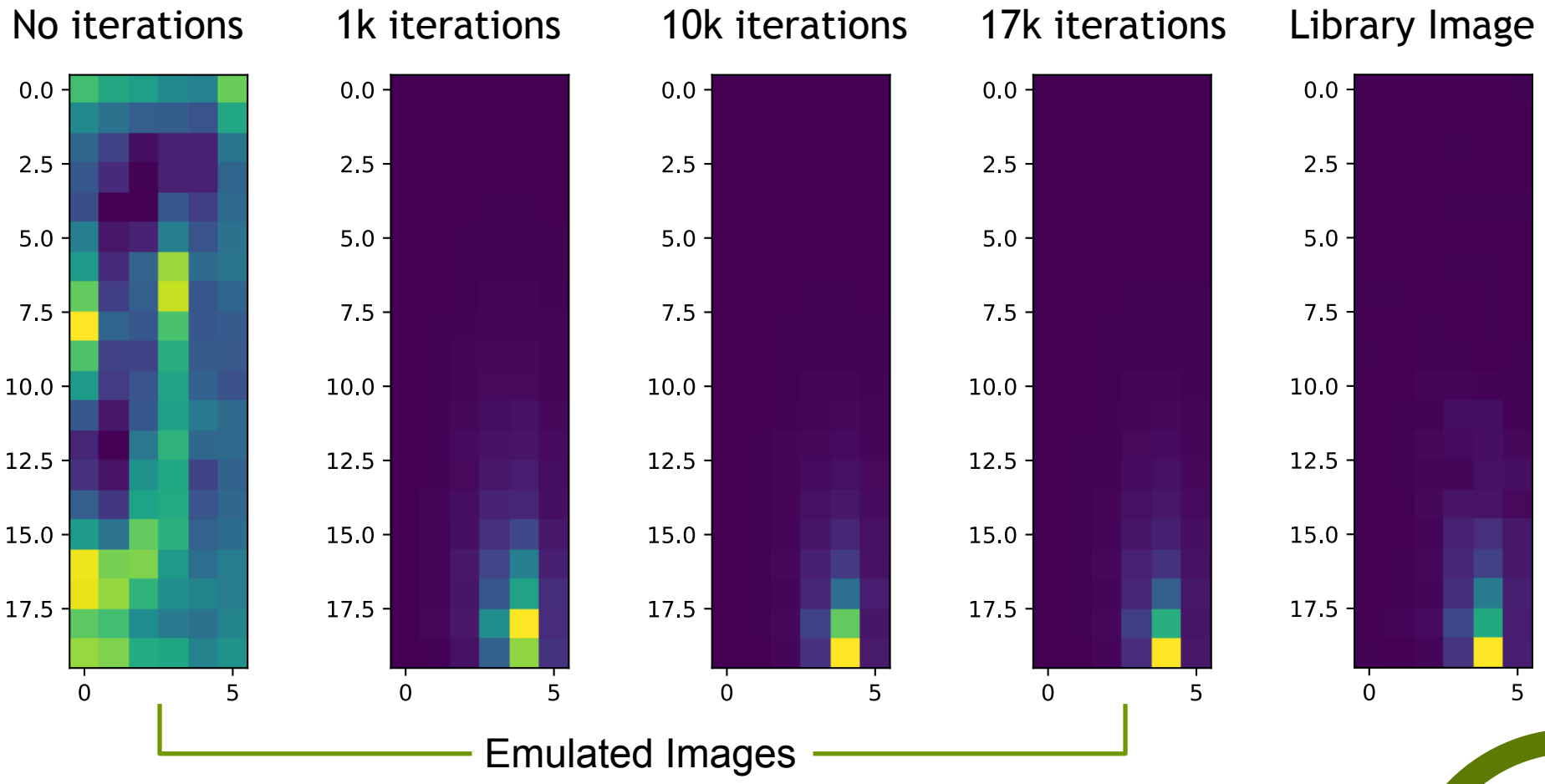
- The images are 20 x 6 pixels
  - Two readout planes high, and six readout planes wide





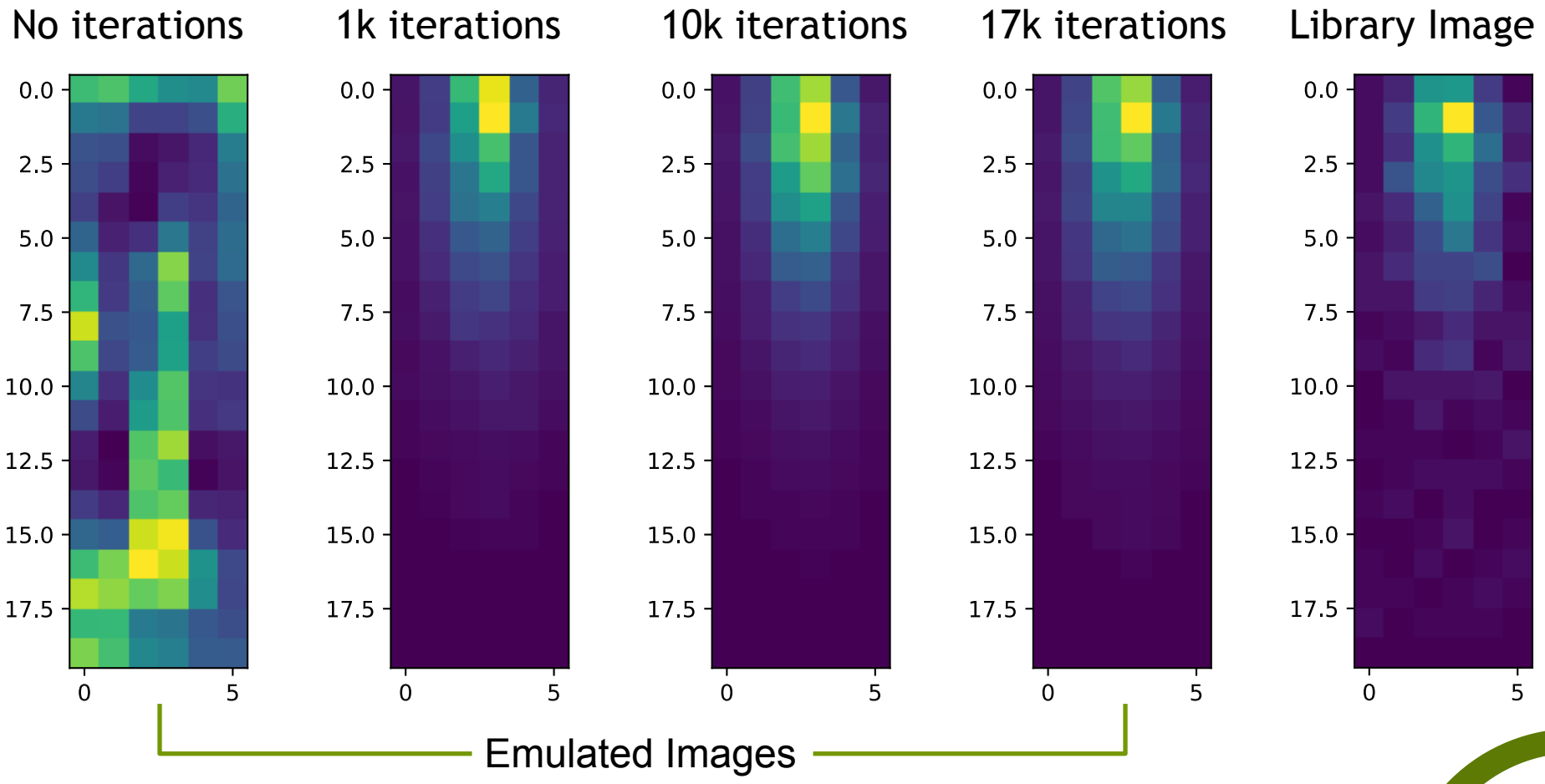
Example Image I

- We trained for roughly 17k iterations



Example Image II

- We trained for roughly 17k iterations



Can the MAGAN do interpolation?

- The size of the voxels used for training was arbitrary
  - Check the behaviour on smaller scales to see if the GAN varies smoothly between training voxels

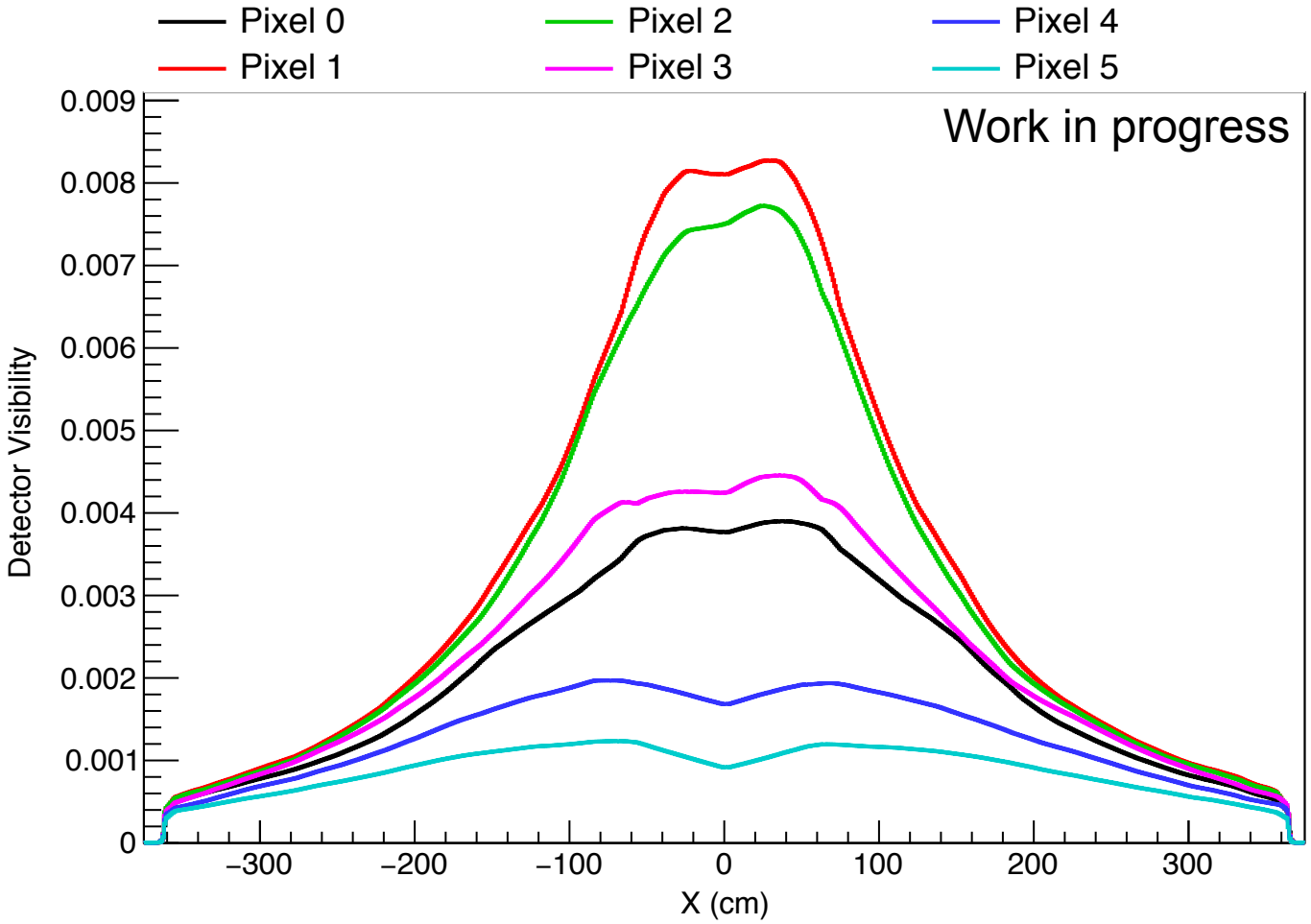
- Test 1: Take 1cm steps in *x* at fixed (250,300) in (*y*,*z*) and plot the light level in six of the photon detectors

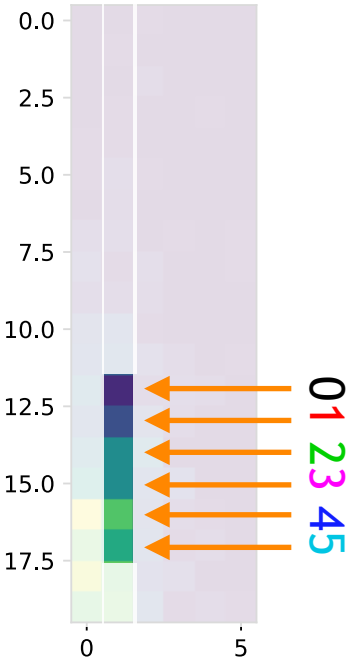
- Test 2: Take 1cm steps in *y* at fixed (200,300) in (*x*,*z*) and plot the light level in six of the photon detectors

- Test 3: Take 1cm steps in *z* at fixed (200,250) in (*x*,*y*) and plot the light level in six of the photon detectors

Test 1: Scan in *x* direction

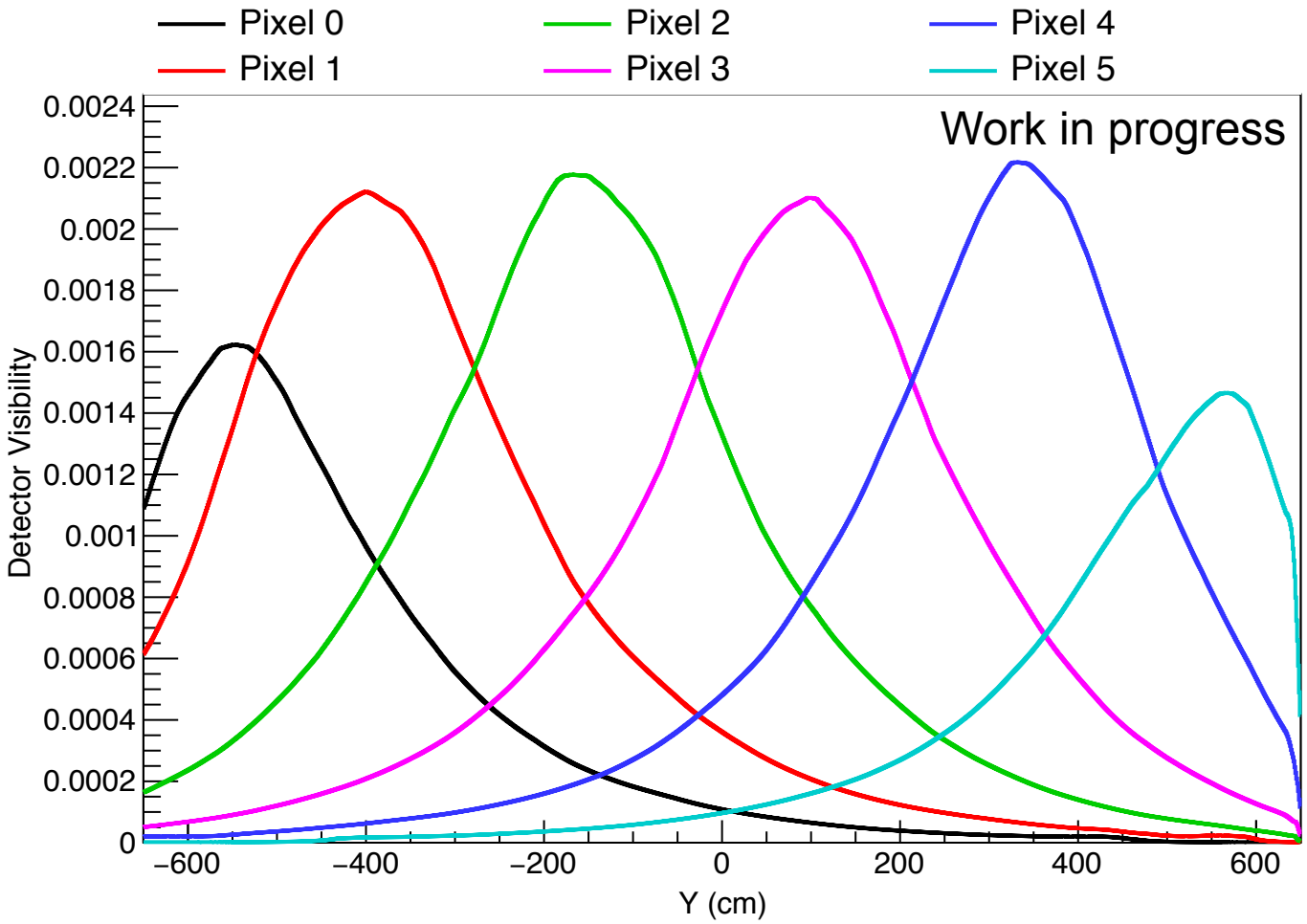
- Take 1cm steps in *x* at fixed (250,300) in (*y*,*z*)

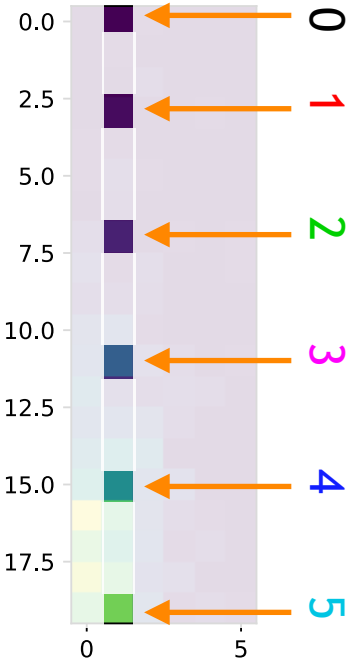




Test 2: Scan in *y* direction

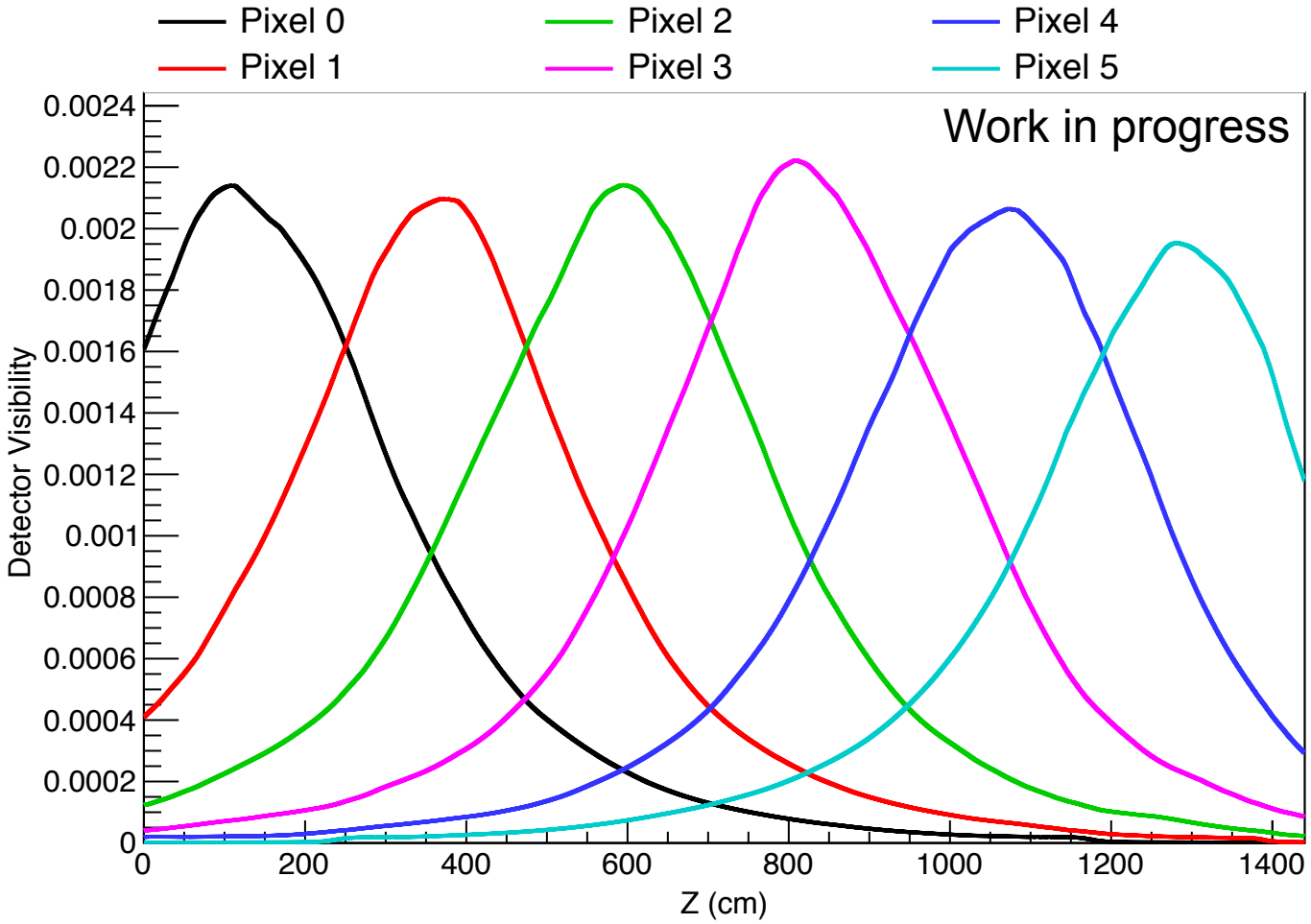
- Take 1cm steps in *y* at fixed (200,300) in (*x*,*z*)

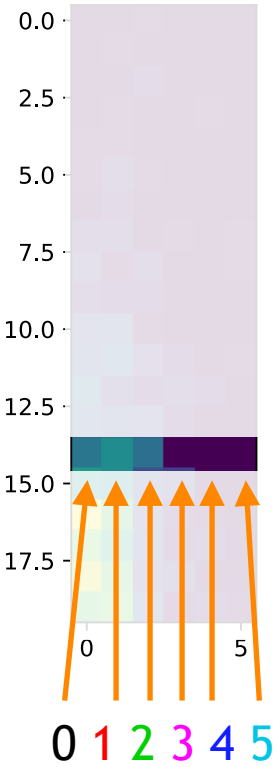




Test 3: Scan in *z* direction

- Take 1cm steps in *z* at fixed (200,250) in (*x*,*y*)





Performance Summary

- The GAN is able to reproduce the light maps from the photon library pretty well
  - We are still working to improve the *emulator* architecture to improve things further
- The *emulator* is fast: one million evaluations (full images) in approximately 40 (220) seconds on a GPU (CPU)
- The behaviour is smooth between the different voxelised training points
- Next we will define some metrics to rigorously define the agreement between the GAN and the library

Outlook and Summary

Ongoing work

- Everything shown here is for images (2D data)
- We are working on some 1D case studies
  - Likely more data best formatted in 1D than 2D
- Worked with an LHCb student at Cambridge to develop a 1D MAGAN for fast simulation of a sub-detector
  - Performance similar to a standard conditional GAN
- Interest from the wider community:
  - Biologist in the US who uses DNA to track populations
  - US astrophysicist modelling galaxy formation

Summary

- Generative Adversarial Networks are a powerful tool
  - They can provide fast simulations in HEP
- I have demonstrated the Model-Assisted GAN both in terms of fast simulation and parameter estimation
- We are happy to help and collaborate with any interested parties

Thank you for your attention!

https://ieeexplore.ieee.org/abstract/document/9032341

S. Alonso-Monsalve and L. H. Whitehead, "Image-Based Model Parameter Optimization Using Model-Assisted Generative Adversarial Networks," in *IEEE Transactions on Neural Networks and Learning Systems, 2020*