# Exploring neural-based 3D reconstruction methods of atom cloud for novel atomic interferometer experiments

Patin Inkaew
Department of Computer Science, Stanford University
pinkaew@stanford.edu

Armaun Sanayei
Department of Computer Science, Stanford University
armaun@stanford.edu

## Abstract

*In this project, we probe the performance of neural-based 3D reconstruction methods of atomic cloud for new atomic interferometer experiment, in particular, the MAGIS-100 experiment. We use simulated dataset of 3D density of atom cloud and corresponding 2D projection views as our dataset. We investigate the performance of two methods: (1) Voxel Reconstruction, and (2) Neural Volume Autoencoder. Our experiment shows that the first model can reconstruct the 3D information of 3D atom cloud from 2D projections while the second fails to capture the 3D geometry due to larger memory requirement and thus limited model size for the same computational resources. This project indicates several challenges in 3D reconstruction of atom cloud, including requirements for carefully crafted neural architecture and optimization procedures.*

## 1. Introduction

In this project, we explore 3D reconstruction of atomic cloud used in the imaging system of atomic interferometers, in particular, for the newly proposed MAGIS-100 experiment. Recently, atomic interferometers and atomic clocks have achieved higher sensitivity and precision, allowing physicists to utilize atomic interferometers to study precision physics: highly-precise testing of fundamental law of physics. Furthermore, atomic interferometers can be used to search for new elusive particles, new forces, as well as detect gravitational wave. Unlike current gravitational wave detectors, such as LIGO and VIRGO, atomic interferometers will have significantly smaller sizes in the order of meters, compared to kilometers. The Matter-wave Atomic Gradiometer Interferometric Sensor or MAGIS-100 is a novel proposed atomic infererometer being constructed at Fermilab. The MAGIS-100 experiment will use the latest technology in atomic interferometry with much more complex and sensitive sensor to probe and test quantum mechanical coherence fluctuations from a wide-variety of sources, including sources from large macroscopic distances.The experimental design of MAGIS-100 is shown in Figure 1.

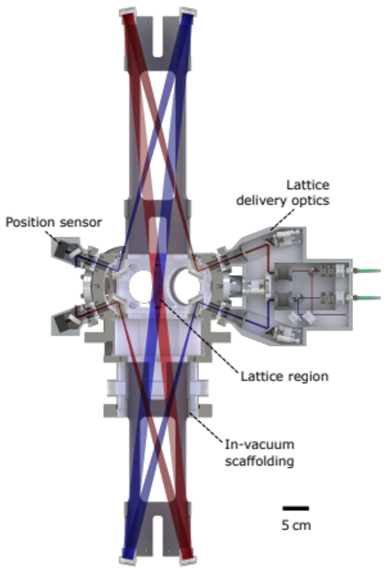


Figure 1. Experimental setup of MAGIS-100 atomic interferometer [1]

Unlike traditional interferometers relying on interference of electromagnetic waves such as laser light, the key principle of atomic interferometer is the interference of matter waves. Wave-matter duality has been proved in many experiments, including the popular double-slit experiment, and this phenomenon has became one of the core principle of quantum physics. In an atomic interferometer, compared

to light, a cloud of atoms is the media used to measure the interference phase difference between paths. Then, laser light is shot to the atom cloud at the right moment to split atoms in two quantum states, acting as if a beam spliter. Subsequent laser light is used to reflect the phase, playing a role of a mirror. By directing the laser lights to the cloud at the precise timing, several types of matter wave interferometers resembling the light interferometers, including Mach–Zehnder and conjugate Ramsey-Borde, can be constructed.

3D tomography of the atom cloud is a key part of the sensing mechanism of an atomic interferometer to measure the phase difference. Interference pattern collected from multiple cameras at different views are taken and used to reconstruct 3D locations of atoms and 3D shape of the cloud. This task is challenging because 2D images of interference patterns are represented as density of atoms and do not have distinct features, such as parallel lines or vanishing points, nor clear methods to compute corresponding points, so classical methods like epipolar geometry cannot be straightforwardly applied. Another limitation include mechanical restrictions: cameras are fitted on the apparatus and the experimental setup restrict the position and number of cameras. The current proposed design of MAGIS-100 detector has only 6 camera views. The examples of simulated 6 views of atom cloud in shown in Figure 2.

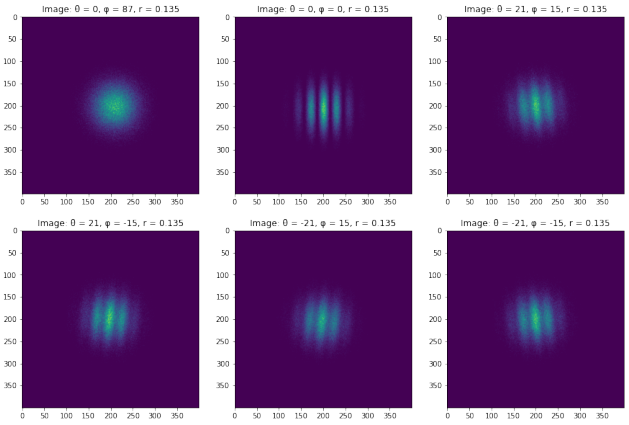


Figure 2. Six views of atomic cloud from simulation

These cameras provide 2D views at different positions of the atom cloud, and the goal is to be able to reconstruct an accurate position of the 3D cloud as well as.

This 3D reconstruction problem is one that computer vision is designed to solve; however, this problem has very unique features that make it more difficult than traditional 3D reconstruction.

## 2. Background and Related Work

### 2.1. 3D Reconstruction of Atom Clouds

The unique obstacles of this 3D reconstructions problems fall into two categories: experimental instrument restrictions and properties of the object being reconstructed. For experimental instrument restrictions, the camera positions are limited due to engineering constraints of the cavity. This limits the number of cameras to six and their positions. A second experimental restriction is Poisson noise from the cameras when imaging the atom cloud. The other category of problems come from the physical properties of the atom cloud. An atom cloud tends to have little structure. Given that the object is a cloud, the atom cloud has no well-defined edges. Moreover, when imaged the atom cloud does not have color; when color is available, it allows for better shape detection given that three channels provide more data; however, this is not available in this case.

A traditional approach to 3D reconstruction from a computer vision perspective is epipolar geometry. This, however, cannot be straightforwardly applied given the nature of the atom cloud as well as the multiple camera views. Moreover, epipolar geometry is object independent; however, in the case of atom cloud, it is clear that a method trained specifically on atom clouds would be more useful and accurate for atom cloud reconstruction. To avoid these difficulties and to make use of all camera views, in this project we explore neural-based approaches.

### 2.2. Related Work

#### 2.2.1 Neural Volume

Neural Volume [5] is a early neural-based methods used to reconstruct input 2D camera views by reconstructing 3D volume. This model is heavily based on autoencoder [2]. The model encoder several camera views to latent space and then decode these latents to recreate 3D voxel output. Then, generated 3D voxel outputs are rendered through volume rendering procedure to produce 2D projection views. These reprojection views are then compared to the input camera views. The model is trained to produce 3D voxel output generating similar 2D projections compared to input camera views.

#### 2.2.2 Neural Radiance Fields (NeRFs)

Neural Radiance Fields (NeRF) [6] is a state-of-the-art method used to tackle novel view synthesis: generate new views of a 3D object given a set of 2D views from specified positions and angles by synthesizing a 3D volume from given 2D projection views. Additionally, NeRF guarantees smoothness which is a desired results for the reconstruction of 3D atom cloud.

NeRF is a neural network algorithm. NeRF model takes in a 5D coordinate (spatial location $(x, y, z)$ and a viewing direction $[\theta, \phi]$ and outputs a volume density and color. To train the neural network, we input a set of images and camera poses, which we have available. For example, this is a given scene generated from the simulation algorithm with camera positions and the atom cloud.

Moreover, NeRF has been shown to be powerful for a sparse set of input views. NeRF is best for static scenes, which is the main problem we are tackling: many views from a few positions.

The architecture of NeRF extends continuous fully-connected layer, also known as a multi-layer perceptron (MLP) to 5D neural radiance fields representation, an MLP layer mapping 3D location and 2D viewing direction to volume density and view-dependent emitted radiance. The model weights of the MLP inside NeRF can capture a volumetric scene representation. To get a true 3D volumetric rendering (e.g. as a point cloud) from these weights, one can use a numerical integration method since the neural network gives us all the views.

## 3. Approaches

In this project, we compare the performance of two reconstruction methods. We use (1) Voxel Reconstruction proposed by SLAC MAGIS group as our baseline model, and (2) Neural Volume Autoencoder.

### 3.1. Reprojection error

Reprojection error is the key error/training loss/metrics used in all approaches. Reprojection error is an error compares the regenerated projections from model outputs with ground truth 2D projection images. Reprojection error uses three inputs, including the proposed 3D voxel density, the position and angle of the camera, and a true image taken from that camera for training. To compute reprojection error, through volume rendering, we project the 3D voxel density output from the model to given position and angle of the camera to obtain 2D reprojection images that would have been observed by the camera at the given camera poses. This 2D image is then compared with the input training image from that camera. In this paper, we use mean square error (MSE) loss with sum reduction to compare reprojections with ground truth images.

### 3.2. Voxel Reconstruction

The baseline method that we used in this project is voxel reconstruction. The voxel reconstruction technique is inspired by voxel carving in that it iteratively goes through the different views and carves out a voxel density model, starting from a full cube and ending with a carved atom cloud denisty voxel grid.

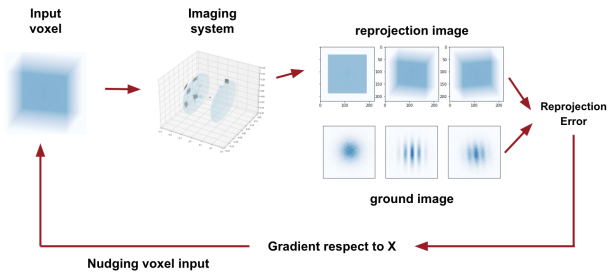


Figure 3. Voxel Reconstruction Process

Reconstruction error is a key part of the voxel reconstruction algorithm. The voxel reconstruction algorithm starts off with a 3D voxel density that is a completely filled in cube. In other words, the voxel density grid is set to all ones. The voxel density grid along with single 2D image and camera position are input. The reprojection error is computed and from this reprojection error, we calculate a gradient on the voxel density. We then uses this gradient to nudge the input voxel density through backpropagation. We repeat this process for multiple iterations using the six different camera views until convergence of the reprojection error.

### 3.3. Neural Volume Autoencoder

We will refer to this method as simply Autoencoder (or AE) for short throughout this paper. This autoencoder neural net is a novel model we explored in this project to tackle atomic cloud reconstruction task. This method is inspired by the Neural Volume discussed in [5] used for face reconstruction; however, we designed the structure of the autoencoder ourselves. The autoencoder can be broken up into two parts: the encoder and decoder. We discuss the specifics of the architecture in the next few sections.

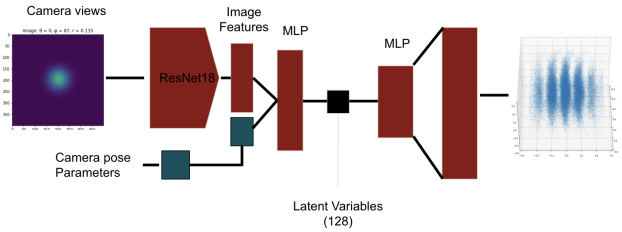


Figure 4. Architecture of the Autoencoder used in this paper

#### 3.3.1 Encoder

The inputs of the auto-encoder are a single camera view and camera pose parameters. A camera view is a gray-scale image of size 220 by 220. The camera pose parameters specify

the position of the camera in 3D space and its angle. Given that the image is a very structured input compared to the the camera pose parameters, the encoder has a pre-processing step as outlined below.

The encoder takes in each camera views. We use a pre-trained ResNet18 convolutional neural network. Since ResNet was trained on ImageNet dataset, we perform transfer learning by replacing the last fully-connected layer of ResNet18 with two linear layers with ReLU activation function. The CNN-encoding features output from this neural network is then concatenated with the camera pose parameters, and then fed through an encoding MLP that encodes all the data into latent variables for our encoding.

#### 3.3.2 Decoder

In the decoding step, we use stack of MLPs to upsampling the number of variables to the final number of voxels. In other words, given that the final voxel grid is of size $(a, b, c)$, we decode the encoding into $a \cdot b \cdot c$ variables and reshape that into a three dimensional tensor of size $(a, b, c)$. This output is the proposal voxel density.

#### 3.3.3 Loss

Similar to the voxel reconstruction method, the loss we use for training and gradient computation is the reprojection error. In this case, it fits in the auto-encoder pipeline since camera view and camera pose parameters are already part of the inputs of the auto-encoder every iteration.

## 4. Experiment, Results, and Discussion

### 4.1. Data Generation

Since MAGIS-100 experiment is still in the development stage, we use simulated dataset of atom cloud together with six camera views as our dataset. We simulate this dataset by simulator provided in diffoptics and Magis-simulator package made available by the MAGIS group at SLAC National Accelerator Laboratory. We use these 2D projection camera views to train our models, both voxel reconstruction and Autoencoder. Since simulating large dataset of atom cloud and corresponding camera views require intensive computation, we use virtual machine on google cloud platform with a single Nvidia Tesla K80 GPU. Examples of camera views are shown in Figure 5.

The simulator allows us to vary many aspect of the scene, including a focal length, magnification, sensor position, sensor resolution, sensor noise, and lens, allowing high diversity in our dataset. We briefly highlight generation processes in the following paragraphs.

Given these parameters listed above, we use the simulator to create a scene object that stores the ground truth 3D atom cloud, the sensor position, the noise parameters, and

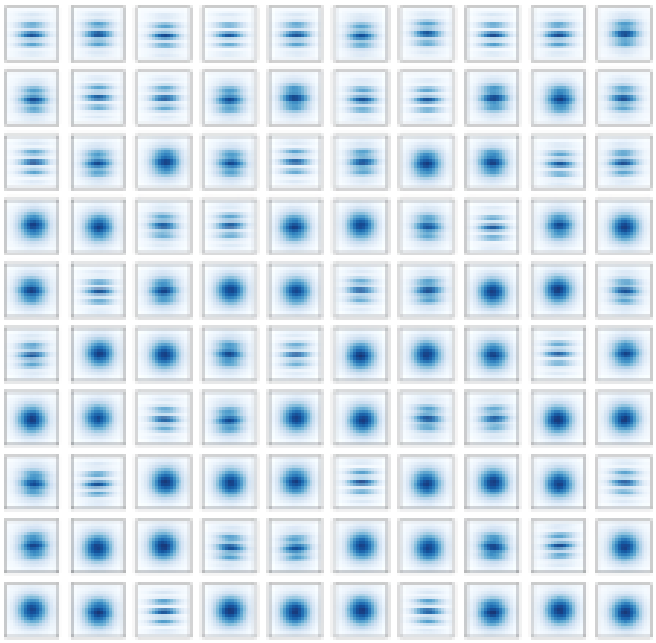

Figure 5. Examples of projection camera views at different angle of an atom cloud

other variables. We can then invoke methods of the object to generate rays from the atom cloud and intersect our rays with the sensor plane and determine the image projection from virtually any position. An example of the scene with a single sensor and a perfect lens is shown in Figure 6.

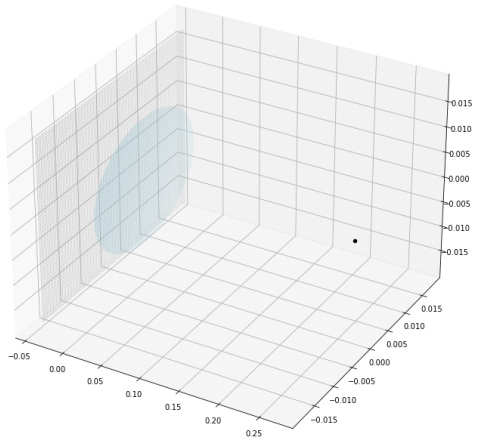


Figure 6. Example of scene setup with a single sensor. A black dot represent atom cloud, a bigger light blue disk represents a view port, a smaller dark disk represents a lens (not shown here), and a gray square represent camera sensor array (not shown here)

The sensor position is determined by a combination of

the focal length and magnification, namely, the x coordinate is $(-f(1+m))$ where $f$ is the focal length and $m$ is the magnification.

We can better visualize how the projections are simulated by sampling and plotting the rays originating from the atom cloud. An example of this visualization with 1000 outgoing rays is shown in Figure 7.

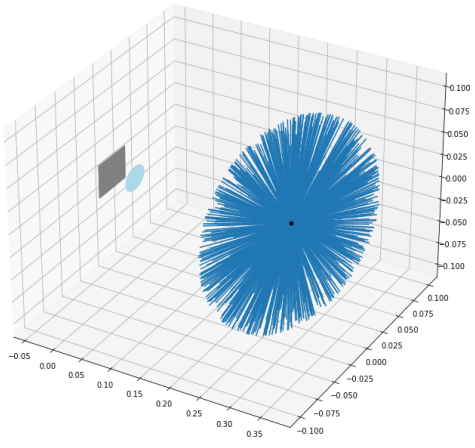

Figure 7. Rays traced from an atom cloud

These rays represent how we are able to get a projection on our simulated sensor. At our sensor location, we perform an intersection and sensor readout of sampled ray crossings and add Poisson noise. We can then save multiple images. In this simple visualization, we only have one image since we have one sensor in the scene. However, in general, we will produce multiple images with multiple camera sensor at different location and viewing angle. The generated 2D projection images will look similar to Figure 2. We perform projections and ray tracing with GPU to accelerate the simulation processes.

We use these 2D projection images as our training dataset and the original 3D configuration from which we are sampling as our ground truth for evaluation. Given the models we introduce in the coming sections, this simulation mechanism is ideal as the simulation mechanism also outputs 2D projections given a sensor position and angle.

In this experiment, we mimic the simulation setup with the proposed setup for MAGIS-100. At the current development stage of MAGIS-100, scientists are investigating the the possibility and quality of reconstruction with only six camera views at six different angles. We will use this camera constraints for our experiment. The visualization of camera setup is shown in Figure 8 and examples of the six reprojection views, together with the proposed cameras' positions and angles, are shown in 2.

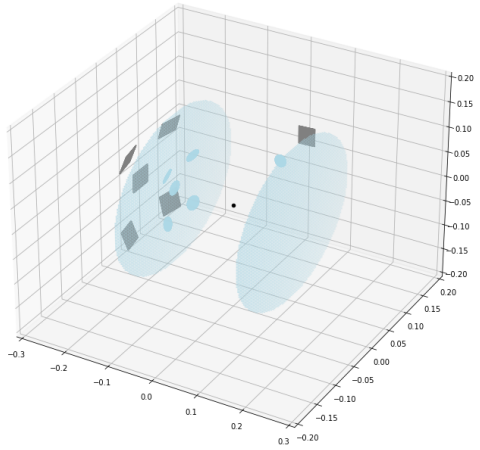

Figure 8. Camera positions and general scene resembling proposed imaging system for MAGIS-100 experiment

## 4.2. Experimental setups

We run all the experiments on google cloud platform with a single Nvidia Tesla K80 GPU. The important parameters of the models are shown in Table 1. Values for each parameter are picked to fit within the memory of the K80 GPU for training.

For training both models, we use ADAM [3] optimization with learning rate $10^{-4}$. We do not use learning rate scheduler in this experiment. We train both models on only one set of six image views to recreate the density producing the simulated 2D projection images. We train voxel reconstruction for 100 epochs and autoencoder for 200 epochs.

| method | parameter | value |
|---|---|---|
| Voxel Reconstruction | voxel size | 40 |
| Autoencoder | CNN-encoding dimension | 128 |
| | Poses dimension | 3 |
| | Latent dimension | 128 |

Table 1. Parameters used in Voxel Reconstruction and Autoencoder

## 4.3. Results and Discussion

### 4.3.1 Voxel Reconstruction

Plot of reprojection error/loss for Voxel Reconstruction is shown in Figure 9. The final reprojection error of Voxel Reconstruction is $30.91783$. Additionally, the evolution of 2D reprojection images generated by density output from the model at epoch $0, 50$, and $100$ are shown in Figure 10. The comparison between final reconstruction and ground truth/target 2D projection is shown in Figure 11.

From the loss plot, we can see that reprojection error gradually decreases and obtain significantly low final repro-

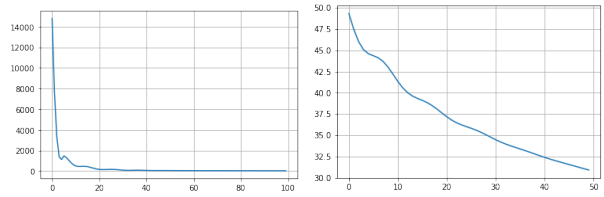

Figure 9. Reprojection error of Voxel Reconstruction across 100 epochs (left) and last 50 epochs (right)

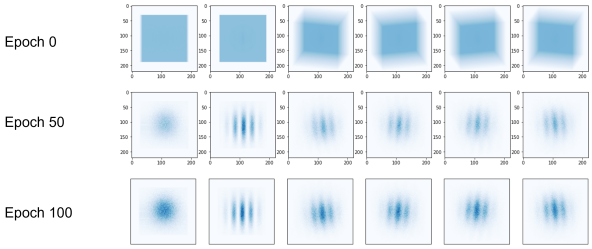


Figure 10. Reprojection images from Voxel Reconstruction at epoch 0, 50, and 100

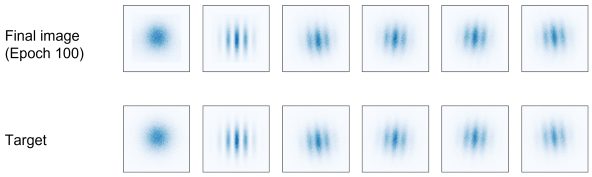


Figure 11. Comparison between final reprojection from Voxel Reconstruction and ground truth/target 2D projection

jection error, suggesting that the model can reconstruct 3D geometry of the atom cloud. To confirm, we can see that reprojection images gradually transform into ground truth reprojection images and the final reprojections are visually comparable to ground truth images. Since this method performs well, we can use the output density from the model to plot the atom cloud density in 3D, as shown in Figure 12. We see that this reconstruction visually agrees with the expect shape of atom cloud seen in atomic interferometer.

### 4.3.2 Autoencoder

Graph of reprojection error/loss for Autoencoder is shown in Figure 13. The final reprojection error is 3468.23633.

Similar to Voxel Reconstruction, we also monitor the evolution of generated projection images at epoch 0, 100, and 200, which are shown in Figure 14. The comparison between final reprojection images and ground truth/target 2D projections is shown in Figure 15.

Even though we also observe that reprojection error during training of autoencoder is decreasing, the final reprojection error is about 3000 which is much larger than that of

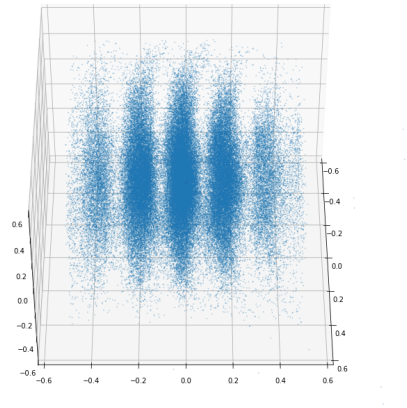

Figure 12. 3D plot of atom cloud density from Voxel Reconstruction

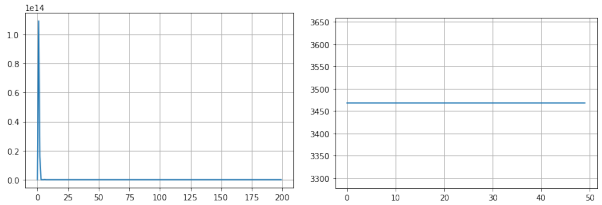

Figure 13. Reprojection error of Autoencoder across 200 epochs (left) and last 50 epochs (right)

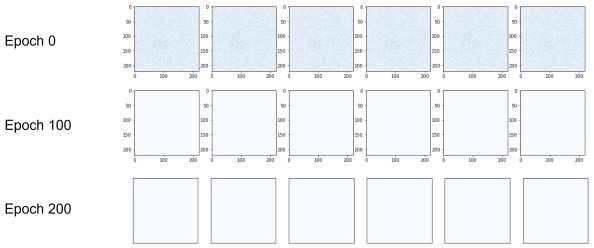


Figure 14. Reprojection images from Autoencoder at epoch 0, 100, and 200

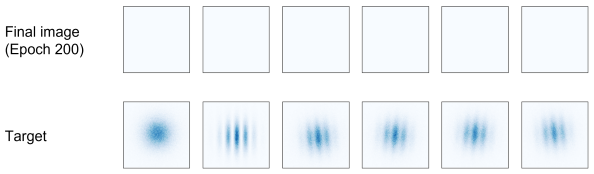


Figure 15. Comparison between final reprojection from Autoencoder and ground truth/target 2D projection

Voxel Reconstruction at about 30. From the loss plot at the last 50 epochs, we see that the reprojection error reaches a plateau and is saturated, suggesting that the training is stuck

in a local minimum.

From the visualization of generated reprojections, we can see that the Autoencoder entirely fails to reconstruct the 3D density of the atom cloud. As shown in the graph of reprojection error and visualization, the generated output reprojections all converge to constant, suggesting the model cannot reconstruct the 3D density of atom cloud and is underfit.

We propose that this underfitting of the model is due to the very limited dimension of CNN-encoding feature dimension and latent dimension used in this experiment. Clearly, these dimensions are significantly smaller than the dimension of the 3D atom cloud. Thus, underfitting behavior would be expected. We have tried to experiment with large latent dimensions; however, the memory of a single K80 GPU is simply not enough to load all model weights. Unfortunately, we did not have access to a GPU with a larger memory. Other potential reasons include: (1) no learning rate schedule, (2) no better training data schedule, and (3) transfer learning gap. ResNet18, used to encode camera views, was trained on the ImageNet dataset with natural images which are largely different from atom cloud projection images.

## 5. Conclusion

This study investigates the performance of two neural-based 3D reconstruction methods of atom cloud for atomic interferometer: (1) Voxel Reconstruction, and (2) Neural Volume Autoencoder. Our experiment shows that Voxel Reconstruction performs well and is able to reconstruct sensible 3D density and generated corresponding projection views that are visually comparable to the provided ground truth projections. On the other hand, the Autoencoder model suffers to reconstruct the 3D density of the atom cloud and the reprojection images are largely misaligned with the ground truth. We propose that this underfitting behavior occurs due to computational constraints limiting the size of the model. Voxel Reconstruction method does not suffer from this limitation because this method does not need to store model weights as in the Autoencoder. We also suspect other potential reasons, such as lack of proper training scheduling and transfer learning gaps.

The main contribution of this project is that we show that 3D reconstruction of atom cloud is a significantly challenging task. First, this is a novel task with a limited number of data and studies; hence, transfer learning is not quite effective. Second, even though 2D projection images only have one channel compared to 3 channels in colored images, these projection images have less prominent features, such as clear edges, boundary, and particularly color. Thus, this task is significantly difficult to pinpoint necessary features for reconstruction. As a result, it is not straightforward to apply traditional methods like epipolar geometry for the reconstruction of atom clouds since they require clear detection of corresponding points in different views or clear edges or boundaries. Thus, we explore neural-based reconstructions. However, this task is still challenging for neural network-based methods as we notice from the performance of the Autoencoder. The 3D reconstruction of atom cloud requires carefully crafted neural architecture, optimization procedures, and potentially high computational resources to obtain high-quality 3D reconstruction.

For future work, one can explore a larger latent dimension (which require larger computational resources) to be able to capture all 3D structure of atom cloud from 2D projection views. Ones can also explore different architectures, including Neural Volume Rendering such as [4], Neural Radiance Fields (NeRFs) [6], and its variances such as [7]. Specifically, NeRF-class architecture may perform better since its encoding of camera views and camera poses are significantly more meaningful in our experiment. Sill, it can suffer from a limited number of camera views. Lastly, ones can also explore better training scheduling, including learning rate adjustment to avoid saturation in a local minimum and better training scheduling, from relatively easy data samples to more difficult datapoints.

## 6. Acknowledgement

We would like to thank Professor Ariel Schwartzman, Murtaza Safdari, Maxime Vandegar, and other members of MAGIS group at the SLAC National Accelerator Laboratory for this project collaboration, providing resources, simulation packages for generating the dataset, and baseline Voxel Reconstruction method.

## References

[1] M. Abe, P. Adamson, M. Borcean, D. Bortoletto, K. Bridges, S. P. Carman, S. Chattopadhyay, J. Coleman, N. M. Curfman, K. DeRose, T. Deshpande, S. Dimopoulos, C. J. Foot, J. C. Frisch, B. E. Garber, S. Geer, V. Gibson, J. Glick, P. W. Graham, S. R. Hahn, R. Harnik, L. Hawkins, S. Hindley, J. M. Hogan, Y. Jiang (), M. A. Kasevich, R. J. Kellett, M. Kiburg, T. Kovachy, J. D. Lykken, J. March-Russell, J. Mitchell, M. Murphy, M. Nantel, L. E. Nobrega, R. K. Plunkett, S. Rajendran, J. Rudolph, N. Sachdeva, M. Safdari, J. K. Santucci, A. G. Schwartzman, I. Shipsey, H. Swan, L. R. Valerio, A. Vasonis, Y. Wang, and T. Wilkason. Matter-wave atomic gradiometer interferometric sensor (magis-100). *Quantum Science and Technology*, 6(4):044003, Jul 2021. 1

[2] G. E. Hinton and R. R. Salakhutdinov. Reducing the dimensionality of data with neural networks. *Science*, 313(5786):504–507, July 2006. 2

[3] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization, 2017. 5

[4] D. B. Lindell, J. N. P. Martel, and G. Wetzstein. Autoint: Automatic integration for fast neural volume rendering, 2020. 7

[5] S. Lombardi, T. Simon, J. Saragih, G. Schwartz, A. Lehrmann, and Y. Sheikh. Neural volumes. *ACM Transactions on Graphics*, 38(4):1–14, Aug 2019. 2, 3

[6] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. In *European conference on computer vision*, pages 405–421. Springer, 2020. 2, 7

[7] T. Müller, A. Evans, C. Schied, and A. Keller. Instant neural graphics primitives with a multiresolution hash encoding. *arXiv:2201.05989*, Jan. 2022. 7