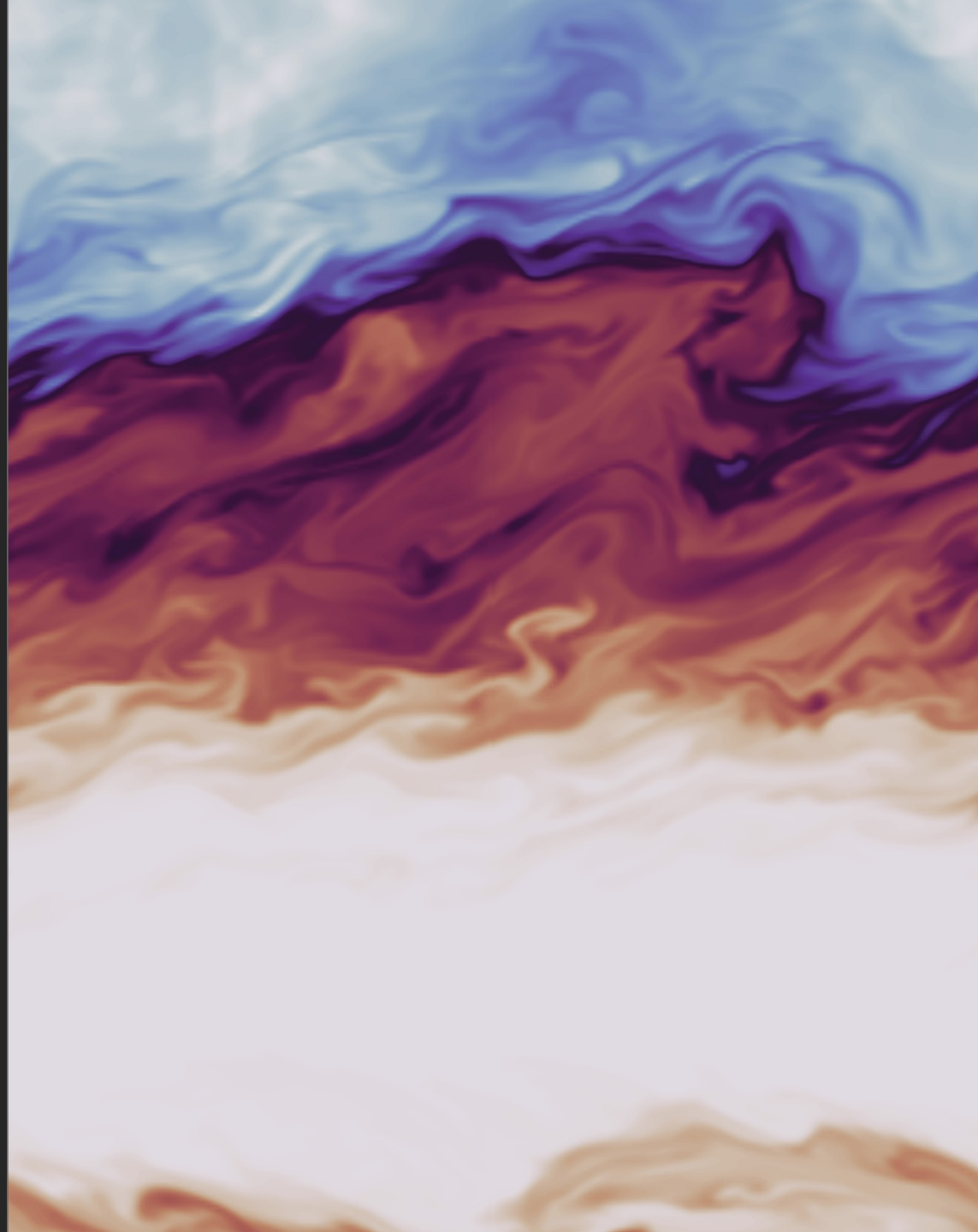


Deep learning for scientific data compression

Tomáš Brzobohatý, Petr Strakoš, Lubomír Říha

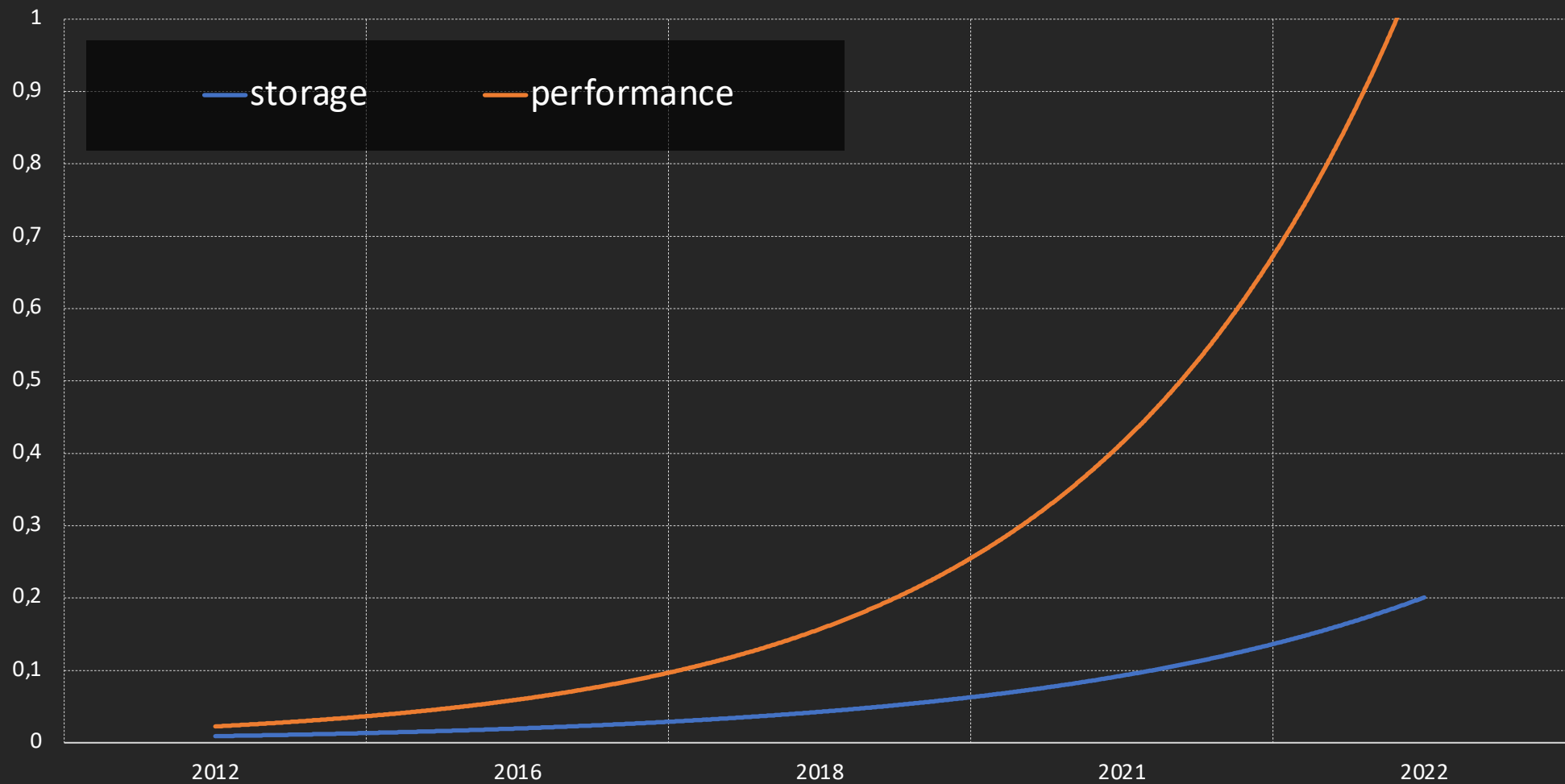
VSB TECHNICAL
UNIVERSITY
OF OSTRAVA

IT4INNOVATIONS
NATIONAL SUPERCOMPUTING
CENTER

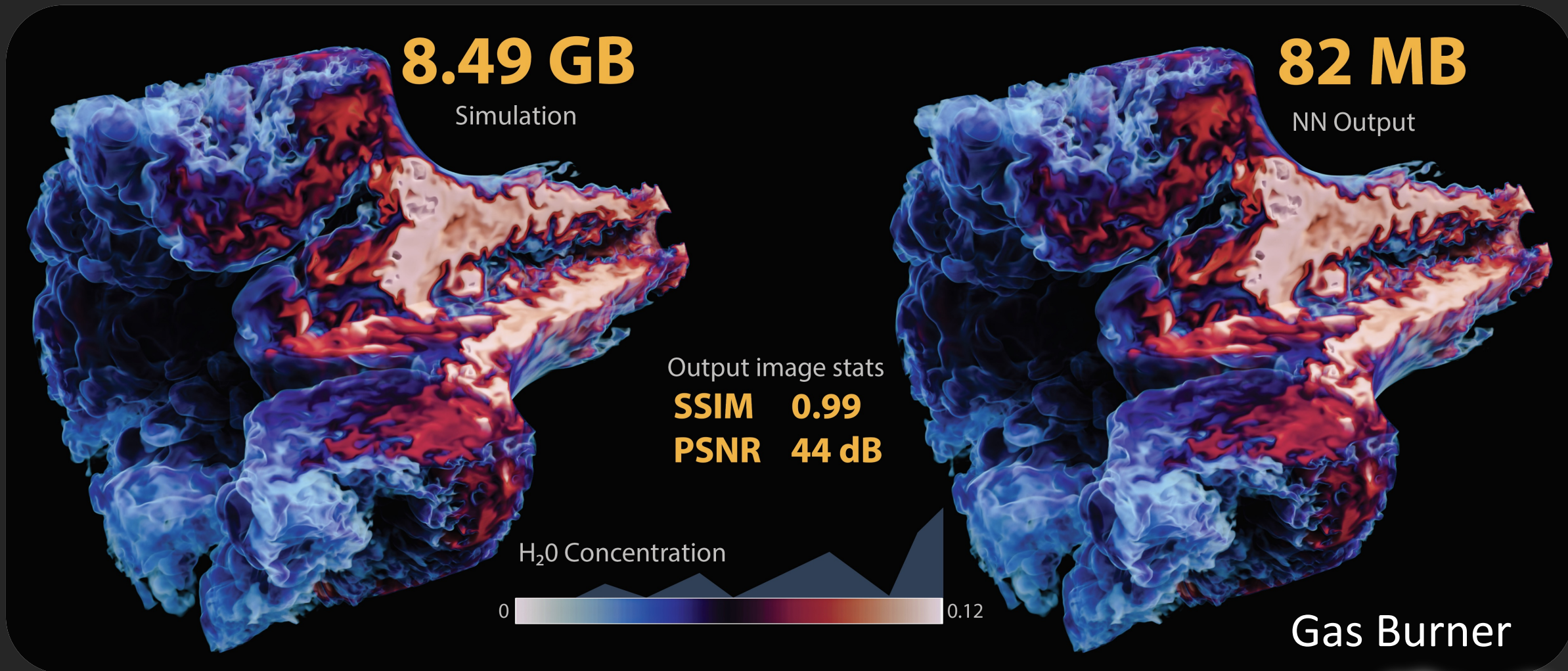


Motivation

HPC resources evolution



Volume Rendering of scalar variable with output image metrics



Structural similarity index (SSIM) is a method for predicting the perceived quality of digital television and cinematic pictures, as well as other kinds of digital images and video.

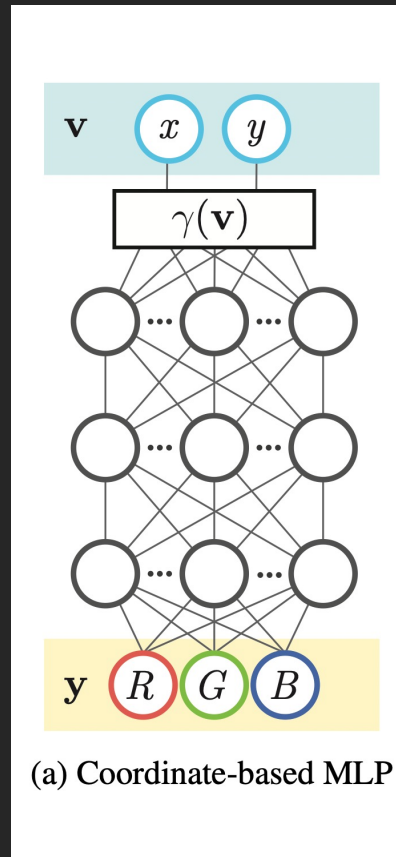
MSE – Mean Square Error

SSIM – Structure Similarity Index (0 – 1), 1 is optimal

PSNR – Peak Signal-to-Noise ratio [dB]

$$\text{MSE} = \frac{1}{n} \sum_{i=1}^n (Y_i - \hat{Y}_i)^2$$

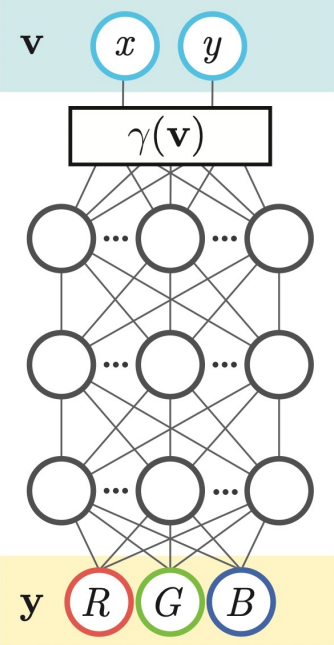
Fourier Features mapping to learn high-frequency functions in low-dimensional problem domains.



- simple, fully connected neural network
- Input: coordinates (pixels of the image)
- Output: R,G,B channels of a given image

Source: M. Tancik, P. Srinivasan, B. Mildenhall, S. Fridovich-Keil, N. Raghavan, U. Singhal, R. Ramamoorthi, J. T. Barron, R. Ng.,
Fourier Features Let Networks Learn High Frequency Functions in Low Dimensional Domains. arXiv:2006.10739, (2020)

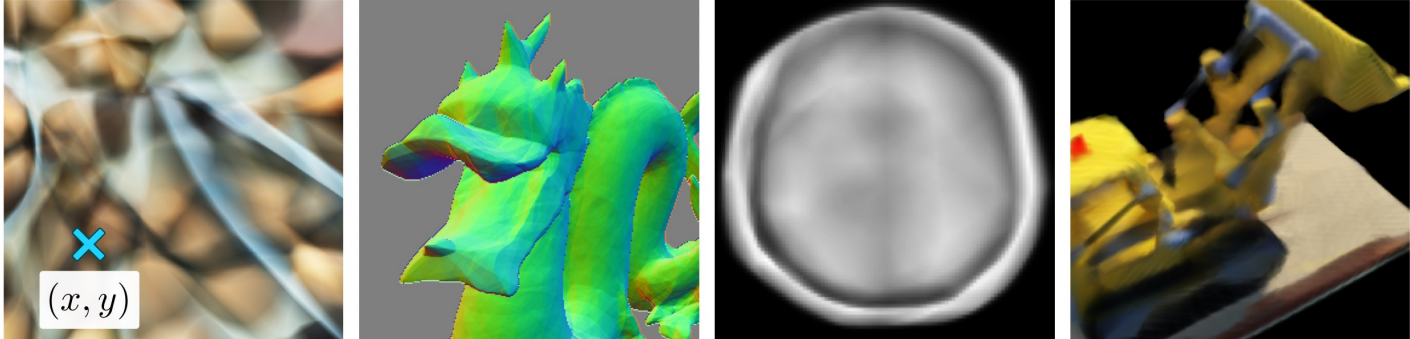
Fourier Features mapping to learn high-frequency functions in low-dimensional problem domains.



The diagram shows a neural network architecture where input coordinates x and y are processed by a mapping function $\gamma(\mathbf{v})$. The output of this function is fed into a series of three fully connected layers, each with three nodes. The final output is a vector \mathbf{y} with components R , G , and B .

(a) Coordinate-based MLP

No Fourier features
 $\gamma(\mathbf{v}) = \mathbf{v}$



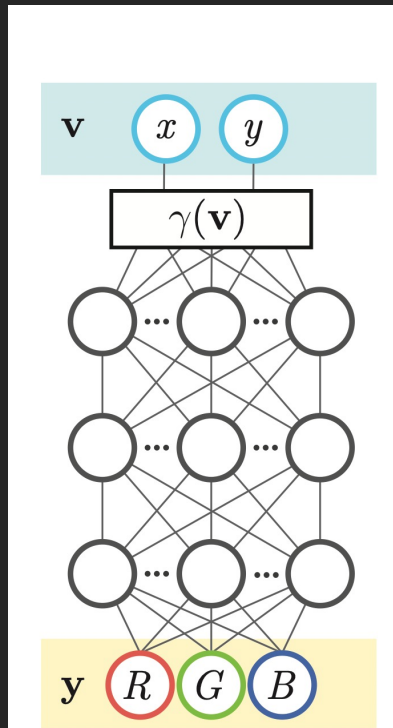
The four images show: 1) A blurry texture of a leaf with a blue 'X' and a box labeled (x, y) indicating the input coordinates. 2) A 3D point cloud of a dragon's head. 3) A blurry grayscale image of a bowl. 4) A blurry image of a yellow mechanical part.

Tancik et. al.:

- Introduced the Fourier Features mapping to learn high-frequency functions in low-dimensional problem domains
- mapping is a transformation of coordinates by a simple multiplication with a matrix with a random distribution
- If the matrix is unitary – the mapping is disabled – the neural network is not capable of capturing high frequencies

Source: M. Tancik, P. Srinivasan, B. Mildenhall, S. Fridovich-Keil, N. Raghavan, U. Singhal, R. Ramamoorthi, J. T. Barron, R. Ng.,
Fourier Features Let Networks Learn High Frequency Functions in Low Dimensional Domains. arXiv:2006.10739, (2020)

Fourier Features mapping to learn high-frequency functions in low-dimensional problem domains.



(a) Coordinate-based MLP

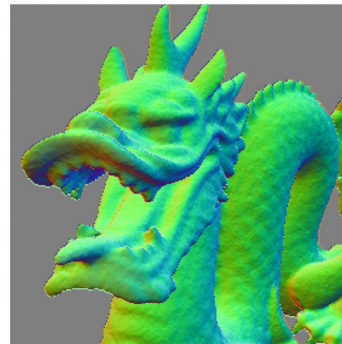
- If the matrix is random and it is multiplied with sin and cos functions the neural network captures also high frequencies
- the size of the matrix affects the number of Fourier Features (FF)
 - it is a parameter that can be defined by the user

We have extended this 2D image-focused approach to scientific datasets

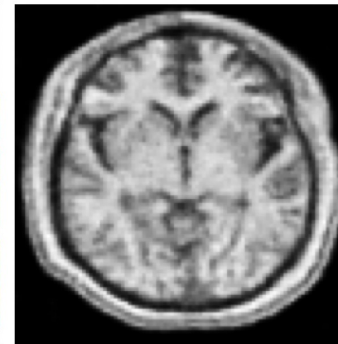
With Fourier features
 $\gamma(\mathbf{v}) = \mathbf{FF}(\mathbf{v})$



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



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



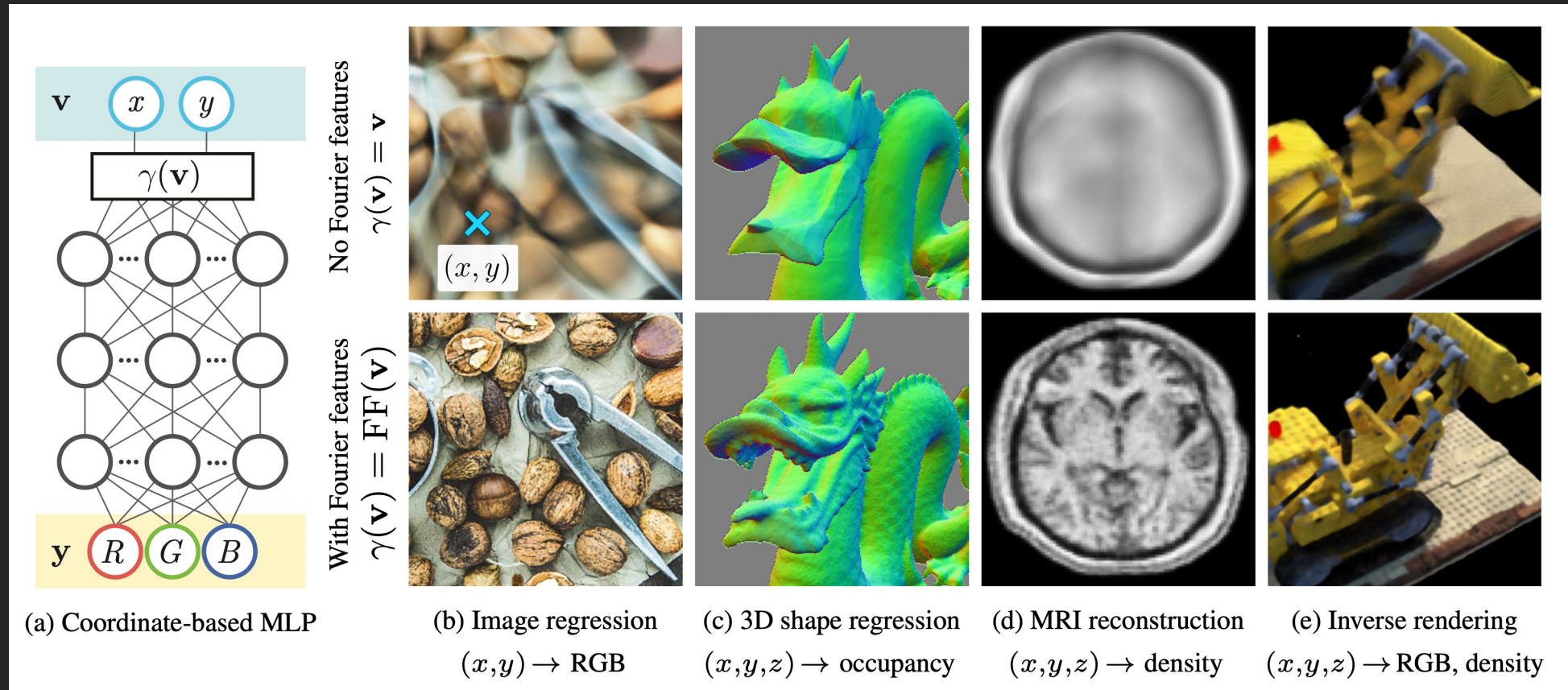
(d) MRI reconstruction
 $(x, y, z) \rightarrow \text{density}$



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

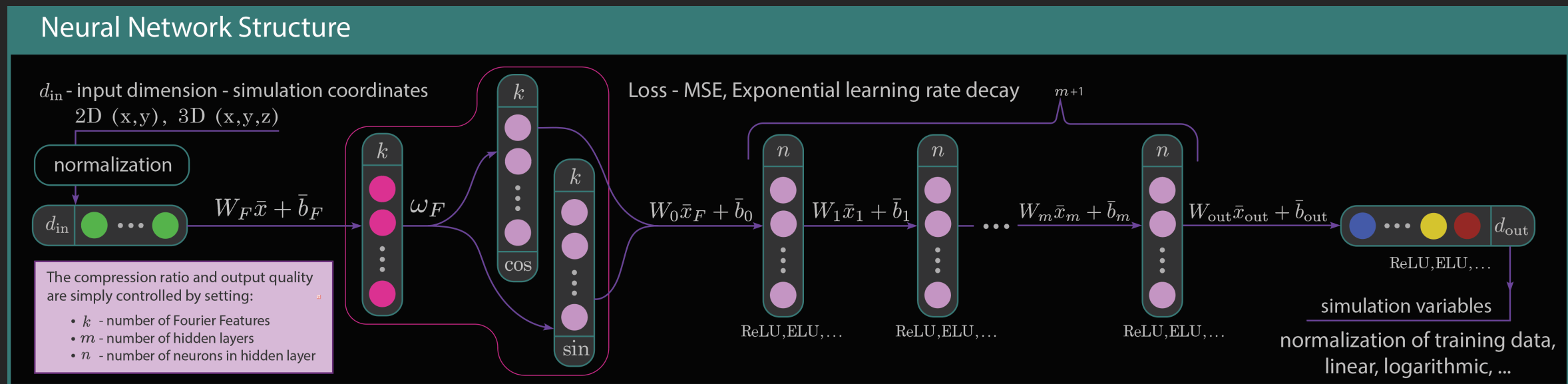
Source: M. Tancik, P. Srinivasan, B. Mildenhall, S. Fridovich-Keil, N. Raghavan, U. Singhal, R. Ramamoorthi, J. T. Barron, R. Ng.,
Fourier Features Let Networks Learn High Frequency Functions in Low Dimensional Domains. arXiv:2006.10739, (2020)

Fourier Features mapping to learn high-frequency functions in low-dimensional problem domains.



Source: M. Tancik, P. Srinivasan, B. Mildenhall, S. Fridovich-Keil, N. Raghavan, U. Singhal, R. Ramamoorthi, J. T. Barron, R. Ng.,
Fourier Features Let Networks Learn High Frequency Functions in Low Dimensional Domains. arXiv:2006.10739, (2020)

Fourier Features mapping to learn high-frequency functions in low-dimensional problem domains.



We present a modified version of Fourier mapping functions for learning the mapping from 2D or 3D coordinates space to an output space containing simulation variables. We use dynamic Fourier features mapping to:

- decrease the input size in the learning process
- enable dynamic learning of the optimal distribution of features mapping.

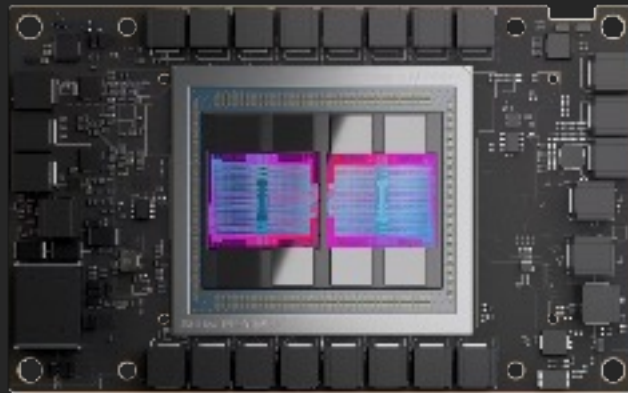
Model Complexity:

$$\#Bytes \cdot (d_{out} + k(d_{in} + 1) + n(2k + m(n + 1) + d_{out}))$$



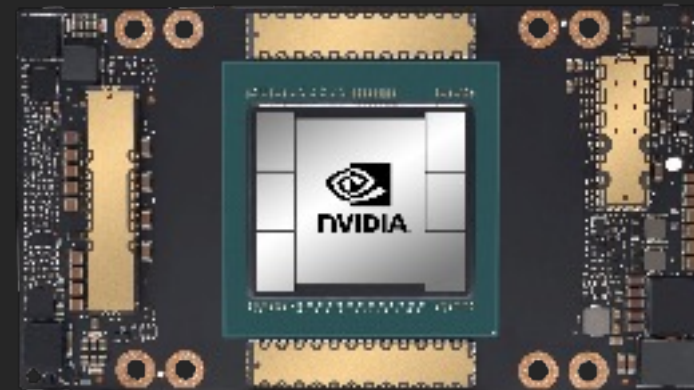
LUMI-G

Radeon Instinct MI250X GPUs

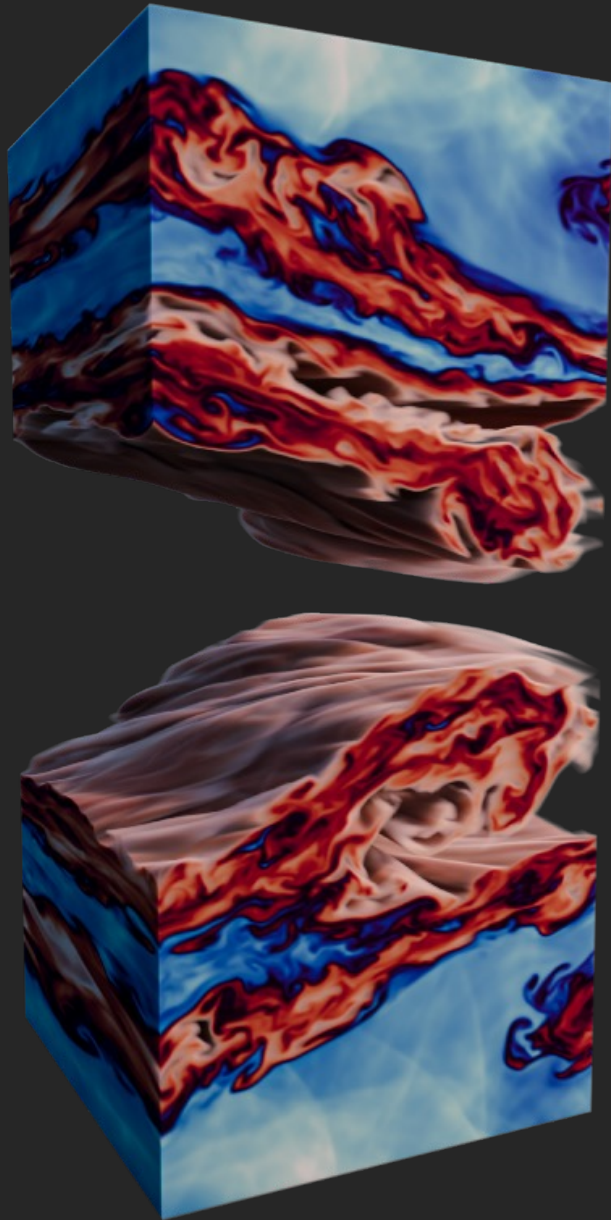


KAROLINA-GPU

Nvidia A100 GPUs on SXM4



Due to the input size, a distributed data-parallel approach for training on multiple GPU accelerators was used. We tested the distributed data-parallel approach up to 256 GPUs on LUMI and up to 32 GPUs on KAROLINA clusters.



Resistive Relativistic Magnetohydrodynamics Simulation by PLUTO code

PLUTO is a freely-distributed software for the numerical solution of mixed hyperbolic/parabolic systems of partial differential equations (conservation laws) targeting high Mach number flows in astrophysical fluid dynamics. The code is designed with a modular and flexible structure whereby different numerical algorithms can be separately combined to solve systems of conservation laws using the finite volume or finite difference approach based on Godunov-type schemes.

<https://plutocode.ph.unito.it>

PLUTO code - 3D dataset

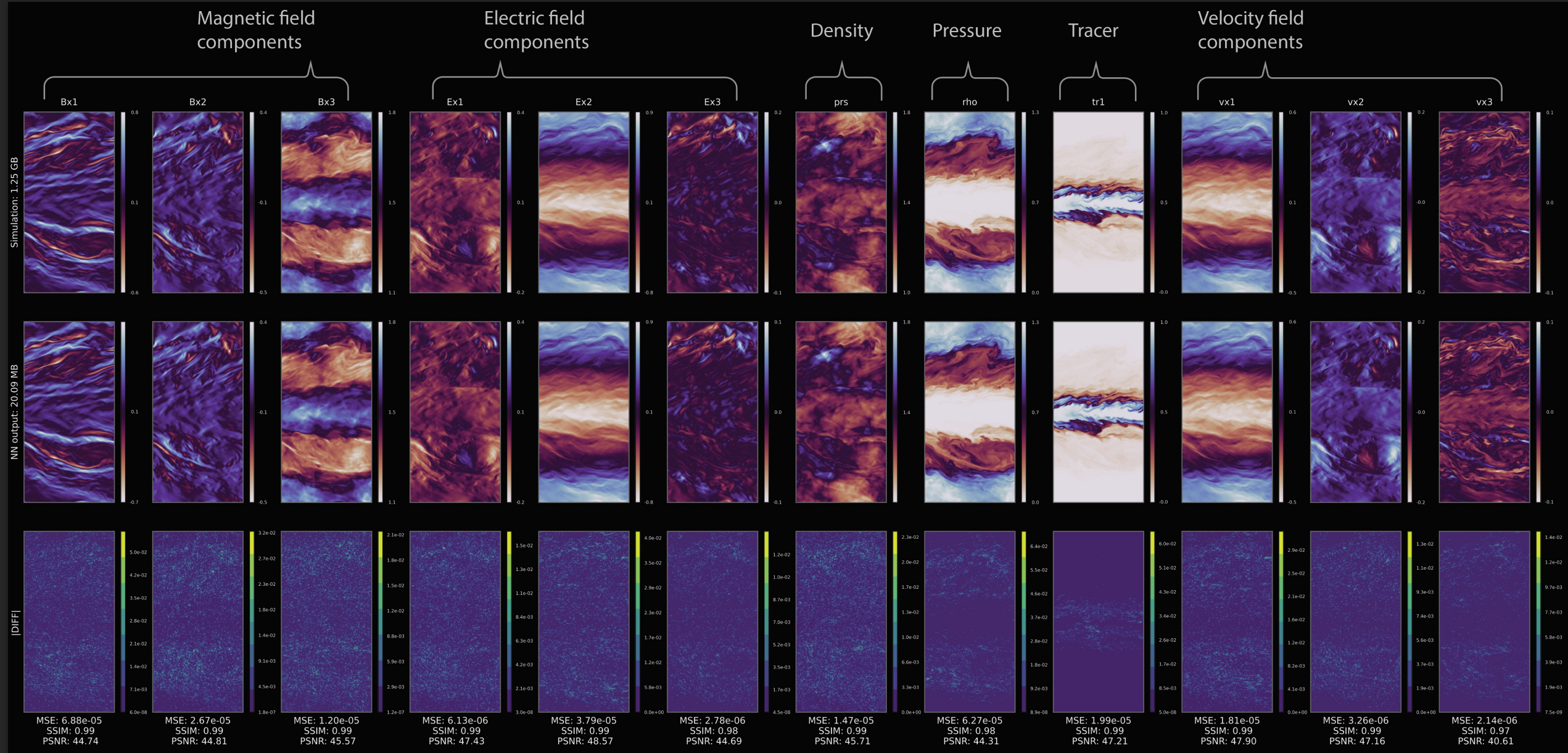
One time step with 12 scalar variables in one neural model

- 85.7 million cells - compression of 3.86GB to 54MB
- 27.6 million cells - compression of 1.25GB to 20.09MB

12 scalar variables in Five time steps in one neural model -

- 27.6 million cells - compression of 6.24GB to 54MB

PLUTO code - 3D dataset, One time step - 12 scalar variables 27.6 million cells



Magnetic field components

Electric field components

Density

Pressure

Tracer

Velocity field components

Bx1

Bx2

Bx3

Ex1

Ex2

Ex3

prs

rho

tr1

vx1

vx2

vx3

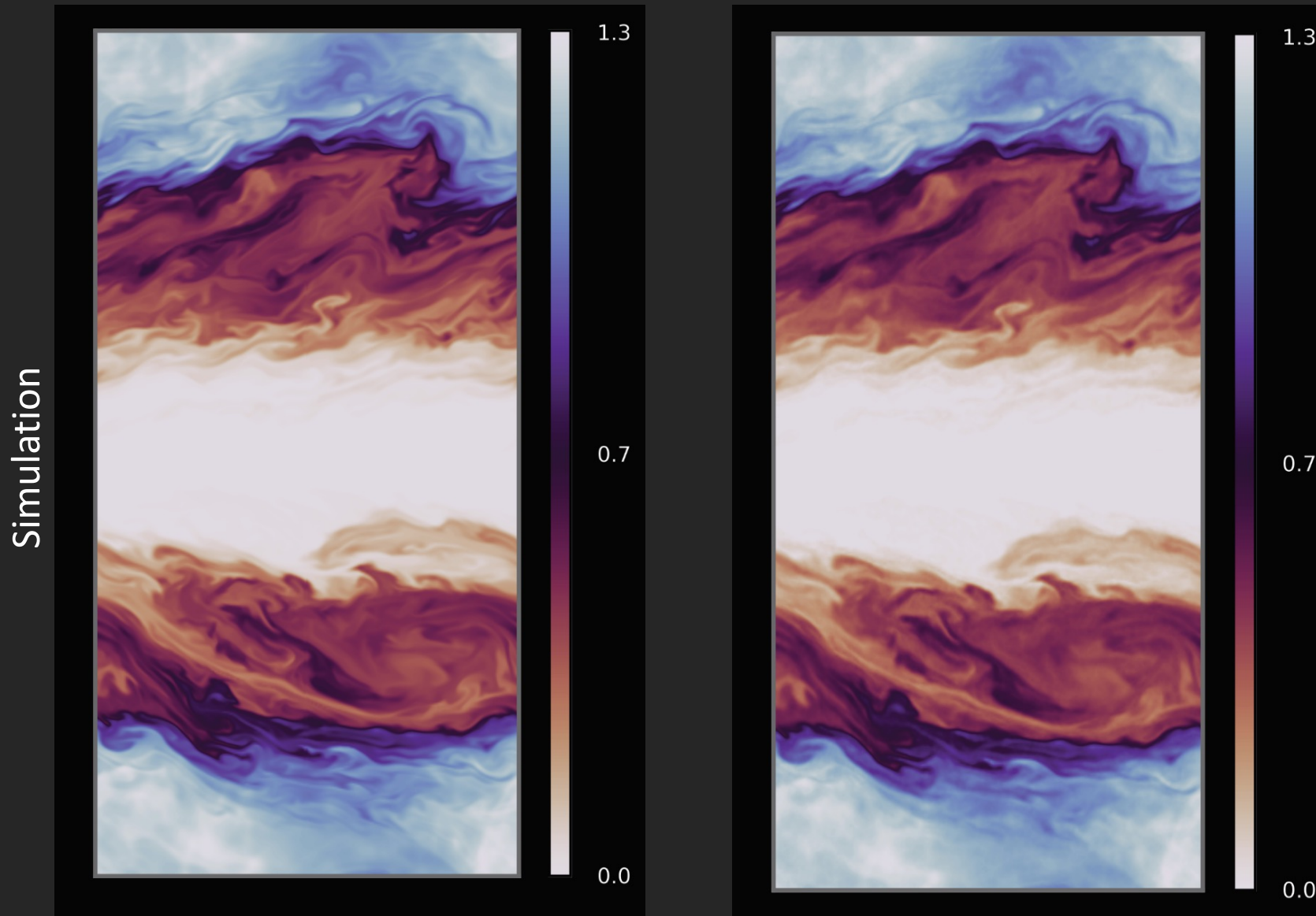
Compression
1.25GB to 20.09MB

Simulation: 1.25 GB

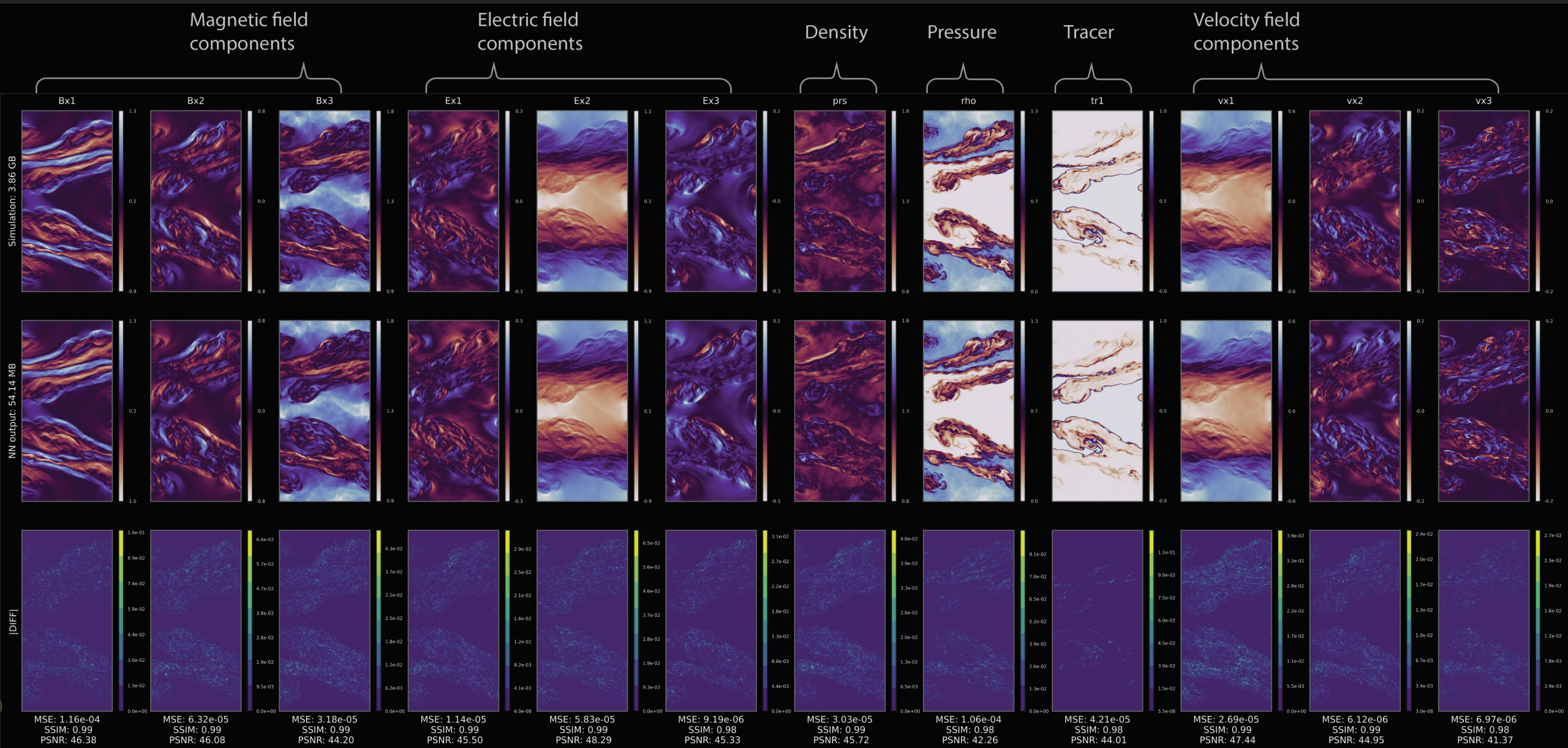
NN output: 20.09 MB

|DIFF|

MSE: 6.88e-05 SSIM: 0.99 PSNR: 44.74	MSE: 2.67e-05 SSIM: 0.99 PSNR: 44.81	MSE: 1.20e-05 SSIM: 0.99 PSNR: 45.57	MSE: 6.13e-06 SSIM: 0.99 PSNR: 47.43	MSE: 3.79e-05 SSIM: 0.99 PSNR: 48.57	MSE: 2.78e-06 SSIM: 0.98 PSNR: 44.69	MSE: 1.47e-05 SSIM: 0.99 PSNR: 45.71	MSE: 6.27e-05 SSIM: 0.98 PSNR: 44.31	MSE: 1.99e-05 SSIM: 0.99 PSNR: 47.21	MSE: 1.81e-05 SSIM: 0.99 PSNR: 47.90	MSE: 3.26e-06 SSIM: 0.99 PSNR: 47.16	MSE: 2.14e-06 SSIM: 0.97 PSNR: 40.61
--	--	--	--	--	--	--	--	--	--	--	--

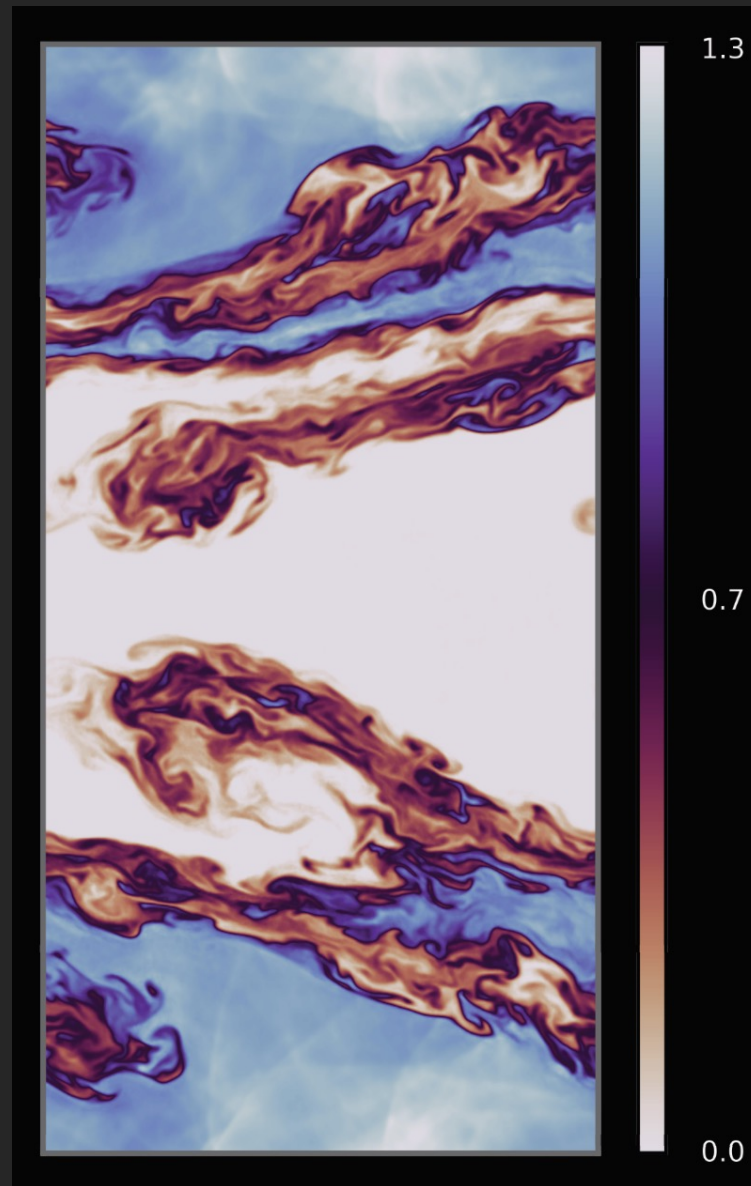
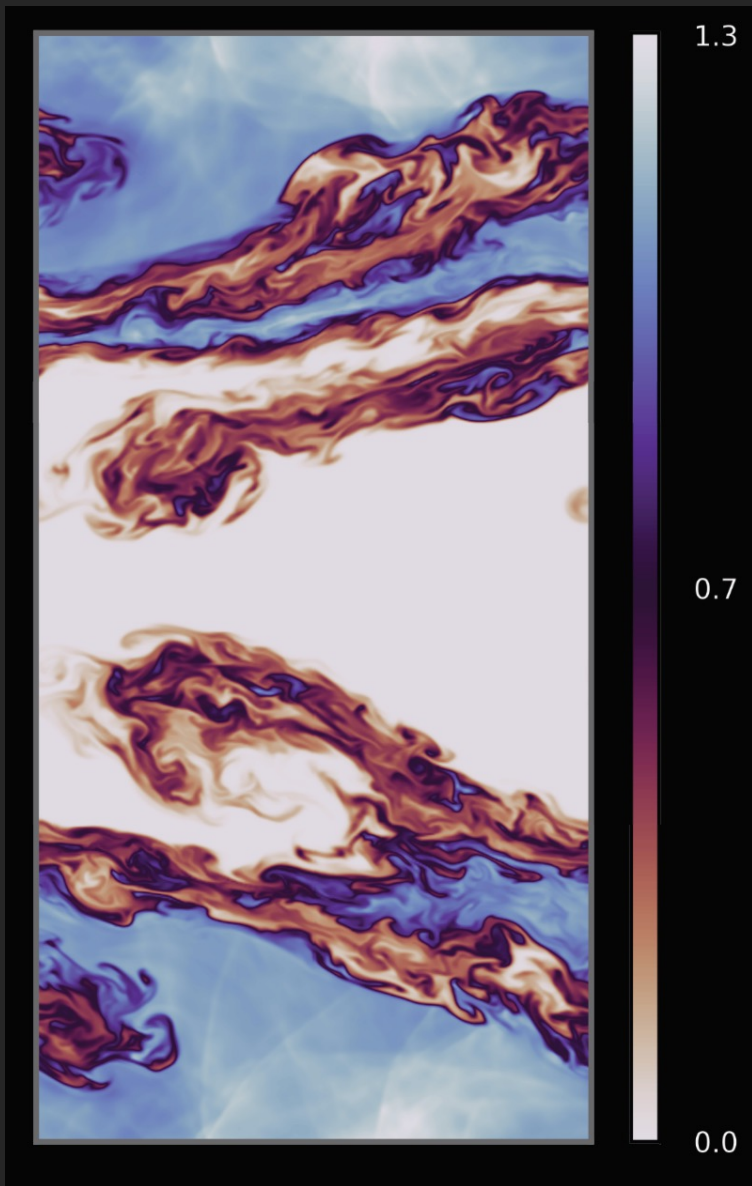


PLUTO code - 3D dataset, One time step - 12 scalar variables 85.7 million cells





Simulation

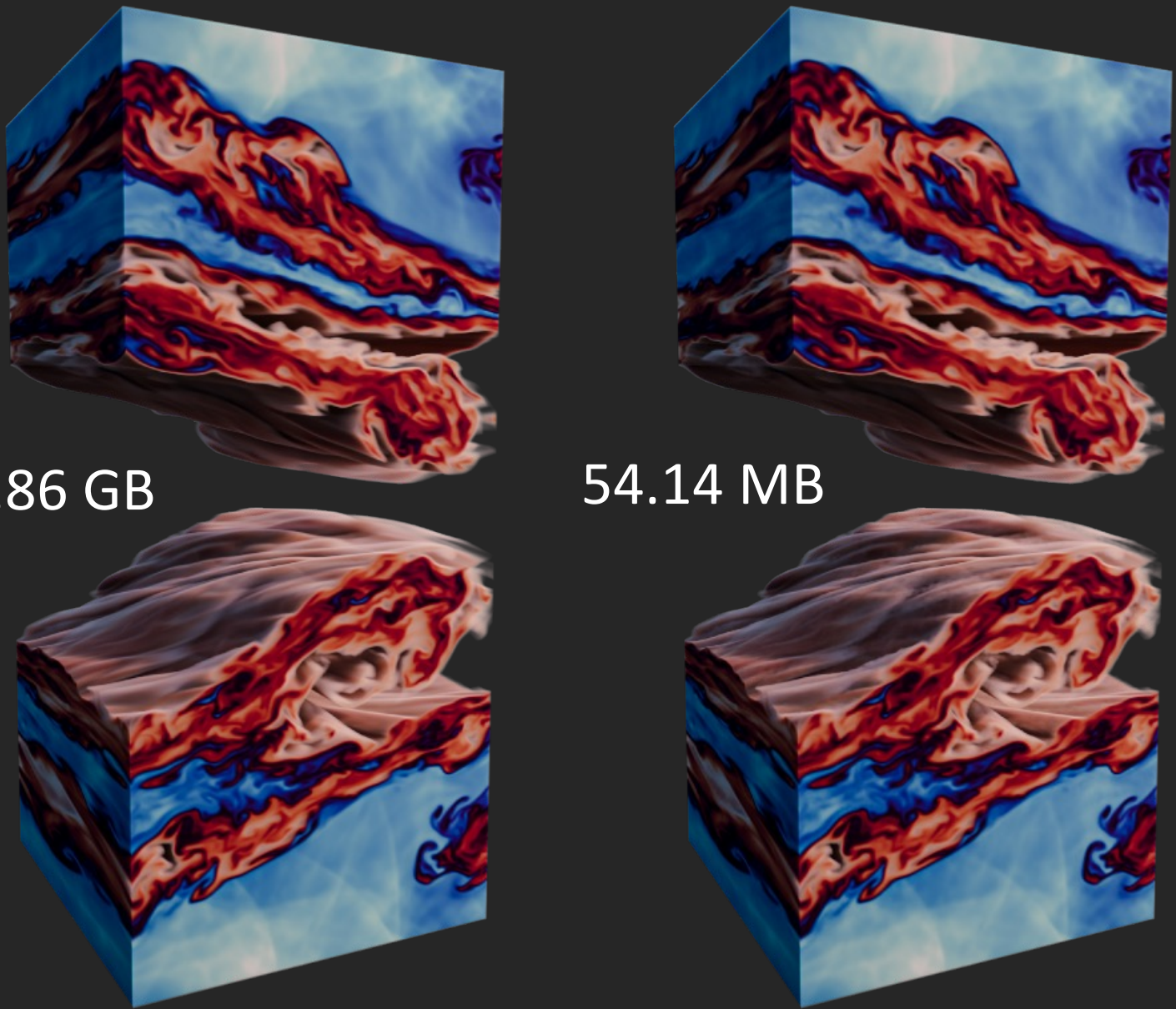


PLUTO code - 3D dataset, One time step - 12 scalar variables 27.6 million cells

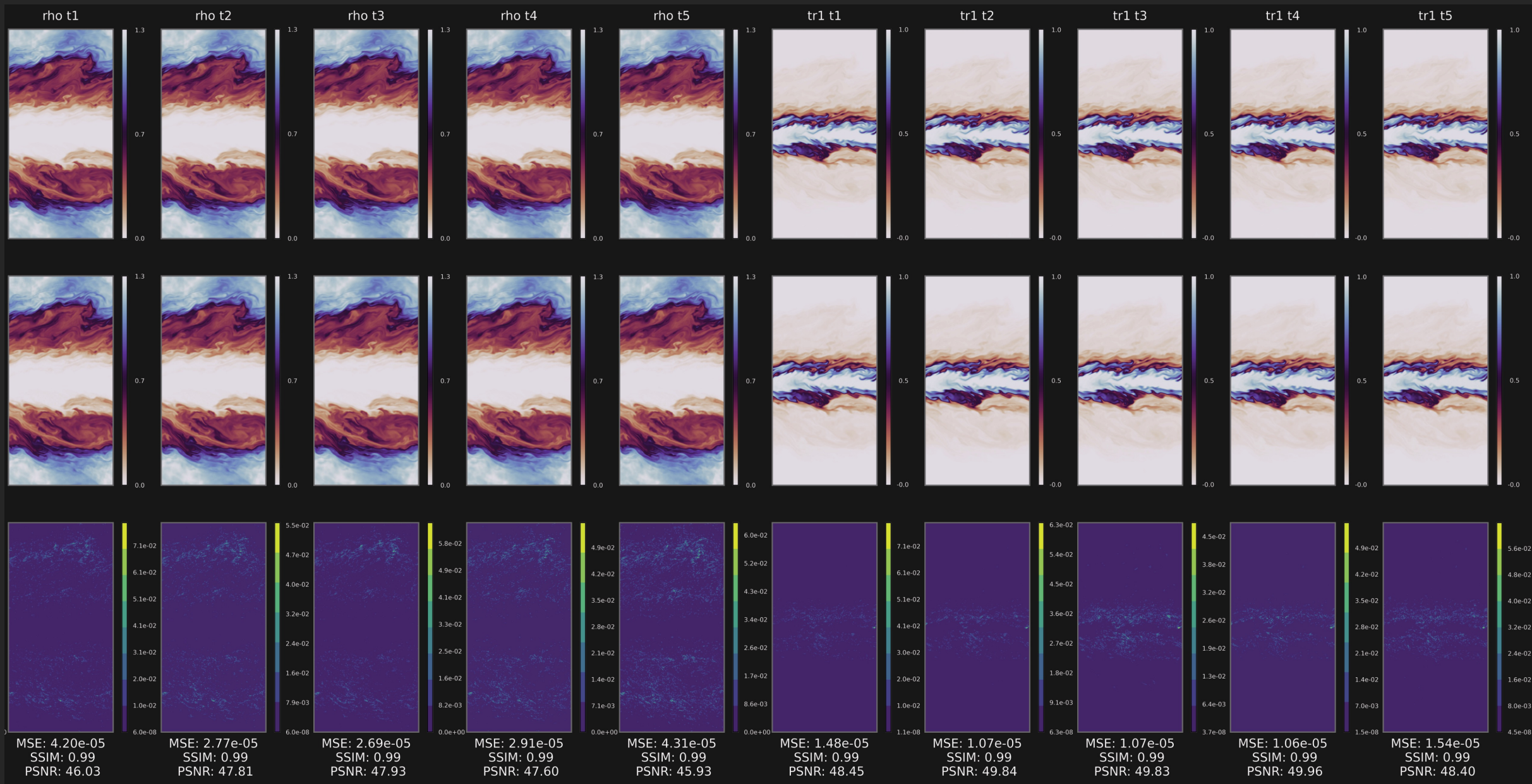
	SSIM		PSNR
Bx1	0.989	Bx1	48.42
Bx2	0.989	Bx2	48.70
Bx3	0.992	Bx3	48.72
Ex1	0.989	Ex1	48.93
Ex2	0.994	Ex2	50.58
Ex3	0.985	Ex3	48.62
prs	0.990	prs	47.13
rho	0.976	rho	43.52
tr1	0.979	tr1	45.58
vx1	0.992	vx1	49.22
vx2	0.988	vx2	48.59
vx3	0.973	vx3	44.43

3.86 GB

54.14 MB

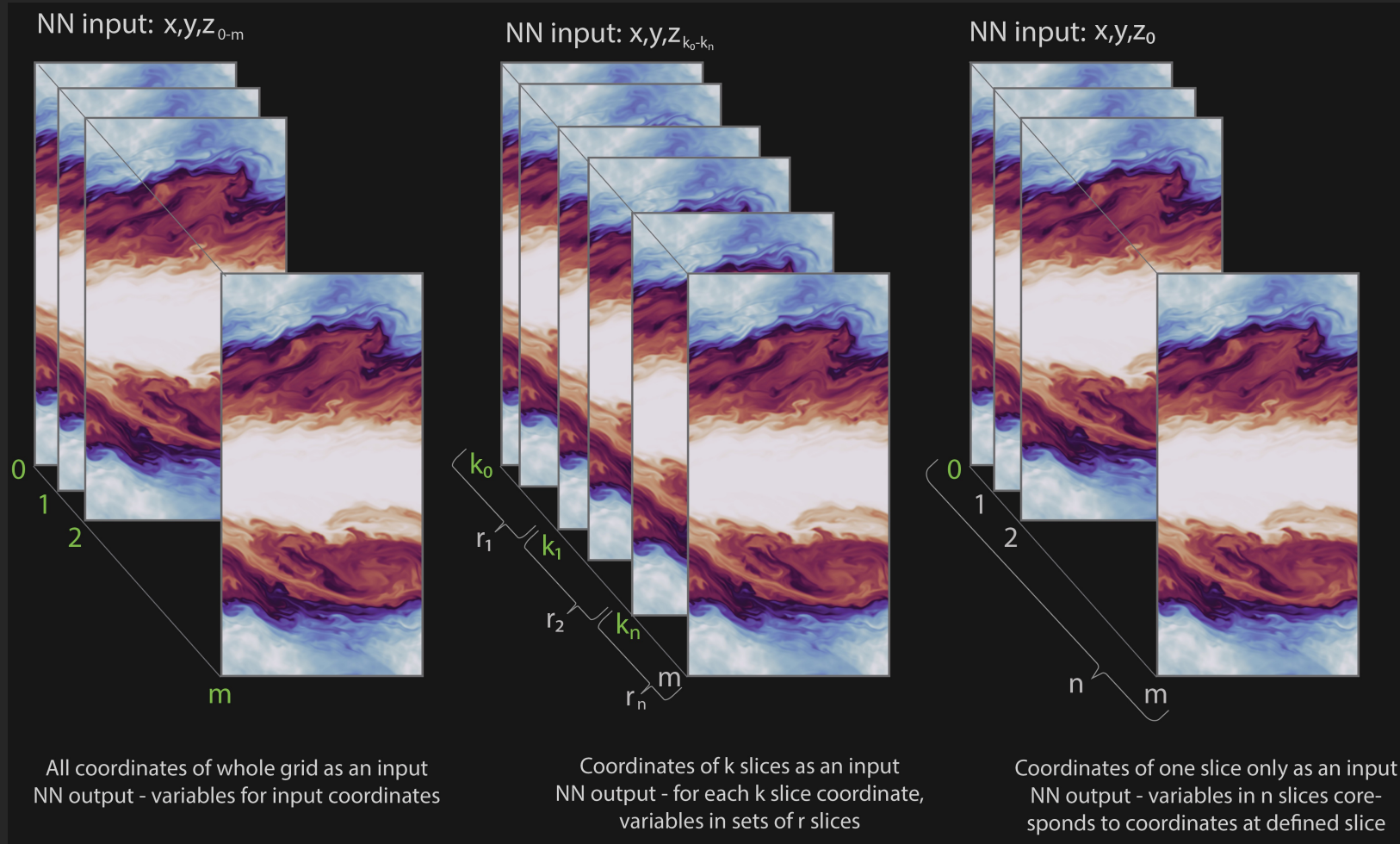


PLUTO code - 3D dataset, 5 time steps - 12 scalar variables 27.6 million cells





For structured datasets, several methodologies of the training process can be selected. Fictitious decomposition of structured data can significantly reduce computational resources and training time but affect result quality.



Using all coordinates for training is inefficient due to the need to use hundreds of GPU-accelerated nodes.

On the other hand, using the coordinates from, for example, only one slice allows the use of only one compute node.

Finding the optimal number of computational resources or the optimal number of input parameters is the main factor that controls the applicability of the proposed solution.

Application potential:

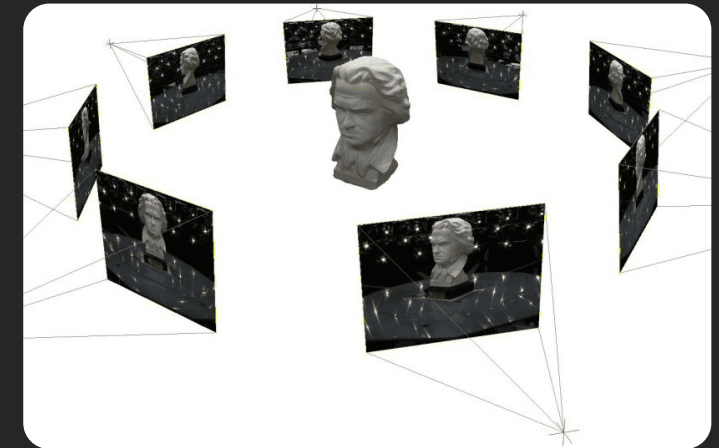
- Storing a larger number of time steps
- Long-term data storage
- Intermediate checkpoints of large-scale simulations for new runs, reducing the runtime overhead of restarted simulations
- Less memory consumption for GPU rendering
- Neural net interpolates results to different scale
- Do all post-processing during the simulation

Interactive data visualization

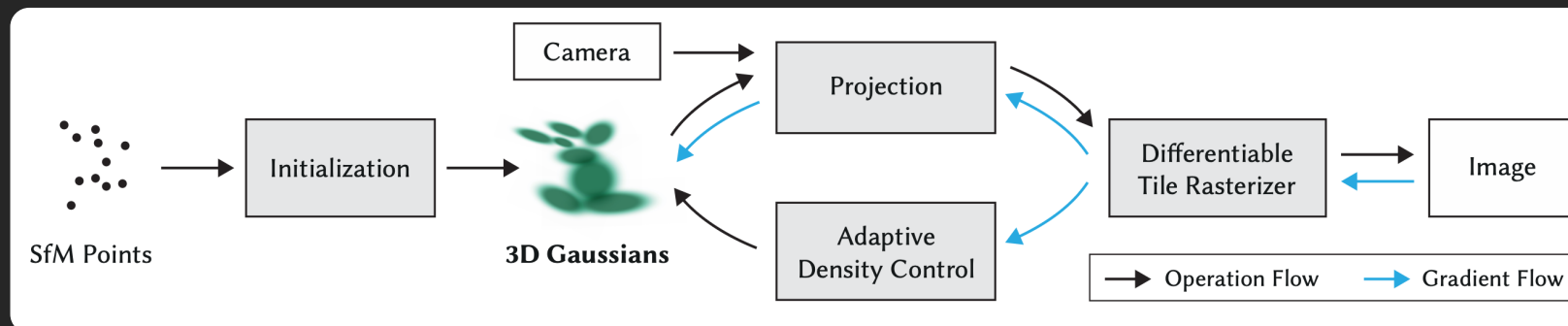
3D Gaussian splatting for real-time visualization

3DGS in a glance

- Based on Neural Radiance Fields methods, which provide a **novel-view synthesis of scenes captured with multiple images.**
- 3DGS maintains competitive training times and allows **real-time** novel-view synthesis at **high visual quality.**
- Commodity HW (PC with a **single GPU**) can be used for rendering **above 30fps at 1080p resolution.**



Source: Convex Variational Methods for Single-View and Space-Time Multi-View Reconstruction



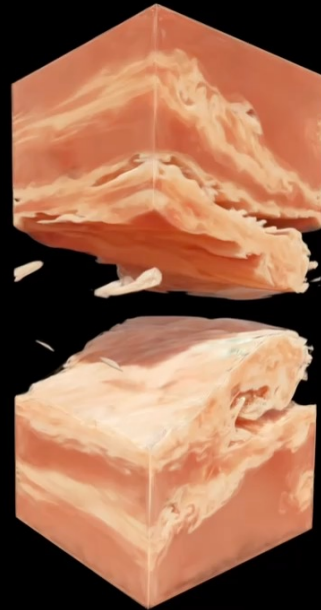
Source: Kerbl, Bernhard, et al. "3D Gaussian Splatting for Real-Time Radiance Field Rendering." ACM Trans. Graph. 42.4 (2023): 139-1.

3D Gaussian splatting for real-time visualization

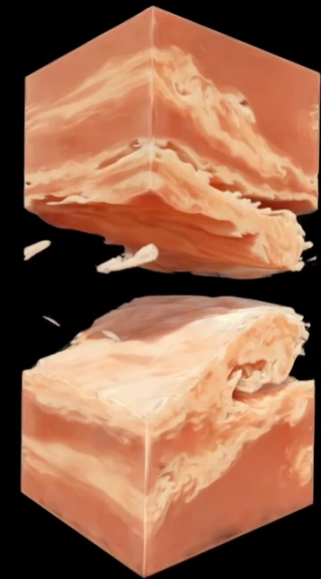
Application on astrophysical data

- PLUTO dataset
 - The original dataset converted into a sparse regular volumetric grid (OpenVDB), 350x700x350 voxels
 - The compressed representation of the original dataset converted into a sparse regular volumetric grid (OpenVDB), 350x700x350 voxels
- Pre-rendered by Cycles renderer (Blender)
- Datasets decoded into 3DGS
- Original and Compressed datasets visualised simultaneously using a single GPU
- Remote display of the results
 - Rendering on the visualization server (1 GPU)
 - Display and interaction on the client's computer over Ethernet

Original dataset



Compressed dataset



Thank you

VSB TECHNICAL UNIVERSITY OF OSTRAVA | IT4INNOVATIONS NATIONAL SUPERCOMPUTING CENTER

Tomáš Brzobohatý tomas.brzobohaty@vsb.cz
Petr Strakoš petr.strakos@vsb.cz
Lubomír Říha lubomir.riha@vsb.cz