Deep Fluids: A Generative Network for Parameterized Fluid Simulations

Byungsoo Kim[1] Vinicius C. Azevedo[1] Nils Thuerey[2]
Theodore Kim[3] Markus Gross[1] Barbara Solenthaler[1]

ETH zürich TUM PIXAR ANIMATION STUDIOS

EG2019

- Physically-based simulations are still slow
  - Obtaining high-quality results is computationally expensive

Kim & James 2009

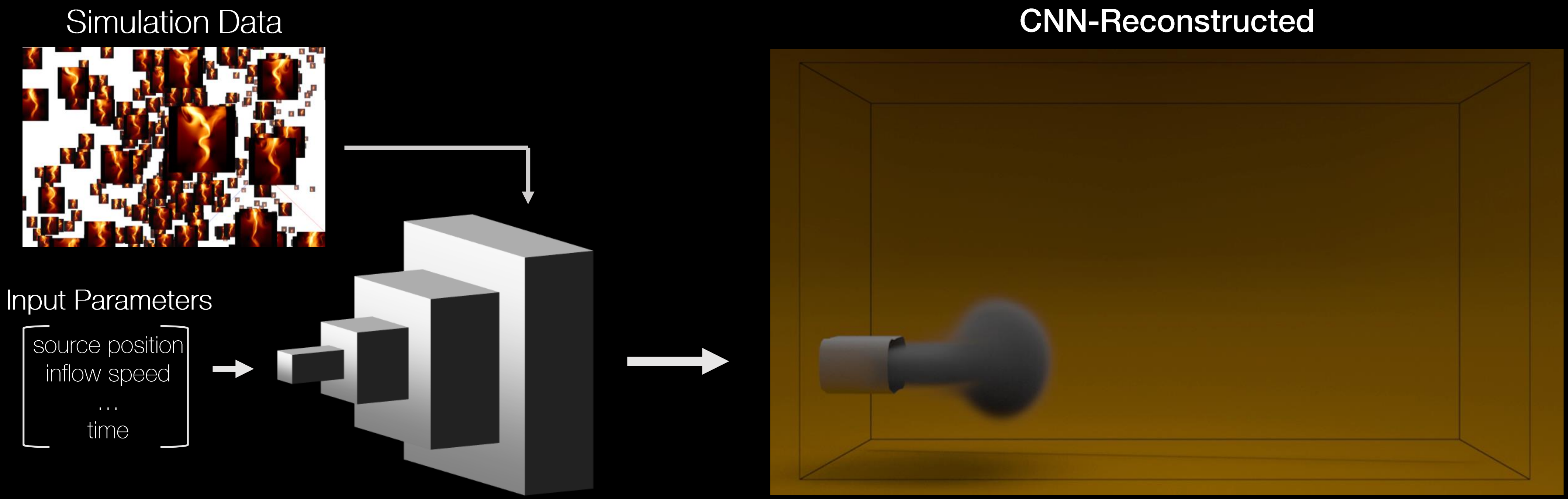
Kim & Delaney 2013

Hahn et al. 2014

- Limited support for artist control
  - Changing an existing simulation entails trial & error

- Production data consumes large amount of storage space
  - Growing need to reuse stored simulation data

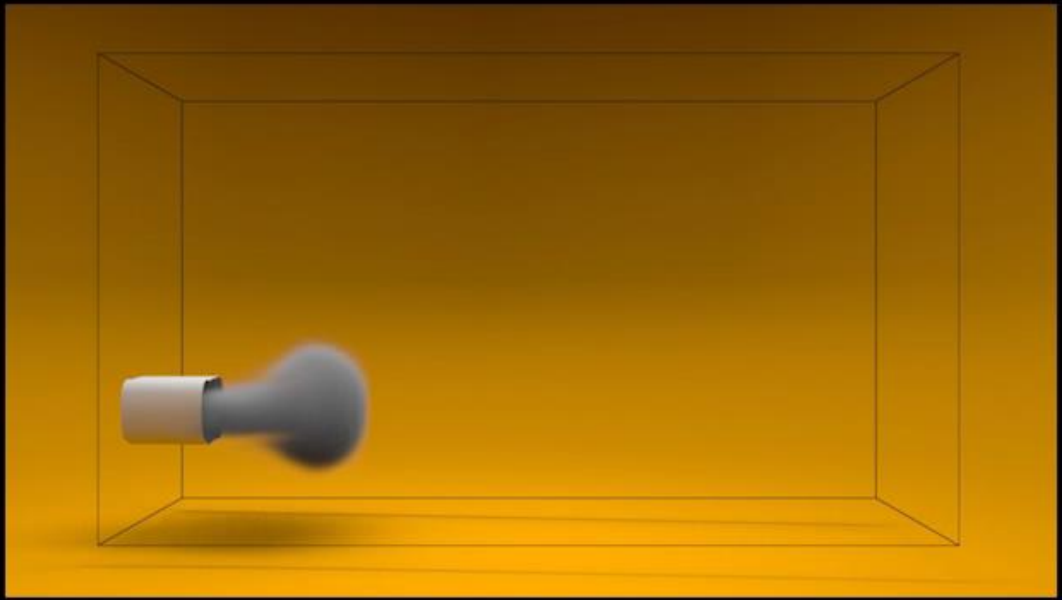
Deep-Fluids: A Generative Network for Parameterized Fluid Simulations



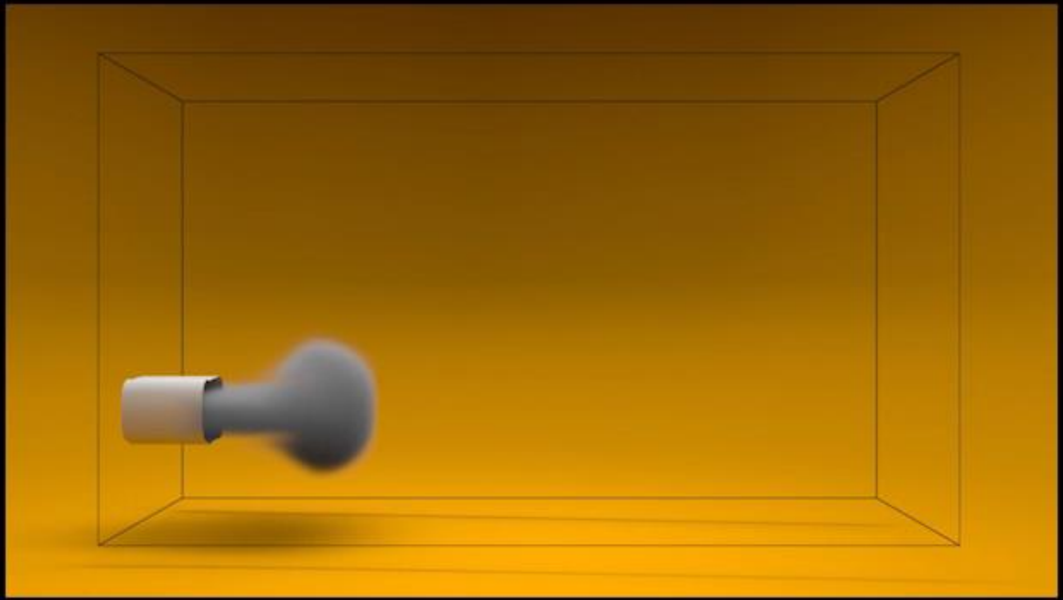
Technical Contributions

- First generative neural network for parametrized Eulerian fluid simulations
- Up to 700x speed-ups compared with underlying CPU solvers for re-simulating the data
- Interpolation between discrete examples across different parameters
- Over 1300x compression ratio for velocity field data
- Novel Latent Space Integrator for complex parameterization

Results

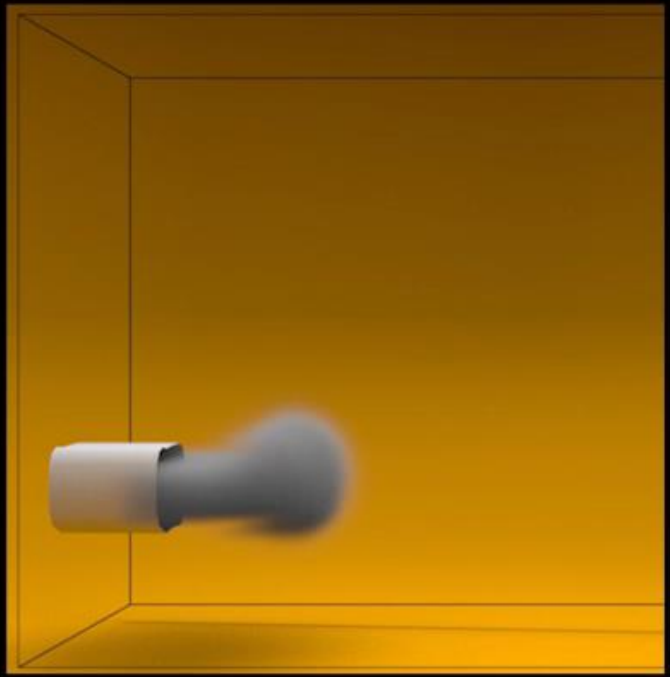


Ground-Truth Simulation

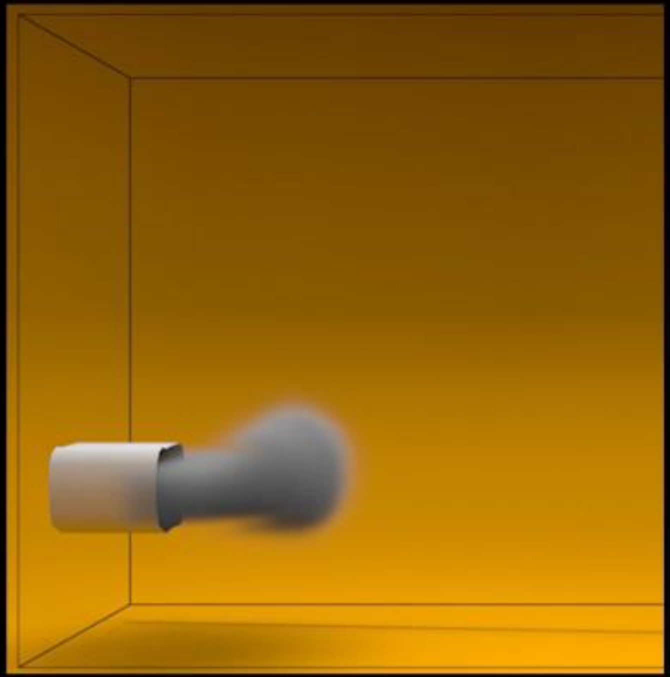


CNN-Reconstructed Simulation

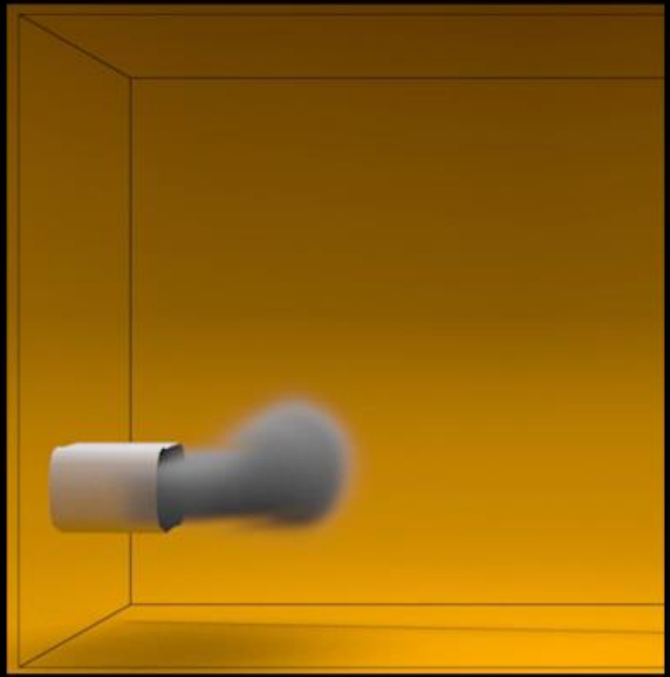
Buoyancy Interpolation Example



Direct correspondence for buoyancy $b = 6 \times 10^{-4}$



Interpolated with $b = 8 \times 10^{-4}$
Not present in the original **data set**

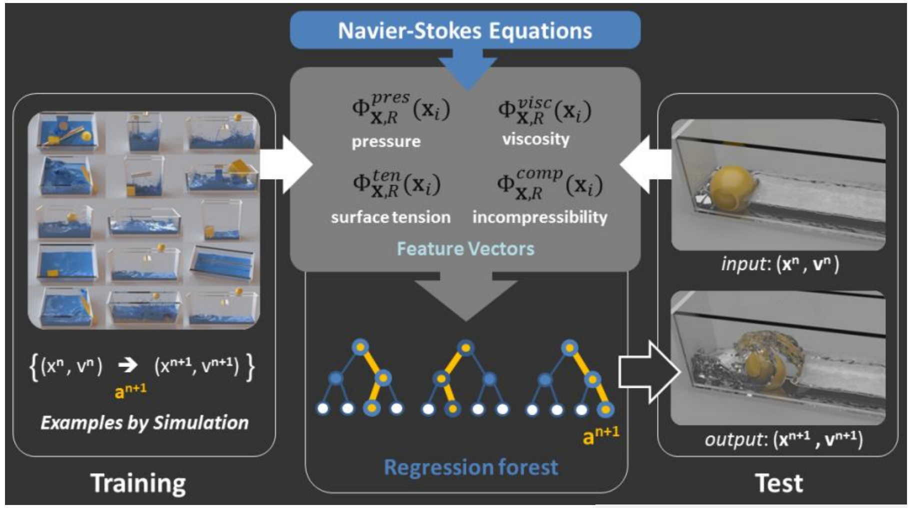


Direct correspondence for buoyancy $b = 1 \times 10^{-3}$

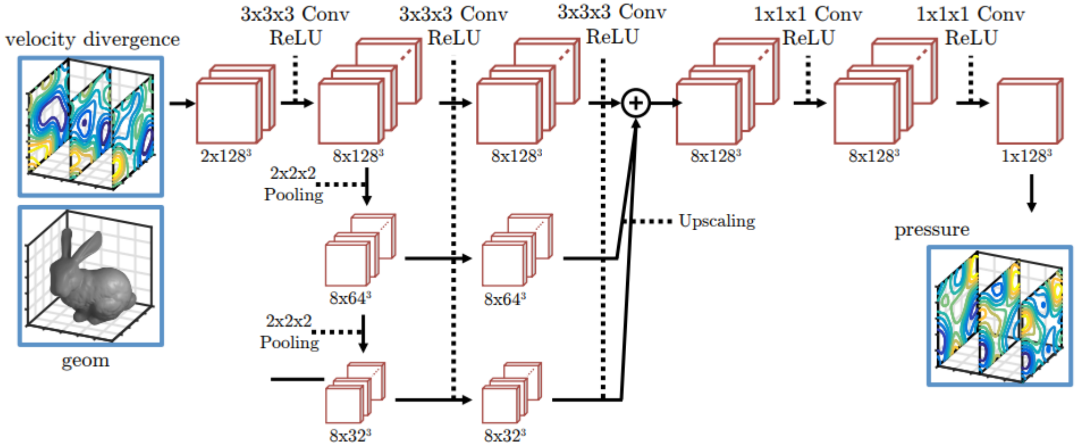
Machine Learning Research in Fluids

Machine Learning for Fluid Simulation

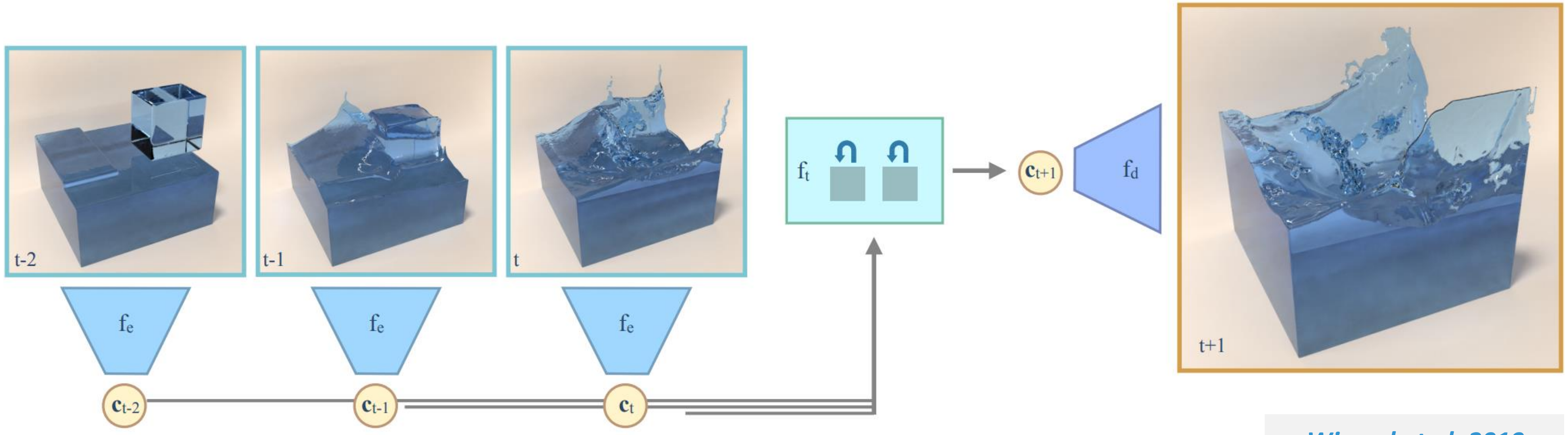
» Speed-Up



Ladicky et al. 2015



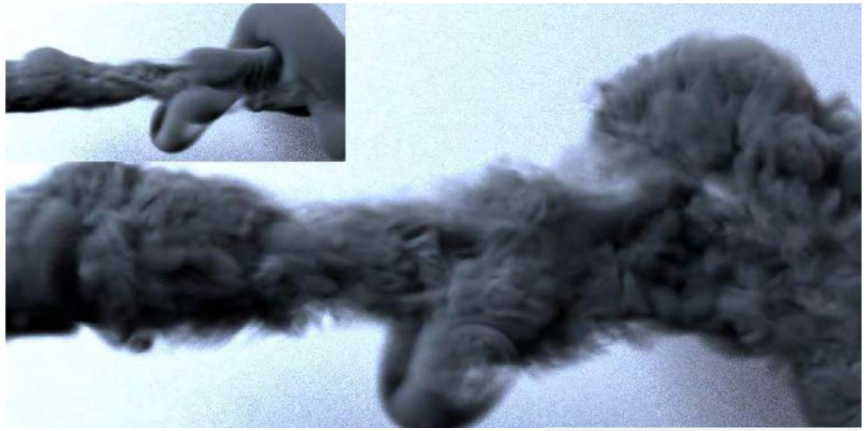
Tompson et al. 2016



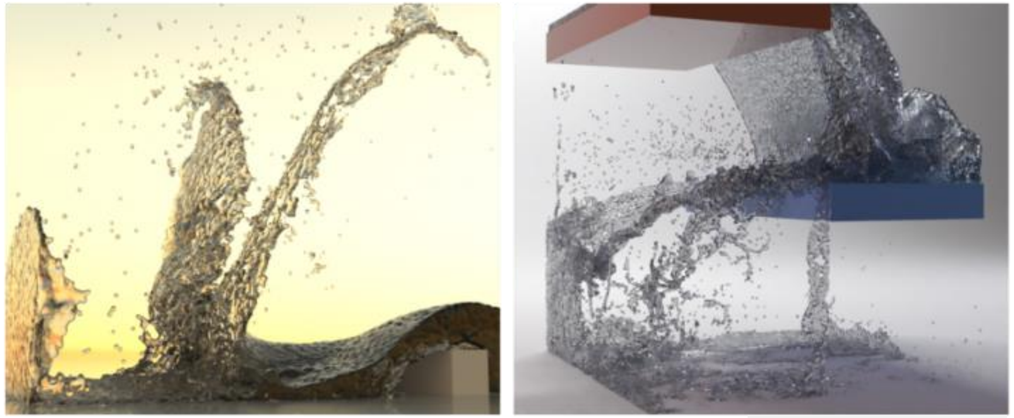
Wiewel et al. 2019

Machine Learning for Fluid Simulation

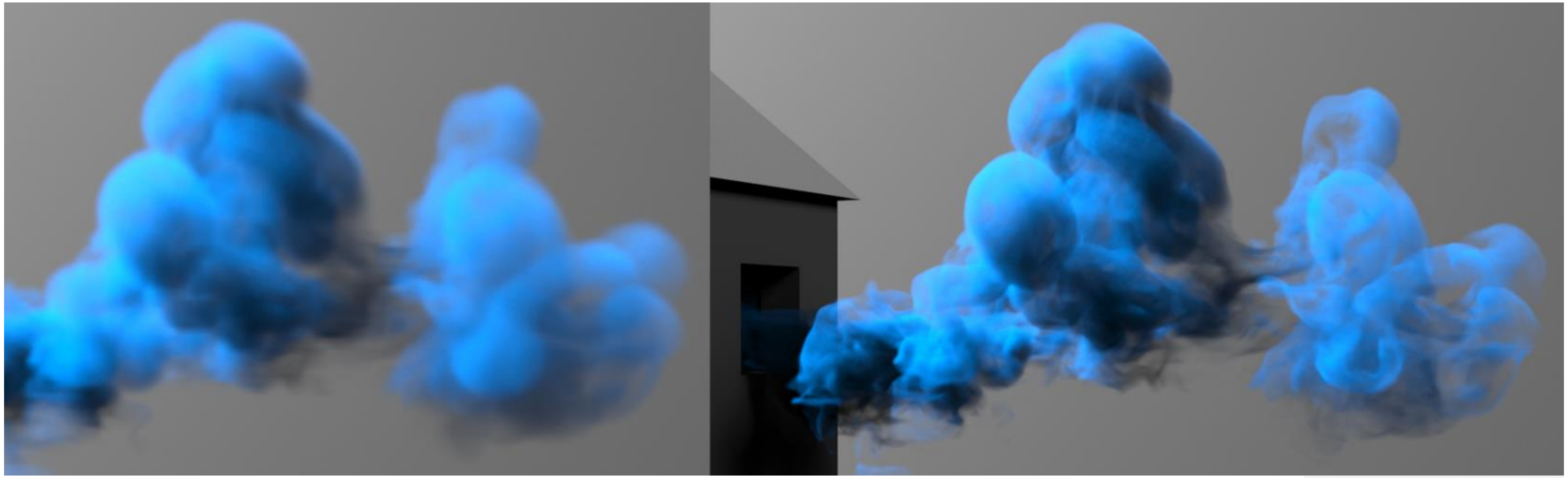
» Enhance Visual Quality



Chu & Thuerey 2017



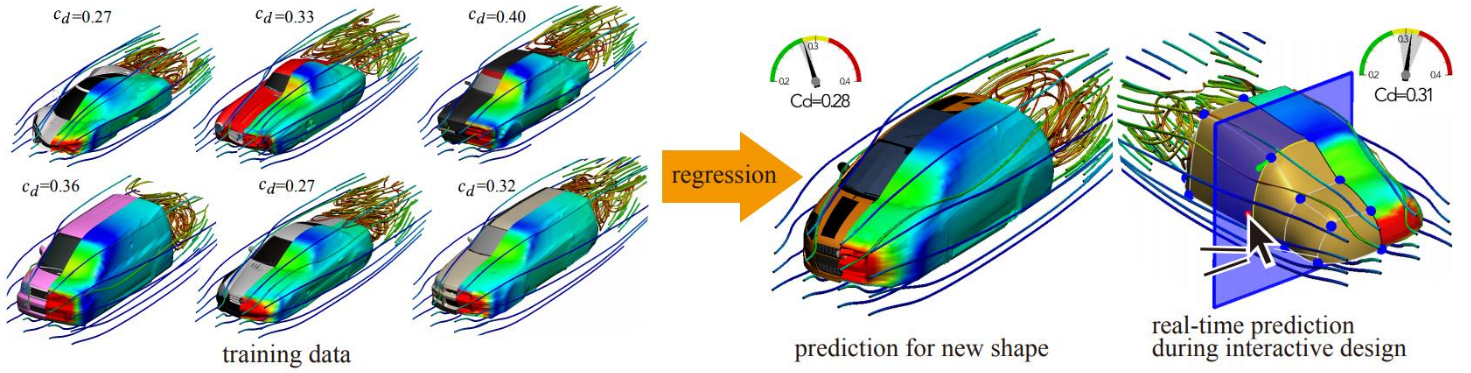
Um et al. 2018



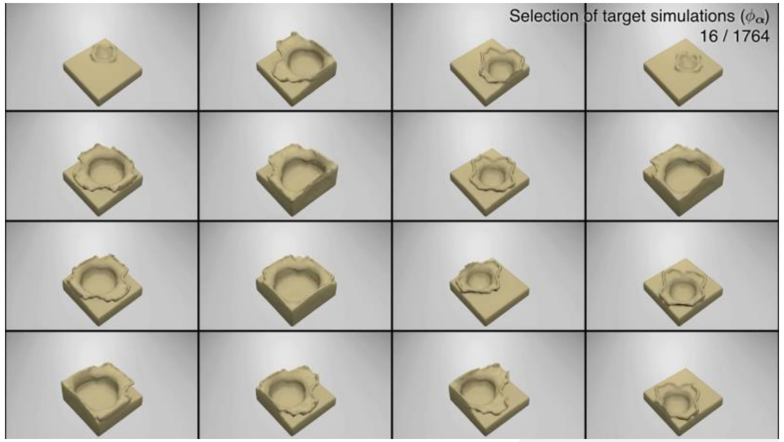
Xie et al. 2018

Machine Learning for Fluid Simulation

» Generative Model



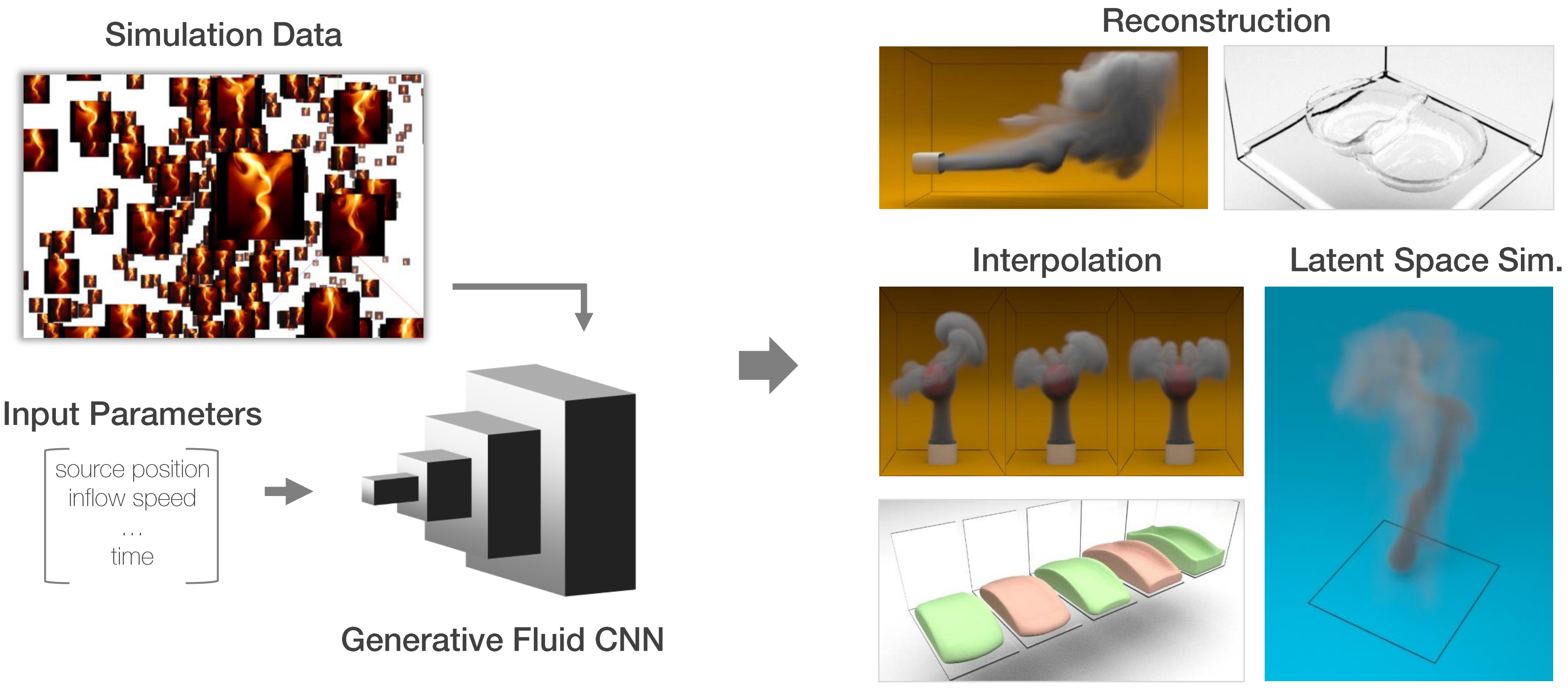
Umetani et al. 2018



Prantl et al. 2019

Machine Learning for Fluid Simulation

» Generative Model

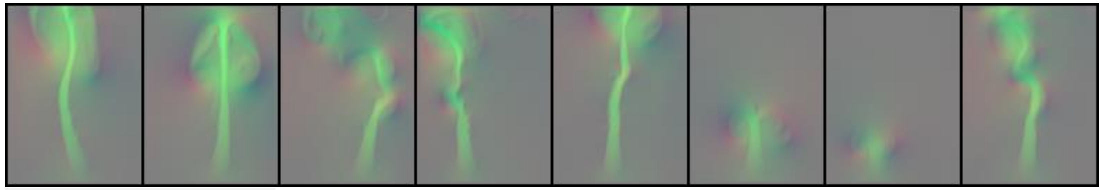


A Generative Model for Fluids

Fluid Simulation Data

» vs. Image Dataset (CelebA)

CelebA

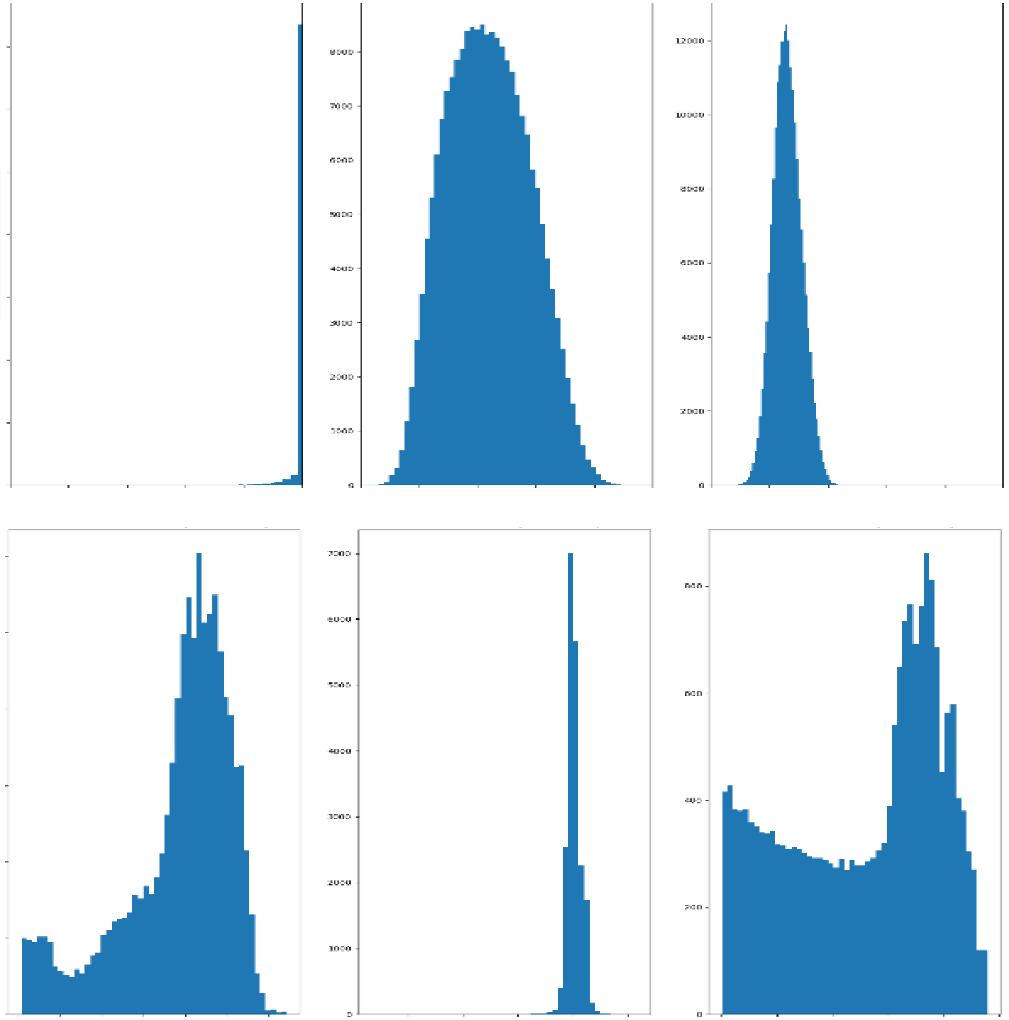


Velocity Data

Histogram plots

CelebA

Velocity Data

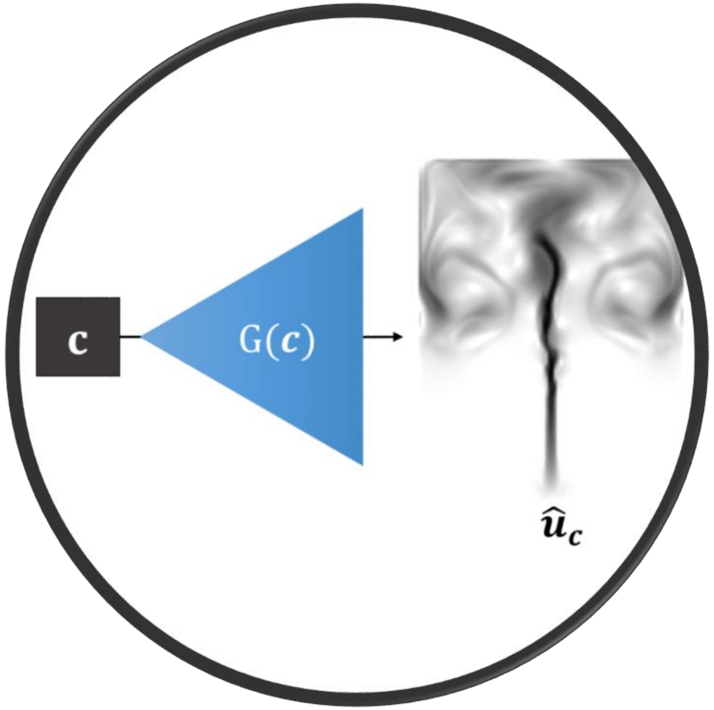


Max | Mean | Standard Dev.

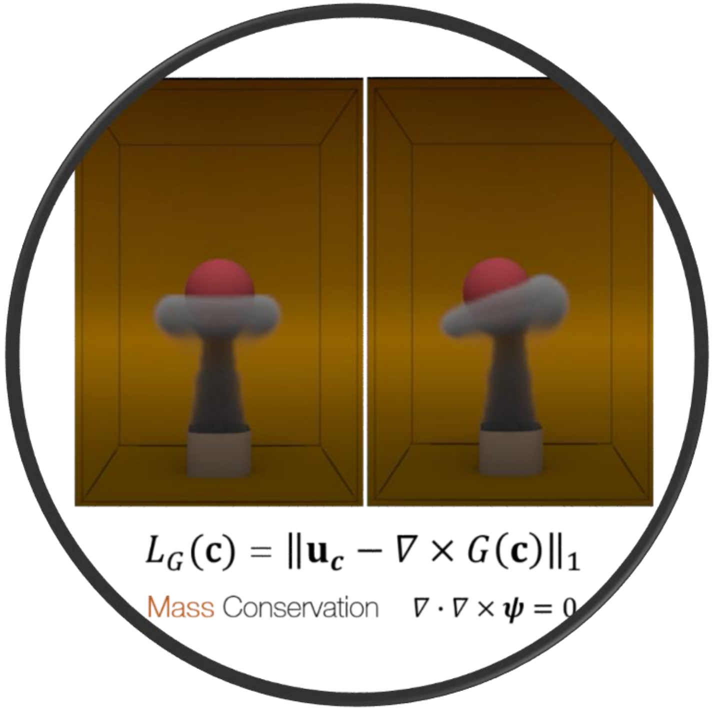
- **Velocity fields** differ from **images**
  - Spatial-temporal data which not possess "Eigenshapes"
  - Different statistical features
  - Standard image-based networks are not optimal

Overview

Dataset



Architecture



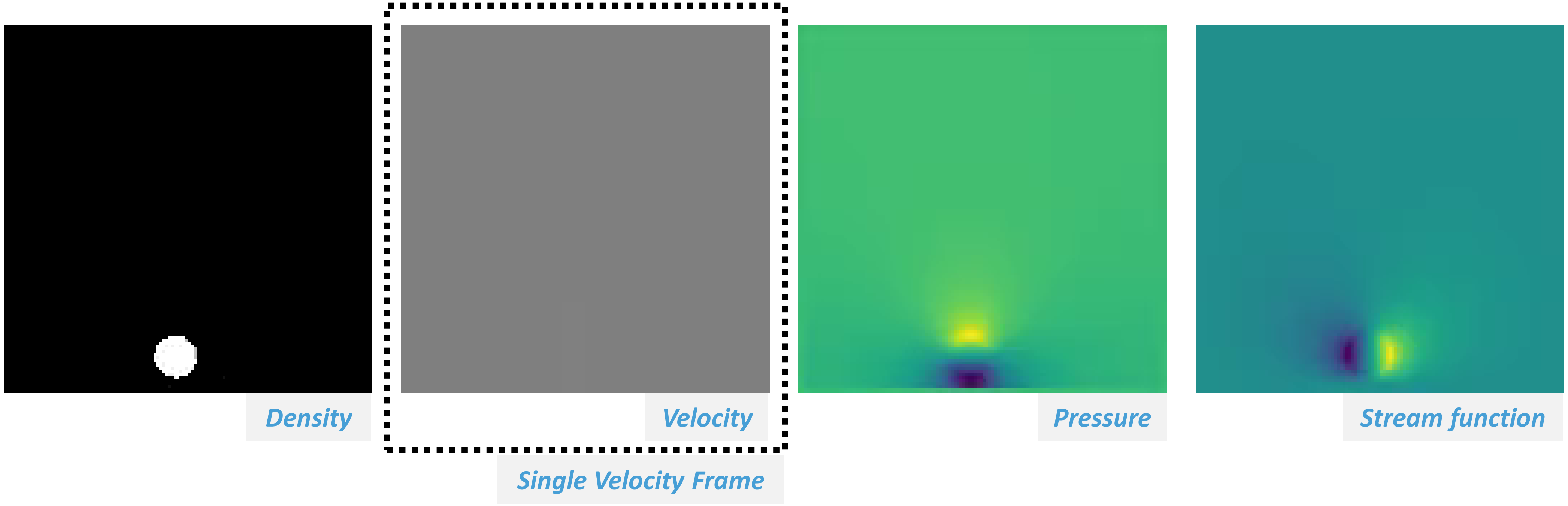
Loss Function

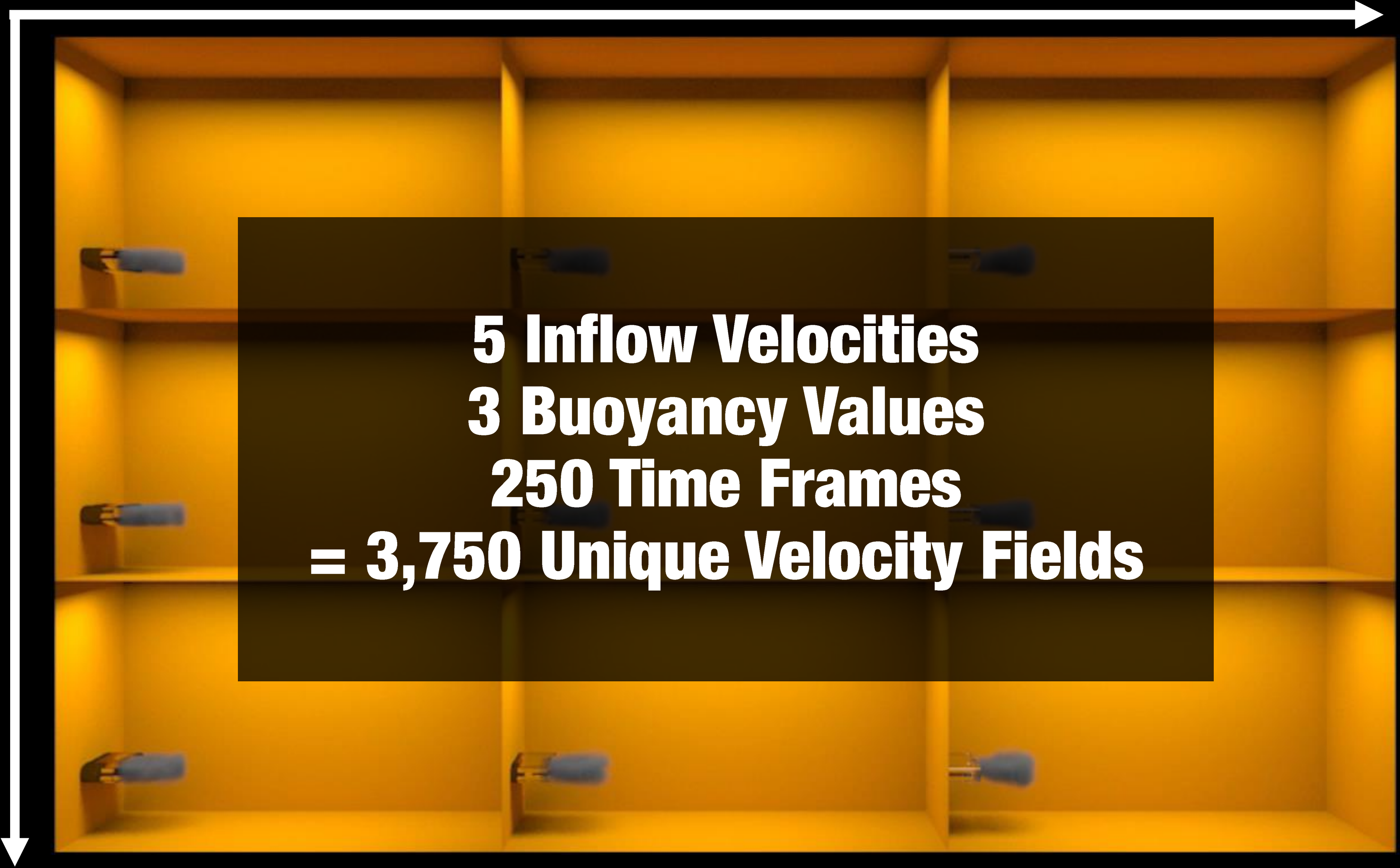
Fluid Simulation Data

» Density, Velocity, Pressure and Stream function

$$\frac{\partial \boldsymbol{u}}{\partial t} = \boldsymbol{g} - \boldsymbol{u} \cdot \nabla \boldsymbol{u} - \frac{1}{\rho} \nabla p + \mu \nabla \cdot \nabla \boldsymbol{u}$$ Momentum Conservation

$$\nabla \cdot \boldsymbol{u} = 0$$ Mass Conservation



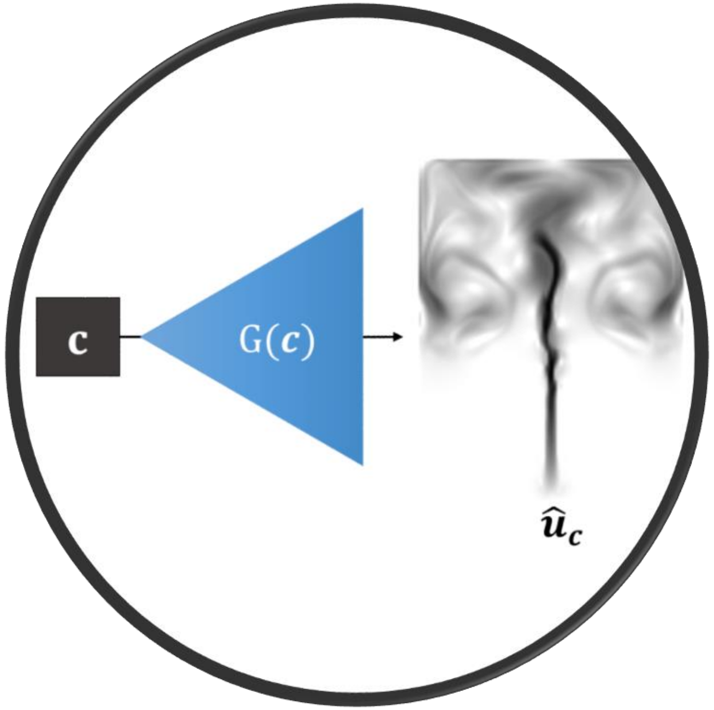


Ground-truth (simulated) velocity-field examples

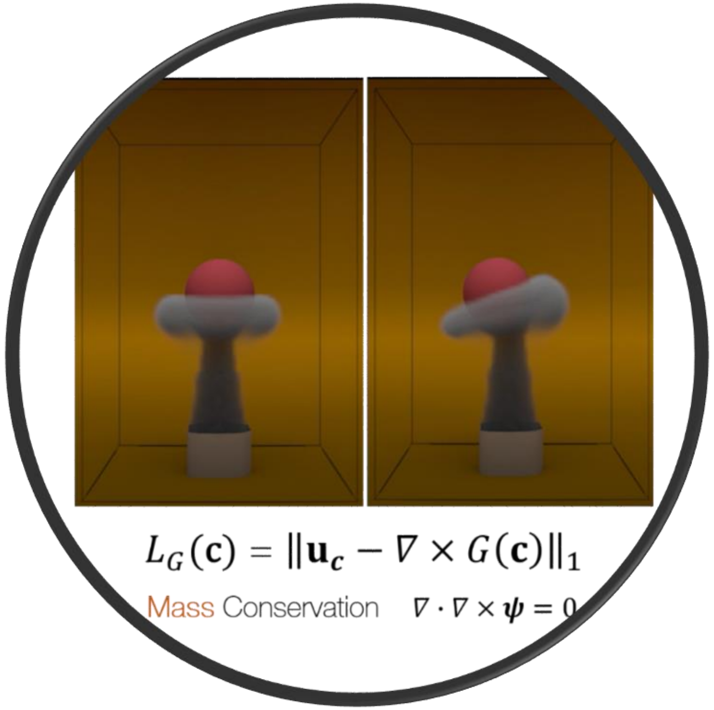
Input Parameters [velocity, buoyancy, time]

Overview

Dataset



Architecture

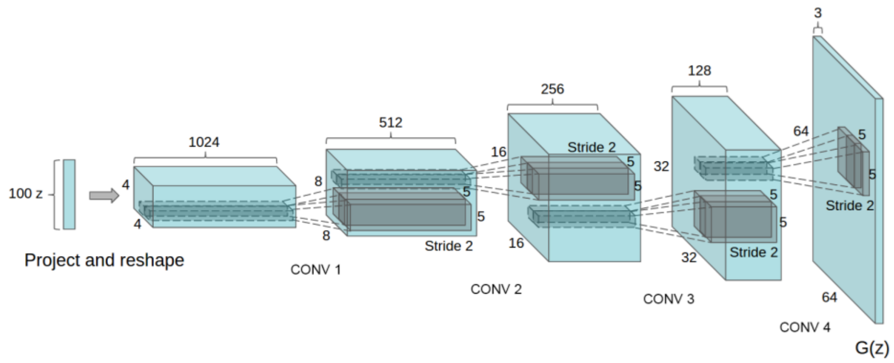


Loss Function

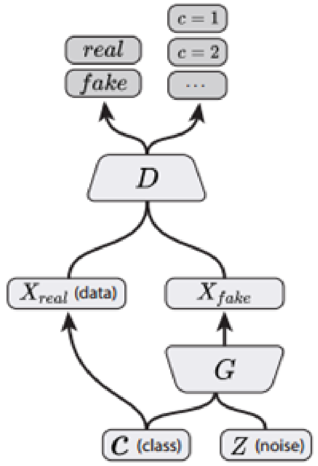
Generative Model

» **Supervised vs. Unsupervised Learning**

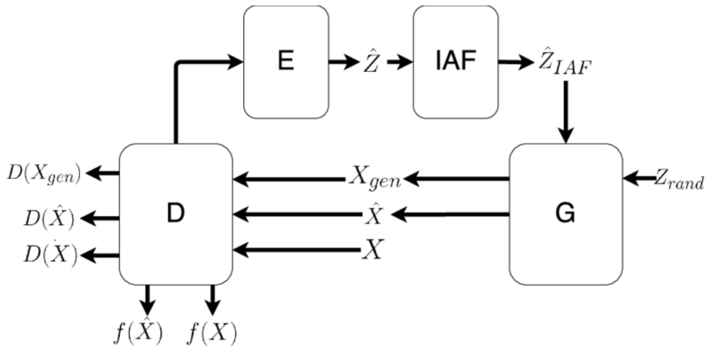
- Tried several unsupervised architectures



DCGAN Generator



ACGAN

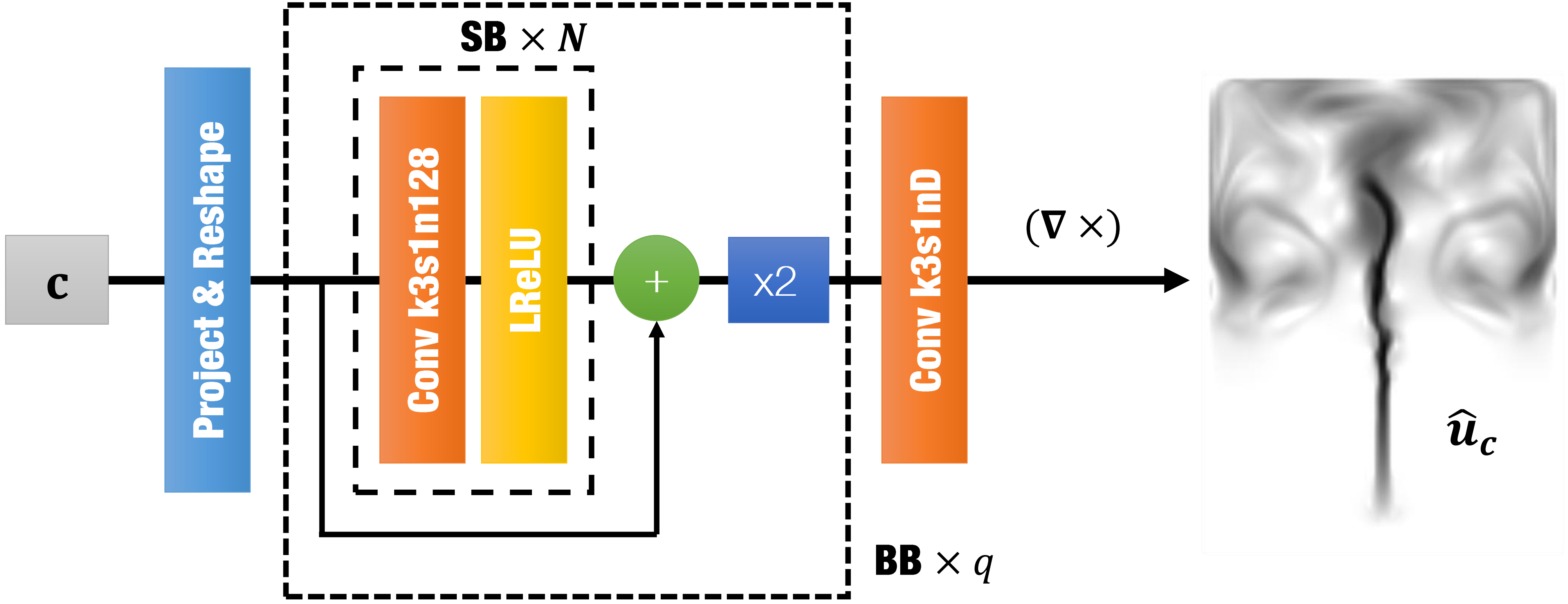


Introspective GAN

- Energy-Based GAN
- Least-Squares GAN
- Boundary Equilibrium GAN

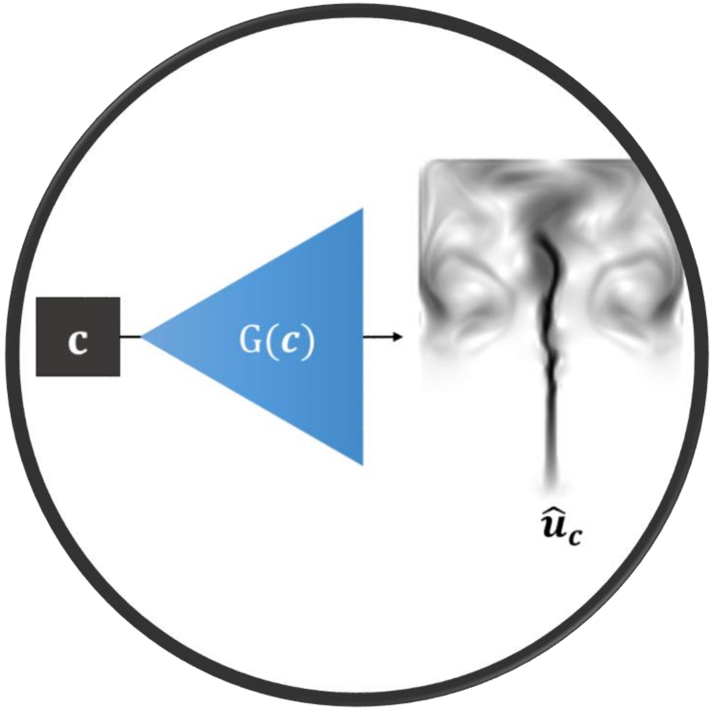
Generative Model

» Architecture

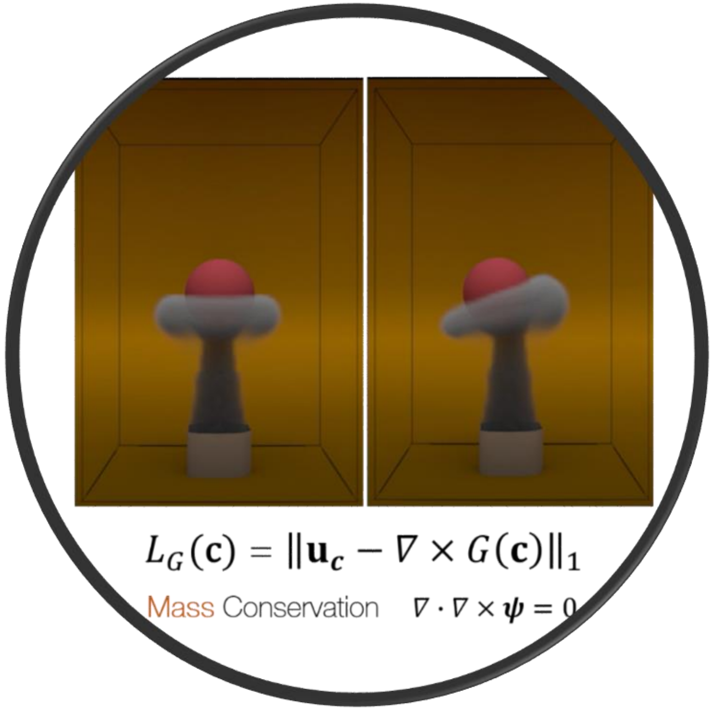


Overview

Dataset



Architecture



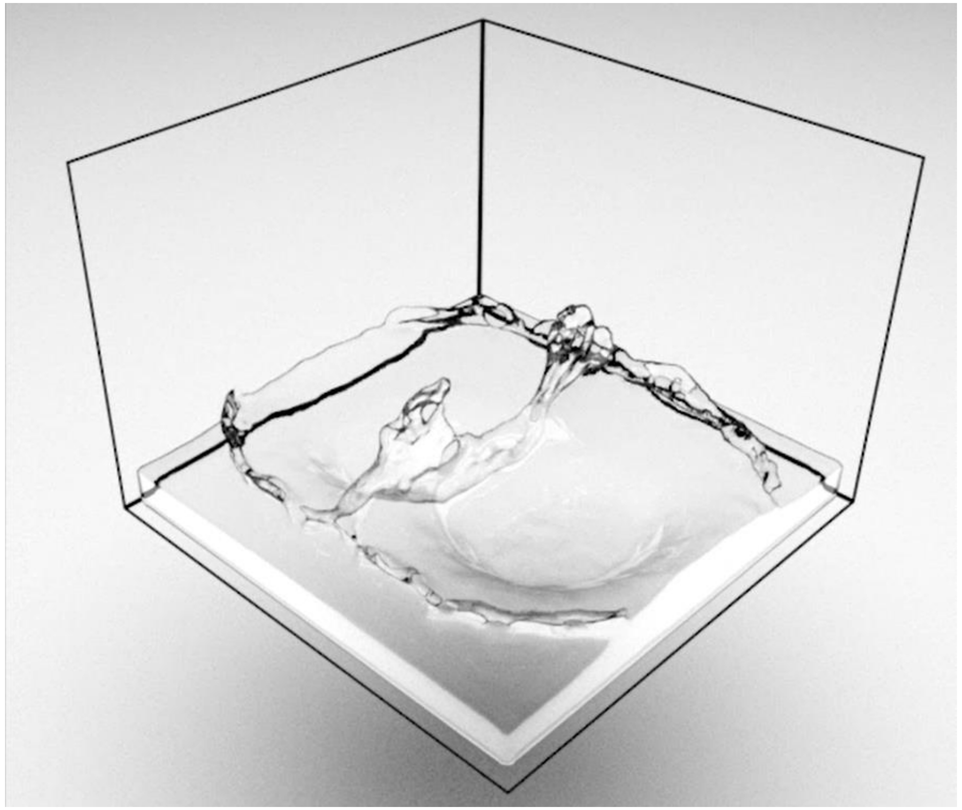
Loss Function

Generative Model

» Stream Function based Loss Function

$$L_G(\mathbf{c}) = \|\mathbf{u}_c - \nabla \times G(\mathbf{c})\|_1$$

Mass Conservation $\nabla \cdot \nabla \times \boldsymbol{\psi} = 0$

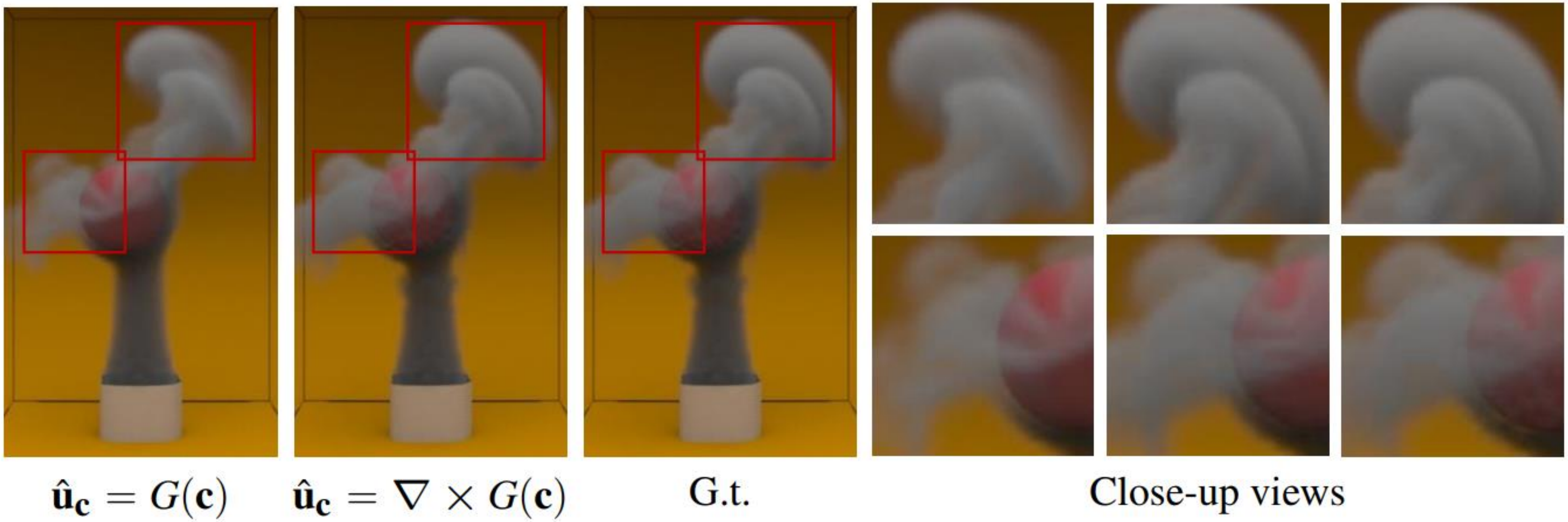


$$L_G(\mathbf{c}) = \|\mathbf{u}_c - G(\mathbf{c})\|_1$$

Partially Divergent Motion (Liquids)

Generative Model

» Comparison of Stream Function and Velocity based Loss Functions

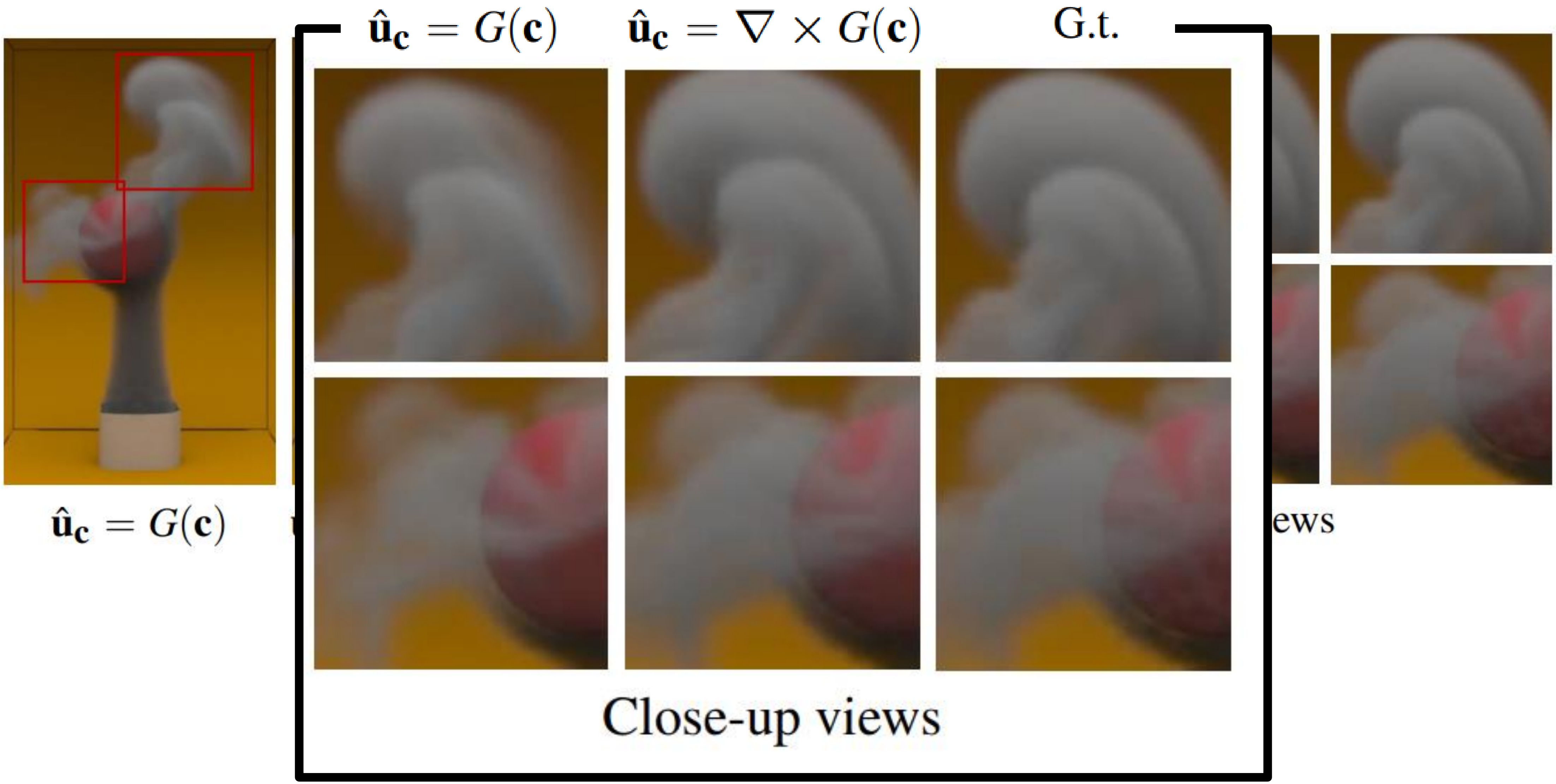


$$L_G(\mathbf{c}) = \|\mathbf{u}_c - \nabla \times G(\mathbf{c})\|_1$$

Mass Conservation $\nabla \cdot \nabla \times \boldsymbol{\psi} = 0$

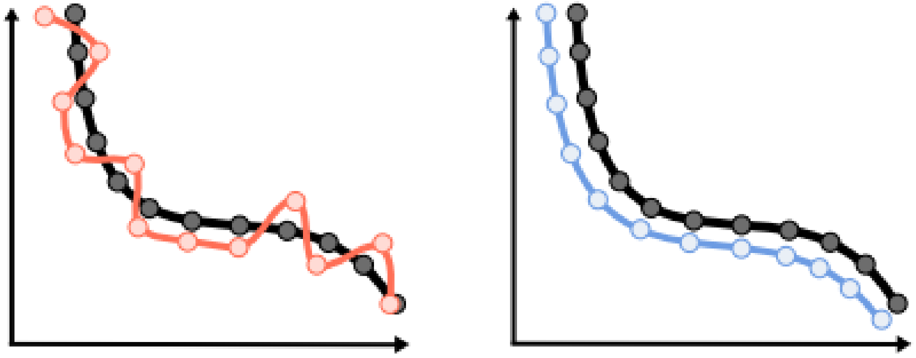
Generative Model

» Comparison of Stream Function and Velocity based Loss Functions



Generative Model

» Jacobian Loss Function for Smooth Data

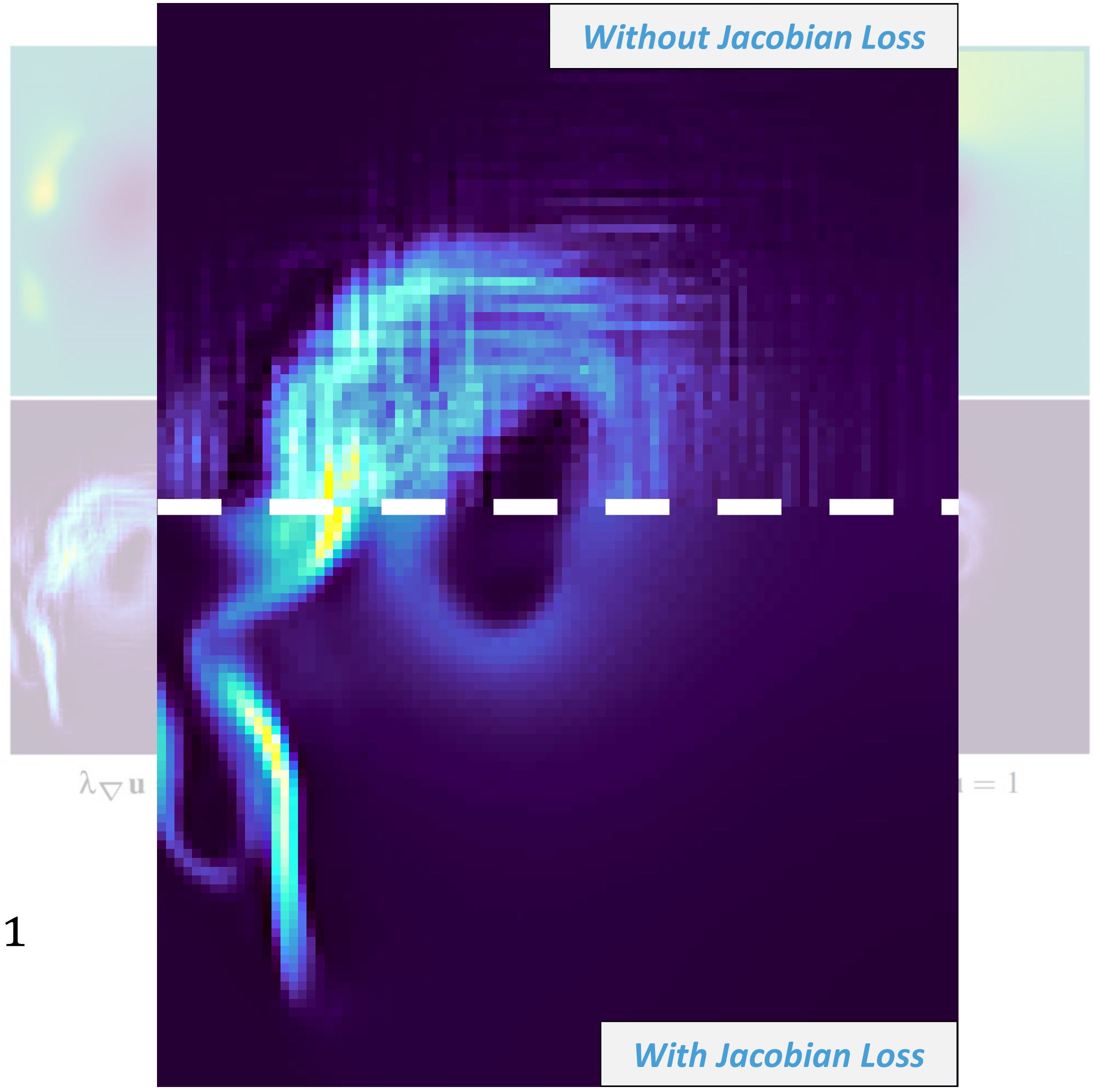


Same L1 Loss Function Value

- Pure L1 loss function does not pin-down derivatives for smooth data
- We increment the loss to also match derivatives of the original data

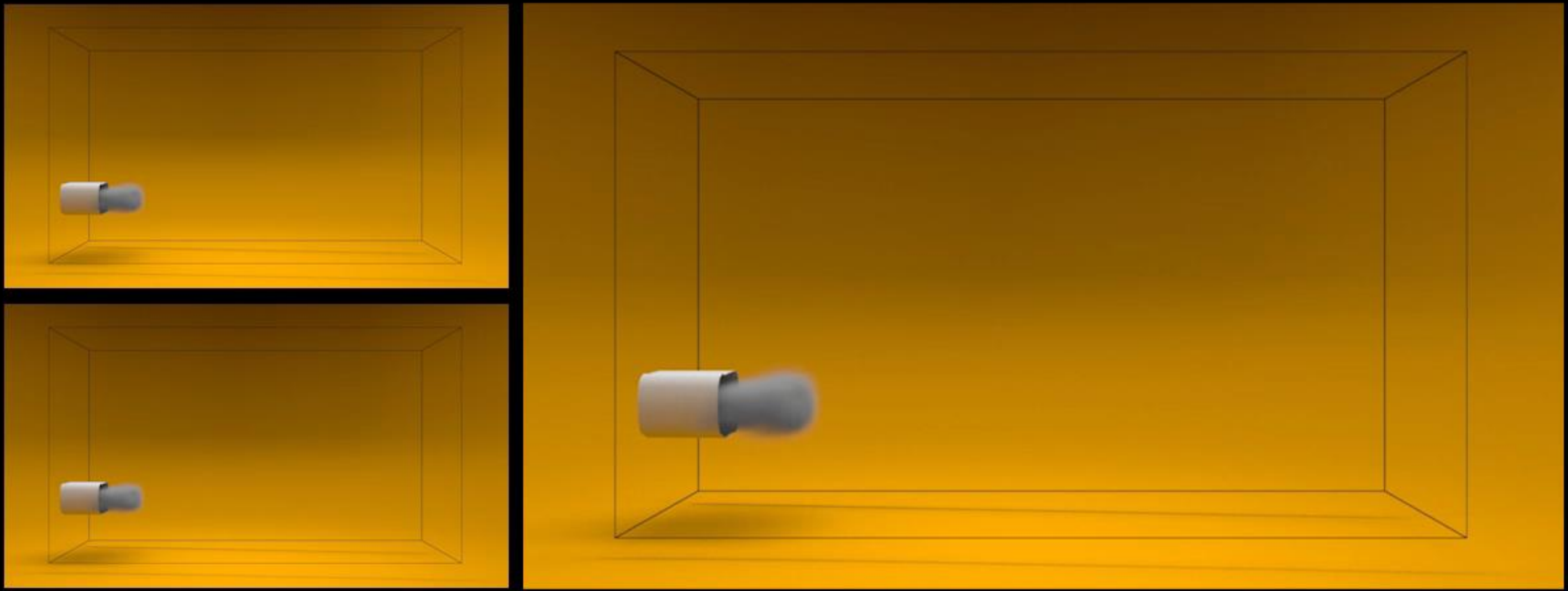
$$L_G(\mathbf{c}) = \lambda_{\mathbf{u}} \|\mathbf{u}_c - \hat{\mathbf{u}}_{\mathbf{c}}\|_1 + \lambda_{\nabla \mathbf{u}} \|\nabla \mathbf{u}_c - \nabla \hat{\mathbf{u}}_{\mathbf{c}}\|_1$$

where $\hat{\mathbf{u}}_{\mathbf{c}} = \nabla \times G(\mathbf{c})$ or $\hat{\mathbf{u}}_{\mathbf{c}} = G(c)$



Results for Parameterizable Scenes

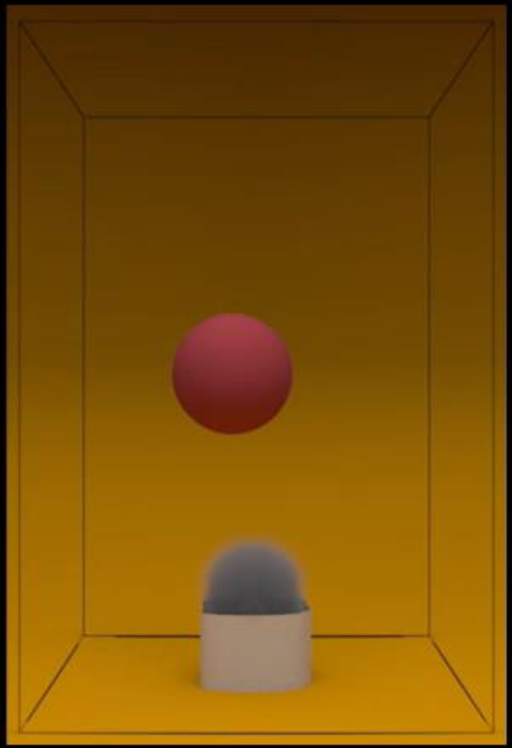
Inflow Velocity Interpolation Example



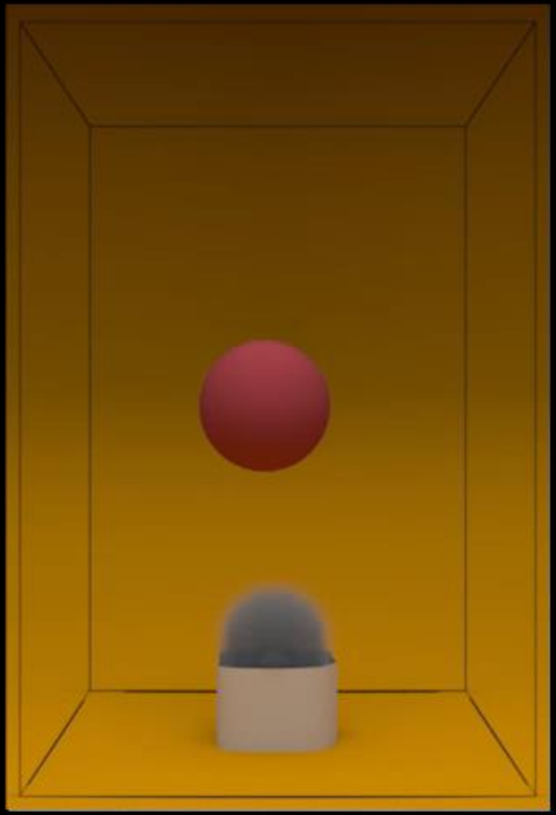
Direct correspondence
for velocities $v_x = 4$ and $v_x = 5$

Interpolated with $v_x = 4.5$, **Not present** in the original **data set**

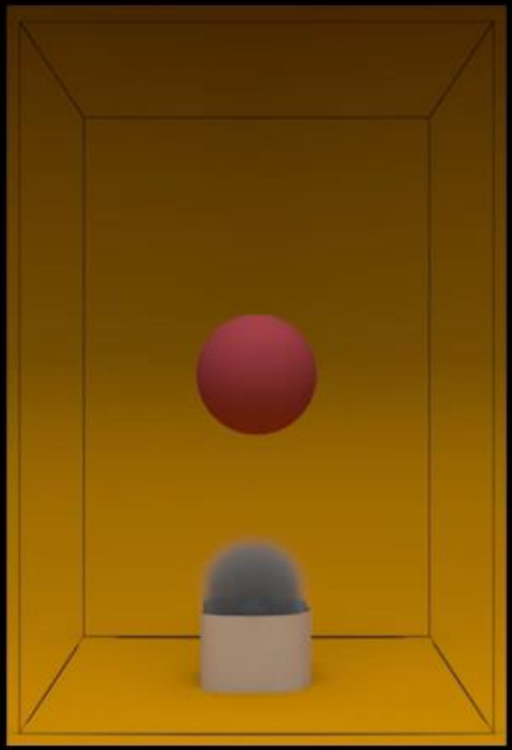
Obstacle Scene Interpolation



CNN Reconstruction for position $p_x = 0.44$

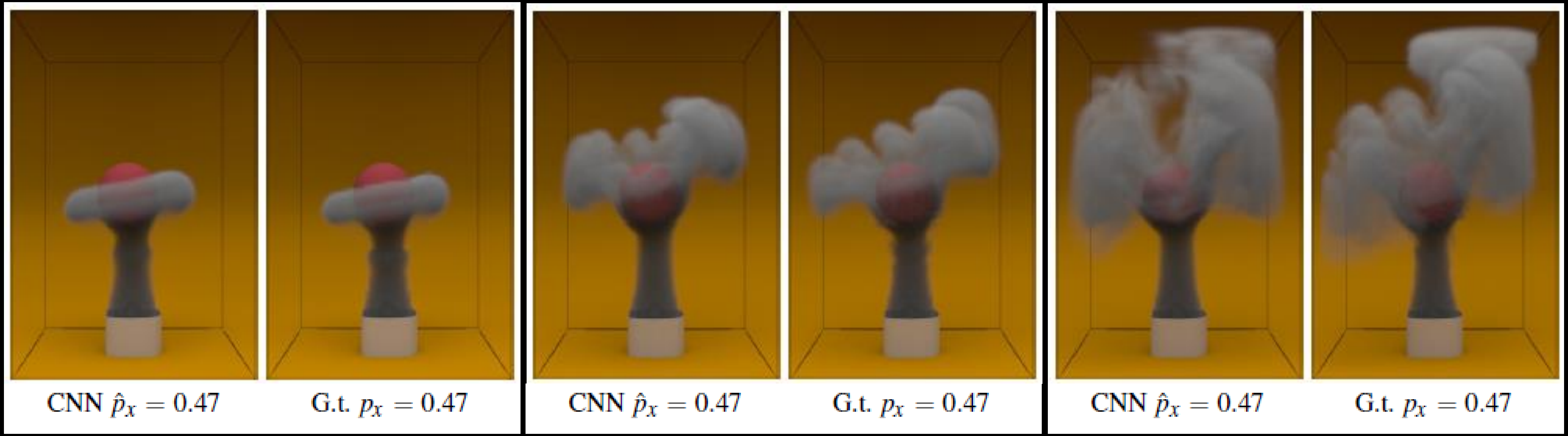


CNN Interpolation for position $p_x = 0.47$

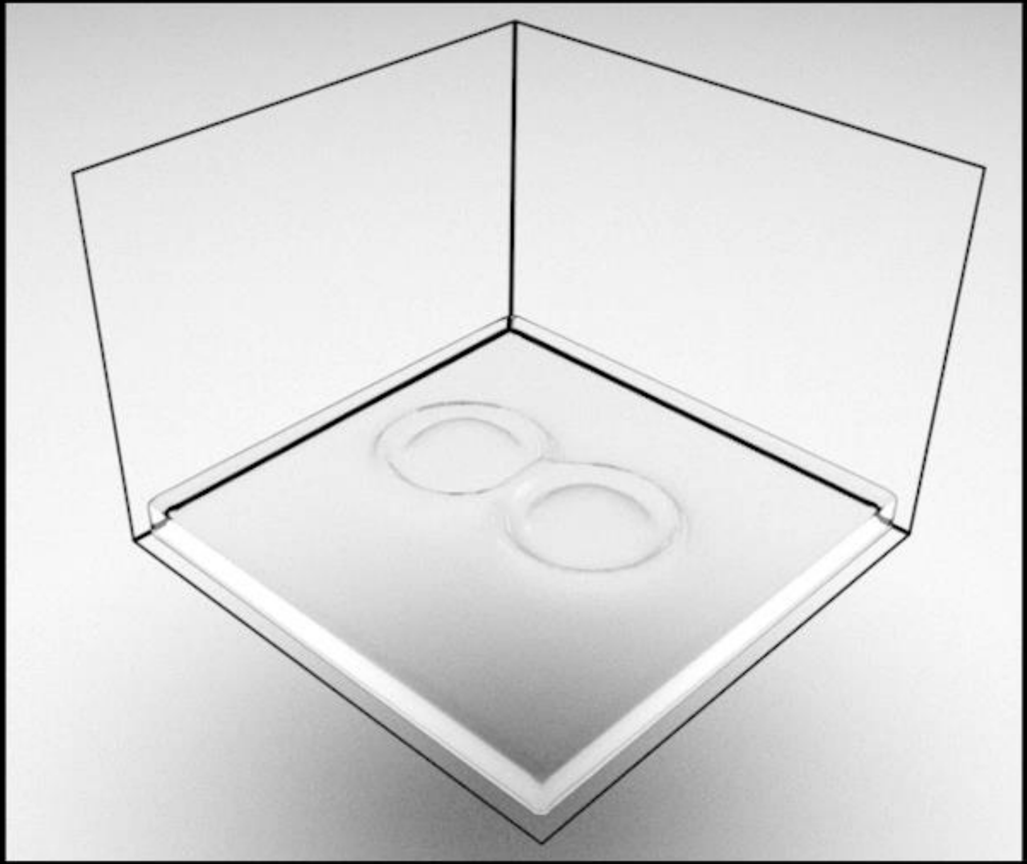


CNN Reconstruction for position $p_x = 0.50$

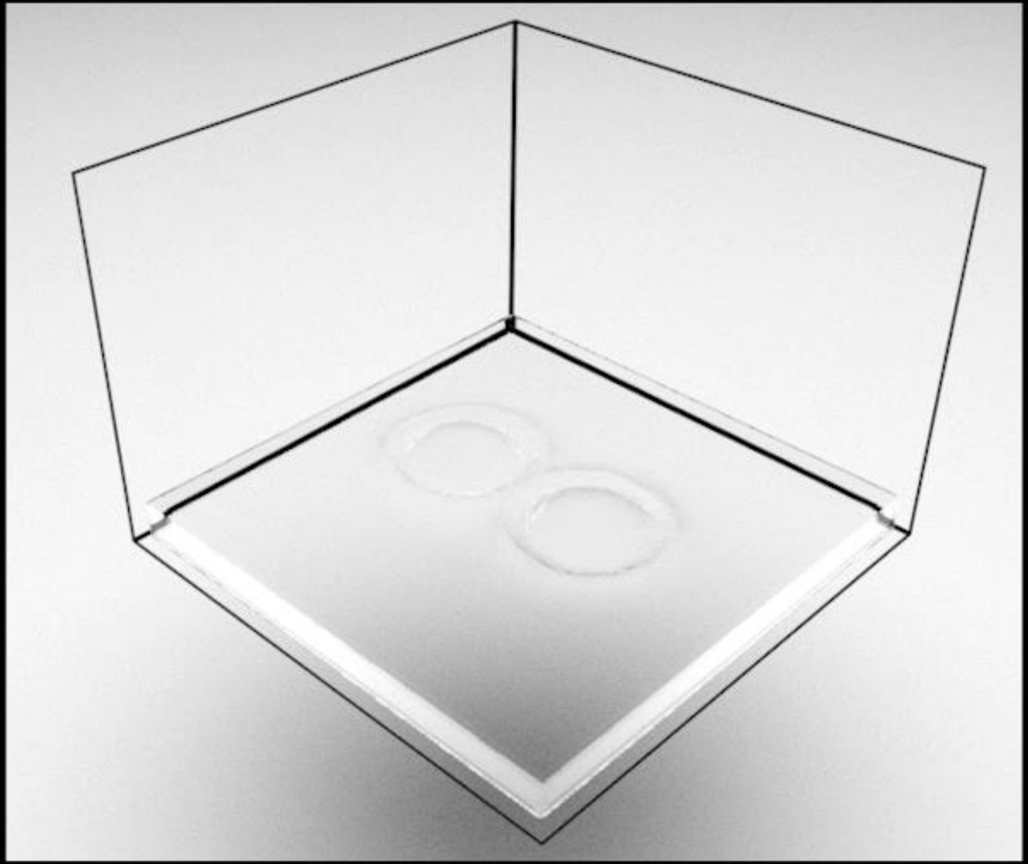
Deep Fluids: Obstacle Scene



Deep Fluids: Liquids

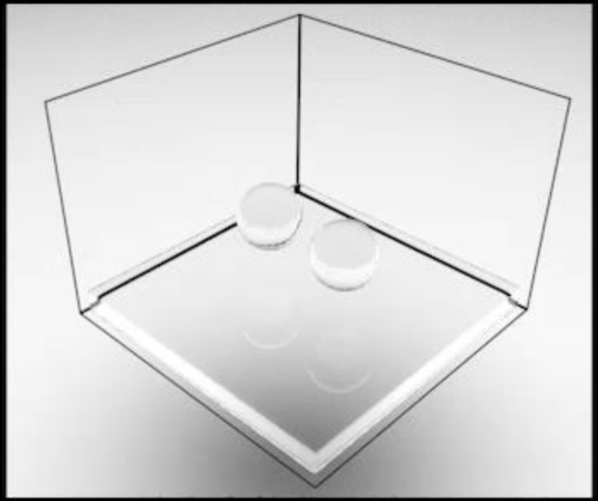


Ground-Truth

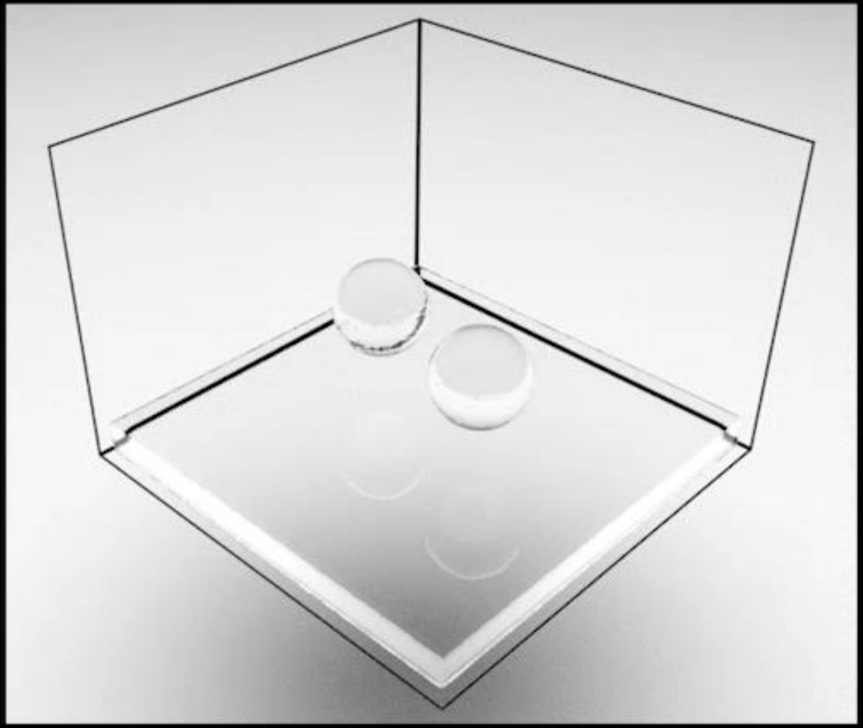


Reconstructed by our CNN

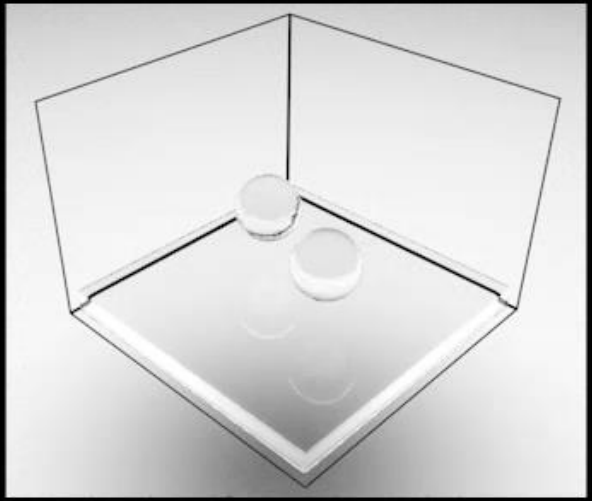
Deep Fluids: Liquids



CNN Reconstruction for distance $d = 0.15$ and angle $\theta = 0^\circ$

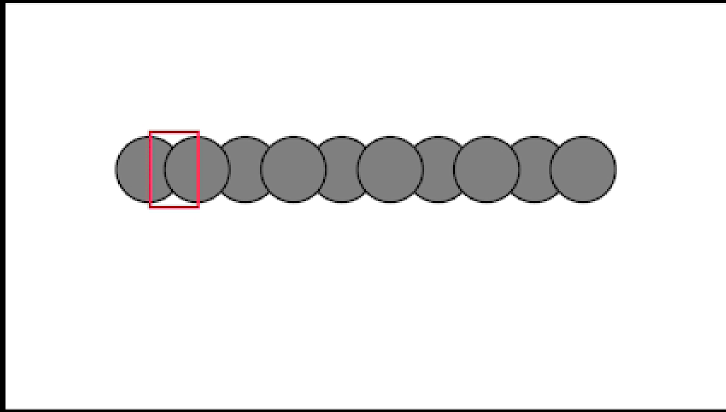


CNN Interpolation for distance $d = 0.1625$ and angle $\theta = 9^\circ$

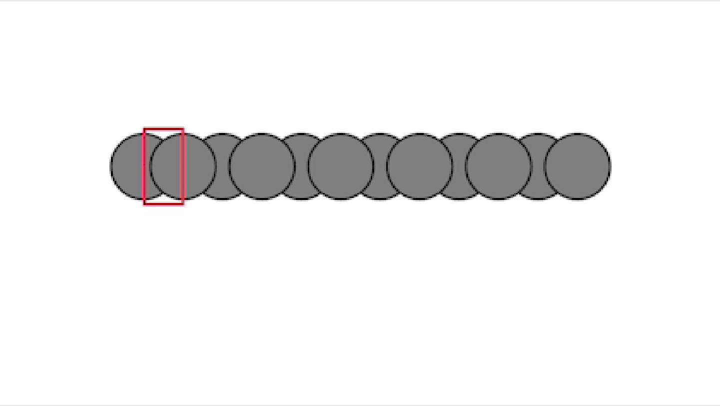


CNN Reconstruction for distance $d = 0.175$ and angle $\theta = 18^\circ$

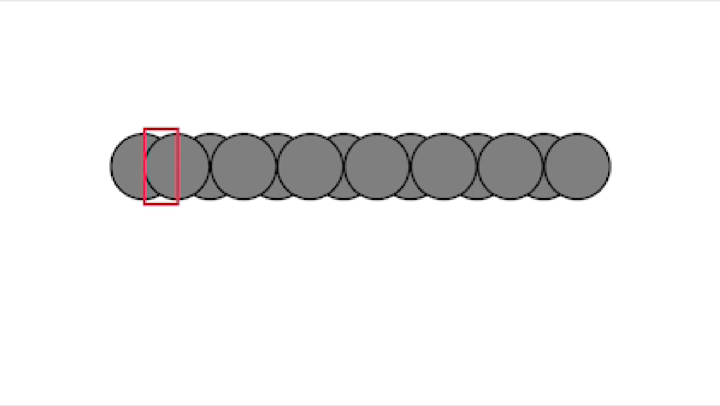
Deep Fluids: Liquids in 2D



10 Positions

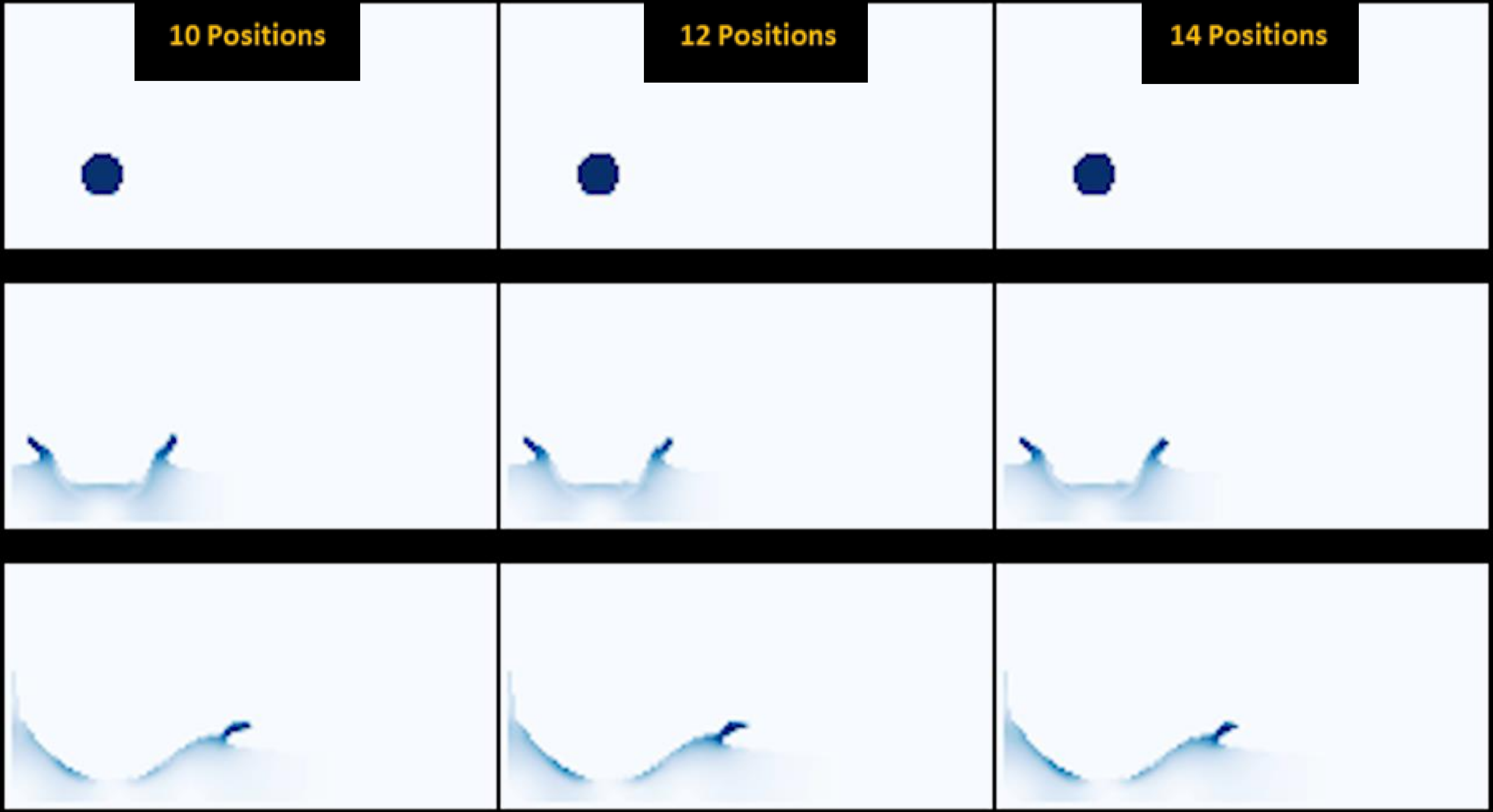


12 Positions

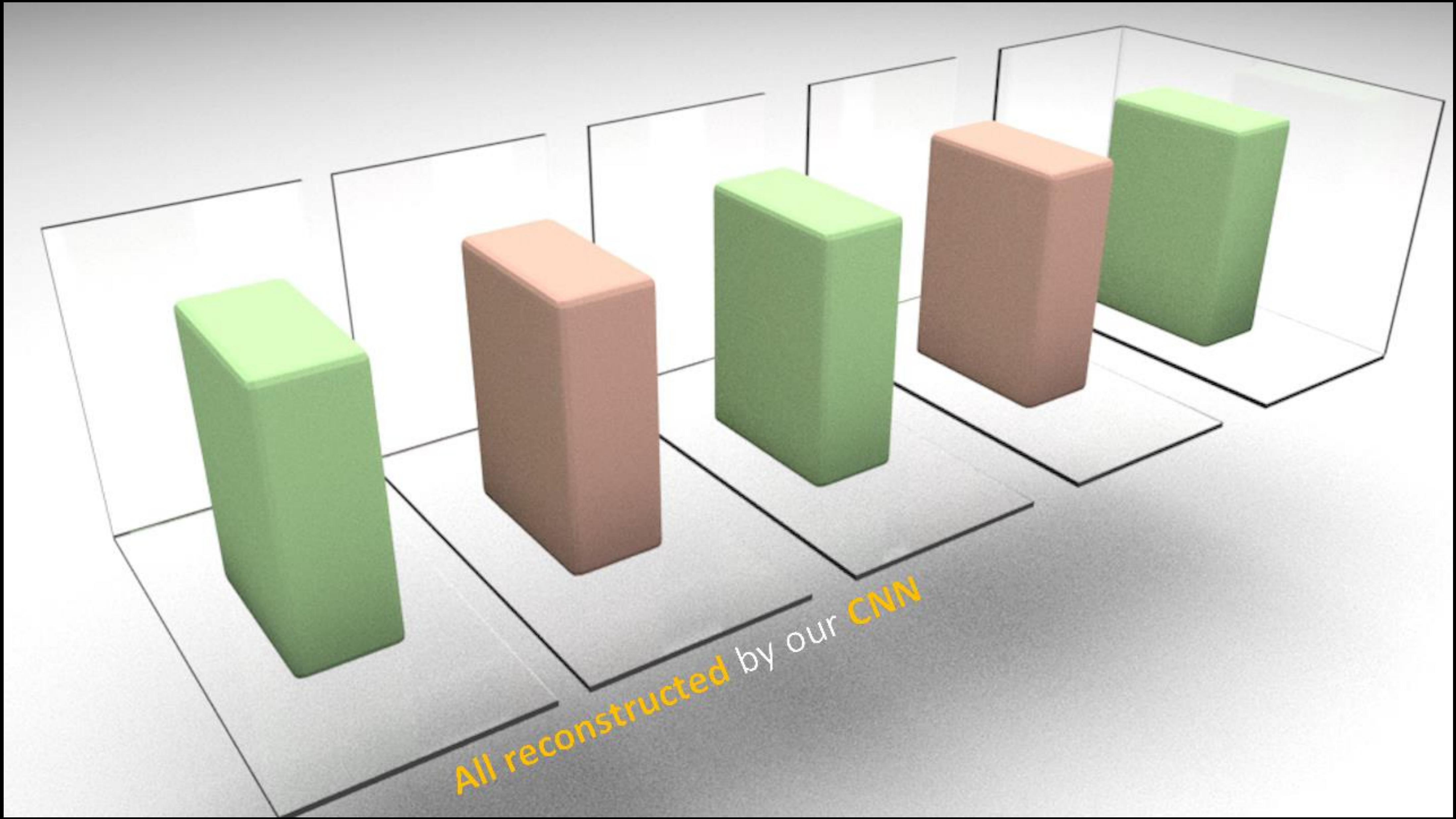


14 Positions

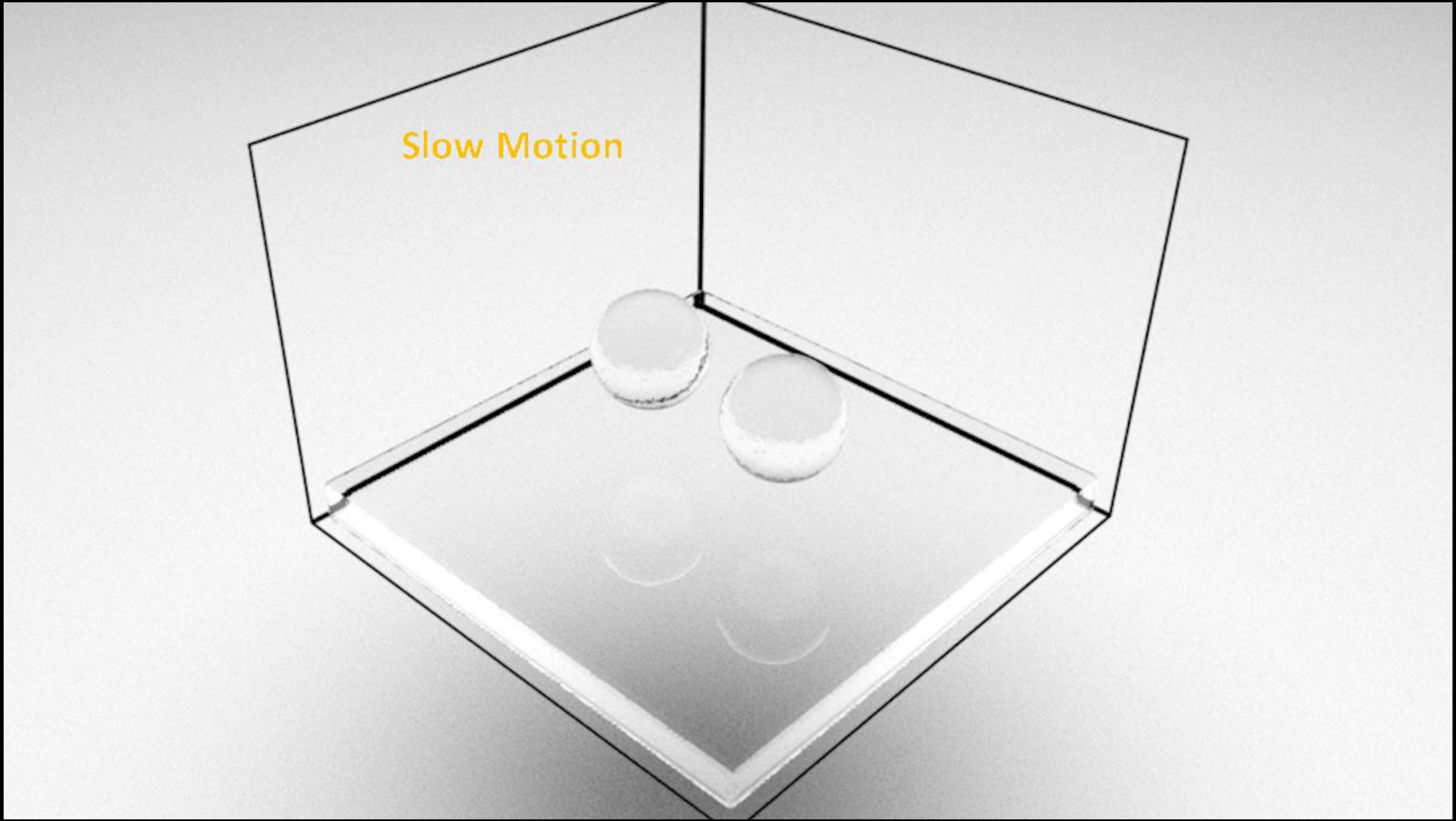
Deep Fluids: Liquids in 2D



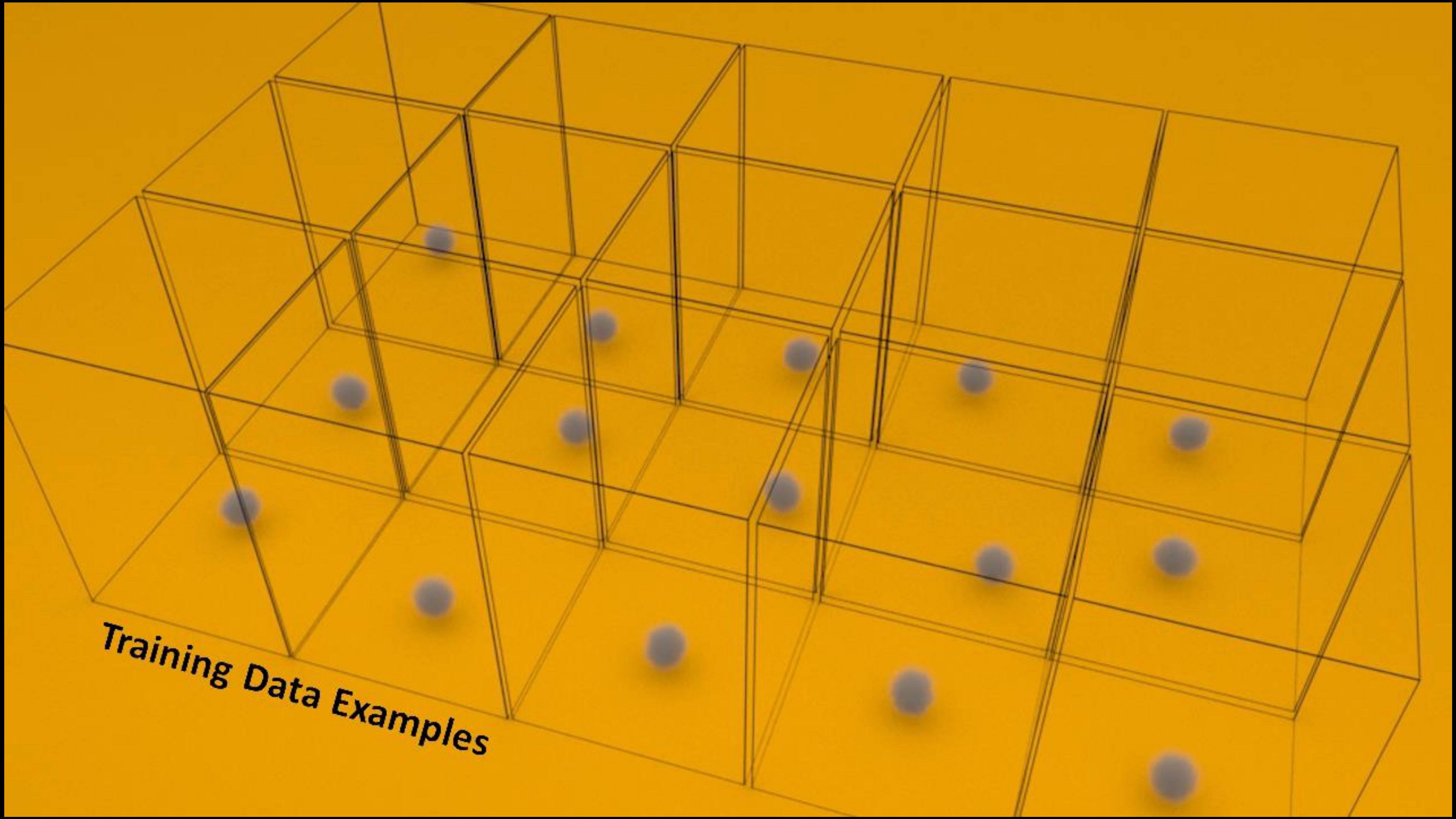
Deep Fluids: Liquids



Deep Fluids: Liquids



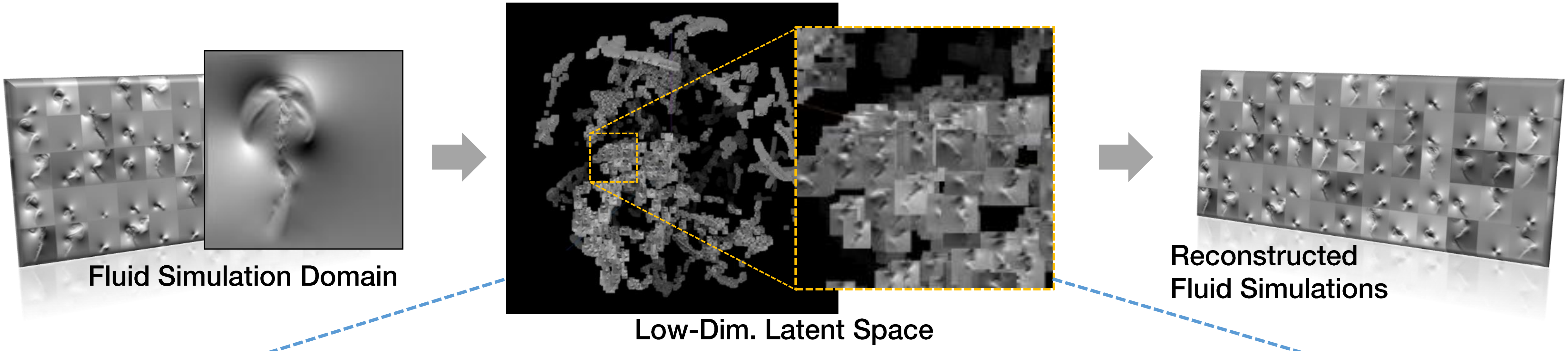
Towards Extended Parameterizations



Input Parameters [history of source positions, time]

Extended Parameterizations

» Learning a Fluid Data Manifold



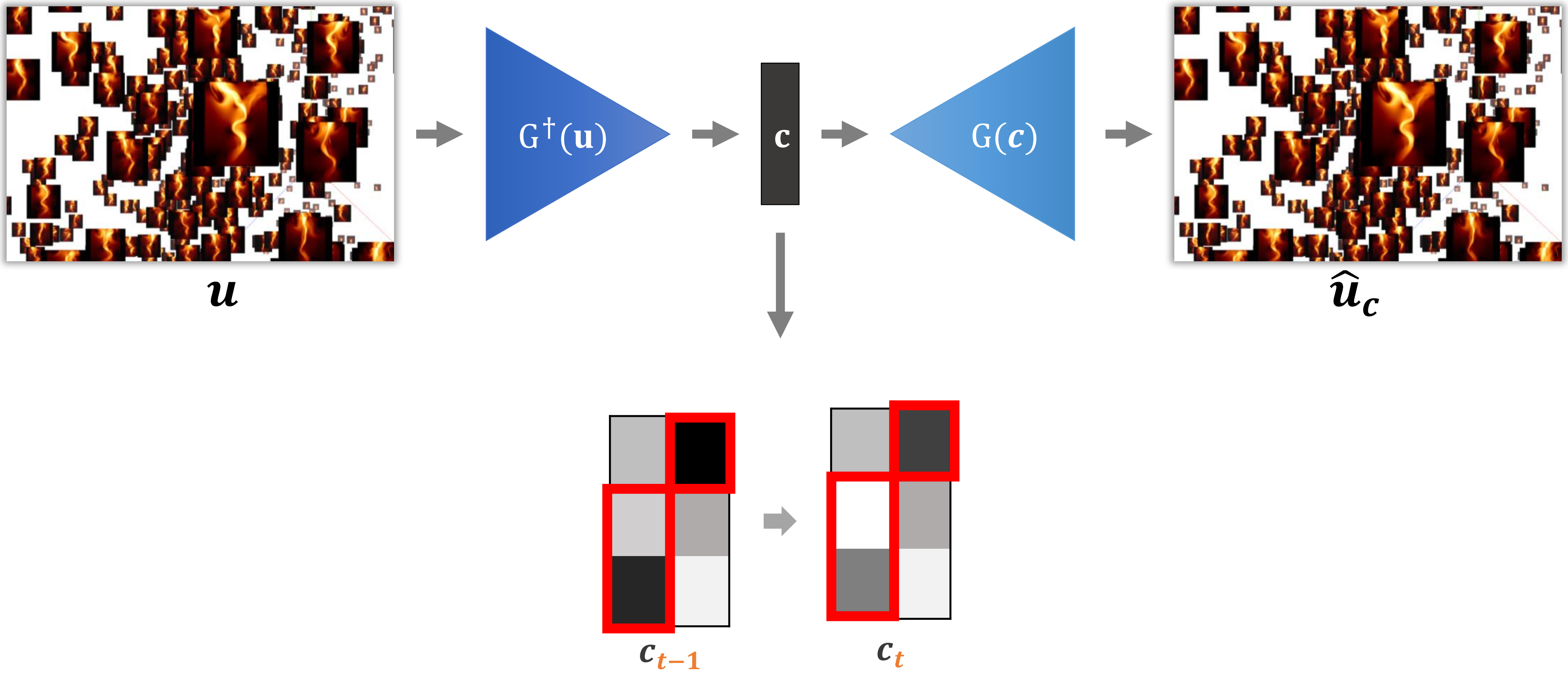
Dimensionality Reduction for Manifold Learning

Linear: PCA, SVD, ICA, Factor Analysis, etc.

Nonlinear: Auto-Encoder, LLE, Laplacian Eigenmaps, etc.

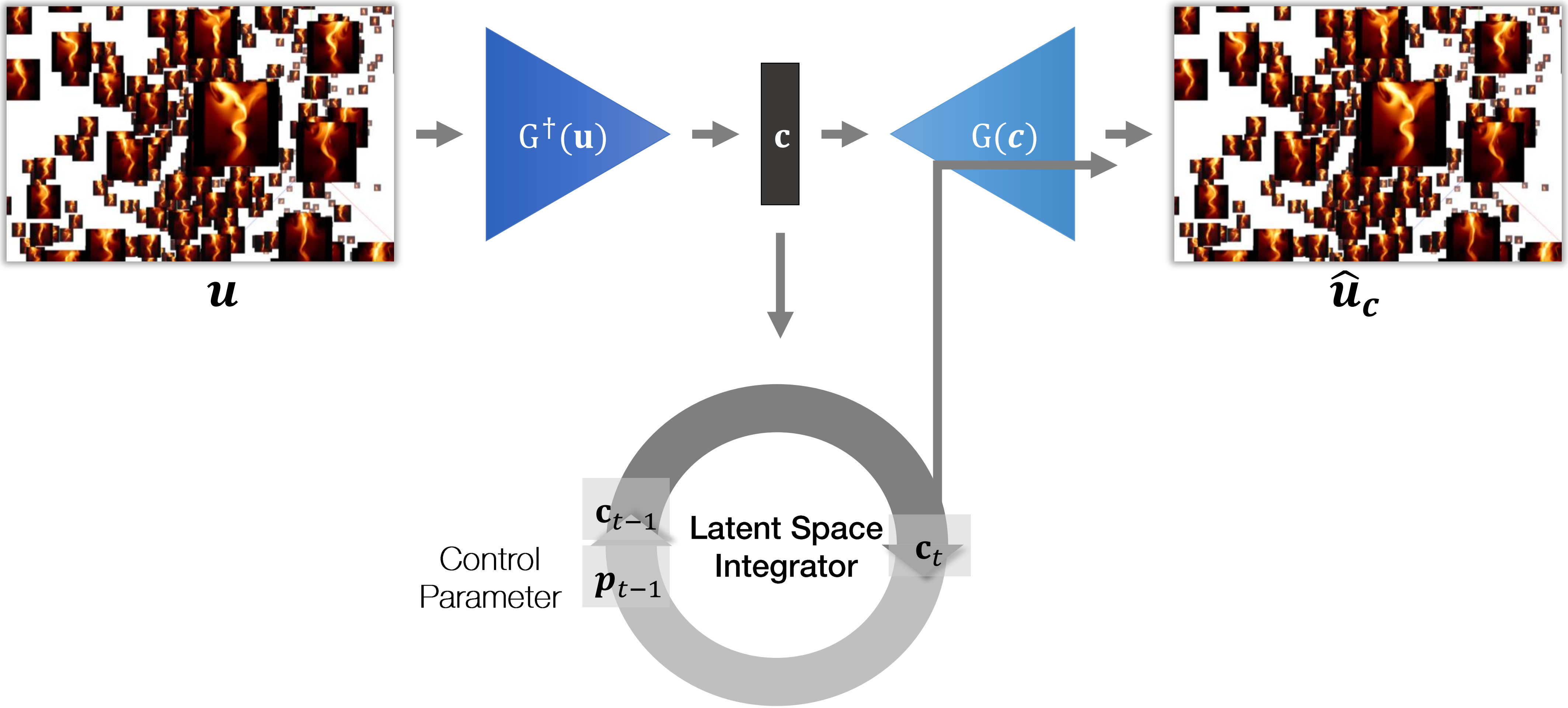
Extended Parameterizations

» Latent-Space Integration

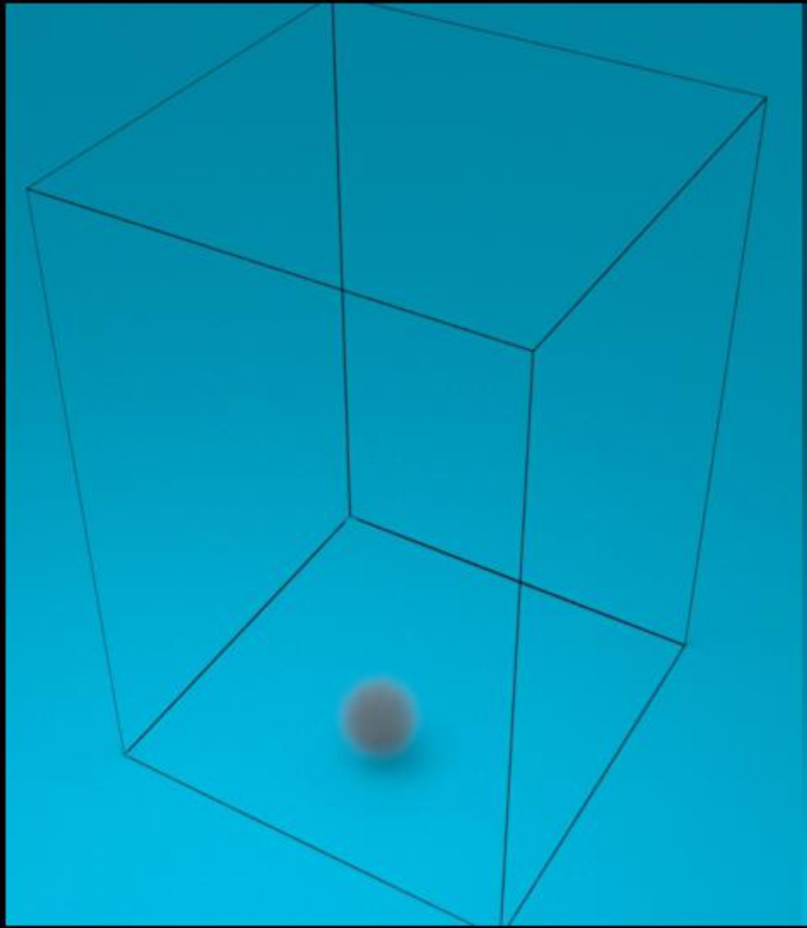


Extended Parameterizations

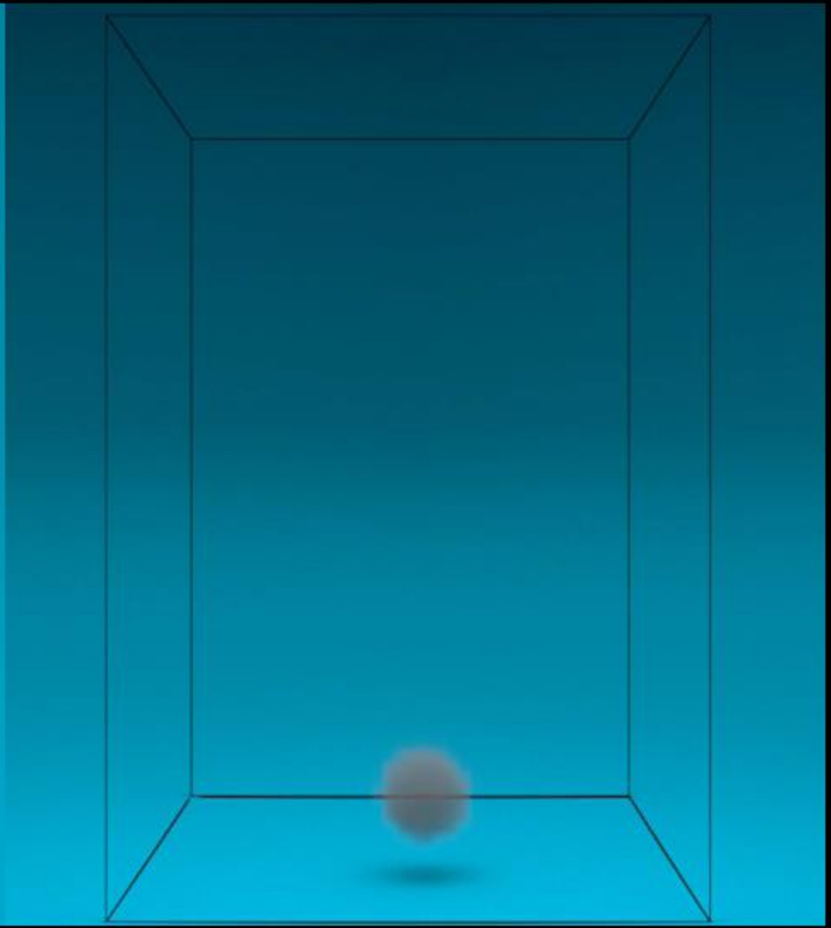
» Latent-Space Integration



Latent Space Simulation: New Source Motion



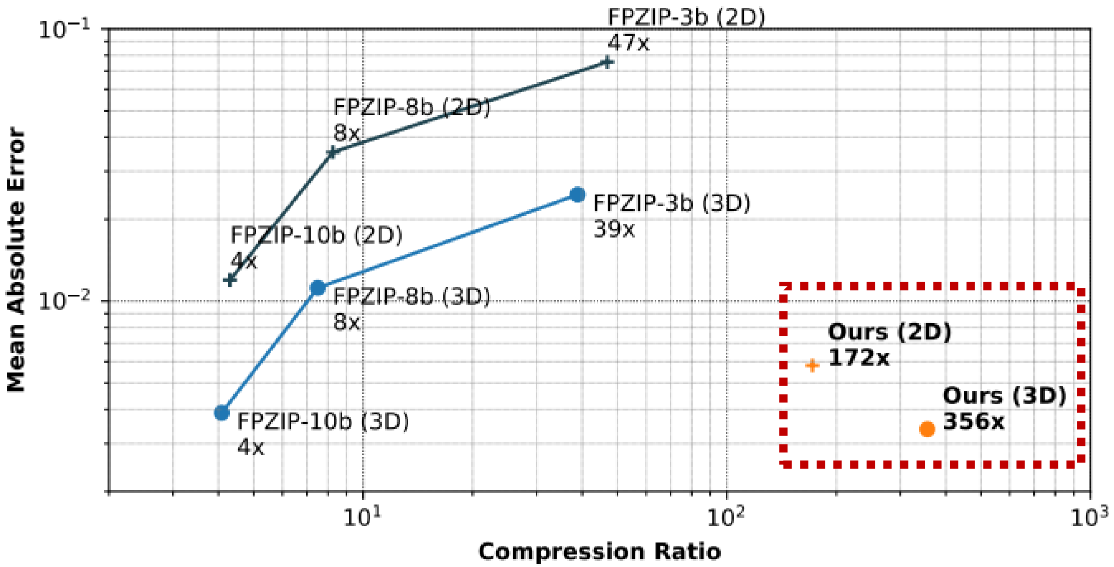
Perspective View



Front View

Performance

Scene	Grid Resolution	# Frames	Simulation Time (s)	Eval. Time (ms) [Batch]	**Speed Up** (×)	Data Set Size (MB)	Network Size (MB)	**Compression Ratio**	Training Time (h)
Smoke Plume	96 × 128	21,000	0.033	0.052 [100]	635	2064	12	172	5
Smoke Obstacle	64 × 96 × 64	6,600	0.491	0.999 [5]	513	31143	30	1038	74
Smoke Inflow	112 × 64 × 32	3,750	0.128	0.958 [5]	128	10322	29	356	40
Liquid Drops	96 × 48 × 96	7,500	0.172	1.372 [3]	125	39813	30	**1327**	134
Viscous Dam	96 × 72 × 48	600	0.984	1.374 [3]	**716**	2389	29	82	100
Rotating Smoke	48 × 72 × 48	500	0.08	0.52 [10]	308	995	38	26	49
Moving Smoke	48 × 72 × 48	80,000	0.08	0.52 [10]	308	159252	38	4191*	49



Limitations

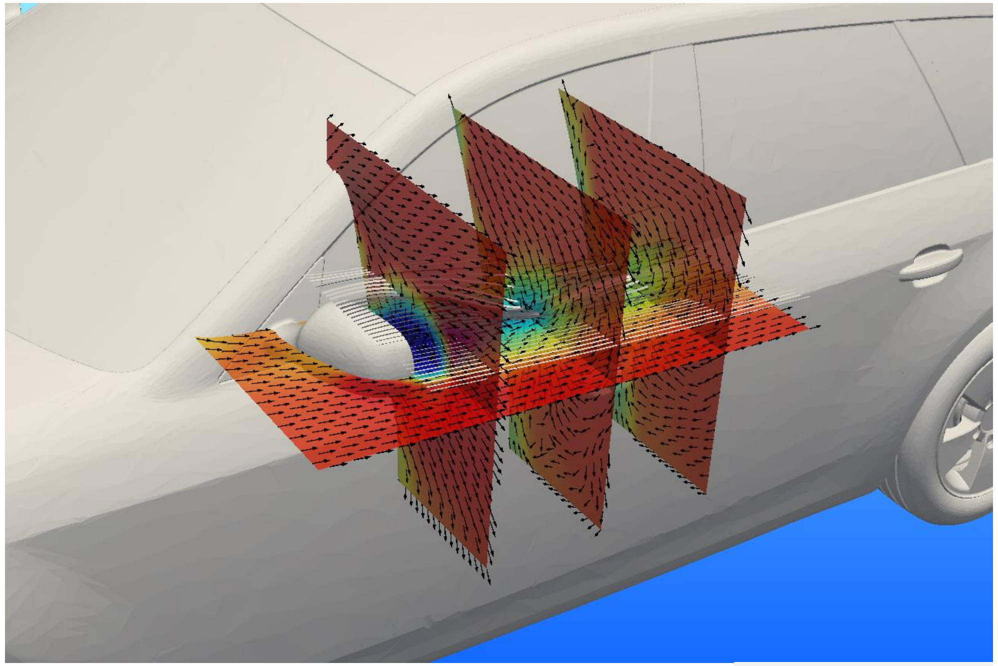
- ## Quality of Reconstruction
  - Reconstructed data is too smooth
    - GANs are useful for hallucinating high-frequency details but not physically plausible
  - Ghosting may happen if data is not sampled with enough density
  - Boundary Conditions introduce discontinuities that might leave liquid particles hanging

- ## Latent Space Integration
  - Learned latent space of AE can be improved
  - Using simply MLP for time integration is not optimal

Summary

- ## Contributions
  - Fast and plausible approximation of parameterized Eulerian fluid simulations with high compression ratio
  - Novel latent space integrator
  - Suitable for games and real-time virtual environments

- ## Future Work
  - Boundary conditions
  - Improved Latent-space integration LSTMs
  - Bypassing modelling by directly reconstructing captured data



@Streamwise

Thanks for your attention [https://github.com/byungsook/deep-fluids]