EUROGRAPHICS 2023 / K. Myszkowski and M. Nießner
(Guest Editors)

COMPUTER GRAPHICS *forum*
Volume 42 (2023), Number 2

Physics-Informed Neural Corrector for Deformation-based Fluid Control

Jingwei Tang[1], Byungsoo Kim[1], Vinicius C. Azevedo[2], Barbara Solenthaler[1]

[1]ETH Zürich, Switzerland
[2]DisneyResearch|Studios, Switzerland

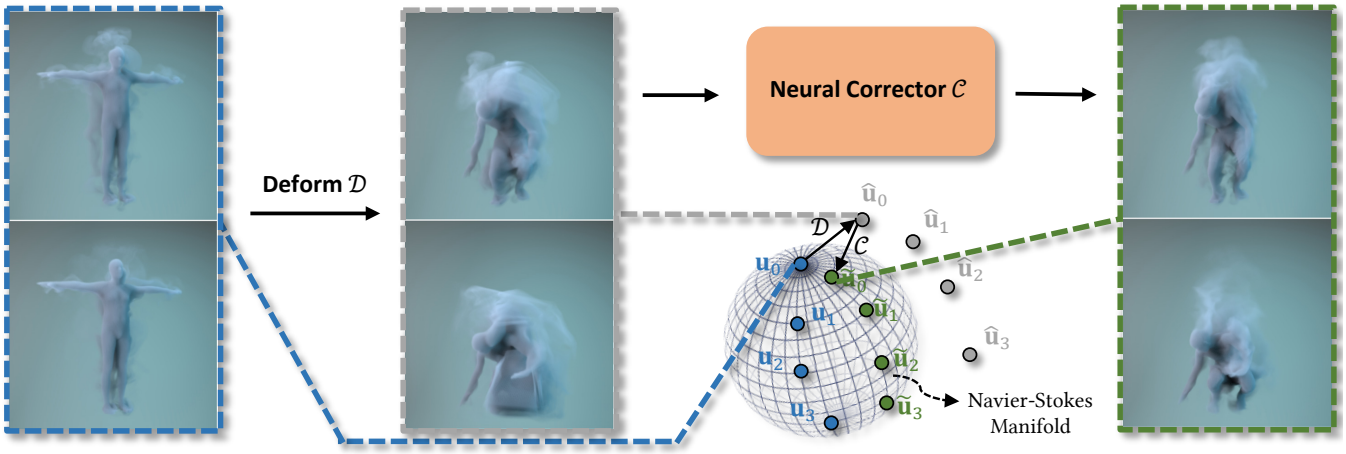

Figure 1: *We present a method to rectify deformed fluid flows using neural networks. Our neural corrector ensures the physical plausibility of edited simulation footprints at test time, enabling interactive control of fluids without re-simulations.*

Abstract

Controlling fluid simulations is notoriously difficult due to its high computational cost and the fact that user control inputs can cause unphysical motion. We present an interactive method for deformation-based fluid control. Our method aims at balancing the direct deformations of fluid fields and the preservation of physical characteristics. We train convolutional neural networks with physics-inspired loss functions together with a differentiable fluid simulator, and provide an efficient workflow for flow manipulations at test time. We demonstrate diverse test cases to analyze our carefully designed objectives and show them leadingthat they lead to physical and eventually visually appealing modifications on edited fluid data.

CCS Concepts

• ***Computing methodologies*** → ***Physical Simulation; Neural networks;***

1. Introduction

Faithfully recreating natural phenomena in virtual environments is one of the most significant topics in the field of computer graphics. Recent developments in fluid simulations have allowed artists to efficiently author realistic effects. However, when it comes to the problem of art-directing, *e.g.*, screen-space control of features or re-timing the simulation, a compromise is assumed between maintaining accessible artistic control and physical plausibility.

Artists often rely on force-based fluid control techniques that have been introduced about two decades ago [FL04, MTPS04]. These artificial force fields can be either computed by optimization [TMPS03, MTPS04, PM17b, TCACS21] or through heuristics [FL04, SY05b, TKPR06]. Optimization methods can accurately match specific objectives at a high computational cost, while heuristics provide an efficient solution that does not exactly satisfy target keyframes. Both approaches define objectives by computing differences between a simulated density field and a target one at a given frame. When simulated and objective density fields do not

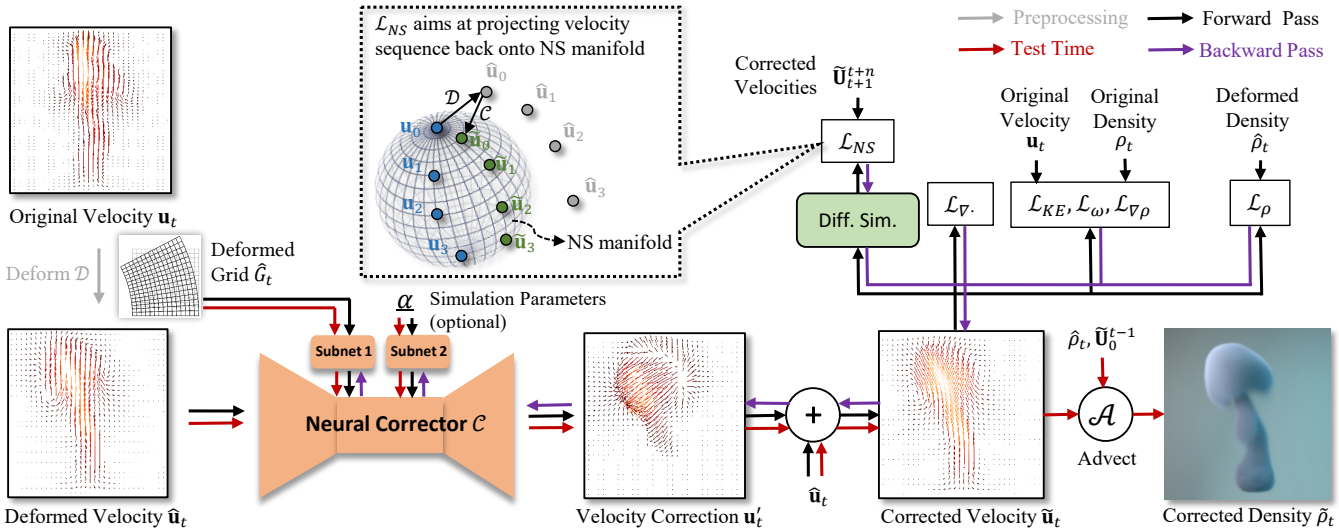

Figure 2: *Overview of our method. Our deformation-based control is a two-step procedure that includes an artist-friendly deformation $\mathcal{D}$, and a neural corrector $\mathcal{C}$ that projects the deformation back onto the Navier-Stokes (NS) manifold. Our neural corrector is implemented by convolutional neural networks and hence enables interactive operations at inference time. The neural corrector is coupled with a differentiable fluid solver and a set of loss functions which provides a strong physics prior in the self-supervised training.*

overlap — if the target smoke goes under extreme deformations — these methods cannot provide meaningful gradients for the optimization or heuristics computation, and artificially computed force fields will not be able to properly guide fluid simulations.

Alternatively, fluids can be controlled by direct manipulation of pre-simulated fluid data. For instance, the volumetric flow data can be deformed with the underlying grid [SDY*15, PM17a], various fluid scenes may be stitched [SDN18] or sculpted for resizing [FHM*21]. While these techniques offer an unprecedented level of post-editing functionality, they rely on computationally expensive optimizations or re-simulations.

To enable efficient artistic control that is less constrained by the type of the deformation, we propose a fluid post-processing pipeline, but with the key difference of enabling target matching in interactive environments while preserving physical constraints. In our pipeline, we simulate a base smoke configuration, deform it with accessible control tools for prototyping, and finally correct the motion to increase physical realism. This is enabled by a self-supervised neural corrector that projects the deformed simulations back onto the manifold of the Navier-Stokes equations with the help of differentiable physics simulation.

Our model is implemented by convolutional neural networks and coupled with a differentiable fluid solver for training *without any reference correction samples*. By considering various physical quantities explicitly, our method offers not only user-friendly control of fluids but also conservation of the original physical characteristics. Thanks to the high speed of feed forward models, there is no necessity of expensive simulations or optimizations except the computation of the baseline simulation for the following deformation. At inference time, our method is thus computationally more efficient than previous target-driven simulation methods and allows users to interactively art-direct fluids. Our contributions can be summarized as follows:

- A neural corrector that projects an artistically deformed velocity field back onto the physical manifold at interactive frame rates. (Sec. 3.1)
- An objective function implemented through a differentiable fluid solver for evaluating physics laws. (Sec. 3.2)
- A set of physics-based loss functions to train the neural corrector in a self-supervised way (Sec. 3.2, Sec. 3.3) and corresponding ablation studies. (Sec. 4)

2. Related Work

Online Control for Fluids. One set of fluid control methods aim at matching given coarse guidance in space or time during simulation. One category of this group is to compute control force fields by evaluating heuristics for guidance. These methods derive control forces based on the difference between current and desired density field [FL04], signed distance functions [SY05a, SDE05, YCZ11], geometric potentials [HK04, SY05b], control particles [REN*04, TKPR06, MM13] and low-resolution simulations [HMK11]. While these techniques are computationally efficient, they often require manual work for delicate control due to the sub-optimal results.

Another category is based on optimization to achieve higher quality matches. Analytic gradients are computed to optimize forces [TMPS03] and are enhanced by the adjoint method [MTPS04], ADMM framework [GITH14, PM17b] and primal-dual algorithm [IEGT17]. Nielsen et al. [NCZ*09, NC10, NDB11] formulated a variational problem to guide high-resolution simulations with low-resolution counterparts. Alternative representations of guidance are utilized such as meshes [RTWT12], vortex

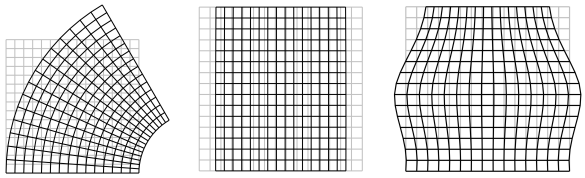

Figure 3: *Different types of deformations performed by manipulating the underlying regular grid. From left to right: bending, uniform scaling and non-uniform scaling.*

structure [WP10], spectral component [RLL*13, FN20, TCACS21] and stream function [SDK21].

Offline Control for Fluids. Unlike online control approaches, offline methods control fluids by post-processing with low computational cost and enhanced controllability at test time.

Reduced order models [TLP06, WST09] build basis functions from simulation snapshots by dimension reduction and span an approximation of the original full model for computational efficiency with a subspace integrator [KD13], Laplacian Eigenfunctions [DWLF12, CSK18] and learning-based models [KAT*19, WKA*20]. Those methods, however, do not provide additional controllability that goes beyond physical parameters.

Turbulence synthesis approaches, another group in offline fluid control, composite turbulent details dissipated during simulation using subgrid [KTJG08], solid boundary [PTSG09] and convolutional neural networks [CT17, XFCT18, UHT18].

On the other hand, Appearance transfer approaches [KEBK05, NKL*07] build on patch-based synthesis to change the appearance of a target object with a source texture by finding correspondences between local regions, and are enhanced to improve temporal coherency [BBRF14, JFA*15], extended to 3D velocity fields [SDKN18], and coupled with neural style transfer techniques [KAGS19, KAGS20]. The main challenge of these methods is to minimize the discontinuity between synthesized patches or frames over time and hence work at the cost of expensive optimization. Contrary to previous approaches, our neural corrector targets *physically-corrupted* fluid deformations rather than subtle feature re-injection or appearance matching on coarse simulations.

Other works in this direction aim at direct fluid authoring. Fluid simulations have been reconstructed from sparse 2D inputs [EHT18, EUT19, FST21, QLWQ21, KHW*22], interpolated [RWTT14, Thu16] and resized [FEHM19, FHM*21]. Pan and Manocha [PM17a] proposed a method for editing smoke simulations, where the underlying grid is deformed and applied in the fluid solver by altering the semi-Lagrangian advection operator. However, the deformed advection operator typically changes the appearance of original fluid details notably, and hence the deformation results of the density fields can also fail to match user-input controls. Sato et al. [SDY*15] presented a divergence-aware method for deforming velocity fields, but it only preserves incompressibility and requires the expensive solution of the Poisson equation to obtain vector potentials. A follow-up work [SDN18] also requires an energy minimization for stitching velocity fields smoothly. In contrast, our method is inherently physics-informed by the coupling with a differentiable fluid solver, and thus can learn a variety of physical characteristics including incompressibility, kinetic energy and vorticity in a self-supervised way and reflect them into edited flows while still matching user control inputs in interactive environments.

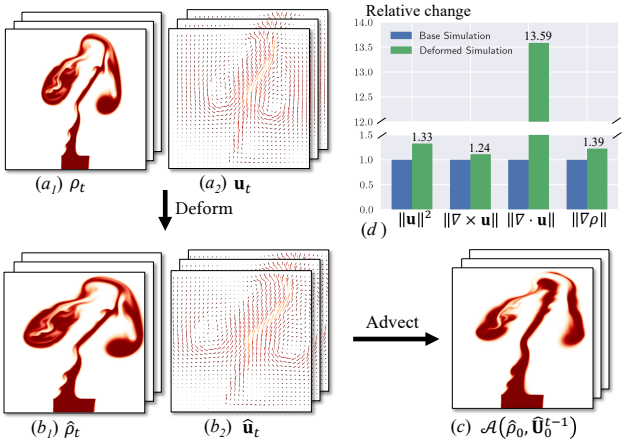

Figure 4: *Direct deformation of a density field (b_1), or of a velocity field (b_2) with subsequent density advection (c), lead to a severe visual degeneration and cause a breakdown of physical quantities (d). The histogram shows deviations of each physical quantities such as kinetic energy, vorticity, divergence and density gradient considered in Sec. 3.2.*

Differentiable Physics. Differentiable simulation frameworks [HKT20, HLS*19, HAL*19] allow us to incorporate gradient-based methods with the help of automatic differentiation, and hence neural networks can be augmented for solving inverse problems more robustly. Our work shares the concept of incorporating differentiable features with [UBF*20], so-called "Solver-in-the-loop", to naturally learn physics-informed corrections via augmented neural networks. We would like to highlight that, while previous work focused on reducing numerical errors compared to given high-resolution *reference samples*, our method focuses on learning the *reference manifold* directly with no samples for the rectification process of deformed fluid flows. Lastly, physics informed neural networks [RPK19] employ automatic differentiation to compute partial gradients in PDEs for training neural networks to solve fluid simulations in a self-supervised way.

3. Physics-Informed Neural Corrector

In this work, we focus on incompressible fluids simulated by solving the Navier–Stokes equations

$$\frac{\partial \mathbf{u}}{\partial t} + (\mathbf{u}\cdot\nabla)\mathbf{u} = -\frac{1}{\rho}\nabla p + \mu\nabla^2\mathbf{u} + \mathbf{f}, \tag{1}$$

$$\nabla\cdot\mathbf{u} = 0, \tag{2}$$

where $\mathbf{u}$, p and $\mathbf{f}$ denote the fluid velocity, pressure and external forces, respectively. External force can be represented through a force function $\mathbf{f} = \mathbf{f}(\rho, \mathbf{u})$ that depends on the marker density

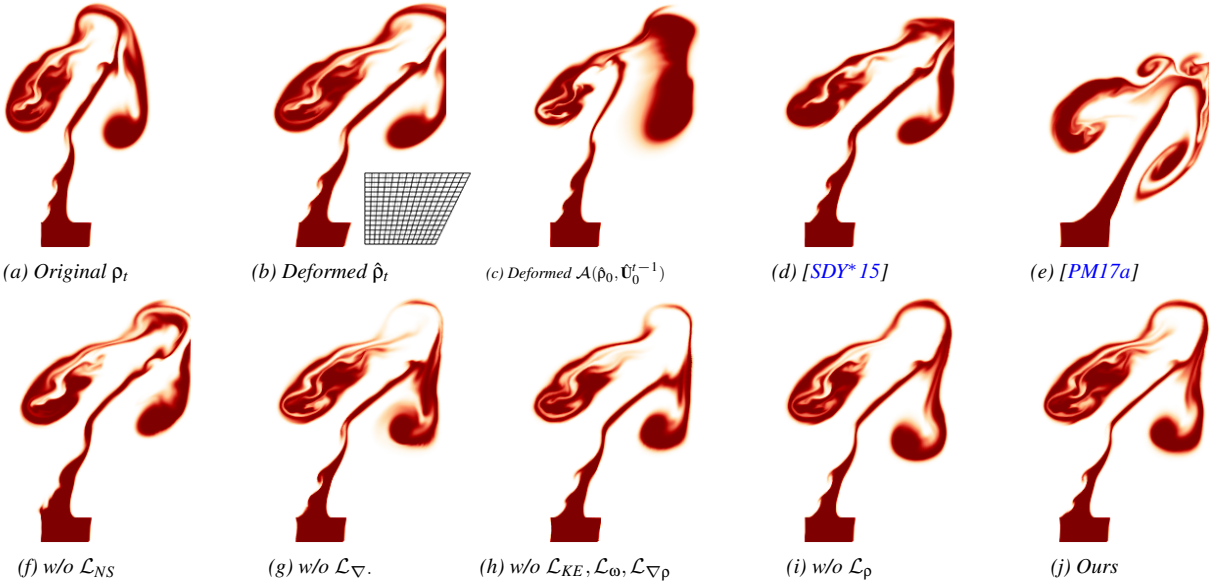

Figure 5: ***Ablation study*** *on loss function terms, computed on simulations of resolution* 256×256 *(a). (b) and (c) show the naive approaches using a deformed density and velocity, respectively. (d) and (e) show state-of-the-art methods that can be directly compared to ours (j). (f)-(i) show our model trained with different subsets of loss terms, which results in various artifacts in overall shape and details, as well as non-physical behaviors.*

and velocity fields. Viscosity terms are omitted due to the inherent dissipation of velocity-pressure fractional stepping methods [ETK*07, MCP*09]. We further denote a fluid solver for multiple time integrations of n from the time frame t as

$$\mathbf{u}_{t+n} = \mathcal{S}^n(\rho_t, \mathbf{u}_t) \tag{3}$$

with a step size of Δt. For simulating smoke, the density field ρ is passively advected $\rho_t = \mathcal{A}(\rho_0, \mathbf{U}_0^{t-1})$ where $\mathbf{U}_0^{t-1} = \{\mathbf{u}_0, ..., \mathbf{u}_{t-1}\}_{(t\geq 1)}$ is the set of velocity fields computed from the solver.

Our deformation-based control consists of two parts: deformation $\mathcal{D}$ and correction $\mathcal{C}$. The deformation process takes user-defined deformations over time as the input control and deforms the density and velocity fields of the original fluid simulation. The deformation is not restricted to any specific operations, it is often performed by manipulating the underlying regular grid G [SDY*15, PM17a, SDN18, FEHM19, FHM*21]. Some examples of deformations can be seen in Fig. 3.

At time step t, the regular grid G is deformed into another grid $\hat{G}_t$ through a displacement field $D_t : \hat{G}_t = G + D_t$.

To deform the simulation fields defined on regular grids σ_t (*e.g.*, $\rho_t, \mathbf{u}_t$), we bilinearly/trilinearly sample the original fields at the deformed grid $\hat{G}_t$ as the deformed fields $\hat{\sigma}_t$. This deformation process is denoted as $\hat{\sigma}_t = \mathcal{D}(\sigma_t, \hat{G}_t)$.

Directly deforming the density fields ρ_t can lead to visually unpleasant results since they can get overly stretched, compressed or deviated from original appearance (Fig. 4, b_1). Alternatively, one could first deform the velocity fields $\mathbf{u}_t$ over time and use them to passively advect density fields (Fig. 4, c). However, such deformation of simulation velocity fields usually cause a breakdown of different physical characteristics as seen in the histogram of Fig. 4, d. For instance, when scaling the original simulation in the x direction by 1.5 times, physical quantities (divergence, velocity magnitude, etc.) vary substantially from 1.2 to 13.6 times compared to their original values. Moreover, naively deforming the velocity field can cause the velocity sequence to deviate significantly from Navier–Stokes Equations, as no specific constraints are applied on user inputs. Both factors eventually lead to visual artifacts in the density fields advected by the deformed velocities.

To deal with the issues created by naively deforming fields, Flynn et al. [FEHM19, FHM*21] uses a seam-carving technique to edit fluid data. To best determine which seam to add/remove, physical characteristics of the fluid such as kinetic energy $\mathbf{u}^2$, vorticity magnitude $|\nabla \times \mathbf{u}|$ and the gradient magnitude of density $|\nabla \rho|$ are considered and formulated into an energy function. Similarly, Sato et al. [SDN18] proposes an interpolation function for "copying and pasting" of fluids that is formulated by the spatial smoothness of the Dirichlet energy $|\nabla \mathbf{u}|^2$, which aims to preserve the divergence $\nabla \cdot \mathbf{u}$ and vorticity magnitude $|\nabla \times \mathbf{u}|$ of the original simulation.

Instead of directly constraining the deformer $\mathcal{D}$ with energy minimization, we use a corrector $\mathcal{C}$ to correct the unphysical motions from the deformation. Formulated as a neural network, the corrector can achieve faster inference at test time than energy minimization. When designing the objectives, we first take the Navier-Stokes equations (Eqns. (1) and (2)) into consideration through a differentiable fluid solver at training time. Thus, applying our corrector leads to deformed simulations that better satisfy temporal advection of quantities and the divergence conditions of the underlying fluid motion.

Moreover, we can also employ previous energy functions [FEHM19,FHM*21,SDN18] for better detail preservation. The design choices of the proposed physically inspired corrector will be detailed in Sec. 3.1.

3.1. Neural Corrector

The corrector $\mathcal{C}$ takes a deformed velocity field $\hat{\mathbf{u}}_t$ and the deformation grid $\hat{G}_t$ as input and outputs the velocity correction $\mathbf{u}'_t$. To differentiate between simulations with distinct properties, the corrector can optionally be conditioned on simulation parameters α (*e.g.*, buoyancy, vorticity confinement scale). The velocity correction term is added to the deformed velocity as the corrected velocity:

$$\tilde{\mathbf{u}}_t = \hat{\mathbf{u}}_t + \mathbf{u}'_t, \quad \mathbf{u}'_t = \mathcal{C}(\hat{\mathbf{u}}_t, \hat{G}_t, \underline{\alpha}). \tag{4}$$

Here the underline indicates the optional input. The correction is performed for all velocity fields in a deformed sequence. The final corrected density fields $\tilde{\rho}_t$ are obtained by t-recursive advections on the first deformed density frame with corrected velocity fields as

$$\tilde{\rho}_t = \mathcal{A}(\hat{\rho}_0, \tilde{\mathbf{U}}_0^{t-1}). \tag{5}$$

We implement our fluid corrector as a convolutional encoder-decoder similar to Chu et al. [CTS*21]. The encoder part downsamples the input field 4x (2D) or 16x (3D) with residual blocks [HZRS16] and strided convolutions. The deformation grid $\hat{G}_t$ is restricted to a lower spatial resolution compared to the original velocities, since reducing the dimensionality of the deformations helps with generalization. This deformation grid is added as an input at the bottleneck of the network after a going through a convolutional layer (Subnet 1). The optional simulation parameter $\underline{\alpha}$, when available, goes through another set of convolution layers (Subnet 2) before being concatenated with the feature map in the bottleneck. Conditioning the architecture to varying simulation parameters allows the artist to have additional control over the final simulation appearance. The decoder part upsamples the bottleneck features back to the original resolution with transposed convolutions. The whole pipeline of our neural corrector can be seen in Fig. 2. We detail the architecture of the neural network in Table (2).

3.2. Physics-informed Loss Functions

As user-input deformations can be arbitrary, it is very difficult to obtain reference corrected velocity fields. We thus define all our loss functions for training the neural corrector in a self-supervised fashion.

Navier–Stokes (NS) Loss. The core of our objective functions is implemented through a differentiable fluid solver for evaluating the network corrections. Under the arbitrary user-input deformations, the original fluid simulations usually go off course to the outside of the NS manifold (Fig. 2). The NS loss aims at projecting the velocity fields back onto the manifold. When advancing the corrected velocity field $\tilde{\mathbf{u}}_t$ by k solver steps, the resulted velocity field $\mathcal{S}^k(\hat{\rho}_t, \tilde{\mathbf{u}}_t)$ should be similar to the corrected velocity field at frame $t+k$:

$$\mathcal{L}_{NS}\left(\tilde{\mathbf{U}}_t^{t+n}\right) = \sum_{k=1}^{n} \left\| \mathcal{S}^k\left(\hat{\rho}_t, \tilde{\mathbf{u}}_t\right) - \tilde{\mathbf{u}}_{t+k} \right\|_2^2. \tag{6}$$

Here $\hat{\rho}_t$ denotes the deformed density. By minimizing the NS loss, we expect the corrected velocities in the time sequence to not only connect themselves through self-advection and external force integration, but also satisfy boundary conditions through the pressure projection. Note that no ground-truth corrected velocity field sequence is required in the NS loss. Only a set of snapshots of deformed velocity fields is present in training. More solver steps will result in a better evaluation of the NS loss, but makes the loss computations more expensive. We found in our experiments that $n = 8$ for 2D and $n = 4$ for 3D are good trade-offs between quality and complexity. We use such settings in all our experiments.

In addition, we emphasize the incompressibility constraint in Eqn. (2) by minimizing the divergence of the corrected velocity fields for n rollout steps:

$$\mathcal{L}_{\nabla\cdot}\left(\tilde{\mathbf{U}}_t^{t+n}\right) = \sum_{k=0}^{n} \|\nabla \cdot \tilde{\mathbf{u}}_{t+k}\|_2^2. \tag{7}$$

Physics Characteristic Losses. While the NS loss implicitly takes the physical features of fluids into account, we additionally consider three extra quantities — kinetic energy, magnitude of vorticity and gradient magnitude of density — in our loss functions to better preserve the detailed characteristics of the original flows. We minimize the differences of the corrected physical fields compared to the original simulation fields:

$$\mathcal{L}_{KE}\left(\tilde{\mathbf{U}}_t^{t+n}\right) = \sum_{k=0}^{n} \left\| \tilde{\mathbf{u}}_{t+k}^2 - \mathcal{D}\left(\mathbf{u}_{t+k}^2, \hat{G}_{t+k}\right) \right\|_2, \tag{8}$$

$$\mathcal{L}_{\omega}\left(\tilde{\mathbf{U}}_t^{t+n}\right) = \sum_{k=0}^{n} \left\| |\tilde{\omega}_{t+k}| - \mathcal{D}\left(|\omega_{t+k}|, \hat{G}_{t+k}\right) \right\|_2^2, \tag{9}$$

$$\mathcal{L}_{\nabla\rho}\left(\tilde{\mathbf{U}}_t^{t+n}\right) = \sum_{k=0}^{n} \left\| \left|\nabla\tilde{\rho}_t^{t+k}\right| - \mathcal{D}\left(|\nabla\rho_{t+k}|, \hat{G}_{t+k}\right) \right\|_2^2. \tag{10}$$

Here $\omega_{t+k} = \nabla \times \mathbf{u}_{t+k}$ is the vorticity field. $\tilde{\rho}_t^{t+k} = \mathcal{A}(\hat{\rho}_t, \tilde{\mathbf{U}}_t^{t+k-1})$ is the corrected density, i.e. deformed density recursively advected by corrected velocity sequence. When comparing the corrected and original simulation fields, we deform each original field (*e.g.*, ω in Eqn. (9)) to the deformed space through $\mathcal{D}$ so that both fields can be compared at a same grid location. Note that deforming a physical quantity does not necessarily imply physical accuracy and these losses are used to preserve features of the original simulation. Alternatively, one could compute features of the velocity and density fields after their mapping to the deformed grid, but this would require evaluating derivatives on curvilinear grids [AO13]. Similar to the NS loss, we evaluate our loss functions over n-rollouts.

Density Guidance Loss. The above losses push the corrected velocity sequence into a physically-correct direction. However, only incurring them can potentially undo the user-input deformations. Therefore we additionally define a density guidance loss to instruct the corrector to follow the general look of the given deformed den-

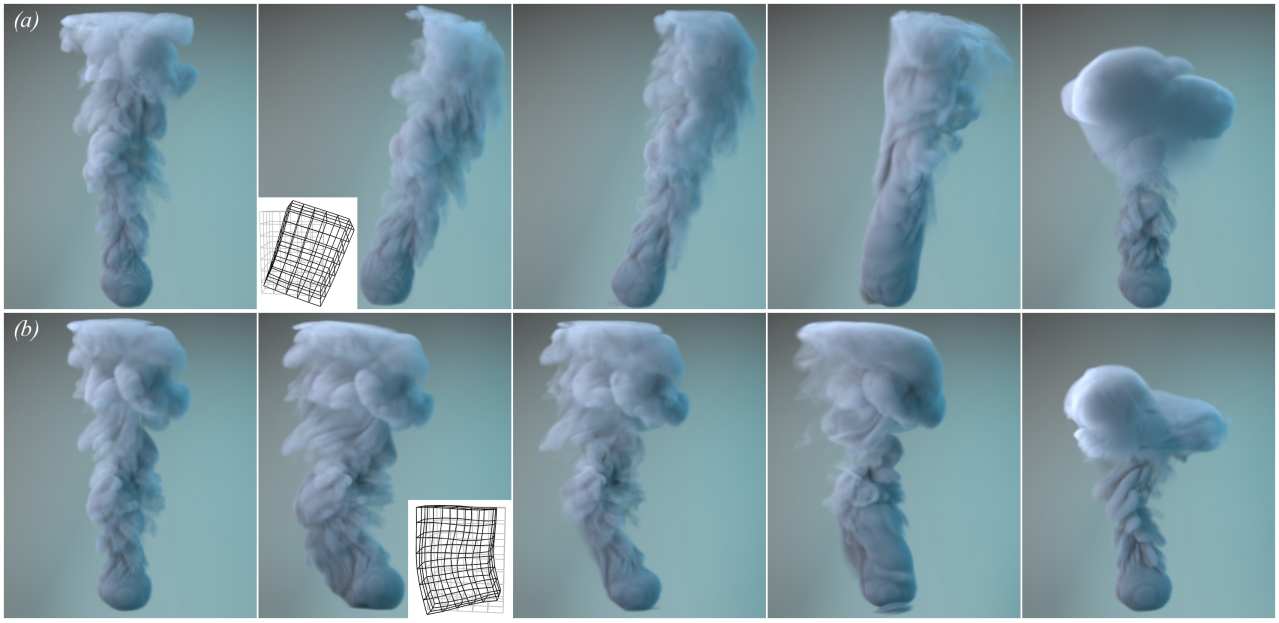

Figure 6: ***Bending*** *of rising plume simulations in resolution* $192 \times 256 \times 128$*. The deformation grids for bending are shown as inset images. Our model can correct the nonphysical motions resulted from the bending.*

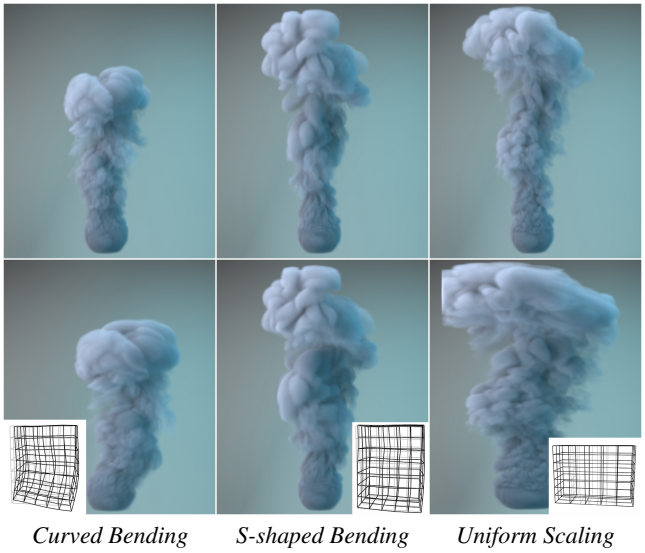

Figure 7: ***Examples from the 3D plume dataset.*** *The top row shows the original simulations of different simulation parameters at their last frames. The bottom row shows the deformed density field of the corresponding original simulation, with the deformation grid shown in the inset images.*

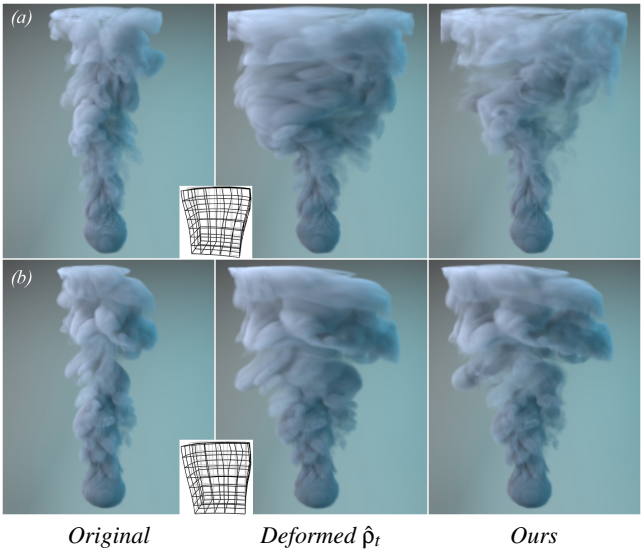

Figure 8: ***Non-uniform scaling*** *of rising plume simulations in* $192 \times 256 \times 128$*. The deformation grids are shown as inset images. Our model can create plausible corrections for spatially varying transformations that both stretch and twist the original simulation.*

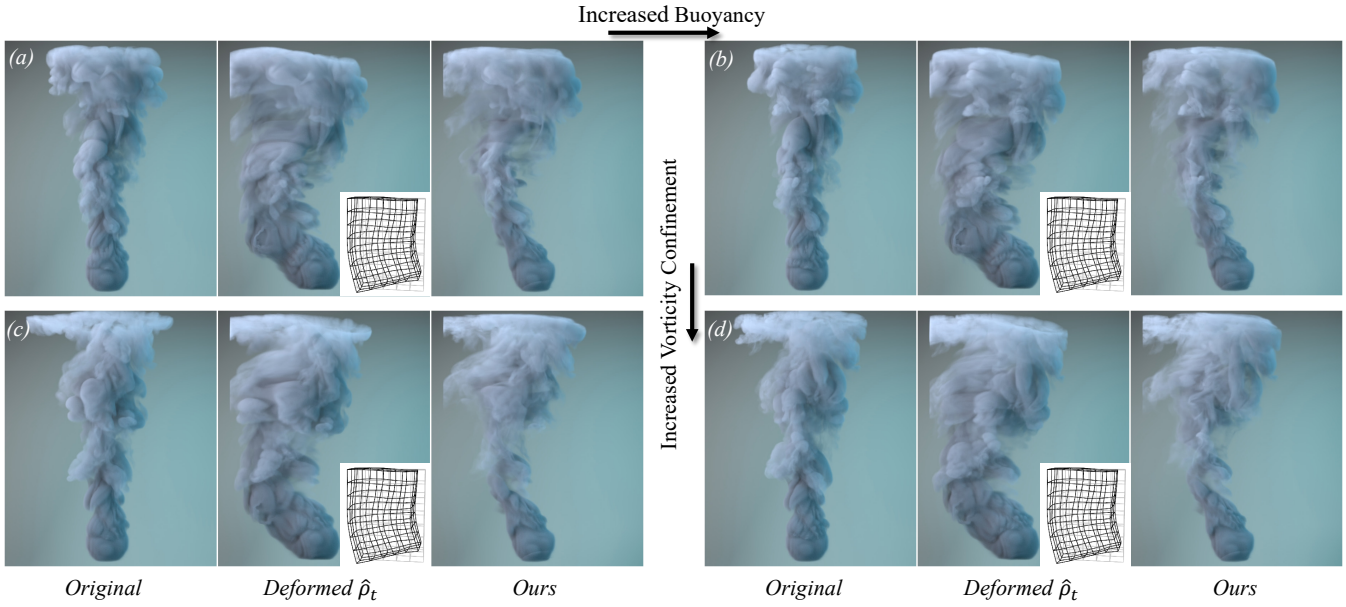

Figure 9: ***Different simulation sequences*** *in resolution* $288 \times 384 \times 192$ *are deformed by the same bending operation (inset image). From (a) to (b) and (c) to (d), the buoyancy of the original simulation increases. From (a) to (c) and (b) to (d), stronger vorticity confinement is used. The model trained on the lower resolution dataset can correct the deformation from all four simulation settings well.*

sity fields:

$$\mathcal{L}_\rho(\tilde{\mathbf{U}}_t^{t+n}) = \sum_{k=1}^{n} ||\tilde{\rho}_t^{t+k} - \hat{\rho}_{t+k}||, \tag{11}$$

Finally, the full objective for learning physics-induced rectifications on deformed fluid flows is defined as a weighted sum of above-mentioned loss functions:

$$\mathcal{L} = \mathcal{L}_{NS} + \lambda_{\nabla\cdot}\mathcal{L}_{\nabla\cdot} + \lambda_{KE}\mathcal{L}_{KE} + \lambda_\omega\mathcal{L}_\omega + \lambda_{\nabla\rho}\mathcal{L}_{\nabla\rho} + \lambda_\rho\mathcal{L}_\rho \tag{12}$$

where λ-s are the weights for each loss term. With all loss terms combined together, the training helps the model find a better trade-off where physical properties and user-input deformation signals are both satisfied to a better extent. For all our experiments, we use $\lambda_{\nabla\cdot} = 10$, $\lambda_{KE} = 0.1$, $\lambda_\omega = 1$, $\lambda_{\nabla\rho} = 10$ and $\lambda_\rho = 10$ resulting from the ablation study on the effect of each loss term in Fig. 5.

3.3. Datasets and Training Summary

Our *2D plume dataset* consists of buoyancy driven smoke simulations. The simulation is initialized by a rectangular density source on the bottom, an open boundary on the top, and closed boundaries on both sides. Vorticity confinement [SU94] is used to reintroduce small scale details lost due to advection. We vary the buoyancy forces from 1.5×10^{-4} to 3.0×10^{-4} and vorticity confinement scale from 0.10 to 0.19, generating 24 simulation sequences of 150 frames. Both buoyancy and vorticity confinement are used as inputs to the network during training. In addition, we generate 30 deformation sequences of bending and uniform scaling. The parameters used to generate those deformations are randomized, in order to create a diverse set of deformation patterns that will enhance the network ability to generalize to unseen deformations. During training, each iteration randomly samples clips from both the simulation and deformation sequences, and deforms the simulation fields on-the-fly, effectively traversing all combinations of simulation parameters and deformation sequences in the dataset. The *3D plume dataset* is likewise computed with a resolution of $192 \times 256 \times 128$ and examples of the 3D dataset can be seen in Fig. 7.

Besides plume simulations, we prepare a *smoky character dataset* to test our model on more complicated deformations in 3D. We first convert a static T-pose human body mesh from the AMASS dataset [MGFT*19] into a volume representing the smoke source and simulate for 100 steps in resolution of $208 \times 208 \times 112$. To conform the smoke motion to the source character animation, the density is dissipated with a rate of 0.9 in the region outside the character volume at each simulation step. Additionally, the velocity is initialized to follow the animated character surface and small perturbations are added to create turbulence patterns. Since only one simulation setting is used in this dataset, no simulation parameters are used as input to the neural network. Nine motion sequences of the same body mesh are extracted from AMASS dataset and used to deform the original smoke simulation. The deformation is done by tracking the differences of the corresponding mesh vertices using Point Deform Node in Houdini [hou]. One example of the dataset can be seen in Fig. 11.

4. Experiments and Results

In this section, we show our results on test dataset and explore several smoke editing applications. We compare with the results from [SDY*15] and [PM17a]. These works are the most similar to ours as they directly use deformation grids to edit smoke sim-

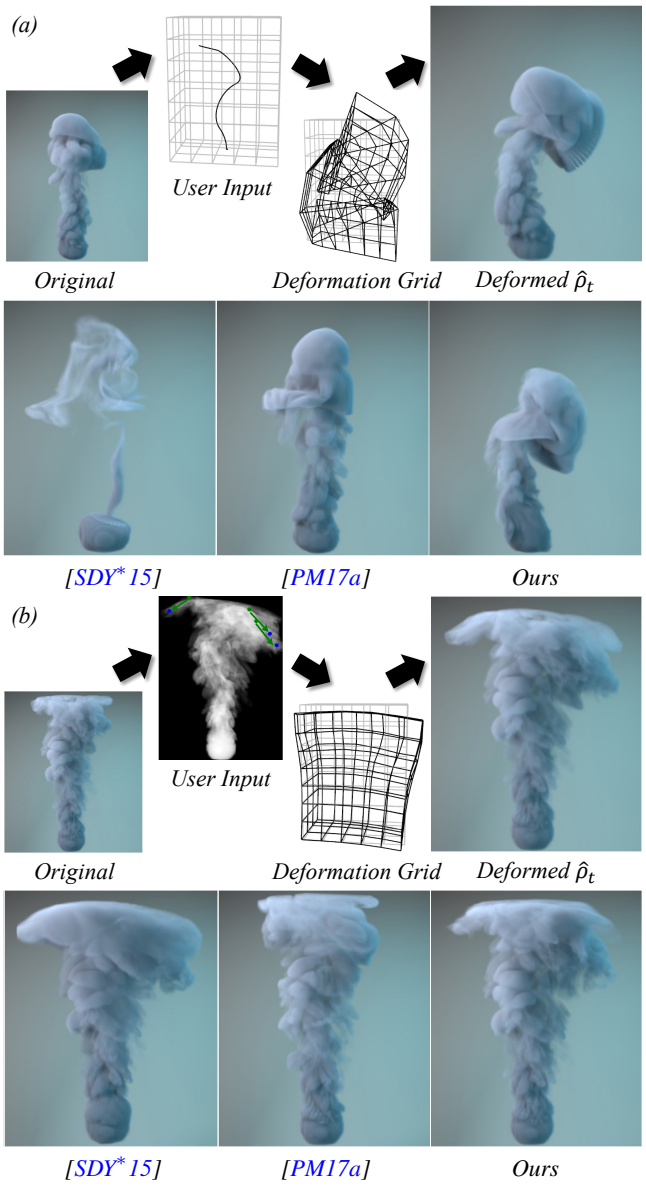

Figure 10: ***Editing*** *plume simulation in screen space. (a) The original simulation is deformed with a 3D curve through the Houdini Curve Deformation Node. (b) Point handles are dragged in space (green to blue). The corresponding deformation grid is generated by 3D Moving Least Square method [ZG07].*

ulations. The vector potential computation in [SDY*15] is implemented with a PCG solver of low tolerance (10^{-6}). The deformation transfer operator in [PM17a] is directly implemented in the advection operator with MacCormack method as suggested by the authors for reduced numerical dissipation, rather than formulated as additional force field. Our accompanying video shows animated sequences for the results presented in this section. The runtime of each method is shown in Table (1).

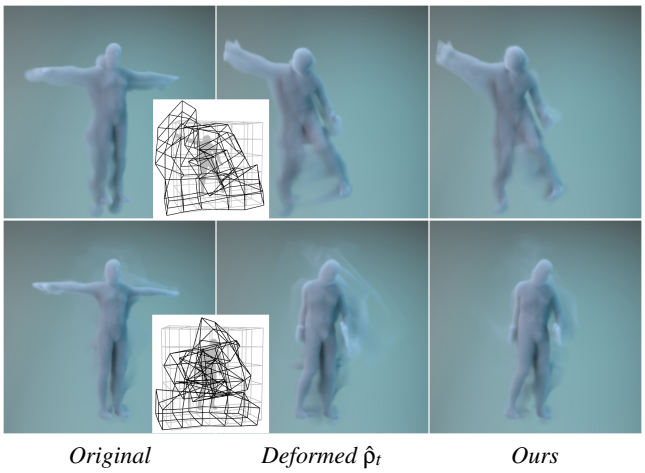

Figure 11: ***Smoky character motion*** *in* $208 \times 208 \times 112$ *from the training dataset. The original simulation is generated by using a character standing still in T-pose as smoke source. The smoke volume is then deformed according to the character motions.*

4.1. Implementation Details

We implement our fluid simulation solver and the neural networks in Python with PyTorch [PGM*19] for its embedded automatic differentiation tool as well as easy GPU deployment. We run all our 2D experiments on an NVIDIA GeForce RTX 2080Ti GPU with 11 GB of dedicated memory, and our 3D experiments on two NVIDIA GeForce RTX 3090 GPUs with 24 GB dedicated memory. The network is trained with an Adam Optimizer [KB15], with the learning rate fixed at 10^{-4}. The training takes 24-48 hours for 2D models and 72-96 hours for 3D models.

For the differentiable fluid simulator, we use the orthogonal and equidistant staggered MAC grid [HW65] to discretize the density, pressure and velocity fields. The MacCormack method is employed to solve the advection step with an 3^{rd}-order Runge-Kutta integrator. For the incompressibility constraint of the fluid solver in Eqn. (2), we implement pressure projection through a Preconditioned Conjugate Gradient (PCG) solver with the Incomplete Poisson Preconditioner [SKF11]. In particular, the backpropagation of the PCG solver is implemented by directly computing the adjoint through another linear solve [MTPS04], rather than a naive automatic differentiation, which can result in significant memory bottleneck. We use our simulator for both the data generation process and the Navier-Stokes loss in Eqn. (6).

4.2. 2D Test Cases for Ablation Studies

We first show 2D test results in Fig. 5 to illustrate the effect of each loss term and compare our method with previous works. Some of the sequences shown here are better represented on the accompanying video, since the Navier–Stokes loss can improve physical realism of transported quantities. The original density (a) is warped into its deformed configuration (b). We also show the result of just naively deforming velocities and advecting the original densities in (c). Simply warping density or velocity fields clearly creates visual artifacts. Our model trained with a loss function including all

Figure 12: ***Smoky character motion*** *in* $320 \times 320 \times 160$ *from the test set. The model trained on the lower resolution dataset is used for the correction.*

terms (j) can better preserve the shape and physical properties of the original smoke.

The method of Sato et al. [SDY*15] (d) can preserve the incompressibility of the velocity field, but deviates from the original appearance of the plume, while the method of Pan et al. [PM17a] (e) does not match the user-input deformed density in (b). When our model is trained without Navier-Stokes Loss (Eqn. (6)) (f), fluid behaviours got lost, and visual artifacts appear. Divergence Loss (Eqn. (7)) is necessary as a helper term as the Navier-Stokes Loss usually does not get minimized to zero, and it enforces the incompressibility of the corrected velocity (g). Leaving out the physical characteristic losses (Eqns. (8), (9) and 10), less turbulent details from the original simulation are reconstructed (h). When Density Guidance Loss (Eqn. (11)) is removed (i), the final appearance of the plume is further away from the user-input deformation (b).

4.3. 3D Applications

Plume Dataset. Fig. 6 shows the results on test examples similar to the training dataset. More extreme deformation examples are shown in Fig. 8. Our correction removes the unphysical motions such smoke sinks in the top-right corner (a), overly stretched densities (b), while also simultaneously matching the user-input deformation. The method of Sato et al. [SDY*15] can provide incompressible velocity sequences, but fails to match the small structures presented in the original simulation; lastly, the method of Pan et al. [PM17a] fails to match the deformation, and creates visual artifacts on the upper region of the smoke.

We additionally test the same model on higher resolution test examples of $288 \times 384 \times 192$. Fig. 9 shows our correction results on the same bending deformation as in Fig. 6(b). The deformation is used on simulations generated with different buoyancy and vorticity confinement coefficients that are not present in the training set. Our model can seamlessly handle all four simulation settings.

To test our model on fluid editing applications, we use the screen-space editing tools to generate deformation grids in Fig. 10. In (a), we use the Path Deform Node from Houdini to deform the simulation fields with a 3D curve. A more extreme deformation grid is generated from this method. In (b) Point handles are defined in 3D (green points), and dragged by the user to the target positions (blue points). A Moving Least Square method [SMW06, ZG07] is used to generate the deformation grid. Our method can correct the unphysical deformations in both cases while the methods of Sato et al. [SDY*15] and Pan et al. [PM17a] create visual artifacts or fail to match user inputs.

Smoky Character Dataset. To test our method on simulations other than plumes and more complicated deformations, we train the model on the smoky character dataset. We show model corrections on the training example in Fig. 11 and on higher resolution ($320 \times 320 \times 160$) test examples in Fig. 12. Our model can remove the clear visual artifacts from extreme deformations.

5. Conclusion and Discussion

We have presented a method for physics-informed correction on deformed fluid data. The recovery of deteriorated physical characteristics during deformation is promptly performed by learned neural networks with physics-inspired losses implemented with a differentiable simulation framework.

One limitation of our method is that once trained, it can only generalize to similar simulation setups. For example, the model trained on the *3D plume dataset* cannot be used to predict corrections on the *smoky character dataset*. Subdividing the input fields into smaller patches during training can potentially help improve generalization, as the required receptive fields effectively decrease. Additionally, we empirically fix the magnitude of each loss term through a hyper-parameter search. The same set of loss weights might not be optimal for other datasets, but can serve as a good starting point for the tuning.

Several other aspects can be improved in the future work: we only tested our method on smoke simulations, but liquid simulations could be similarly corrected as our method it is not restricted to smoke by design. There are other types of corrections that we have not tested, for instance, detail enhancement as presented in [UBF*20] or flow interpolations as shown in [SDN18]. Other than our current version of linearly performing the correction on

Table 1: ***Performance comparison*** *between ours and previous work. All deformations and corrections are run on a NVIDIA GeForce RTX 2080Ti GPU. The runtime of our method is evaluated per frame and excludes the training time of the networks. The deformations in Fig. 11 and Fig. 12 are performed in Houdini on CPU beforehand, thus the runtime is not reported.*

Scene	Resolution	Deformation [s]	Ours [s]	[SDY*15] [s]	[PM17a] [s]
2D Plume (Fig. 5)	256×256	1.00×10^{-3}	1.10×10^{-2}	0.38	0.32
3D Plume (Fig. 6, Fig. 10)	$192 \times 256 \times 128$	4.00×10^{-3}	0.22	11.82	5.14
3D Plume (Fig. 9)	$288 \times 384 \times 192$	3.20×10^{-2}	1.41	63.09	24.17
Smoky Character (Fig. 11)	$208 \times 208 \times 112$	–	0.12	9.04	5.35
Smoky Character (Fig. 12)	$320 \times 320 \times 160$	–	0.44	44.69	19.48

Table 2: ***Neural network architecture****. Conv stands for convolution, IN stands for instance normalization, ConvT stands for transposed convolution. We take a* 256×256 *input as example to state the sizes.*

Layer	Output Size	Operations	Kernel, Channels	Input
Subnet 1 (Deformation Grid)				
input_g	64×64	–	–, 2	$\hat{G}$
conv_g	64×64	Conv	$3 \times 3, 32$	input_g
Subnet 2 (Optional Simulation Parameters)				
input_σ	64×64	–	–, 2	σ
conv_σ1	1×1	Conv	$1 \times 1, 8192$	input_σ
up_σ1	64×64	Reshape-Upsample	–, 16	conv_σ1
conv_σ2	64×64	Conv	$7 \times 7, 16$	up_σ1
Neural Corrector				
input	256×256	–	–, 2	$\hat{\mathbf{u}}$
conv0	256×256	Conv–IN–Relu	$7 \times 7, 32$	input
conv1	128×128	Conv–IN–Relu	$3 \times 3, 64$	conv0
conv2	64×64	Conv–IN–Relu	$3 \times 3, 128$	conv1
res3 – res5	64×64	Conv–IN–Relu	$\begin{bmatrix} 3 \times 3, 128 \\ 3 \times 3, 128 \end{bmatrix} \times 3$	[conv2, conv_g]
res6 – res8	64×64	Conv–IN–Relu	$\begin{bmatrix} 3 \times 3, 128(+16) \\ 3 \times 3, 128(+16) \end{bmatrix} \times 3$	[res5, (conv_σ2)]
conv9	128×128	ConvT–IN–Relu	$4 \times 4, 64$	res8
conv10	256×256	ConvT–IN–Relu	$4 \times 4, 32$	conv9
output	256×256	ConvT	$7 \times 7, 32$	conv10

the deformed velocity fields, we tested using advection operator for non-linear corrections but didn't get satisfying results. More non-linear corrections could be tested for better corrections in extreme deformations. In terms of architectural choices, there is also room for improvement by using coordinate-based networks [SMB*20].

Appendix A: Neural Network Architecture

We detail our neural network architectures in Table 2. The table shows architectures in 2D, with a 256×256 input example to state the sizes. When using the network in 3D, we change the 2D layers to 3D counterparts. We also downsample the input $4\times$ with strided convolutions before *conv0* layer, and upsample the output $4\times$ with transposed convolution after *output* layer.

References

[AO13] AZEVEDO V. C., OLIVEIRA M. M.: Efficient Smoke Simulation on Curvilinear Grids. *Computer Graphics Forum 32*, 7 (10 2013), 235–244. doi:10.1111/cgf.12231. 5

[BBRF14] BROWNING M., BARNES C., RITTER S., FINKELSTEIN A.: Stylized keyframe animation of fluid simulations. In *NPAR Symposium on Non-Photorealistic Animation and Rendering* (2014), vol. 2014-Janua, ACM, pp. 63–70. doi:10.1145/2630397.2630406. 3

[CSK18] CUI Q., SEN P., KIM T.: Scalable laplacian eigenfluids. *ACM Transactions on Graphics 37*, 4 (8 2018), 1–12. doi:10.1145/3197517.3201352. 3

[CT17] CHU M., THUEREY N.: Data-driven synthesis of smoke flows with CNN-based feature descriptors. *ACM Transactions on Graphics 36*, 4 (7 2017), 1–14. URL: https://dl.acm.org/doi/10.1145/3072959.3073643, doi:10.1145/3072959.3073643. 3

[CTS*21] CHU M., THUEREY N., SEIDEL H. P., THEOBALT C., ZAYER R.: Learning meaningful controls for fluids. *ACM Transactions on Graphics 40*, 4 (2021). doi:10.1145/3450626.3459845. 5

[DWLF12] DE WITT T., LESSIG C., FIUME E.: Fluid simulation using Laplacian eigenfunctions. *ACM Transactions on Graphics 31*, 1 (2012), 1–11. URL: http://dl.acm.org/citation.cfm?doid=2077341.2077351, doi:10.1145/2077341.2077351. 3

[EHT18] ECKERT M.-L., HEIDRICH W., THUEREY N.: Coupled Fluid Density and Motion from Single Views. *Computer Graphics Forum 37*, 8 (12 2018), 47–58. URL: https://onlinelibrary.

wiley.com/doi/10.1111/cgf.13511, doi:10.1111/cgf.13511. 3

[ETK*07] ELCOTT S., TONG Y., KANSO E., SCHRÖDER P., DESBRUN M.: Stable, circulation-preserving, simplicial fluids. *ACM Transactions on Graphics 26*, 1 (1 2007), 4. doi:10.1145/1189762.1189766. 4

[EUT19] ECKERT M.-L., UM K., THUEREY N.: ScalarFlow: A large-scale volumetric data set of real-world scalar transport flows for computer animation and machine learning. *ACM Transactions on Graphics 38*, 6 (2019). doi:10.1145/3355089.3356545. 3

[FEHM19] FLYNN S., EGBERT P., HOLLADAY S., MORSE B.: Fluid carving: intelligent resizing for fluid simulation data. *ACM Transactions on Graphics 38*, 6 (12 2019), 1–14. doi:10.1145/3355089.3356572. 3, 4, 5

[FHM*21] FLYNN S., HART D., MORSE B., HOLLADAY S., EGBERT P.: Generalized fluid carving with fast lattice-guided seam computation. *ACM Transactions on Graphics 40*, 6 (12 2021), 1–15. doi:10.1145/3478513.3480544. 2, 3, 4, 5

[FL04] FATTAL R., LISCHINSKI D.: Target-driven smoke animation. In *ACM SIGGRAPH 2004 Papers, SIGGRAPH 2004* (New York, New York, USA, 2004), ACM Press, pp. 441–448. URL: http://portal.acm.org/citation.cfm?doid=1186562.1015743, doi:10.1145/1186562.1015743. 1, 2

[FN20] FOROOTANINIA Z., NARAIN R.: Frequency-domain smoke guiding. *ACM Transactions on Graphics 39*, 6 (12 2020), 1–10. doi:10.1145/3414685.3417842. 3

[FST21] FRANZ E., SOLENTHALER B., THUEREY N.: Global Transport for Fluid Reconstruction with Learned Self-Supervision. 3

[GITH14] GREGSON J., IHRKE I., THUEREY N., HEIDRICH W.: From capture to simulation - Connecting forward and inverse problems in fluids. *ACM Transactions on Graphics 33*, 4 (2014), 1–11. doi:10.1145/2601097.2601147. 2

[HAL*19] HU Y., ANDERSON L., LI T.-M., SUN Q., CARR N., RAGAN-KELLEY J., DURAND F.: DiffTaichi: Differentiable Programming for Physical Simulation. In *ICLR* (2019). URL: http://arxiv.org/abs/1910.00935. 3

[HK04] HONG J.-M., KIM C.-H.: Controlling fluid animation with geometric potential. *Computer Animation and Virtual Worlds 15*, 34 (7 2004), 147–157. doi:10.1002/cav.17. 2

[HKT20] HOLL P., KOLTUN V., THUEREY N.: Learning to Control PDEs with Differentiable Physics. In *International Conference on Learning Representations* (2020). URL: http://arxiv.org/abs/2001.07457. 3

[HLS*19] HU Y., LIU J., SPIELBERG A., TENENBAUM J. B., FREEMAN W. T., WU J., RUS D., MATUSIK W.: ChainQueen: A real-time differentiable physical simulator for soft robotics. In *Proceedings - IEEE International Conference on Robotics and Automation* (2019), vol. 2019-May, pp. 6265–6271. doi:10.1109/ICRA.2019.8794333. 3

[HMK11] HUANG R., MELEK Z., KEYSER J.: Preview-based sampling for controlling gaseous simulations. In *Proceedings of the 2011 ACM SIGGRAPH/Eurographics Symposium on Computer Animation - SCA '11* (New York, New York, USA, 2011), ACM Press, p. 177. doi:10.1145/2019406.2019430. 2

[hou] Houdini. URL: https://www.sidefx.com/products/houdini/. 7

[HW65] HARLOW F. H., WELCH J. E.: Numerical calculation of time-dependent viscous incompressible flow of fluid with free surface. *Physics of Fluids 8*, 12 (1965), 2182–2189. URL: https://aip.scitation.org/doi/10.1063/1.1761178, doi:10.1063/1.1761178. 8

[HZRS16] HE K., ZHANG X., REN S., SUN J.: Deep residual learning for image recognition. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition 2016-Decem* (2016), 770–778. doi:10.1109/CVPR.2016.90. 5

[IEGT17] INGLIS T., ECKERT M.-L., GREGSON J., THUEREY N.: Primal-Dual Optimization for Fluids. *Computer Graphics Forum 36*, 8 (2017), 354–368. doi:10.1111/cgf.13084. 2

[JFA*15] JAMRIŠKA O., FIŠER J., ASENTE P., LU J., SHECHTMAN E., SÝKORA D.: LazyFluids: Appearance transfer for fluid animations. *ACM Transactions on Graphics 34*, 4 (2015), 92. doi:10.1145/2766983. 3

[KAGS19] KIM B., AZEVEDO V. C., GROSS M., SOLENTHALER B.: Transport-based neural style transfer for smoke simulations. *ACM Transactions on Graphics 38*, 6 (12 2019), 1–11. doi:10.1145/3355089.3356560. 3

[KAGS20] KIM B., AZEVEDO V. C., GROSS M., SOLENTHALER B.: Lagrangian neural style transfer for fluids. *ACM Transactions on Graphics 39*, 4 (8 2020). doi:10.1145/3386569.3392473. 3

[KAT*19] KIM B., AZEVEDO V. C., THUEREY N., KIM T., GROSS M., SOLENTHALER B.: Deep Fluids: A Generative Network for Parameterized Fluid Simulations. *Computer Graphics Forum 38*, 2 (5 2019), 59–70. doi:10.1111/cgf.13619. 3

[KB15] KINGMA D. P., BA J. L.: Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings* (2015), vol. 5. 8

[KD13] KIM T., DELANEY J.: Subspace fluid re-simulation. *ACM Transactions on Graphics 32*, 4 (2013), 1. URL: http://dl.acm.org/citation.cfm?doid=2461912.2461987, doi:10.1145/2461912.2461987. 3

[KEBK05] KWATRA V., ESSA I., BOBICK A., KWATRA N.: Texture optimization for example-based synthesis. In *ACM SIGGRAPH 2005 Papers on - SIGGRAPH '05* (New York, New York, USA, 2005), ACM Press, p. 795. URL: http://portal.acm.org/citation.cfm?doid=1186822.1073263, doi:10.1145/1186822.1073263. 3

[KHW*22] KIM B., HUANG X., WUELFROTH L., TANG J., CORDONNIER G., GROSS M., SOLENTHALER B.: Deep Reconstruction of 3D Smoke Densities from Artist Sketches. *Computer Graphics Forum (Procs. Eurographics 2022) 41*, 2 (5 2022). 3

[KTJG08] KIM T., THÜREY N., JAMES D., GROSS M.: Wavelet turbulence for fluid simulation. *ACM Transactions on Graphics 27*, 3 (8 2008), 1–6. doi:10.1145/1360612.1360649. 3

[MCP*09] MULLEN P., CRANE K., PAVLOV D., TONG Y., DESBRUN M.: Energy-preserving integrators for fluid animation. *ACM Transactions on Graphics 28*, 3 (7 2009), 1–8. doi:10.1145/1531326.1531344. 4

[MGFT*19] MAHMOOD N., GHORBANI N., F. TROJE N., PONS-MOLL G., BLACK M. J.: AMASS: Archive of Motion Capture as Surface Shapes. In *The IEEE International Conference on Computer Vision (ICCV)* (10 2019). URL: https://amass.is.tue.mpg.de. 7

[MM13] MADILL J., MOULER D.: Target particle control of smoke simulation. In *Proceedings - Graphics Interface* (2013), GI '13, Canadian Information Processing Society, pp. 125–132. 2

[MTPS04] MCNAMARA A., TREUILLE A., POPOVIĆ Z., STAM J.: Fluid control using the adjoint method. In *ACM SIGGRAPH 2004 Papers, SIGGRAPH 2004* (New York, New York, USA, 2004), ACM Press, pp. 449–456. URL: http://portal.acm.org/citation.cfm?doid=1186562.1015744, doi:10.1145/1186562.1015744. 1, 2, 8

[NC10] NIELSEN M. B., CHRISTENSEN B. B.: Improved variational guiding of smoke animations. *Computer Graphics Forum 29*, 2 (2010), 705–712. doi:10.1111/j.1467-8659.2009.01640.x. 2

[NCZ*09] NIELSEN M. B., CHRISTENSEN B. B., ZAFAR N. B., ROBLE D., MUSETH K.: Guiding of smoke animations through variational coupling of simulations at different resolutions. In *Computer Animation, Conference Proceedings* (New York, New York, USA, 2009), ACM Press, pp. 217–226. URL: http://portal.acm.org/citation.cfm?doid=1599470.1599499, doi:10.1145/1599470.1599499. 2

[NDB11] NIELSEN M. B., DIGITAL W., BRIDSON R.: Guide shapes for high resolution naturalistic liquid simulation. In *ACM Transactions on Graphics* (New York, New York, USA, 2011), vol. 30, ACM Press, p. 1. URL: http://portal.acm.org/citation.cfm?doid=1964921.1964978, doi:10.1145/1964921.1964978. 2

[NKL*07] NARAIN R., KWATRA V., LEE H.-P., KIM T., CARLSON M., LIN M. C.: Feature-Guided Dynamic Texture Synthesis on Continuous Flows. In *Proc. Eurographics Symposium on Rendering (EGSR)* (Aire-la-Ville, Switzerland, Switzerland, 2007), EGSR'07, Eurographics Association, pp. 361–370. URL: http://dx.doi.org/10.2312/EGWR/EGSR07/361-370, doi:10.2312/EGWR/EGSR07/361-370. 3

[PGM*19] PASZKE A., GROSS S., MASSA F., LERER A., BRADBURY J., CHANAN G., KILLEEN T., LIN Z., GIMELSHEIN N., ANTIGA L., DESMAISON A., KOPF A., YANG E., DEVITO Z., RAISON M., TEJANI A., CHILAMKURTHY S., STEINER B., FANG L., BAI J., CHINTALA S.: PyTorch: An Imperative Style, High-Performance Deep Learning Library. In *Advances in Neural Information Processing Systems 32*, Wallach H., Larochelle H., Beygelzimer A., d Alché-Buc F., Fox E., Garnett R., (Eds.). Curran Associates, Inc., 2019, pp. 8024–8035. URL: http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf. 8

[PM17a] PAN Z., MANOCHA D.: Editing smoke animation using a deforming grid. *Computational Visual Media 3*, 4 (12 2017), 369–378. URL: http://link.springer.com/10.1007/s41095-017-0096-2, doi:10.1007/s41095-017-0096-2. 2, 3, 4, 6, 7, 8, 9, 10

[PM17b] PAN Z., MANOCHA D.: Efficient solver for spacetime control of smoke. *ACM Transactions on Graphics 36*, 5 (2017). doi:10.1145/3016963. 1, 2

[PTSG09] PFAFF T., THUEREY N., SELLE A., GROSS M.: Synthetic turbulence using artificial boundary layers. *ACM Transactions on Graphics 28*, 5 (12 2009), 1–10. doi:10.1145/1618452.1618467. 3

[QLWQ21] QIU S., LI C., WANG C., QIN H.: A Rapid, End-to-end, Generative Model for Gaseous Phenomena from Limited Views. *Computer Graphics Forum 40*, 6 (9 2021), 242–257. doi:10.1111/cgf.14270. 3

[REN*04] RASMUSSEN N., ENRIGHT D., NGUYEN D., MARINO S., SUMNER N., GEIGER W., HOON S., FEDKIW R.: Directable photorealistic liquids. In *Computer Animation 2004 - ACM SIGGRAPH / Eurographics Symposium on Computer Animation* (New York, New York, USA, 2004), ACM Press, pp. 193–202. URL: http://portal.acm.org/citation.cfm?doid=1028523.1028549, doi:10.1145/1028523.1028549. 2

[RLL*13] REN B., LI C. F., LIN M. C., KIM T., HU S. M.: Flow field modulation. *IEEE Transactions on Visualization and Computer Graphics 19*, 10 (2013), 1708–1719. doi:10.1109/TVCG.2013.73. 3

[RPK19] RAISSI M., PERDIKARIS P., KARNIADAKIS G. E.: Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational Physics 378* (2019), 686–707. URL: https://doi.org/10.1016/j.jcp.2018.10.045, doi:10.1016/j.jcp.2018.10.045. 3

[RTWT12] RAVEENDRAN K., THUEREY N., WOJTAN C., TURK G.: Controlling liquids using meshes. In *Computer Animation 2012 - ACM SIGGRAPH / Eurographics Symposium Proceedings, SCA 2012* (Goslar Germany, Germany, 2012), SCA '12, Eurographics Association, pp. 255–264. URL: http://dl.acm.org/citation.cfm?id=2422356.2422393. 2

[RWTT14] RAVEENDRAN K., WOJTAN C., THUEREY N., TURK G.: Blending liquids. *ACM Transactions on Graphics 33*, 4 (7 2014), 1–10. doi:10.1145/2601097.2601126. 3

[SDE05] SCHPOK J., DWYER W., EBERT D. S.: Modeling and animating gases with simulation features. In *Proceedings of the 2005 ACM SIGGRAPH/Eurographics symposium on Computer animation - SCA '05* (New York, New York, USA, 2005), ACM Press, p. 97. doi:10.1145/1073368.1073381. 2

[SDK21] SATO S., DOBASHI Y., KIM T.: Stream-guided smoke simulations. *ACM Transactions on Graphics 40*, 4 (8 2021), 1–7. doi:10.1145/3450626.3459846. 3

[SDKN18] SATO S., DOBASHI Y., KIM T., NISHITA T.: Example-based turbulence style transfer. *ACM Transactions on Graphics 37*, 4 (2018), 84. doi:10.1145/3197517.3201398. 3

[SDN18] SATO S., DOBASHI Y., NISHITA T.: Editing Fluid Animation Using Flow Interpolation. *ACM Transactions on Graphics 37*, 5 (11 2018), 1–12. URL: https://dl.acm.org/doi/10.1145/3213771, doi:10.1145/3213771. 2, 3, 4, 5, 9

[SDY*15] SATO S., DOBASHI Y., YUE Y., IWASAKI K., NISHITA T.: Incompressibility-preserving deformation for fluid flows using vector potentials. *The Visual Computer 31*, 6-8 (6 2015), 959–965. doi:10.1007/s00371-015-1122-y. 2, 3, 4, 6, 7, 8, 9, 10

[SKF11] SIDERIS C., KAPADIA M., FALOUTSOS P.: Parallelized Incomplete Poisson Preconditioner in Cloth Simulation. 2011, pp. 389–399. doi:10.1007/978-3-642-25090-3{_}33. 8

[SMB*20] SITZMANN V., MARTEL J. N. P., BERGMAN A. W., LINDELL D. B., WETZSTEIN G.: Implicit Neural Representations with Periodic Activation Functions. In *NeurIPS* (2020), Larochelle H., Ranzato M., Hadsell R., Balcan M.-F., Lin H.-T., (Eds.). URL: https://proceedings.neurips.cc/paper/2020/hash/53c04118df112c13a8c34b38343b9c10-Abstract.html. 10

[SMW06] SCHAEFER S., MCPHAIL T., WARREN J.: Image deformation using moving least squares. In *ACM SIGGRAPH 2006 Papers on - SIGGRAPH '06* (New York, New York, USA, 2006), ACM Press, p. 533. doi:10.1145/1179352.1141920. 9

[SU94] STEINHOFF J., UNDERHILL D.: Modification of the Euler equations for "vorticity confinement": Application to the computation of interacting vortex rings. *Physics of Fluids 6*, 8 (8 1994), 2738–2744. doi:10.1063/1.868164. 7

[SY05a] SHI L., YU Y.: Controllable smoke animation with guiding objects. *ACM Transactions on Graphics 24*, 1 (1 2005), 140–164. doi:10.1145/1037957.1037965. 2

[SY05b] SHI L., YU Y.: Taming liquids for rapidly changing targets. In *Proceedings of the 2005 ACM SIGGRAPH/Eurographics symposium on Computer animation - SCA '05* (New York, New York, USA, 2005), ACM Press, p. 229. doi:10.1145/1073368.1073401. 1, 2

[TCACS21] TANG J., C. AZEVEDO V., CORDONNIER G., SOLENTHALER B.: Honey, I Shrunk the Domain: Frequency-aware Force Field Reduction for Efficient Fluids Optimization. *Computer Graphics Forum 40*, 2 (5 2021), 339–353. doi:10.1111/cgf.142637. 1, 3

[Thu16] THUEREY N.: Interpolations of Smoke and Liquid Simulations. *ACM Transactions on Graphics 36*, 1 (9 2016), 1–16. URL: http://dl.acm.org/citation.cfm?doid=2996392.2956233, doi:10.1145/2956233. 3

[TKPR06] THUREY N., KEISER R., PAULY M., RÜDE U.: Detail-Preserving Fluid Control. In *Proceedings of the 2006 ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (2006), SCA '06, Eurographics Association, p. 7–12. doi:10.5555/1218064.1218066. 1, 2

[TLP06] TREUILLE A., LEWIS A., POPOVIĆ Z.: Model reduction for real-time fluids. *ACM Transactions on Graphics 25*, 3 (2006), 826. URL: http://portal.acm.org/citation.cfm?doid=1141911.1141962, doi:10.1145/1141911.1141962. 3

[TMPS03] TREUILLE A., MCNAMARA A., POPOVIĆ Z., STAM J.: Keyframe control of smoke simulations. *ACM Transactions on Graphics 22*, 3 (7 2003), 716–723. URL: http://portal.acm.org/citation.cfm?doid=882262.882337, doi:10.1145/882262.882337. 1, 2

[UBF*20] UM K., BRAND R., FEI Y. R., HOLL P., THUEREY N.: Solver-in-the-Loop: Learning from Differentiable Physics to Interact with Iterative PDE-Solvers. In *Advances in Neural Information Processing Systems* (2020), Larochelle H., Ranzato M., Hadsell R., Balcan M. F., Lin H., (Eds.), vol. 33, Curran Associates, Inc., pp. 6111–6122. URL: https://proceedings.neurips.cc/paper/2020/file/43e4e6a6f341e00671e123714de019a8-Paper.pdf. 3, 9

[UHT18] UM K., HU X., THUEREY N.: Liquid Splash Modeling with Neural Networks. *Computer Graphics Forum 37*, 8 (12 2018), 171–182. URL: https://onlinelibrary.wiley.com/doi/10.1111/cgf.13522, doi:10.1111/cgf.13522. 3

[WKA*20] WIEWEL S., KIM B., AZEVEDO V. C., SOLENTHALER B., THUEREY N.: Latent Space Subdivision: Stable and Controllable Time Predictions for Fluid Flow. *Computer Graphics Forum 39*, 8 (12 2020), 15–25. doi:10.1111/cgf.14097. 3

[WP10] WEISSMANN S., PINKALL U.: Filament-based smoke with vortex shedding and variational reconnection. In *ACM SIGGRAPH 2010 papers on - SIGGRAPH '10* (New York, New York, USA, 2010), ACM Press, p. 1. doi:10.1145/1833349.1778852. 3

[WST09] WICKE M., STANTON M., TREUILLE A.: Modular bases for fluid dynamics. In *ACM SIGGRAPH 2009 papers on - SIGGRAPH '09* (New York, New York, USA, 2009), ACM Press, p. 1. URL: http://portal.acm.org/citation.cfm?doid=1576246.1531345, doi:10.1145/1576246.1531345. 3

[XFCT18] XIE Y., FRANZ E., CHU M., THUEREY N.: TempoGAN: A temporally coherent, volumetric GAN for super-resolution fluid flow. *ACM Transactions on Graphics 37*, 4 (8 2018), 1–15. URL: https://dl.acm.org/doi/10.1145/3197517.3201304, doi:10.1145/3197517.3201304. 3

[YCZ11] YUAN Z., CHEN F., ZHAO Y.: Pattern-guided smoke animation with lagrangian coherent structure. *ACM Transactions on Graphics 30*, 6 (12 2011), 1–8. doi:10.1145/2070781.2024170. 2

[ZG07] ZHU Y., GORTLER S. J.: *3D deformation using moving least squares*. Tech. rep., 2007. 8, 9