FLINT: Learning-based Flow Estimation and Temporal Interpolation for Scientific Ensemble Visualization

Hamid Gadirov, Jos B.T.M. Roerdink, and Steffen Frey

***Abstract*—We present FLINT (learning-based FLow estimation and temporal INTerpolation), a novel deep learning-based approach to estimate flow fields for 2D+time and 3D+time scientific ensemble data. FLINT can flexibly handle different types of scenarios with (1) a flow field being partially available for some members (e.g., omitted due to space constraints) or (2) no flow field being available at all (e.g., because it could not be acquired during an experiment). The design of our architecture allows to flexibly cater to both cases simply by adapting our modular loss functions, effectively treating the different scenarios as flow-supervised and flow-unsupervised problems, respectively (with respect to the presence or absence of ground-truth flow). To the best of our knowledge, FLINT is the first approach to perform flow estimation from scientific ensembles, generating a corresponding flow field for each discrete timestep, even in the absence of original flow information. Additionally, FLINT produces high-quality temporal interpolants between scalar fields. FLINT employs several neural blocks, each featuring several convolutional and deconvolutional layers. We demonstrate performance and accuracy for different usage scenarios with scientific ensembles from both simulations and experiments.**

***Index Terms*—Flow Estimation, Interpolation, Deep Learning, Spatiotemporal Data.**

I. Introduction

Technological advancements allow to capture and simulate time-dependent processes at high resolution, both spatially and temporally. This facilitates the acquisition of results from for a large number of runs, forming spatio-temporal ensembles. Such ensembles enable scientists to search a parameter space or estimate the impact of stochastic factors. However, the large volume of generated data often cannot entirely be preserved due to storage or bandwidth limitations [1], and, e.g., timesteps and/or variables need to be omitted. Data acquired via experiments are also typically restricted regarding available modalities (e.g., only camera images or scans are available). Reconstructing missing information post-hoc can greatly support visual analysis in such scenarios. Even if all variables of interest are captured and the full data can be preserved, assessing how well data can be reconstructed has been demonstrated to be useful for diverse purposes in visualization, including the choice of techniques in flow visualization [2], timestep selection [3], and ensemble member comparison [4].

In this paper, we propose FLINT (learning-based FLow estimation and temporal INTerpolation), a new deep learning-based approach to supplement scientific ensembles with an otherwise missing flow field—in scenarios when it had to be omitted or could not be captured, as is often the case for experiments. FLINT not only estimates a corresponding flow field for each timestep but can further reconstruct high-quality temporal interpolants between scalar fields.

Technically speaking, we consider two types of flow: *physical flow* and *optical flow*. In the general case of spatiotemporal *nD* scalar fields (like density of a fluid flow or luminance in 3D light microscopy image sequences) representing an underlying physical phenomenon, we define optical flow as the observed change of the *nD* scalar field patterns. By contrast, physical flow directly denotes the velocity of the actual material objects (moving particles in a fluid, motion of cells in a tissue volume, etc.). Physical flow is usually directly given as output of flow solvers, but can also be determined experimentally where feasible (e.g., via particle displacement velocimetry).

Optical flow can be estimated from arbitrary scalar fields, also ones that are only indirectly affected by physical flow (like a temperature field). Of course, like in the case of optical flow in computer vision we need to assume that 1) there exist observable temporal changes of the field under investigation; and 2) these changes are correlated with physical changes (in order to be meaningful). If no physical flow information is given at all—e.g., in the case of an experiment without dedicated flow measurements or when such information could not be stored due to space constraints—FLINT can estimate high-quality optical flow from scientific data as an additional modality for analysis (e.g., using flow glyphs or integral lines allows to effectively present temporal changes). If physical flow is at least partially given for some members of the ensemble (at the time of model training), FLINT is capable of making use of this and adds estimated physical flow to members during inference.

FLINT is versatile and can operate with any scalar field data, whether 2D or 3D. In cases where only scalar data is available, FLINT can estimate optical flow. When vector data are available for some members, it can estimate the missing physical flow. Even if the scalar field is correlated but not directly governed by the vector field, FLINT can learn the vector field, although the optical and physical flow may differ in this case. Fig. 1 shows an overview of the FLINT pipeline.

FLINT draws inspiration from recent state-of-the-art advancements in computer vision research [5]–[8]. FLINT especially builds upon RIFE (Real-time Intermediate Flow Estimation) [8]—a method for video frame interpolation based

Hamid Gadirov, University of Groningen. E-mail: h.gadirov@rug.nl.
Jos Roerdink, University of Groningen. E-mail: j.b.t.m.roerdink@rug.nl.
Steffen Frey, University of Groningen. E-mail: s.d.frey@rug.nl.

on optical flow —and extends it in several ways to enable the accurate learning of flow information, and yield high-quality temporal interpolants for scientific data.

FLINT substantially improves over RIFE for better temporal interpolation with scientific ensembles while being significantly faster, and enables the estimation of meaningful flow fields from spatiotemporal data, which would otherwise be missing, to open up new opportunities for analysis. Its flexible design allows FLINT to be applied to different scenarios without the need for architectural modifications — regardless of whether the flow field is (partially) available or not — simply by adapting the loss functions used in training the neural network. This makes FLINT suitable for both simulation and experimental ensembles, where the latter may typically lack available flow field data. We refer to scenarios with and without ground-truth flow as "flow-supervised" and "flow-unsupervised", respectively. FLINT performs flow field estimation and temporal interpolation in an integrated way.

A first application of flow estimation by FLINT is its facilitation of applying flow visualization techniques to datasets initially consisting solely of scalar data, which often occurs in practice for experimental data (e.g., [9], [10]). This broadens the scope of visualization possibilities and supports the presentation of dynamic processes. Another application arises in large-scale data analysis. For example, simulations on supercomputers produce massive amounts of data, only a small fraction of which can practically be stored [11]. This often necessitates omitting (i) certain variables (like the flow field) and/or (ii) timesteps, either uniformly or via adaptive selection [3], [12]. With FLINT we can reconstruct data reduced both by (i) and (ii), thus enabling comprehensive analysis and understanding of complex phenomena.

In summary, we propose FLINT, a new flexible method for the estimation of flow for 2D+time and 3D+time scientific ensembles. The FLINT code will be made available publicly. We consider our main contributions to be as follows:

- To the best of our knowledge, FLINT is the first approach to achieve high-quality flow estimation in scenarios (1) where flow information is partially available (*flow-supervised*, e.g., simulations where flow is omitted due to storage constraints) or (2) entirely unavailable (*flow-unsupervised*, e.g., experimental image sequences captured via cameras).
- FLINT produces high-quality temporal interpolants between scalar fields, such as density or luminance.
- FLINT can handle both 2D+time and 3D+time ensembles without requiring domain-specific assumptions, complex pre-training, or intermediate fine-tuning on simplified datasets.

II. Related Work

Flow field estimation in scientific visualization. Kappe *et al.* [13] reconstructed and visualized 3D local flow from 3D+time microscopy data using both image processing and optical flow methods. Kumpf *et al.* [14] utilized ensemble sensitivity analysis (ESA) to obtain the evolution of sensitivity features in 2D or 3D geo-spatial data. They used forward and backward optical flow-based feature assignment for tracing the evolution of sensitivities through time. Manandhar *et al.* [15] developed a 3D optical flow estimation method for

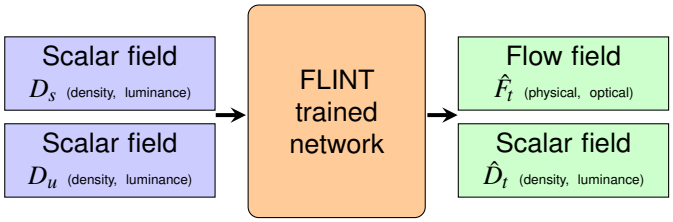


Fig. 1: Overview of FLINT pipeline during inference. The trained deep neural network performs flow field estimation $\hat{F}_t$ and temporal (scalar) field interpolation $\hat{D}_t$, where $s < t < u$, by utilizing the available densities D_s and D_u from the previous and following timesteps.

light microscopy image volumes. A dense 3D flow field was obtained from the 3D displacement vectors in a pair of such volumes. Gu *et al.* introduced Scalar2Vec [16]—a deep learning-based framework that translates scalar fields into velocity vector fields, using a k-complete bipartite translation network (kCBT-Net). We conceptually compare our findings to Scalar2Vec in Sec. VI. However, these approaches involve case-specific assumptions [13]–[15] or rely on ground truth (GT) [16] from synthetic data, limiting their generalizability to scientific ensembles derived from physical experiments. In contrast, FLINT does neither require data-specific assumptions nor an available GT flow field.

Optical flow learning in computer vision. In computer vision, optical flow is defined as the observed change of the pixel brightness pattern in two (or more) images of a moving object. The goal is to compute an approximation to the 2D motion field—a projection of the 3D velocities of surface points of the object onto the image plane—from spatiotemporal patterns of image intensity. Ideally, the optical flow is the same as the motion field, but this need not always be the case. Optical flow methods estimate physical velocity components by relying on constraints such as brightness constancy and imposing local or global spatial smoothness of the estimated velocity field, while aiming to preserve motion discontinuities and detect occlusions. Optical flow estimation has become a crucial component in computer vision applications, including tasks such as video frame interpolation. Recent deep learning-based methods such as RAFT and RAFT-3D [5], [17], FlowNet (2.0) [6], [18], PWC-Net [19], and ProFlow [20] achieved remarkable results, yet require complex multi-stage optimization (involving pre-training on simpler datasets and fine-tuning on target datasets), and/or need GT flow vectors for training. Estimating optical flow in scientific ensembles, characterized by unbounded structures and density-only representations, is difficult, and manual labeling is impractical. While effective on benchmark datasets, generalization to scientific ensembles poses a substantial challenge.

FLINT is based on RIFE (Real-Time Intermediate Flow Estimation for Video Frame Interpolation) [8] which estimates optical flow using a student-teacher architecture. However, RIFE targets 2D real-world RGB datasets, such as Vimeo90K [21], Middlebury [22], UCF101 [23], and HD [24], while FLINT can also handle 3D ensembles. For learning physical flow, we employ a loss function inspired by the RAFT

loss [5]. In scenarios where the flow field is unavailable during model training, we add additional loss components, such as photometric loss [25], to facilitate flow-unsupervised learning of the model.

Machine learning-based upscaling & super-resolution. ML-based upscaling and super-resolution approaches have gained significant attention across various domains, including image processing, computer vision, and scientific visualization [26], [27]. They can be categorized into three main groups: spatial super-resolution (SSR), temporal super-resolution (TSR), and spatio-temporal super-resolution (STSR). SRCNN [28], SRFBN [29], and SwinIR [30] are SSR techniques that aim to improve the spatial resolution of images by generating realistic and fine-grained details. TSR techniques focus on interpolating intermediate timesteps from subsampled time sequences while maintaining the spatial resolution. Methods such as phase-based interpolation [31], SepConv [32], and SloMo [33] tackle the challenge of synthesizing high-quality intermediate frames to enhance the temporal resolution of videos. While previous approaches, like STNet [34], can upscale data with a fixed spatial or temporal scale factor, they do not simultaneously address both spatial and temporal super-resolution. Han *et al.* proposed a recurrent generative network, TSR-TVD [35], designed to be trained on one variable and then applied to generate a temporally higher-resolution version of another variable from its lower-resolution counterpart. Recent methods like Filling the Void [36], SSR-TVD [37], FFEINR [38], HyperINR [39], CoordNet [40], and STSR-INR [41] have made significant strides in enabling TSR or SSR of data fields at arbitrary resolution. While FLINT achieves state-of-the-art accuracy in temporal interpolation between two given fields, its primary objective and notable strength lies in flow estimation. This task involves predicting flow fields directly from given scalar fields, assuming no flow information is available during the inference stage. Methods like CoordNet or STSR-INR, which are powerful in TSR or SSR tasks, are not designed for predicting vector fields from scalar ones. Our FLINT method not only excels in reconstructing temporal data with high precision but also produces impressive results in supplementing the data with accurate flow information.

Student-Teacher Learning. In machine learning, techniques have been developed under the name of teacher-student architecture. One approach is *knowledge distillation* [42] to transfer the knowledge (parameters) learnt by a larger model (*teacher model*) and transfer it to a smaller model (*student model*). A separate strand of research uses the concept of *privileged information* [43], where the teacher provides the student during training with additional "privileged" information for knowledge transfer. Both approaches have been unified [44]. The learning can be *offline*, where student networks learn the knowledge from pre-trained teacher networks, or *online*, where student and teacher networks are simultaneously trained, so that the whole knowledge learning process can be end-to-end trainable [45]. We have implemented an online student-teacher model and instead of a fully separate teacher network, we employ only one additional block that corresponds to the teacher network, akin to Huang et al. [8].

III. Method

FLINT is, to the best of our knowledge, the first method capable of performing flow estimation from available scalar fields in scientific ensembles, while simultaneously achieving temporal super-resolution. Leveraging recent deep learning advancements in optical flow estimation, FLINT can perform interpolation between scalar fields of different timesteps without strictly relying on GT flow vectors during training. This enhances its applicability to scientific ensembles, especially where GT flow data is unavailable.

FLINT implements a student-teacher architecture [45] which has several key advantages. It enables more accurate estimations as it is trained on GT temporal scalar fields (as demonstrated in Sec. V). Guidance through the teacher model further enhances the student model's learning process, resulting in stable training, faster convergence, and improved model robustness. FLINT does not require pre-training and intermediate fine-tuning on simplified datasets; it already converges on the target datasets, in contrast to previous comparable works in computer vision [5]–[7]. FLINT further yields significantly faster convergence and achieves an accuracy of more than 90% within the initial 30% of the total training time. Below, we describe the neural network architecture employed in FLINT (Sec. III-A), and present the temporal interpolation and flow estimation pipeline, both for the training and inference phases (Sec. III-B). Our loss function for training can flexibly adapt to different scenarios as described in Sec. III-C. FLINT builds upon RIFE [8] and modifies as well as extends it in several ways; a detailed comparison is presented in Sec. III-D.

A. FLINT Network Architecture

The network architecture is a feed-forward CNN, see Fig. 2. The student network consists of N stacked convolutional blocks (*Conv Block*), each incorporating convolutional (*Conv*) and deconvolutional (*Deconv*) layers (the middle column in the orange box of Fig. 2 shows an expanded view of a convolution block). There are 256 feature channels in all convolutional layers of the first block, 192 of the second and third, and 128 of the last block. The number of channels in the layers of the teacher block *Conv Block*^{teach} is set to 128, similarly to the last block of the student model. We utilize a PReLU activation function [46] in all layers except for the last one. The teacher model crucially features a dedicated Conv Block^{teach} which enables it to directly consider a GT scalar field by receiving D_t^{GT} through an additional channel. Note that traditionally such information is not available directly to the network architecture but is only considered in the loss function. The loss components that drive the training process are shown on the right side in Fig. 2. Student and teacher models are *jointly* optimized during training (i.e., following an online scheme [45]), with the teacher model refining the student model's results. According to our experiments, this streamlined one-stage online training approach significantly reduces training time compared to two-stage optimization, where the teacher model is trained first, and a student network is subsequently trained to align with the teacher's outputs. As demonstrated in Sec. VI-C, our one-stage approach does not yield any degradation in performance while significantly reducing training time. The proposed FLINT

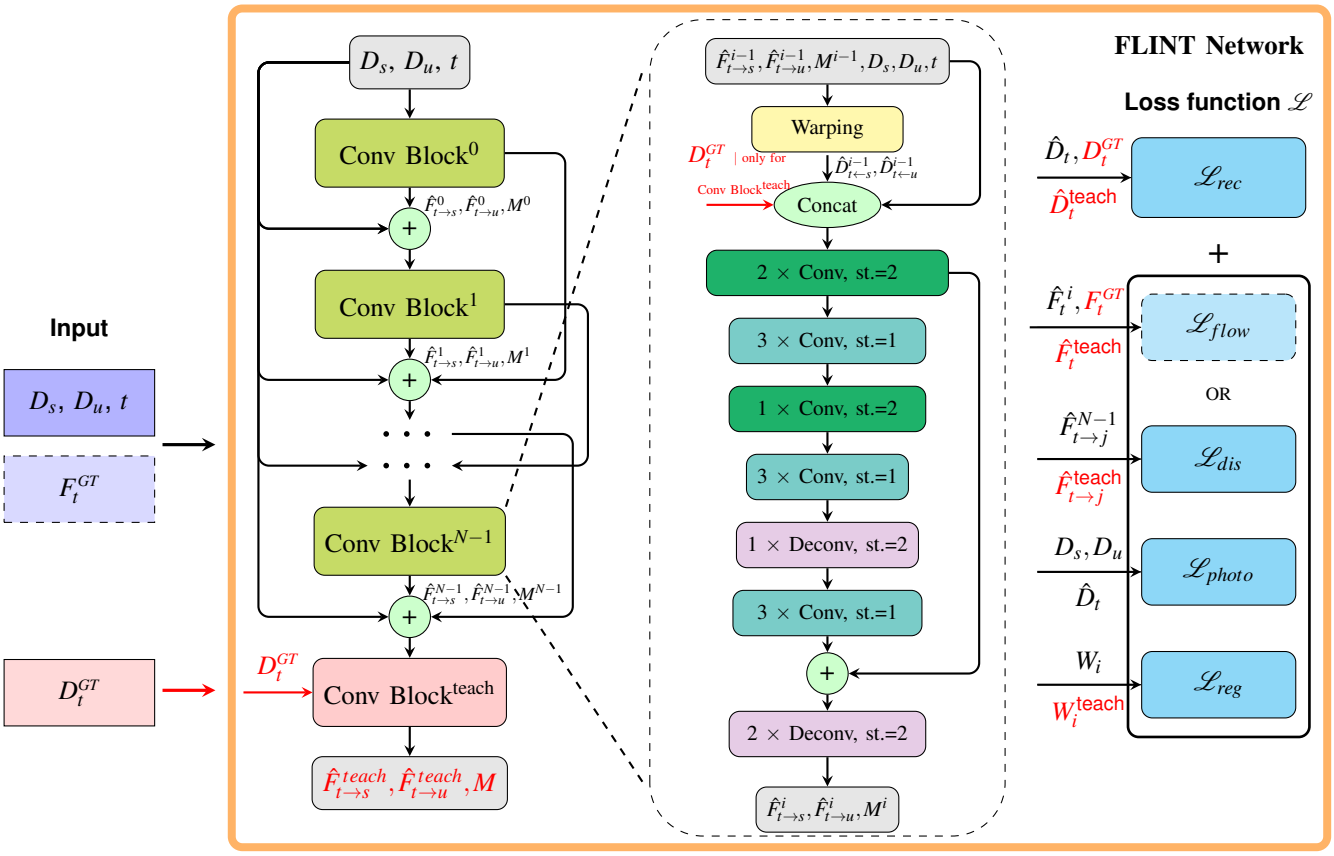


Fig. 2: FLINT network architecture and pipeline during training (see corresponding Fig. 1 for inference): given the input fields D_s, D_u, D_t^{GT} (GT scalar field at time t) and F_t^{GT} (GT flow at time t) in scenarios in which the latter is available, FLINT predicts the $\hat{D}_t$ scalar field and $\hat{F}_t^i$ flow fields used in the loss function for optimizing network parameters. The FLINT model architecture and loss function are shown in the orange box. The model consists of several stacked blocks of the convolutional network, which takes D_s, D_u, and t as input and in the i^{th} Conv Block computes estimated flows $\hat{F}^i_{t\to s}, \hat{F}^i_{t\to u}$, and fusion mask M^i used for interpolation. We obtain the best results with four ($N = 4$) blocks. The teacher block Conv Block^{teach}, which receives D_t^{GT} (the GT field at time t, see red arrows) as additional input, is only used during training. The zoomed-in view highlights the structure of a generic Conv Block consisting of backward warping, concatenation, as well as convolutional and deconvolutional layers with specified strides ($st.$). The D_t^{GT} input at the concatenation stage is only used for the teacher Conv Block. The loss function, which can be adjusted depending on the scenario, is shown on the right within the orange box. W_i and W_i^{teach} represent the i^{th} weight matrix of the last convolutional block in the student and teacher network, respectively. FLINT uses the same model architecture for ensembles with and without available GT flow fields. The GT flow F_t^{GT} is only used in the loss function $\mathscr{L}_{flow}$.

model architecture serves as the foundation for tasks with and without available GT flow fields, differing only in the applied loss functions (Sec. III-C).

B. Flow Estimation and Scalar Field Interpolation

As input, FLINT receives two scalar fields D_s and D_u of the same ensemble member at timesteps $s < u$ and an intermediate timestep t, where $s < t < u$. As output, FLINT (1) provides interpolants $\hat{D}_t$ at time $t \in [s,u]$ and (2) predicts the corresponding optical or physical flow field $\hat{F}_t$. First, intermediate flow fields $\hat{F}_{t\to s}$ and $\hat{F}_{t\to u}$ are computed. The *time-backward* flow $\hat{F}_{t\to s}$ refers to the intermediate flow field vectors from a frame at time t to an earlier frame at time s, while the *time-forward* flow $\hat{F}_{t\to u}$ is from the frame at time t to a later frame at time u. In the process, intermediate warped scalar fields are computed and fused by a fusion mask M, as follows.

Warping. In computer graphics warping means changing a source image into a target image. In *forward* warping a mapping is used to specify where each pixel from the source image ends up in the target image; however, holes may occur in the target image. This can be resolved by using *backward* warping. We utilize this technique in FLINT through a reverse mapping that finds, for each pixel p_t in the target image, the point p_s in the source image where it originated. Then resampling around this point p_s is applied by bilinear interpolation of source pixel values to determine the value of the target pixel p_t. The warping operator $\overleftarrow{W}$ denotes the combined effect of reverse mapping and bilinear interpolation. In our case the intermediate flow fields define the mappings, see Fig. 3. Here $\hat{D}_{t\leftarrow s}$ and $\hat{D}_{t\leftarrow u}$ are target images at time t with source images D_s and D_u, respectively. This results in two warped scalar

fields $\hat{D}_{t\leftarrow s}$ and $\hat{D}_{t\leftarrow u}$:

$$\hat{D}_{t\leftarrow s} = \overleftarrow{W}(D_s, \hat{F}_{t\to s}), \quad \hat{D}_{t\leftarrow u} = \overleftarrow{W}(D_u, \hat{F}_{t\to u}). \tag{1}$$

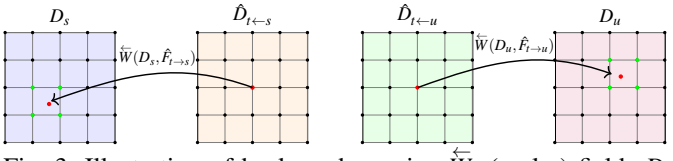


Fig. 3: Illustration of backward warping $\overleftarrow{W}$: (scalar) fields D_s and D_u are reversely mapped according to the flow fields $\hat{F}_{t\to s}$ and $\hat{F}_{t\to u}$. The fields $\hat{D}_{t\leftarrow s}$ and $\hat{D}_{t\leftarrow u}$ are then reconstructed using bilinear interpolation considering the values at the coordinates shown in green.

Fusion mask. The fusion mask M, where $M(i,j) \in [0,1], \forall i,j$, combines two intermediate warped scalar fields $\hat{D}_{t\leftarrow s}$ and $\hat{D}_{t\leftarrow u}$ at successive timesteps into an interpolated scalar field $\hat{D}_t$. The values in M are learned by FLINT to ensure a smooth transition between $\hat{D}_{t\leftarrow s}$ and $\hat{D}_{t\leftarrow u}$, minimizing artifacts and preserving the structural integrity of the original fields across space.

Refining the intermediate flows. The intermediate flow fields are computed via N convolutional blocks by iterative refinement, see Fig. 2. This coarse-to-fine process iteratively and jointly refines both the flow fields and the fusion mask at each Conv Block, ensuring progressively consistent and high-quality updates throughout the network. Superscripts i in Fig. 2 denote the various quantities at iteration i. Given two input fields D_s and D_u and a timepoint t, with $s < t < u$, Conv Block^0 computes a rough estimation of intermediate flow fields $\hat{F}^0_{t\to s}, \hat{F}^0_{t\to u}$ and fusion mask M^0 to capture large motions. Then in Conv Block^i, $i > 0$, D_s and D_u are first backward-warped using Eq. (1), based on the intermediate flows $\hat{F}^{i-1}_{t\to s}, \hat{F}^{i-1}_{t\to u}$ and mask M^{i-1} of the previous iteration (see the yellow Warping block in Fig. 2, middle column). Next, D_s and D_u, warped frames $\hat{D}^{i-1}_{t\leftarrow s}$ and $\hat{D}^{i-1}_{t\leftarrow u}$, intermediate flows $\hat{F}^{i-1}_{t\to s}$ and $\hat{F}^{i-1}_{t\to u}$, mask M^{i-1}, and timestep t are concatenated and processed by the stack of convolution and deconvolution layers in Conv Block^i. This results in updated $\hat{F}^i_{t\to s}, \hat{F}^i_{t\to u}$, and M^i, which then enter the next Conv Block. The process continues until the last Conv Block^{N-1} has finished computation, producing final estimates $\hat{F}^{N-1}_{t\to s}, \hat{F}^{N-1}_{t\to u}$, and M^{N-1}.

Interpolation and flow estimation. Interpolated scalar field $\hat{D}_t$, intermediate flow fields $\hat{F}_t{}^i$, and estimated flow field $\hat{F}_t$ are obtained via:

$$\hat{D}_t = \hat{D}^{N-1}_{t\leftarrow s} \odot M + \hat{D}^{N-1}_{t\leftarrow u} \odot (\mathbf{I} - M) \tag{2a}$$

$$\hat{F}_t{}^i = \hat{F}^i_{t\to u}, \quad \hat{F}_t = \hat{F}^{N-1}_t \quad (N=4) \tag{2b}$$

where $\odot$ denotes element-wise multiplication, and $\mathbf{I}$ is the identity matrix. We determined $N = 4$ as the optimal value, see Sec. VI-C.

Refinement by teacher module. During training, a dedicated *Conv Block*^{teach} implements a "privileged distillation scheme" [44] in the form of a teacher module that has access to a GT scalar field by receiving D_{GT} through an additional channel during concatenation; see Fig. 2. This produces flow fields $\hat{F}^{teach}_{t\to s}, \hat{F}^{teach}_{t\to u}$ and mask M^{teach}. By inserting these in Eq. (2a) and Eq. (2b), an interpolated field $\hat{D}^{teach}_t$ and estimated flow field $\hat{F}^{teach}_t$ are obtained for the teacher. The outputs for student $(\hat{D}_t, \hat{F}_t)$ and teacher $(\hat{D}^{teach}_t, \hat{F}^{teach}_t)$ are used in the loss functions to compute the prediction error which will then be backpropagated to update the network parameters. The process is repeated for many inputs during training (for more details, see Sec. IV-A).

Inference. In the inference phase, the scalar fields D_s and D_u are processed by the *trained* network, meaning that all GT information, the teacher block, and the loss functions are absent. The interpolated scalar field $\hat{D}_t$ and reconstructed flow field $\hat{F}_t$ as computed by Eqs. (2a), (2b) constitute the final output, see Fig. 1.

C. Loss Function

FLINT uses several loss components that can be combined depending on the scenario: (1) with available GT flow field and (2) without (the components are listed in Fig. 2 and described further below).

(1) The total loss for ensembles with available GT flow field is a linear combination of reconstruction loss $\mathscr{L}_{rec}$ and flow loss $\mathscr{L}_{flow}$:

$$\mathscr{L} = \mathscr{L}_{rec} + \lambda_{flow}\mathscr{L}_{flow}, \tag{3}$$

where $\lambda_{flow} = 0.2$ for balancing total loss scale w.r.t. the reconstruction component (determined via hyperparameter search, see Sec. VI-C).

(2) The total loss of FLINT for ensembles without available flow fields is a linear combination of the reconstruction $\mathscr{L}_{rec}$, distillation $\mathscr{L}_{dis}$, photometric $\mathscr{L}_{photo}$, and regularization $\mathscr{L}_{reg}$ losses:

$$\mathscr{L} = \mathscr{L}_{rec} + \lambda_{dis}\mathscr{L}_{dis} + \lambda_{photo}\mathscr{L}_{photo} + \lambda_{reg}\mathscr{L}_{reg}, \tag{4}$$

where $\lambda_{dis} = 10^{-4}, \lambda_{photo} = 10^{-6}, \lambda_{reg} = 10^{-8}$ for balancing total loss scale w.r.t. other loss components (see Sec. VI-C).

Scalar field interpolation (flow-unsupervised). To temporally interpolate between fields, we utilize student and teacher blocks (Sec. III-A), incorporating loss components aimed at improving the accuracy of the interpolated density from Eq. (2a). The reconstruction loss $\mathscr{L}_{rec}$ measures the L_1 distance between the GT D^{GT}_t and the reconstructed field representation from both the student and teacher:

$$\mathscr{L}_{rec} = \|D^{GT}_t - \hat{D}_t\|_1 + \|D^{GT}_t - \hat{D}^{teach}_t\|_1. \tag{5}$$

The first term in Eq. (5) is evaluated using D^{GT}_t but only for the loss calculation; D^{GT}_t is never used as input to the student network, only to the teacher module.

Physical flow estimation (flow-supervised). When we have access to GT flow information during training, we incorporate a flow loss component to enhance the quality of the learned flow field (physical flow). This loss function comprises reconstruction and teacher components, as well as the flow loss component. The supervised flow loss measures the L_1 distance between the estimated flow from each block of the neural network and the GT flow. In our experiments, we found that accumulating this measure based on all blocks rather than just the last one yields better results. We also adopt the concept of exponentially increasing weights from RAFT [5]. This loss

equation for physical flow estimation can be expressed as follows:

$$\mathscr{L}_{\text{flow}} = \sum_{i=1}^{N} \gamma^{N-i} \|F_t^{GT} - \hat{F}_t^i\|_1 + \|F_t^{GT} - \hat{F}_t^{teach}\|_1, \quad (6)$$

where F_t^{GT} is the GT flow at time t, $\hat{F}_t^i$ is the flow output from the corresponding i^{th} block of the student network (Eq. (2b)), $N = 4$ is the number of blocks in the model, and $\hat{F}_t^{teach}$ is the flow output from the teacher block. We experimentally established the value of γ as 0.8, aligning with the RAFT loss and validating this choice through our hyperparameter search.

Again, F_t^{GT} is only used for calculation of the error; it is never used as input to the student-teacher network.

Optical flow estimation (flow-unsupervised). When the flow field is not available for model training, we add distillation [8], photometric [25], and regularization loss components as conceptual replacements of the supervised physical flow loss. These components help the model to reconstruct the desired timesteps based on the optical flow field, which in this case is learned in a flow-unsupervised mode.

The distillation loss is based on the fact that the model outputs more accurate flow when it receives the GT timesteps of the different field which is available (e.g., density). It is computed as L_2 distance between the intermediate flows of the teacher and student networks:

$$\mathscr{L}_{dis} = \sum_{j \in \{s,u\}} \|\hat{F}_{t\to j}^{N-1} - \hat{F}_{t\to j}^{teach}\|_2, \quad (7)$$

We also conducted experiments incorporating the smoothness loss component, commonly used in optical flow estimation tasks [25]. However, in our case, where there are generally no clearly distinguishable objects within our ensembles, the smoothness loss did not improve the results. The metric results obtained by incorporating the smoothness loss component are reported in Table VI under the "FLINT *smooth*" row. The Charbonnier penalty function is used in both photometric and smoothness loss components to provide robustness against outliers [47]. The photometric loss is computed as the difference between the fields D_s, D_u, and the reconstructed field $\hat{D}_t$:

$$\mathscr{L}_{photo}(\vec{v}; D_j, \hat{D}_t) = \frac{1}{2} \sum_{j \in \{s,u\}} \sum_{\vec{p} \in P} \rho(D_j(\vec{p}) - \hat{D}_t(\vec{p} + \vec{v}(\vec{p}))), \quad (8)$$

where $\vec{v}$ are the components of the estimated optical flow field and $\rho(x) = \sqrt{x^2 + \varepsilon^2}$ is the Charbonnier penalty function with $\varepsilon = 10^{-9}$. The summation $\sum_{\vec{p} \in P}$ indicates that the loss is computed over all pixel coordinates $\vec{p} \in P \subset \mathbb{N}^d$ of the domain in a $d-$dimensional space, and the expression $\vec{p} + \vec{v}(\vec{p})$ represents the updated pixel coordinates after applying the estimated flow vectors $\vec{v}$. We apply the photometric loss component to the last block of FLINT, i.e., $ConvBlock^{N-1}$ in Fig. 2.

Additionally, we apply L_1 regularization [48] to the weight matrix of the last convolutional block of the student and teacher networks to prevent overfitting of the flow field learning:

$$\mathscr{L}_{reg} = \sum_{i=1}^{L} \left(\|W_i\|_1 + \|W_i^{teach}\|_1 \right), \quad (9)$$

where W_i and W_i^{teach} represent the i^{th} weight matrix of the last convolutional block in the student and teacher network, respectively, with L denoting the total number of weights.

D. Comparison between FLINT and RIFE methods

We now compare the architectures of FLINT and RIFE in detail (their respective performances are discussed in Sec. V and Sec. VI).

Similarities. RIFE and FLINT both receive two scalar input fields and employ several convolution blocks to produce approximate intermediate flows and a fusion mask. Warped fields are computed from the input frames with the help of the intermediate flows. Merging the two warped frames with the help of the fusion mask via Eq. (2a) yields the interpolated field. RIFE and FLINT feature an online end-to-end trainable student-teacher architecture using a special teacher convolution block. To calculate the model residual error, both use an optimization that minimizes a loss function, consisting of various terms.

Improvements and Extensions. In RIFE the number of student convolution blocks is fixed to three, while in FLINT we can flexibly adapt the number and configuration of blocks for the most optimal performance. Furthermore, RIFE's use of different scales for different blocks with bilinear resize is replaced by convolution and deconvolutional layers, resulting in more learnable parameters. Additionally, FLINT comprises convolutional and deconvolutional layers with varying strides in each block, unlike RIFE's eight convolutional layers with a stride of one per block. This autoencoder-like design effectively captures key input features [49], as demonstrated by the performance gains shown in Table VI.

Crucially, FLINT features new kinds of loss functions for handling different scenarios. When no flow fields are available (case **(2)** in Sec. III-C), FLINT uses additional loss functions, such as photometric loss [25] (RIFE has only the first two terms in Eq. (4)). In contrast to RIFE, FLINT can utilize available GT flow fields during training (case **(1)**) to improve estimated flow fields quality during inference.

In RIFE, the intermediate flow fields are only used to obtain the interpolated video frames which appear blurred and discontinuous (see [8, Fig. 5]), especially when the teacher module is omitted. In FLINT we achieve flow fields of high quality so they can serve as meaningful supplements for spatiotemporal data analysis. FLINT further exhibits significantly lower computational cost: eliminating the need for RIFE's encoder-decoder CNN (RefineNet [33], [50]) allowed FLINT to halve its training time without impacting result quality according to our experiments. Last but not least, while RIFE is limited to 2D image sequences, FLINT can handle arbitrary 2D scalar fields as well as 3D+time data to open up new opportunities for analysis in scientific visualization. To enable this, we implemented a new 3D warping technique, integrated 3D convolutional and deconvolutional layers, and utilized 3D loss functions.

IV. Study Setup

In this section, we describe the training setup, provide an overview of the datasets used in our experiments and discuss the evaluation methods employed to assess the FLINT results.

A. Training

We apply the standard prepossessing step of normalization to [0, 1] when necessary before the start of the training. FLINT is optimized using AdamW [51] — an adaptive gradient descent method with weight decay used in back-propagation algorithms for training feed-forward neural networks that combines the benefits from both the Adam optimizer [52] and L_2-regularization. We employ early stopping with a patience parameter of 30, which is equivalent to regularization [53] and helps to prevent overfitting on the training data. We use an experimentally determined learning rate of 6×10^{-4} for the 2D case and 1×10^{-4} for the 3D case respectively with a cosine annealing scheduler that gradually decreases the learning rate to 6×10^{-6} and 1×10^{-6} respectively by the end of the training. We train FLINT with mini-batches of size 32 for both 2D ensemble datasets and mini-batches of size 2 for 3D datasets used in our study (see Sec. IV-B). We split the set of all available data into training, validation (for monitoring training progress), and test subsets.

Note that training and test data are obtained by subsampling the original dataset, which is assumed to provide the GT scalar fields. Then FLINT performs interpolation at missing timesteps.

To support arbitrary interpolation, we chose $t \in [s,u]$ randomly at the training stage. Throughout this work, we use a maximum time window of size 12 to sample the triplets D_s, D_t, D_u for the training set, which was determined via hyperparameter search, to support arbitrary interpolation and flow estimation during the training stage. This window determines the maximum time gap between the timesteps that are used for interpolation (i.e., timesteps s and u). Our proposed FLINT model is trained for 120 epochs, resulting in a trained model size of 79MB. The training process takes no more than 12 hours on a single Nvidia Titan V GPU with 12GB of VRAM to converge for all datasets.

B. Datasets

We consider four scientific datasets in our study.

LBS (flow-supervised). The first is generated by a Lattice Boltzmann Simulation [54]. This ensemble is similar to a classic Kármán vortex street simulation and yields density as well as flow fields with a spatial resolution of 100×400. We consider an ensemble comprising 21 members, each consisting of 3K timesteps, with varying cylinder size, position, radius, collision timescale, and the dynamic viscosity of the fluid within the simulation. We utilize different members of the ensemble for validation and testing of FLINT. We randomly sample a training subset of 40K timesteps from a 12-sized window in the case of the LBS ensemble as discussed above.

Droplets (flow-unsupervised). The second is a Drop Dynamics (Droplets) ensemble derived from a physical experiment investigating the impact of a falling droplet on a film [9]. The experiment employs shadowgraphy imaging to study the splash crown and secondary droplets formed after the primary droplet's impact. Shadowgraphs reflect changes in the second derivative in density, which lead to variations in the refractive index of the medium and can therefore be detected optically. The experimental dataset consists of monochrome videos with a spatial resolution of 160×224, totaling 135K timesteps from 1K ensemble members. This dataset was collected to analyze various droplet impact regimes in relation to parameters like fluid viscosity, droplet velocity, film thickness, and Weber number. We interpret changes in luminosity in the video images as variations in the density field. Since no flow information is available, this constitutes a flow-unsupervised setup. We use different sets of 3K timesteps for validation and testing of FLINT and a randomly sampled subset of 40K timesteps for training.

5Jets (flow-supervised). The third one is a 3D+time 5Jets dataset generated by a Navier Stoke flow solver originally obtained from UC Davis. This simulation models five jets entering a cubical region and consists of 2000 timesteps with a spatial resolution of $128 \times 128 \times 128$. It contains density and the x, y, z components of velocity. We utilize every 10^{th} timestep of the simulation and randomly sample a subset of 500 timesteps for training, validation, and testing.

Nyx (flow-supervised). The fourth dataset is a 3D+time ensemble based on the compressible cosmological hydrodynamics simulation Nyx, developed by Lawrence Berkeley National Laboratory [55]. We consider an ensemble comprising 18 members, each consisting of a maximum of 1600 timesteps with a spatial resolution of $128 \times 128 \times 128$. It contains density and the x, y, z components of velocity. Akin to InSituNet [56], we vary three parameters for ensemble generation: the total matter density ($\Omega_m \in [0.12, 0.155]$), the total density of baryons ($\Omega_b \in [0.0215, 0.0235]$), and the Hubble constant ($h \in [0.55, 0.7]$). We randomly sample a training subset of 500 timesteps after selecting every 5th timestep across different ensemble members for training, validation, and testing.

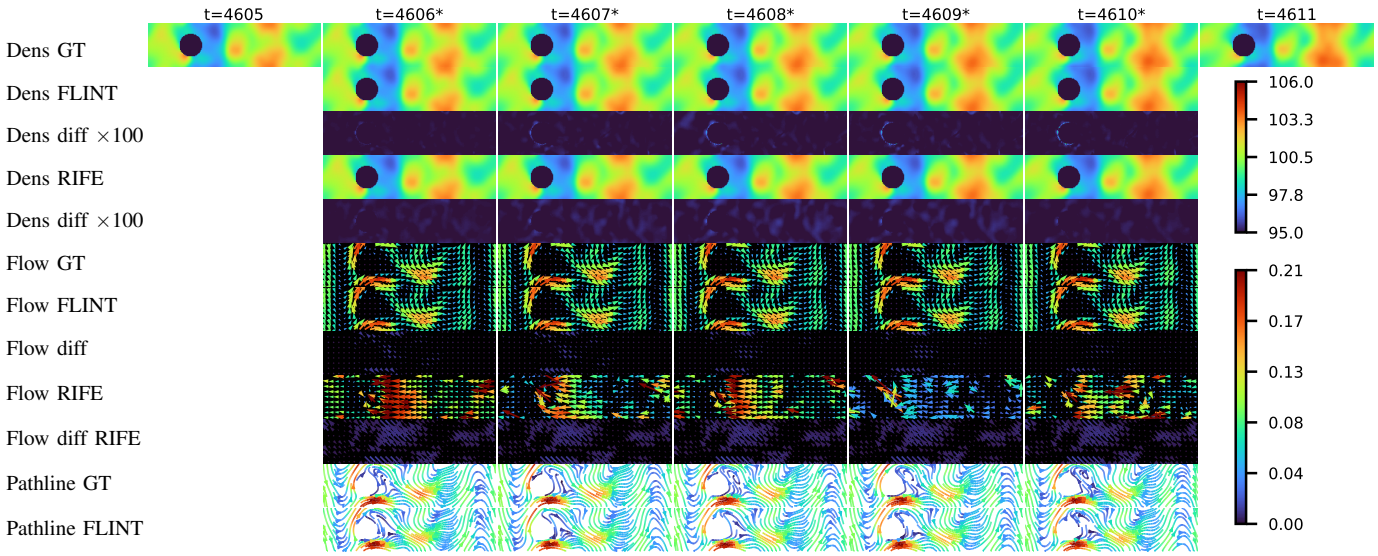
To visualize a simulation ensemble field, we use the *Turbo* colormap [57]. We visualize the 2D flow field via arrow glyphs and 3D flow via volume rendering with a transfer function that indicates flow direction. Examples of the LBS and Droplets ensemble fields, as well as 5Jets and Nyx transfer functions, are in the supplementary material.

C. Evaluation

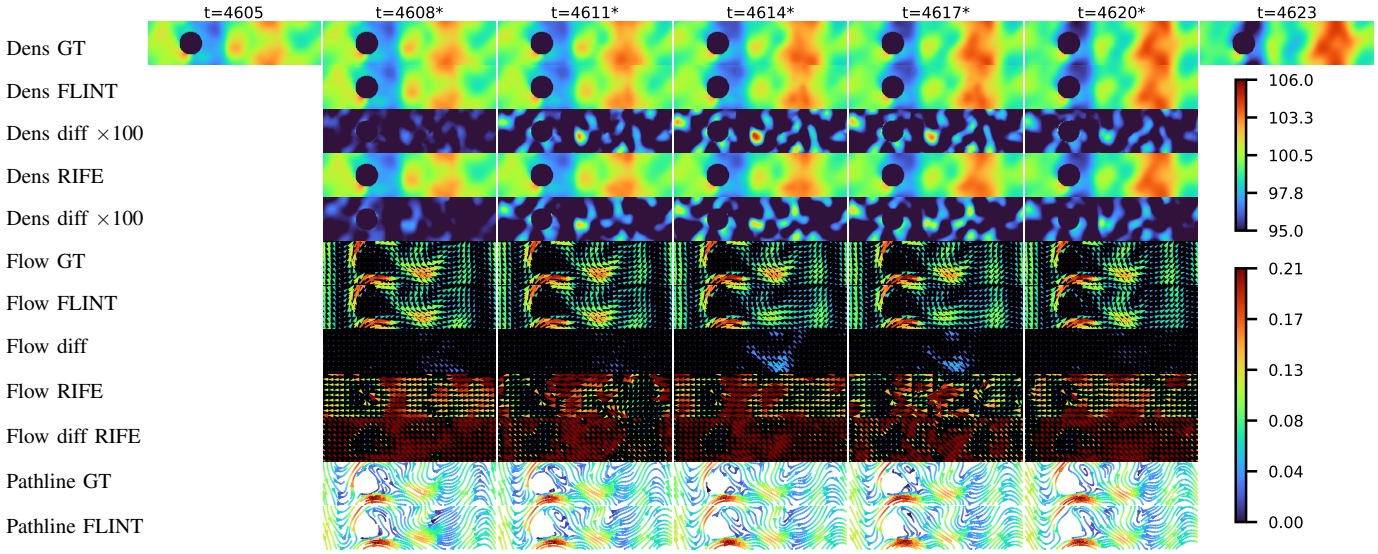
We evaluate the performance of FLINT for reconstructing the scalar field both qualitatively and quantitatively. For quantitative evaluation, we utilize two different metrics: *peak signal-to-noise ratio* (PSNR) and *learned perceptual image patch similarity* (LPIPS) [58]. LPIPS measures similarity between the activations of two images using a pre-defined network, a lower score indicates greater perceptual similarity.

We evaluate the accuracy of the learned flow field using the *endpoint error* (EPE), which measures the average Euclidean distance between estimated and GT flow vectors—a lower EPE indicates higher accuracy. For qualitative assessment, we visualize the flow field outcomes and analyze difference plots for both simulation and experimental ensembles. In our evaluation, we consider different interpolation (or subsampling) rates, where a rate of x means that we only consider each x^{th} timestep of the original data as input.

For the 3D datasets, we utilize 3D PSNR and 3D EPE in the volume domain rather than the image domain. This ensures that the evaluation metrics are appropriately adapted to the spatial characteristics of the 3D data, providing a more



(a) Interpolation rate of 6× (between $s = 4605$ and $u = 4611$).



(b) Interpolation rate of 18× (between $s = 4605$ and $u = 4623$).

Fig. 4: LBS ensemble: FLINT flow field estimation and temporal density interpolation at the timesteps with an asterisk (*)—(a) 6× and (b) 18× interpolation. Each subfigure, from top to bottom shows GT density, FLINT interpolated density, difference with GT density (magnified by ×100), RIFE interpolated density, difference with GT density (magnified by ×100); GT flow, FLINT flow estimation, difference with GT flow, flow estimated by RIFE, difference with GT flow, GT pathlines, and FLINT pathlines. The colorbar on the top right maps density, and the one on the bottom right maps flow magnitude.

accurate assessment of the performance and accuracy of FLINT in handling volumetric datasets.

V. Qualitative Results

We evaluate FLINT via four different datasets (Sec. IV-B) in different scenarios regarding flow estimation (i.e., supplementing density with flow in both flow-supervised and flow-unsupervised cases) as well as density interpolation (i.e., temporal super-resolution).

A. LBS: Flow Field Available for Some Members

First, we focus on the scenario in which we have GT flow fields available for some ensemble members (i.e., flow-supervised learning scenario). As can be seen in Fig. 4 overall, our approach achieves accurate performance in terms of density field interpolation and crucially flow field estimation. In this example, results were obtained at the interpolation rates of 6 and 18. For both cases, the error between the reconstructed density field and its GT is very small (note the magnification by a factor of 100 in the FLINT difference plots, third row). Moreover, it shows that the model is able to effectively learn a flow field that closely resembles simulated flow, resulting in a low error when compared to GT data. Comparing our FLINT method to RIFE, we can see that interpolation of the density field worked well for both, however, the flow learned by RIFE is not accurate and is far away structurally from the GT flow. (See Sec. III-D for a summary of the differences between the FLINT and RIFE methods.) Upon closer examination of the density difference error between FLINT and GT in Fig. 4b, it

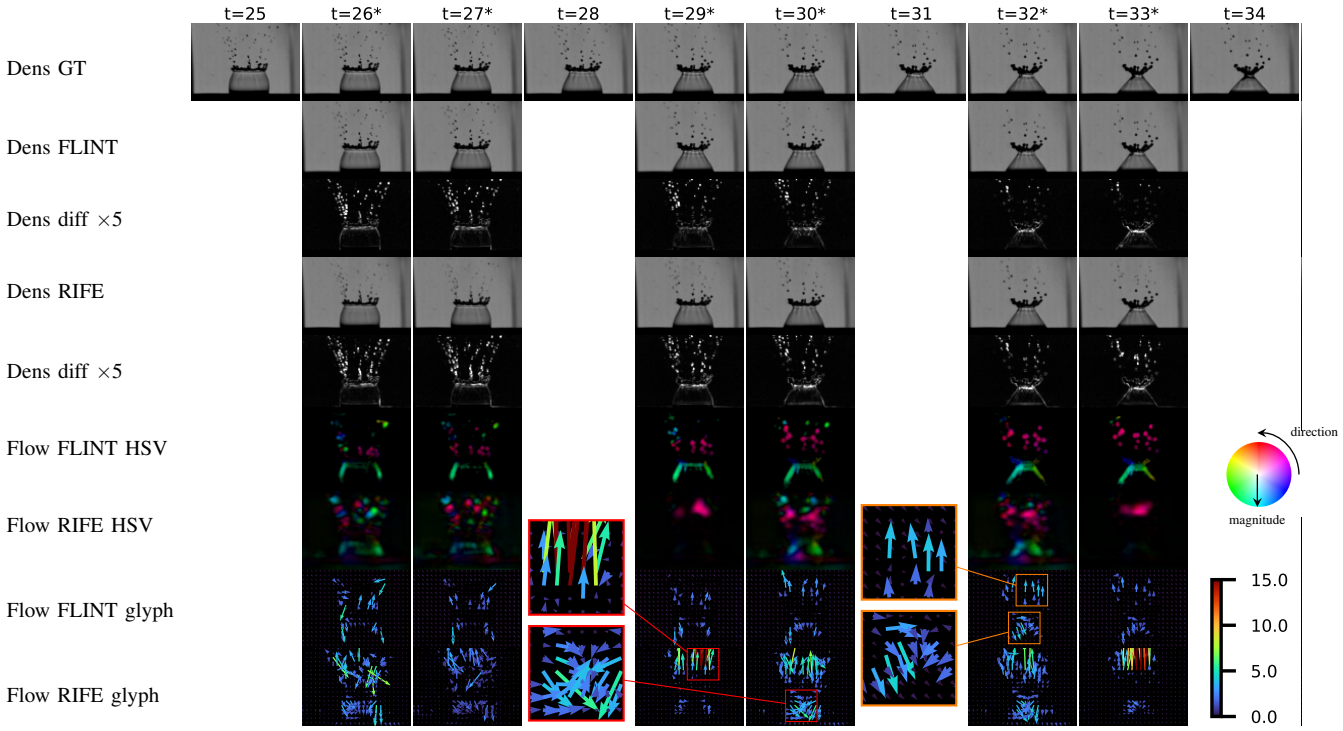


Fig. 5: Droplets ensemble: FLINT flow field estimation and temporal density interpolation during inference—at the timesteps with an asterisk (*)—3× interpolation. From top to bottom, the rows show GT density, FLINT interpolated density, difference to GT density (magnified by ×5), RIFE interpolated density, difference to GT density (magnified by ×5), FLINT flow estimation in HSV (see bottom right), RIFE flow estimation in HSV, flow glyphs for FLINT, and flow glyphs for RIFE.

becomes apparent that the highest error occurs in the middle of the frame. This aligns with an observed error between the GT flow and FLINT flow estimation in the same region. This observation suggests that as the model begins to produce flow with a certain degree of error in this example, the density also experiences a similar error.

In general, going from an interpolation rate of 6 (Fig. 4a) to 18 (Fig. 4b), the density interpolation remains quite accurate (differences are only visible due to significant contrast enhancement). Naturally, there are more errors in the flow estimations of both FLINT and RIFE when increasing the interpolation rate, but FLINT is still able to maintain comparably high accuracy. Further examples including different interpolation rates can be found in the supplementary material. This includes evaluations at very high interpolation rates, such as 32×, where the quality of flow estimation begins to degrade due to the significant structural differences between widely separated timesteps. Despite this, the scalar field interpolation remains relatively robust, demonstrating FLINT's ability to handle challenging conditions effectively.

Additionally, we perform a pathline analysis to further assess the accuracy of the flow estimated by FLINT. Pathlines were generated to visualize the flow field. Results for FLINT are shown in the last row and GT pathlines in the last but one row in Fig. 4. It can be seen that the pathline patterns produced by FLINT align closely with the GT for an interpolation rate of 6 (Fig. 4a), indicating a high level of accuracy in the flow field estimation. Even at an increased interpolation rate of 18 (Fig. 4b), the pathline patterns closely reflect the GT, although some small deviations become noticeable (e.g., for $t = 4608$, $t = 4614$, and $t = 4617$). These findings further validate the robustness of FLINT in capturing complex flow dynamics across varying interpolation rates, demonstrating the utility of reconstructing this otherwise missing flow information for analyzing flow around a cylinder.

B. Droplets: No GT Flow Field Available

The effectiveness of FLINT in interpolating the luminance field—captured by cameras during an experiment—based on the estimated optical flow, even in the absence of GT flow information, is evident from Fig. 5. The results demonstrate camera image interpolation by FLINT of high quality for the Droplets ensemble, in the second row. We compare FLINT's temporal interpolation performance to the one achieved with RIFE in Fig. 5, fourth row. RIFE's interpolation exhibits slightly higher variation compared to the GT, as can be seen from the difference plots, in the third and fifth rows. It is noteworthy that this concerns experimental data that naturally contains noise. For instance, in the top row of Fig. 5, the noise is visible as small, scattered inconsistencies in the background throughout the luminance field, giving it a slightly grainy appearance. Additionally, shadows can be observed on the right side of the images, where darker regions obscure portions of the luminance field. Despite this, FLINT demonstrates robust performance, effectively handling these imperfections and delivering high-quality interpolation results.

When examining the optical flow results generated by FLINT in Fig. 5 (sixth row), it is clear that the model successfully captures meaningful flow patterns that correspond to the

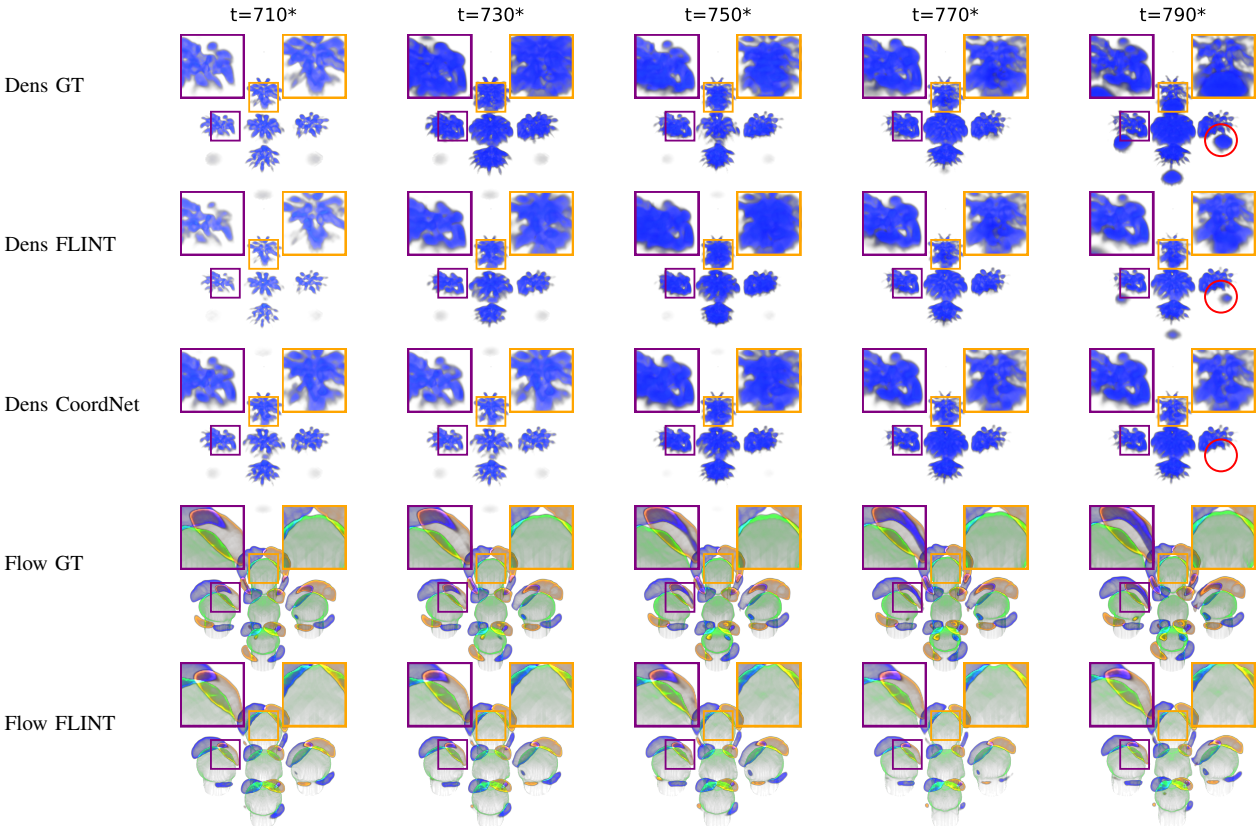


Fig. 6: 5Jets: FLINT flow field estimation and temporal density interpolation during inference, 20×. From top to bottom, the rows show GT density, FLINT interpolated density, CoordNet interpolated density, GT flow, and FLINT estimated flow. 3D rendering was used for the density and flow field visualization (● ● ● colors representing x, y, and z flow directions respectively), transfer functions are in the supplementary material.

direction of fluid particle movements in the majority of cases. The visualization shows droplets moving upward with a distinct red hue, while the parts of the splashes collapsing downward are characterized by a green hue. The optical flow estimated by RIFE, in comparison, is significantly less accurate as it fails to capture the finer details of particle movements, resulting in a blurrier flow. This indicates that FLINT is able to learn and predict flow information that aligns with the underlying dynamics of the fluid. In Fig. 5 (the second-to-last row for FLINT and the last row for comparison with RIFE), we further present arrow glyphs depicting flow direction and magnitude. With FLINT, these visualizations accurately reflect the movements of the bubble, crown, and droplets, and with this directly capture the evolution of the underlying physical phenomena in a static image (complementing the experimental images in the top row(s)). For example, in the orange zoom-ins ($t = 32$), the arrow glyphs effectively illustrate how the bottom part of the crown collapses downward, while the droplets predominantly move upward as they splash. This provides a clear and insightful visualization of the contrasting dynamics within the scene. When comparing FLINT to RIFE, it becomes evident that our method offers superior accuracy in capturing the flow dynamics. RIFE tends to generate excessive flow in the area of the crown, particularly in timesteps such as $t = 26$, 27, 30, and 32. For example, the red bottom zoom-in ($t = 30$) demonstrates such a case where in the original experimental images only the boundaries of the bubble are visibly moving (see top row, Dens GT). Additionally, RIFE produces less accurate flow representations for the droplet splashes in the top part of the scene, as clearly seen in timesteps $t = 26$, 29 (red zoom-in), and 33. In contrast, FLINT adequately captures these flow patterns, providing an expressive representation for the analysis of fluid dynamics. In sum, FLINT generates flow fields that align with observed movements for analysis, even without GT flow.

C. 5Jets: Density & Flow Available for Some Timesteps

Third, we consider the scenario in which we have GT density and velocity fields available for some timesteps sampled from the whole 3D dataset. In this case, the goal is to perform unsupervised density interpolation (i.e., density TSR) and flow estimation via a flow-supervised learning scenario. As Fig. 6 shows, our approach achieves accurate performance in terms of density field interpolation and flow field estimation. In this example, results were obtained at an interpolation rate of 20×. Visually, the error between the renderings of the reconstructed density field and its GT is very small. Moreover, it shows that even at a comparably large interpolation rate of 20×, the model is able to effectively learn a flow field that structurally resembles the GT flow. Comparing our FLINT method to CoordNet, we can see that interpolation of the density field worked well for both (although some differences can be observed, especially highlighted with the red circles).

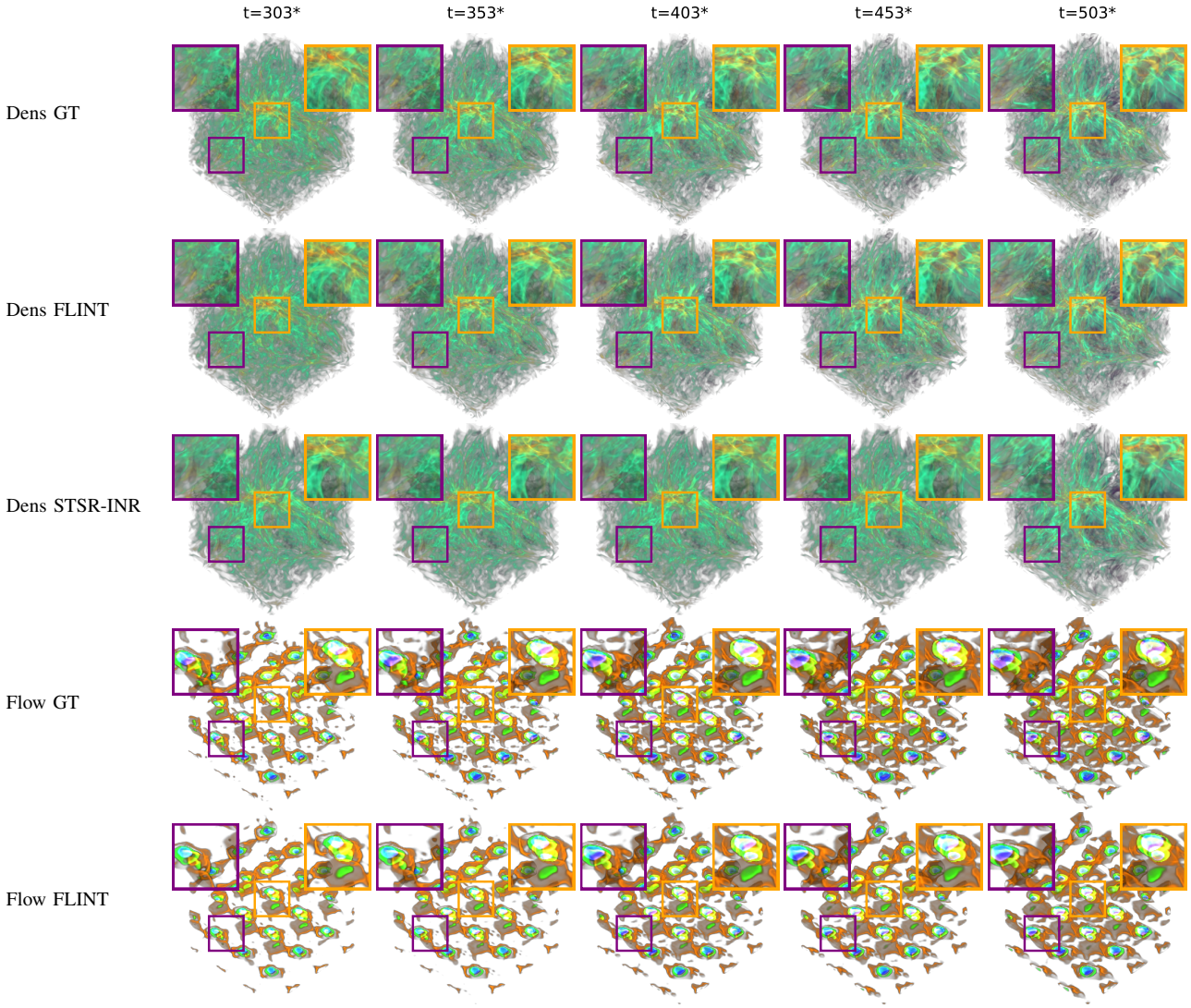


Fig. 7: Nyx: FLINT flow field estimation and temporal density interpolation during inference, 5×. From top to bottom, the rows show GT density, FLINT interpolated density, STSR-INR interpolation, GT flow, and FLINT flow estimation. 3D rendering was used for the density and flow field visualization (● ● ● colors representing x, y, and z flow directions respectively), transfer functions are in the supplementary material.

We are using a similar model size of both FLINT and CoordNet for this comparison. Crucially, however, FLINT additionally supplements the density with accurate flow information (as can be seen in Fig. 6), a feature that CoordNet lacks. In this dataset, the flow reveals the evolution of five distinct jets along the x, y, and z directions. Observing the flow allows us to interpret how density evolves, particularly in the purple and orange zoom-ins from timesteps $t = 710$ to $t = 790$. Here, we can see that the density concentrates and intensifies along the paths of the jets, forming more pronounced structures as the jets push the material outward. This results in higher density regions aligned with the jet flows, indicating areas where the flow is driving the accumulation of matter. This enhances our understanding of the underlying physical processes and provides a more comprehensive system analysis. Examples of the metric scores for the interpolation rates of 10×, 15×, and 20× can be found in Table I.

D. Nyx: Density & Flow Available for Some Timesteps

Fourth, we consider the scenario where GT density and velocity fields are available for some members of the entire 3D ensemble. As Fig. 7 illustrates, FLINT achieves accurate performance in terms of both density field interpolation and flow field estimation. Visually, the difference between the renderings of the reconstructed density field and its GT is minimal. Moreover, even at a relatively high interpolation rate of 5×, FLINT effectively learns a flow field that structurally resembles the GT flow. When comparing our FLINT method to STSR-INR for temporal density interpolation (top three rows), it is evident that the interpolation of the density field is more accurate with FLINT. For example, at $t = 503$, STSR-INR shows a different structure and less dark matter density compared to FLINT, which preserves density. This trend is consistently observed across all shown timesteps in Fig. 7. For this comparison, we used the same model size for both FLINT and STSR-INR. By default, Implicit Neural Representation (INR)-based

models like CoordNet and STSR-INR have smaller model sizes compared to FLINT due to their simpler architecture, where data is represented as a continuous function parameterized by a compact set of weights. In contrast, FLINT employs a more complex design with multiple convolutional blocks, enabling it to achieve accurate flow estimation and scalar field interpolation. To ensure a fair comparison below, we configured STSR-INR to have the same model size as FLINT, allowing us to directly assess their performance under comparable conditions.

Crucially, FLINT not only reconstructs the density field but also supplements it with accurate flow information, as shown in Fig. 7 (last row), a feature that STSR-INR lacks. When examining the flow, we observe circular swirling patterns, indicating the complex dynamics of the baryonic gas. As these flows intensify, we see evidence of dark matter moving outward—reflected in both the GT and FLINT density, especially in the orange zoom-ins—consistent with an expanding universe. This underlines the utility of FLINT in capturing not just the static density fields but also the dynamic evolution of the cosmic structures. Examples of the metric scores for interpolation rates of $3\times$, $5\times$, and $8\times$ can be found in Table II.

E. Nyx: Domain Expert Evaluation

A domain expert—an assistant professor in astronomy with a research focus on cosmological simulations and observational data evaluation—provided insights into FLINT's utility for analyzing scalar fields and velocity information in the Nyx dataset. He highlighted that FLINT's ability to estimate velocity fields alongside density fields offers significant advantages. The expert remarked, "compared to other reconstruction methods, with FLINT you get information not only on the *density* in Nyx simulations but also on the *velocity*". This dual capability is particularly important in cosmology, where velocity data derived from "redshift" measurements are critical for estimating distances and analyzing spatial relationships between objects.

The expert stated that "having a way to estimate accurately the peculiar velocities would help when comparing with observations". He further underlined the relevance of velocity information in creating "lightcones", which combine simulation timesteps to mimic observations of the universe at different epochs. These lightcones are crucial for studying phenomena such as the intergalactic medium and the 3D distribution of galaxies, where "the distance information comes really from a velocity shift". He emphasized that "having a way to estimate accurately the velocities would help when comparing with observations". By accurately reconstructing both density and velocity fields, FLINT enhances the fidelity of simulation outputs aligned with observational data, bridging gaps between simulations and real-world measurements, and enabling a deeper understanding of cosmological dynamics.

VI. Quantitative and Comparative Evaluation

In this section, we present quantitative results and compare against baseline methods to demonstrate the improvement achieved with our proposed method, followed by ablation and parameter studies to explore different configurations and hyperparameters.

A. Comparison Against Baselines

We illustrate quantitative results using boxplots in Fig. 8 for LBS and Droplets ensembles. FLINT (blue) is compared against two baselines, RIFE (red) and linear interpolation (green), across various interpolation factors ranging from 2 to 64, showcasing representative outcomes for both ensembles. Similarly, in LPIPS-based evaluation for the Droplets, FLINT demonstrates better performance. Furthermore, examining the flow estimation (supplementing density with flow information during inference) results for the LBS ensemble in Fig. 8b reveals that FLINT achieves a significantly lower endpoint error with minimal variance when compared to RIFE. This underscores FLINT's proficiency in learning an accurate flow representation that closely aligns with the GT physical flow, particularly evident in the LBS ensemble scenario. As linear interpolation cannot provide flow estimation results, no statistics are shown for it, in the case of the LBS ensemble.

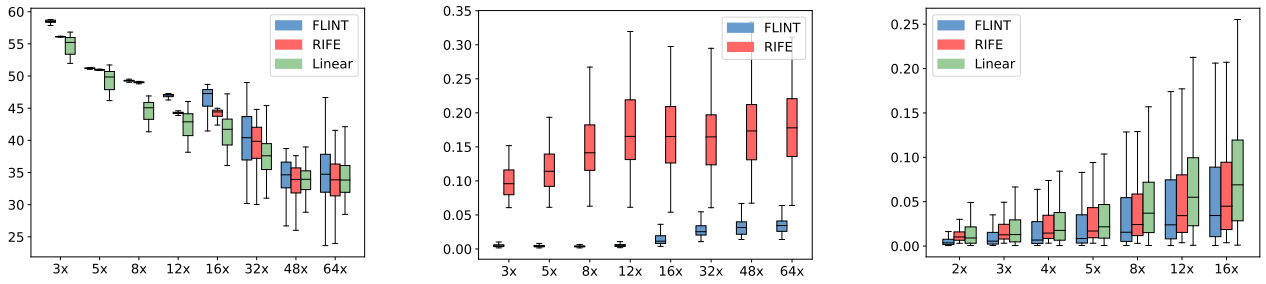
In the case of the Droplets ensemble, FLINT significantly outperforms linear interpolation for both LPIPS and PSNR metrics, confirming our expectations, as shown in Fig. 8c and supplementary material. This superior performance can be attributed to FLINT's capacity to learn robust optical flow, which in turn enhances its ability to interpolate and reconstruct complex density fields. FLINT consistently performs better than RIFE as well, across all presented interpolation rates, for both PSNR and LPIPS metrics. This performance reaffirms that our proposed model is well-suited for the challenging task of reconstructing density fields within scientific ensembles. These findings underscore the potential of FLINT in spatiotemporal data interpolation tasks, with application for scientific visualization and data analysis.

CoordNet [40], another method we considered for comparative evaluation, offers an advanced framework for various tasks in time-varying volumetric data visualization, including TSR. It has shown improvements over the TSR-TVD method [35] discussed in Sec. II, positioning it as a relevant benchmark for temporal interpolation performance. We trained CoordNet for the 3D+time 5Jets dataset and compared across various interpolation rates, see Table I. FLINT demonstrates competitive performance in terms of PSNR score for density interpolation while also serving the dual purpose of enabling flow estimation. While CoordNet facilitates TSR, it does not extend to flow field estimation, which we consider to be the main contribution of this work.

TABLE I: Comparison against baselines, 5Jets

Method	10×		15×		20×	
	PSNR ↑	EPE ↓	PSNR ↑	EPE ↓	PSNR ↑	EPE ↓
FLINT	46.72	0.7271	**45.41**	0.7286	**44.93**	0.7292
Linear	44.70	—	42.28	—	38.11	—
CoordNet	**48.45**	—	44.89	—	43.74	—

STSR-INR [41], another method we considered for comparative evaluation, was applied to the 3D+time Nyx dataset and compared across various interpolation rates, as shown in Table II. STSR-INR is an INR-based approach that employs a variable embedding scheme to learn latent vectors for different variables. This method utilizes a variational auto-



(a) LBS: PSNR ↑ scores (b) LBS: EPE ↓ scores (c) Droplets: LPIPS ↓ scores

Fig. 8: Comparison of FLINT, RIFE, and linear interpolation at various interpolation rates. For PSNR plots for the Droplets dataset and LPIPS scores for the LBS dataset, see Figures 15 and 16 of the supplementary material, respectively.

decoder to optimize the learnable latent vectors, enabling latent-space interpolation. STSR-INR shows improvements over both STNet [34] and CoordNet [40], positioning it as a relevant benchmark for TSR performance. FLINT demonstrates superior performance in terms of PSNR score for density interpolation while also serving the dual purpose of enabling flow estimation, a capability that extends beyond the functionalities provided by STSR-INR.

TABLE II: Comparison against baselines, Nyx

Method	3×		5×		8×	
	PSNR ↑	EPE ↓	PSNR ↑	EPE ↓	PSNR ↑	EPE ↓
FLINT	**53.17**	0.0310	**52.31**	0.0310	**49.39**	0.0309
Linear	47.49	—	41.92	—	37.51	—
STSR-INR	49.63	—	44.21	—	41.09	—

Evaluation on Additional Scalar Field. In addition to evaluating FLINT for density and flow fields, we extended our analysis to another scalar field to demonstrate its versatility. Specifically, we evaluated FLINT's performance on the temperature field from the Nyx simulation ensemble. We examined FLINT's ability to interpolate the temperature field and estimate the corresponding flow fields. Qualitative results are shown in Fig. 18 of the supplementary material, and quantitative evaluations are provided in Table III. The temperature with flow results indicate that the temperature interpolation performance remains robust across different interpolation rates, although the PSNR values are slightly lower compared to density results. This difference reflects the increased complexity of interpolating temperature fields due to their distinct data characteristics. Similarly, the flow results show lower EPE scores. This is expected, as estimating flow from temperature is inherently more challenging in comparison to density due to its weaker correlation with movement or physical flow dynamics.

Our findings align with observations by Gu *et al.* [16], where the performance of vector field reconstruction can vary depending on the input scalar field. Similarly, FLINT's performance showed variations when applied to the temperature field, reflecting the challenges associated with different data characteristics.

These results further demonstrate FLINT's adaptability across diverse scalar fields while highlighting the influence of field-specific dynamics on reconstruction accuracy.

Impact of Noise. To evaluate FLINT's robustness to noise, we introduced random Gaussian noise into the 3D Nyx simulation ensemble. Even with the added noise in the GT density, FLINT achieved reliable results, maintaining an average PSNR of 46.36 dB and an EPE of 0.0356 at a 5× interpolation rate. In comparison, without noise, the model achieved a higher PSNR of 52.31 dB and a slightly lower EPE of 0.0310, demonstrating a modest decline in performance due to noise. This highlights FLINT's resilience to noisy data and reinforces its applicability to real-world scenarios where imperfections in scientific datasets are common. For further details, please refer to the supplementary material.

TABLE III: Comparison of different fields, Nyx

Field	3×		5×		8×	
	PSNR ↑	EPE ↓	PSNR ↑	EPE ↓	PSNR ↑	EPE ↓
Density + Flow	53.17	0.0310	52.31	0.0310	49.39	0.0309
Temperature + Flow	49.32	0.0446	46.91	0.0451	44.43	0.0467

Large Interpolation Rates. At large interpolation rates, such as 32× (Fig. 13 of the supplementary material), we observe that while density interpolation maintains acceptable quality (average PSNR of 37.71 dB), the accuracy of flow estimation noticeably degrades (average EPE is 0.0479). In comparison, at 8× interpolation, the model achieved a PSNR of 49.39 dB and an EPE of 0.0309, while at 16×, the PSNR was 43.42 dB with an EPE of 0.0338. The degradation at a large interpolation rate is expected due to the increased structural differences between flows at widely separated time steps, which make flow estimation inherently more challenging. Nevertheless, FLINT delivers satisfactory results for scalar field interpolation even at these demanding rates. Qualitatively, while the main structure, such as the distribution of dark matter, is preserved and remains visually consistent with the GT, finer details and intricate patterns become less accurate. Further details are provided in the supplementary material.

Inference time comparison. We conducted an inference time comparison of our model, FLINT, against RIFE, CoordNet, and STSR-INR across all datasets. Specifically, we compared FLINT with RIFE on the 2D ensembles (LBS and Droplets) and with CoordNet and STSR-INR on the 3D datasets (5Jets and Nyx). For each evaluation, we measured the inference time over 3K timesteps for the 2D ensembles and 300 timesteps for the 3D ensembles from the test set. Our findings indicate that FLINT achieves faster inference time compared to RIFE, with an average of 0.025 seconds per timestep on the 2D datasets, as opposed to RIFE's 0.035 seconds. On the 3D datasets, FLINT significantly outperforms both CoordNet and STSR-INR in inference speed, even achieving interactive rates

with an average time of 0.2 seconds per timestep for FLINT. In comparison, it takes 2.1 seconds for CoordNet and 1.5 seconds for STSR-INR. The faster inference time of FLINT compared to INR-based approaches like CoordNet and STSR-INR can be attributed to the inherent efficiency of convolutional neural networks (CNNs), which excel in parallel processing and optimized memory access, while INR-based methods require solving implicit functions at each point, adding computational complexity.

B. Ablation Studies

Our proposed FLINT method is a deep neural network that has various loss components for training the network. To assess the impact of these in achieving optimal results, we conducted a series of ablation studies.

TABLE IV: Ablation of FLINT (flow-supervised)

Method	**LBS, 3×**		**5Jets, 10×**		**Nyx, 5×**	
	PSNR ↑	**EPE ↓**	**PSNR ↑**	**EPE ↓**	**PSNR ↑**	**EPE ↓**
FLINT no *flow*	57.41	0.0842	37.80	3.4985	51.97	0.1461
FLINT no *rec*	49.89	0.0119	32.32	0.7332	44.67	0.0497
FLINT w/o s-t	56.37	0.0064	41.99	0.7323	52.08	0.0334
FLINT	**58.44**	**0.0051**	**46.72**	**0.7271**	**52.31**	**0.0310**

TABLE V: Ablation of FLINT, Droplets, 2× (flow-unsupervised)

Method	**PSNR ↑**	**LPIPS ↓**
FLINT no *rec*	32.09	0.0373
FLINT no *dis*	39.87	0.0129
FLINT no *photo*	40.47	0.0105
FLINT no *reg*	40.71	0.0098
FLINT no *dis, photo*	40.36	0.0116
FLINT no *dis, reg*	40.37	0.0117
FLINT no *reg, photo*	40.11	0.0110
FLINT no *dis, reg, photo*	39.81	0.0119
FLINT w/o s-t	40.25	0.0202
FLINT	**41.16**	**0.0087**

The ablation studies conducted on FLINT provide valuable insights into the impact of different loss components on the overall performance. In Table IV, which presents the results for the LBS, 5Jets, and Nyx datasets with available GT flow fields for the model training, four variants of FLINT are compared: "FLINT no *flow*", "FLINT no *rec*", "FLINT w/o s-t", and "FLINT". "FLINT no *flow*" refers to FLINT without the inclusion of the flow-related loss component. As expected, this variant demonstrates considerably less accurate physical flow, measured by EPE metric, despite a relatively high density interpolation score, measured by PSNR metric. "FLINT no *rec*" refers to FLINT without the reconstruction-related loss component. In this case, the flow learned by the model is relatively good, however, the density interpolation suffers due to the absence of its related loss component. "FLINT w/o s-t" refers to FLINT where the teacher block (see Fig. 2) and the teacher-related terms in the loss functions are removed. This scenario leads to slightly less accurate interpolation and flow estimation due to the fact that the model does not receive GT densities during the training. Finally, "FLINT" represents the complete FLINT model with all loss components. It achieves the highest scores across all metrics, including PSNR and EPE, outperforming all other variants. This demonstrates the significance of incorporating the proposed loss components in achieving superior interpolation results.

Similar ablation studies were performed for the Droplets ensemble, for which no GT flow fields are available, as shown in Table V. Multiple variants of FLINT were evaluated, including "FLINT no *rec*", "FLINT no *dis*" (without distillation), "FLINT no *photo*" (without photometric loss), "FLINT no *reg*" (without regularization), ablation of combinations of these losses, and "FLINT w/o s-t". Each variant is compared to the complete FLINT method. The results indicate that each loss component contributes to the overall performance of FLINT. The complete FLINT model outperforms all the variants in terms of PSNR and LPIPS scores. This showcases the importance of reconstruction, distillation, photometric, and regularization loss components in achieving the best possible reconstructions for the Droplets ensemble.

C. Parameter Studies

Our proposed FLINT method involves several hyperparameters, and we performed extensive parameter studies to understand their impact on achieving optimal results. Table VI displays the outcomes of these studies regarding 2D+time and 3D+time datasets.

TABLE VI: Hyperparameter search for FLINT

Method	**LBS, 3×**		**5Jets, 10×**		**Nyx, 5×**		**Droplets, 2×**	
	PSNR ↑	**EPE ↓**	**PSNR ↑**	**EPE ↓**	**PSNR ↑**	**EPE ↓**	**PSNR ↑**	**LPIPS ↓**
FLINT *128*	49.86	0.0135	43.57	0.7397	51.68	0.0315	36.89	0.0162
FLINT *Lapl*	57.23	0.0112	42.82	0.7387	51.73	0.0319	39.44	0.0192
FLINT $\lambda_{flow} = 0.3$	58.03	0.0109	44.89	0.7308	52.29	0.0311	—	—
FLINT $\lambda_{flow} = 0.1$	58.17	0.0113	45.11	0.7549	52.19	0.0320	—	—
FLINT $\gamma = 0.9$	55.81	0.0130	44.84	0.7327	52.14	**0.0310**	—	—
FLINT $\gamma = 0.7$	56.28	0.0137	46.08	0.7384	52.28	0.0317	—	—
FLINT $\lambda_{reg} = 10^{-9}$	—	—	—	—	—	—	40.80	0.0089
FLINT $\lambda_{reg} = 10^{-7}$	—	—	—	—	—	—	40.06	0.0122
FLINT $\lambda_{photo} = 10^{-7}$	—	—	—	—	—	—	40.79	0.0107
FLINT $\lambda_{photo} = 10^{-5}$	—	—	—	—	—	—	38.98	0.0144
FLINT $\lambda_{dis} = 1 \times 10^{-5}$	—	—	—	—	—	—	37.86	0.0157
FLINT $\lambda_{dis} = 1 \times 10^{-3}$	—	—	—	—	—	—	40.34	0.0114
FLINT *smooth*	—	—	—	—	—	—	36.96	0.0125
FLINT *stride = 1*	51.33	0.0147	43.41	0.8116	52.19	0.0418	38.64	0.0149
FLINT *3 Blocks*	53.08	0.0136	43.09	0.7346	52.28	0.0312	39.74	0.0144
FLINT *4 Blocks*	**58.44**	**0.0051**	**46.72**	**0.7271**	**52.31**	**0.0310**	**41.16**	**0.0087**
FLINT *5 Blocks*	57.20	**0.0051**	45.57	0.7420	52.15	**0.0310**	41.04	0.0113
FLINT *6 Blocks*	56.17	0.0106	44.61	0.7691	51.98	0.0313	39.79	0.0134
FLINT *two-stage*	56.87	0.0069	45.78	0.7409	52.07	**0.0310**	40.95	0.0108

The parameter search investigates various configurations of FLINT , each labeled with a specific identifier. To determine the model's optimal architecture for our task, we conducted a series of experiments, adjusting the model's capacity by varying its width and depth. For example, "FLINT *128*" represents a configuration with 128 channels in all convolutional layers across all blocks. Through hyperparameter optimization, we found that the best-performing model, named "FLINT" or "FLINT *4 Blocks*", increases both the channel capacity and the number of blocks. This configuration, with 4 blocks and channel counts in the convolutional layers ranging from 256 to 128, is designed to capture more intricate and detailed features within the data. Another configuration is "FLINT *smooth*", where a smoothness loss component was incorporated into the training process for Droplets, however, not yielding the best results. Additionally, "FLINT *stride = 1*" represents an architecture where each block uses convolutional and deconvolutional layers with a stride of one, instead of varying strides. Furthermore, we explored two different loss functions for the interpolation part of the model. The configuration "FLINT *Lapl*" indicates the use of the Laplacian loss, which measures the L_1-loss between two Laplacian pyramid representations (pyramidal

level is 5) of the reconstructed density field and GT density field. The configuration "FLINT" employs the simple L_1-distance as the loss function, as described in Sec. III-C. In addition, we varied the number of blocks of FLINT (*Conv Block* in Fig. 2), ranging from three to six. As can be seen from Table VI, the most optimal configuration is the one with four blocks ("FLINT *4 Blocks*"). Models with fewer than four blocks lack sufficient capacity, while those with more begin to overfit. The configuration "FLINT *two-stage*" employs a two-stage optimization approach where a teacher model is trained first, followed by a student network trained to align with the teacher's outputs. In contrast, "FLINT" achieves slightly better performance overall while reducing training time by nearly half (12 vs. 20 hours), demonstrating its efficiency.

The results in Table VI demonstrate optimized FLINT's superior performance over other configurations, with the highest scores in PSNR and EPE metrics. This underscores the importance of channel capacity, number of blocks, and optimized loss functions in capturing the intricate dynamics of fluid ensembles, enabling FLINT's best density interpolation and flow estimation capabilities.

VII. Discussion and Conclusion

In this work, we proposed FLINT, a learning-based approach for the estimation of flow information and scalar field interpolation for 2D+time and 3D+time scientific ensembles. FLINT offers flexibility in handling various data availability scenarios. It can perform flow-supervised learning to estimate flow fields for members when partial flow data is available. In cases where no flow data is provided, common in experimental datasets, FLINT employs a flow-unsupervised approach, generating flow fields based on optical flow concepts. Additionally, FLINT generates high-quality temporal interpolants between scalar fields like density or luminance, outperforming recent state-of-the-art methods. It achieves fast inference and does not require complex training procedures, such as pre-training or fine-tuning on simplified datasets, commonly required by other flow estimation methods [5]–[7]. Its effectiveness in producing high-quality flow and scalar fields has been validated across both simulation and experimental data.

For future work, we aim to improve the FLINT method and expand its application to various problems and domains. FLINT is generally applicable to various fields, including density, flow, energy, as well as to grayscale or RGB datasets, and is valuable for visualization applications, and we aim to study a larger range of applications in future work. FLINT extensions could include expanding support for more timesteps as input, exploring extrapolation in addition to interpolation, or ensemble parameter space exploration. Another promising direction for future work is harnessing the learned flow field for efficient dimensionality reduction (DR). The compressed representation of the estimated flow could be used in order to perform DR of the ensemble members and compared against standard and ML-based DR techniques [59]. Furthermore, in the medical domain, learned optical flow can serve as additional input to classification models [60]. This strengthens disease classification tasks by incorporating motion fields, which convey crucial information. As a result, leveraging estimated flow enhances the accuracy and robustness of disease classification models, ultimately contributing to improved healthcare outcomes.

Acknowledgments

We thank the Center for Information Technology of the University of Groningen for their support and for providing access to the Hábrók high performance computing cluster. We also thank the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) for supporting this work by funding SFB 1313 (Project Number 327154368). We thank dr. Maxime Trebitsch (Institut d'Astrophysique de Paris) for his help in interpreting the flow visualizations of the Nyx cosmological simulation and assessing their usefulness for data analysis.

References

[1] H. Childs, J. Bennett, C. Garth, and B. Hentschel, "In situ visualization for computational science," *IEEE Comp. Graph. Appl.*, vol. 39, no. 6, pp. 76–85, 2019.

[2] H. Jänicke, T. Weidner, D. Chung, R. S. Laramee, P. Townsend, and b. M. Chen, "Visual reconstructability as a quality metric for flow visualization," *Comp. Graph. Forum*, vol. 30, no. 3, pp. 781–790, 2011.

[3] S. Frey and T. Ertl, "Flow-based temporal selection for interactive volume visualization," *Comp. Graph. Forum*, vol. 36, no. 8, pp. 153–165, 2017.

[4] G. Tkachev, S. Frey, and T. Ertl, "Local prediction models for spatiotemporal volume visualization," *IEEE Trans. Vis. Comp. Graph.*, vol. 27, no. 7, pp. 3091–3108, 2021-07.

[5] Z. Teed and J. Deng, "RAFT: Recurrent all-pairs field transforms for optical flow," in *Computer Vision–ECCV 2020: 16th European Conference, Proceedings, Part II 16*. Springer, 2020, pp. 402–419.

[6] A. Dosovitskiy, P. Fischer, E. Ilg, P. Hausser, C. Hazirbas, V. Golkov, P. Van Der Smagt, D. Cremers, and T. Brox, "Flownet: Learning optical flow with convolutional networks," in *Proc. IEEE Int. C. Comput. Vis. (ICCV)*, 2015, pp. 2758–2766.

[7] K. Luo, C. Wang, S. Liu, H. Fan, J. Wang, and J. Sun, "UPFlow: Upsampling pyramid for unsupervised optical flow learning," in *Proc. IEEE C. Comp. Vis. Pattern Rec. (CVPR)*, 2021, pp. 1045–1054.

[8] Z. Huang, T. Zhang, W. Heng, B. Shi, and S. Zhou, "Real-time intermediate flow estimation for video frame interpolation," in *European Conference on Computer Vision*. Springer, 2022, pp. 624–642.

[9] A. Geppert, D. Chatzianagnostou, C. Meister, H. Gomaa, G. Lamanna, and B. Weigand, "Classification of impact morphology and splashing/deposition limit for n-hexadecane," *Atomization and Sprays*, vol. 26, no. 10, 2016.

[10] S. Frey, S. Scheller, N. Karadimitriou, D. Lee, G. Reina, H. Steeb, and T. Ertl, "Visual analysis of two-phase flow displacement processes in porous media," in *Computer Graphics Forum*, vol. 41, no. 1. Wiley Online Library, 2022, pp. 243–256.

[11] H. Childs, S. D. Ahern, J. Ahrens, A. C. Bauer, J. Bennett, E. W. Bethel, P.-T. Bremer, E. Brugger, J. Cottam, M. Dorier *et al.*, "A terminology for in situ visualization and analysis systems," *The Int. J. of High Performance Computing Applications*, vol. 34, no. 6, pp. 676–691, 2020.

[12] Y. Yamaoka, K. Hayashi, N. Sakamoto, and J. Nonaka, "In situ adaptive timestep control and visualization based on the spatio-temporal variations of the simulation results," in *Proc. of the Workshop on In Situ Infrastruct. for Enabling Extreme-Scale Analysis and Vis.*, 2019, pp. 12–16.

[13] C. P. Kappe, L. Schütz, S. Gunther, L. Hufnagel, S. Lemke, and H. Leitte, "Reconstruction and visualization of coordinated 3d cell migration based on optical flow," *IEEE Trans. Vis. Comp. Graph.*, vol. 22, no. 1, pp. 995–1004, 2015.

[14] A. Kumpf, M. Rautenhaus, M. Riemer, and R. Westermann, "Visual analysis of the temporal evolution of ensemble forecast sensitivities," *IEEE Trans. Vis. Comp. Graph.*, vol. 25, no. 1, pp. 98–108, 2018.

[15] S. Manandhar, P. Bouthemy, E. Welf, P. Roudot, and C. Kervrann, "A sparse-to-dense method for 3D optical flow estimation in 3D light-microscopy image sequences," in *2018 IEEE 15th International Symposium on Biomedical Imaging (ISBI)*. IEEE, 2018, pp. 952–956.

[16] P. Gu, J. Han, D. Z. Chen, and C. Wang, "Scalar2vec: Translating scalar fields to vector fields via deep learning," in *2022 IEEE 15th Pacific Visualization Symposium (PacificVis)*. IEEE, 2022, pp. 31–40.

[17] Z. Teed and J. Deng, "RAFT-3D: Scene flow using rigid-motion embeddings," in *Proc. IEEE C. Comp. Vis. Pattern Rec. (CVPR)*, 2021, pp. 8375–8384.
[18] E. Ilg, N. Mayer, T. Saikia, M. Keuper, A. Dosovitskiy, and T. Brox, "Flownet 2.0: Evolution of optical flow estimation with deep networks," in *Proc. IEEE C. Comp. Vis. Pattern Rec. (CVPR)*, 2017, pp. 2462–2470.
[19] D. Sun, X. Yang, M.-Y. Liu, and J. Kautz, "PWC-Net: CNNs for optical flow using pyramid, warping, and cost volume," in *Proc. IEEE C. Comp. Vis. Pattern Rec. (CVPR)*, 2018, pp. 8934–8943.
[20] D. Maurer and A. Bruhn, "Proflow: Learning to predict optical flow," *arXiv preprint arXiv:1806.00800*, 2018.
[21] T. Xue, B. Chen, J. Wu, D. Wei, and W. T. Freeman, "Video enhancement with task-oriented flow," *Int. J. Comp. Vis.*, pp. 1106–1125, 2019.
[22] S. Baker, D. Scharstein, J. Lewis, S. Roth, M. J. Black, and R. Szeliski, "A database and evaluation methodology for optical flow," *Int. J. Comp. Vis.*, vol. 92, pp. 1–31, 2011.
[23] K. Soomro, A. R. Zamir, and M. Shah, "A dataset of 101 human action classes from videos in the wild," *Center for Research in Computer Vision*, vol. 2, no. 11, 2012.
[24] W. Bao, W.-S. Lai, X. Zhang, Z. Gao, and M.-H. Yang, "Memc-net: Motion estimation and motion compensation driven neural network for video interpolation and enhancement," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 43, no. 3, pp. 933–948, 2019.
[25] J. J. Yu, A. W. Harley, and K. G. Derpanis, "Back to Basics: Unsupervised learning of optical flow via brightness constancy and motion smoothness," in *European Conference on Computer Vision*. Springer, 2016, pp. 3–10.
[26] C. Ledig, L. Theis, F. Huszár, J. Caballero, A. Cunningham, A. Acosta, A. Aitken, A. Tejani, J. Totz, Z. Wang *et al.*, "Photo-realistic single image super-resolution using a generative adversarial network," in *Proc. IEEE C. Comp. Vis. Pattern Rec. (CVPR)*, 2017, pp. 4681–4690.
[27] W. Shi, J. Caballero, F. Huszár, J. Totz, A. P. Aitken, R. Bishop, D. Rueckert, and Z. Wang, "Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network," in *Proc. IEEE C. Comp. Vis. Pattern Rec. (CVPR)*, 2016, pp. 1874–1883.
[28] C. Dong, C. C. Loy, K. He, and X. Tang, "Image super-resolution using deep convolutional networks," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 38, no. 2, pp. 295–307, 2015.
[29] Z. Li, J. Yang, Z. Liu, X. Yang, G. Jeon, and W. Wu, "Feedback network for image super-resolution," in *Proc. IEEE C. Comp. Vis. Pattern Rec. (CVPR)*, 2019, pp. 3867–3876.
[30] J. Liang, J. Cao, G. Sun, K. Zhang, L. Van Gool, and R. Timofte, "SwinIR: Image restoration using swin transformer," in *Proc. IEEE C. Comp. Vis. Pattern Rec. (CVPR)*, 2021, pp. 1833–1844.
[31] S. Meyer, O. Wang, H. Zimmer, M. Grosse, and A. Sorkine-Hornung, "Phase-based frame interpolation for video," in *Proc. IEEE C. Comp. Vis. Pattern Rec. (CVPR)*, 2015, pp. 1410–1418.
[32] S. Niklaus, L. Mai, and F. Liu, "Video frame interpolation via adaptive separable convolution," in *Proc. IEEE C. Comp. Vis. Pattern Rec. (CVPR)*, 2017, pp. 261–270.
[33] H. Jiang, D. Sun, V. Jampani, M.-H. Yang, E. Learned-Miller, and J. Kautz, "Super slomo: High quality estimation of multiple intermediate frames for video interpolation," in *Proc. IEEE C. Comp. Vis. Pattern Rec. (CVPR)*, 2018, pp. 9000–9008.
[34] J. Han, H. Zheng, D. Z. Chen, and C. Wang, "STNet: An end-to-end generative framework for synthesizing spatiotemporal super-resolution volumes," *IEEE Trans. Vis. Comp. Graph.*, vol. 28, pp. 270–280, 2021.
[35] J. Han and C. Wang, "TSR-TVD: Temporal super-resolution for time-varying data analysis and visualization," *IEEE Trans. Vis. Comp. Graph.*, vol. 26, no. 1, pp. 205–215, 2019.
[36] A. Mishra, S. Hazarika, A. Biswas, and C. Bryan, "Filling the Void: Deep learning-based reconstruction of sampled spatiotemporal scientific simulation data," *arXiv preprint arXiv:2205.12868*, 2022.
[37] J. Han and C. Wang, "SSR-TVD: Spatial super-resolution for time-varying data analysis and visualization," *IEEE Transactions on Visualization and Computer Graphics*, vol. 28, no. 6, pp. 2445–2456, 2020.
[38] C. Jiao, C. Bi, and L. Yang, "Ffeinr: Flow feature-enhanced implicit neural representation for spatio-temporal super-resolution," *arXiv preprint arXiv:2308.12508*, 2023.
[39] Q. Wu, D. Bauer, Y. Chen, and K.-L. Ma, "HyperINR: A fast and predictive hypernetwork for implicit neural representations via knowledge distillation," *arXiv preprint arXiv:2304.04188*, 2023.
[40] J. Han and C. Wang, "CoordNet: Data generation and visualization generation for time-varying volumes via a coordinate-based neural network," *IEEE Trans. Vis. Comp. Graph.*, 2022.
[41] K. Tang and C. Wang, "STSR-INR: Spatiotemporal super-resolution for multivariate time-varying volumetric data via implicit neural representation," *Computers & Graphics*, vol. 119, p. 103874, 2024.
[42] G. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," *arXiv preprint arXiv:1503.02531*, 2015.
[43] V. Vapnik and R. Izmailov, "Learning using privileged information: Similarity control and knowledge transfer," *J. Mach. Learning Res.*, vol. 16, pp. 2023–2049, 2015.
[44] D. Lopez-Paz, L. Bottou, B. Schölkopf, and V. Vapnik, "Unifying distillation and privileged information," in *Proc. Int. C. Learning Representations (ICLR)*, 2016, pp. 1–10.
[45] C. Hu, X. Li, D. Liu, X. Chen, J. Wang, and X. Liu, "Teacher-student architecture for knowledge learning: A survey," *arXiv preprint arXiv:2210.17332*, 2022.
[46] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification," in *Proc. IEEE Int. C. Comput. Vis. (ICCV)*, 2015, pp. 1026–1034.
[47] D. Sun, S. Roth, and M. J. Black, "A quantitative analysis of current practices in optical flow estimation and the principles behind them," *Int. J. Comp. Vis.*, vol. 106, no. 2, pp. 115–137, 2014.
[48] C. M. Bishop and N. M. Nasrabadi, *Pattern recognition and machine learning*. Springer, 2006, vol. 4, no. 4.
[49] Y. Bengio, A. Courville, and P. Vincent, "Representation learning: A review and new perspectives," *IEEE transactions on pattern analysis and machine intelligence*, vol. 35, no. 8, pp. 1798–1828, 2013.
[50] S. Niklaus and F. Liu, "Softmax splatting for video frame interpolation," in *Proc. IEEE C. Comp. Vis. Pattern Rec. (CVPR)*, 2020, pp. 5437–5446.
[51] I. Loshchilov, F. Hutter *et al.*, "Fixing weight decay regularization in adam," *arXiv preprint arXiv:1711.05101*, vol. 5, 2017.
[52] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.
[53] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*. MIT press, 2016.
[54] P. Mocz, "Lattice Boltzmann Python," GitHub repository, Accessed: 2023, https://github.com/pmocz/latticeboltzmann-python.
[55] J. Sexton, Z. Lukic, A. Almgren, C. Daley, B. Friesen, A. Myers, and W. Zhang, "Nyx: A massively parallel amr code for computational cosmology," *The J. of Open S. Software*, vol. 6, no. 63, p. 3068, 2021.
[56] W. He, J. Wang, H. Guo, K.-C. Wang, H.-W. Shen, M. Raj, Y. S. Nashed, and T. Peterka, "InSituNet: Deep image synthesis for parameter space exploration of ensemble simulations," *IEEE Trans. Vis. Comp. Graph.*, vol. 26, no. 1, pp. 23–33, 2019.
[57] C. Crameri, "The Turbo Colormap," 2018, dOI: 10.1038/s41467-019-11512-7.
[58] R. Zhang, P. Isola, A. A. Efros, E. Shechtman, and O. Wang, "The unreasonable effectiveness of deep features as a perceptual metric," in *Proc. IEEE C. Comp. Vis. Pattern Rec. (CVPR)*, 2018, pp. 586–595.
[59] H. Gadirov, G. Tkachev, T. Ertl, and S. Frey, "Evaluation and selection of autoencoders for expressive dimensionality reduction of spatial ensembles," in *ISVC*. Springer, 2021, pp. 222–234.
[60] T. Wang and H. Snoussi, "Detection of abnormal events via optical flow feature analysis," *Sensors*, vol. 15, no. 4, pp. 7156–7171, 2015.

Hamid Gadirov received his MSc degree in computer science from the University of Stuttgart, Germany in 2020. He is currently pursuing a PhD at the Bernoulli Institute at the University of Groningen, the Netherlands. His research focuses on machine learning for scientific visualization, with an emphasis on spatio-temporal data and ensemble analysis.

Jos Roerdink received his PhD degree in theoretical physics from the University of Utrecht in 1983. He is emeritus professor of Scientific Visualization and Computer Graphics at the University of Groningen, the Netherlands. His research interests include mathematical morphology and biomedical visualization.

Steffen Frey received his PhD degree in computer science from the University of Stuttgart, Germany in 2014. He is an assistant professor at the Bernoulli Institute at the University of Groningen, the Netherlands. His research interests are in visualization methods for increasingly large quantities of scientific data.

SUPPLEMENTARY MATERIAL

A. Scientific Ensembles

Examples of fields at different timesteps from various members of both the LBS and Droplets ensembles can be found in Figs. 9 and 10. These figures offer a visual representation of the data from simulations and physical experiments, which are used in our research. One can observe that Fig. 9 illustrates the considerable structural diversity that exists between the density and flow fields of the LBS ensemble, highlighting the intricate variations within these fields.

B. Density Interpolation and Flow Estimation

Illustrative results showcasing FLINT's performance at various interpolation rates for both the LBS and Nyx ensembles are provided in Figs. 11, 12, and 13. These results affirm the positive outcomes in terms of density interpolation and flow estimation achieved at the interpolation rates of 3 and 8, indicating the method's reliability in these scenarios. However, it's worth noting that at a very high rate of 32, as can be seen in Figs. 12d and 13, although density interpolation can still maintain satisfactory quality, there becomes a noticeable degradation in the quality of flow estimation, which is not surprising due to the large variation in structure between the two fields. In Fig. 12d, we observe significantly larger errors for flow estimation compared to lower interpolation rates, highlighting the challenges posed by such high rates. Similarly, in Fig. 13, while the flow retains its overall structure, the accuracy in finer details, particularly around the circular swirling patterns, is noticeably reduced.

C. Comparison Against Baselines

The flow estimated by FLINT achieves much better results on the test set consisting of one unseen ensemble member compared to the baseline RIFE method. The comparison examples are shown in Fig. 14.

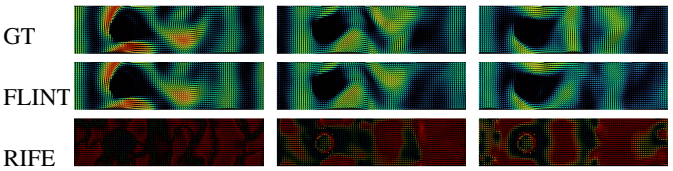


Fig. 14: LBS examples of flow results comparison, rows from top to bottom: GT, FLINT, and RIFE flow, respectively.

In the context of the Droplets ensemble, the results presented in Fig. 15 provide evidence of FLINT's superior performance compared to linear interpolation. This observation holds true across a spectrum of interpolation rates, according to our expectations. FLINT also consistently surpasses the RIFE method in terms of both PSNR and, previously shown in the paper, LPIPS metrics.

In the context of the LBS ensemble, the results presented in Fig. 16 demonstrate FLINT's superior performance compared to both linear interpolation and the RIFE method, also when evaluated using the LPIPS metric. Across varying interpolation rates, FLINT consistently achieves lower LPIPS scores, signifying better perceptual similarity to the ground truth. These findings underscore FLINT's ability to preserve fine details and produce visually coherent interpolations, even under challenging conditions.

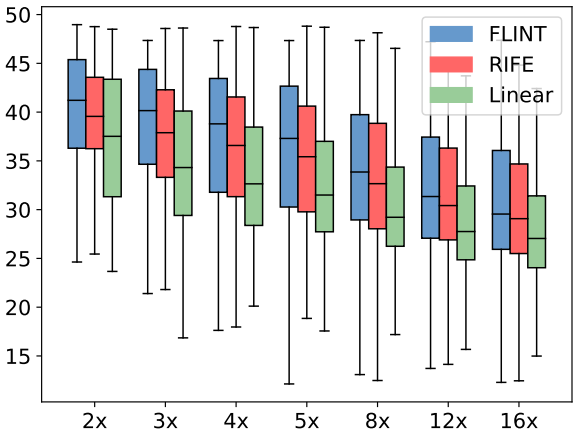


Fig. 15: Droplets ensemble: comparison of PSNR scores of FLINT, RIFE, and linear interpolation at various rates.

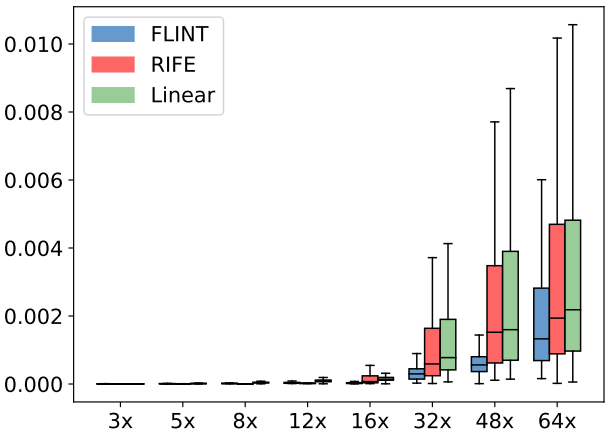
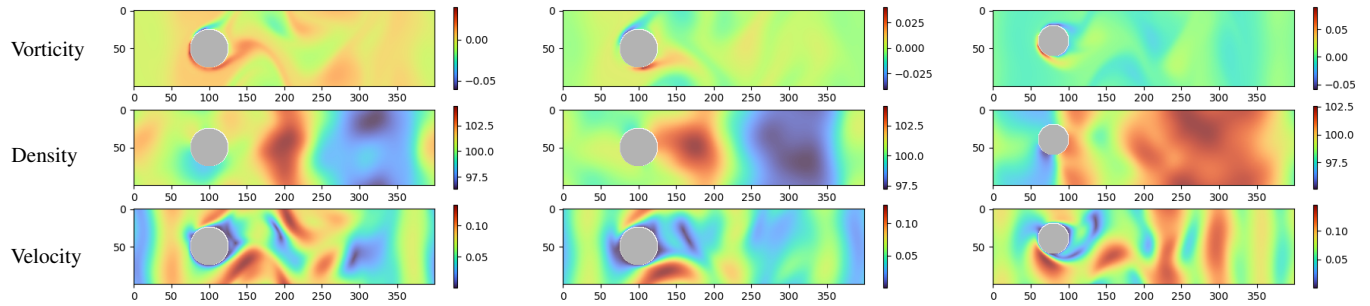


Fig. 16: LBS ensemble: comparison of LPIPS scores of FLINT, RIFE, and linear interpolation at various interpolation rates.

D. Noise Handling

FLINT's robustness to noisy data was evaluated in both 2D and 3D scientific ensembles. The 2D Droplets ensemble naturally contains noisy input data, as evidenced by the density plots in Fig. 5 top row, where background noise is clearly visible. Despite this inherent noise, FLINT successfully reconstructs density and flow fields, maintaining high accuracy and capturing the essential dynamics of the dataset.

Additionally, to further test FLINT's robustness, we introduced random Gaussian noise into the 3D Nyx simulation ensemble. Specifically, we added noise with a mean of zero and a standard deviation of 0.025 to the density field (ranging within $[0, 1]$ after normalization). These parameters were selected to simulate realistic imperfections without overwhelming the dataset, allowing us to assess FLINT's ability to maintain accuracy under noisy conditions. The results, illustrated in Fig. 17 (second row corresponds to the GT density with added noise), demonstrate that FLINT's performance does not significantly degrade under these conditions. The model continues to produce accurate density and flow estimations (third and fifth rows, respectively), even in the presence of added noise. These findings highlight FLINT's capability to handle real-world imperfections in data, making it a reliable solution for noisy scientific ensembles.



(a) member 1, $t = 9000$ (b) member 2, $t = 9000$ (c) member 3, $t = 9000$

Fig. 9: LBS samples from three different ensemble members at $t = 9000$: vorticity, density, and velocity fields.

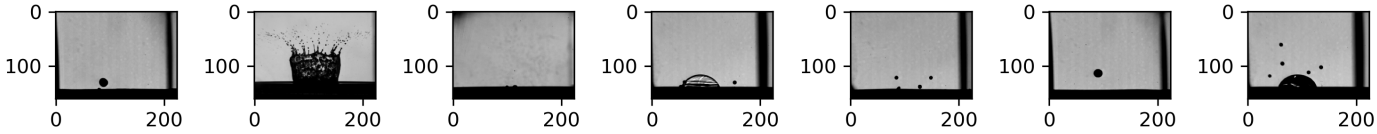


Fig. 10: Drop Dynamics samples from different ensemble members at different timesteps.

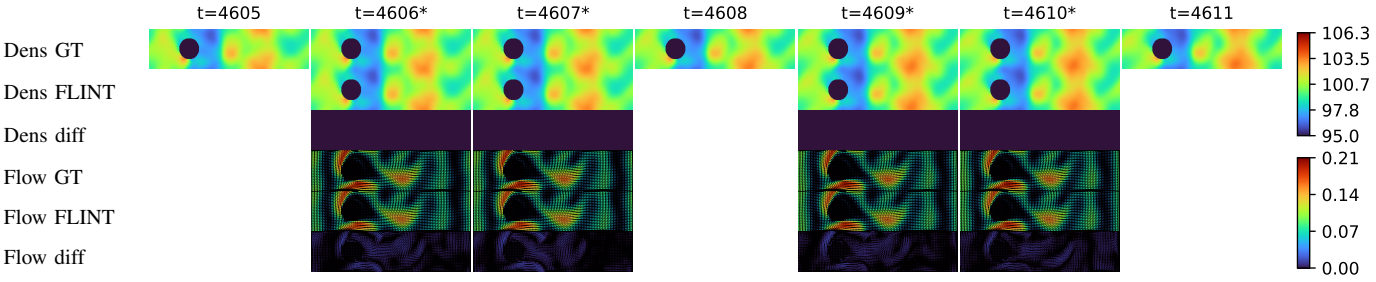
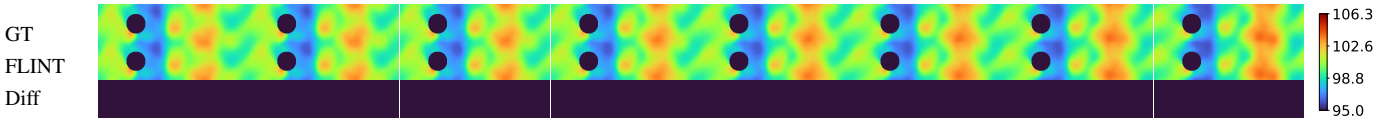
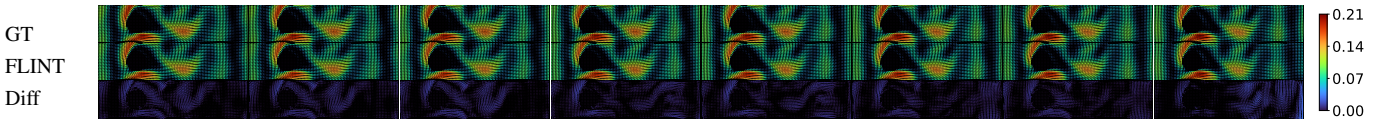


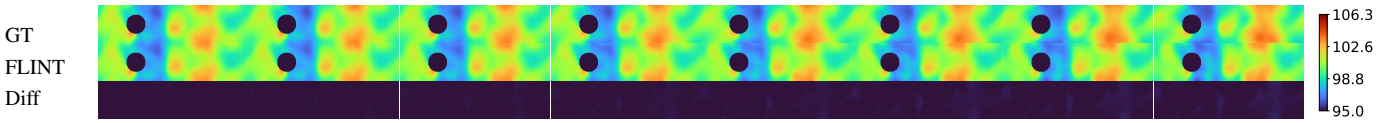
Fig. 11: FLINT results for LBS ensemble, 3× interpolation: labels of the rows (on the left) from top to bottom: density GT, interpolation, difference; flow GT, estimation, and difference; colorbars: top - density, bottom - flow (on the right). The use of an asterisk symbol (*) following the timestep numbers indicates the density interpolation and flow estimation (second and fifth rows) carried out at that particular timestep. These can be compared against the actual GT fields (first and fourth rows).



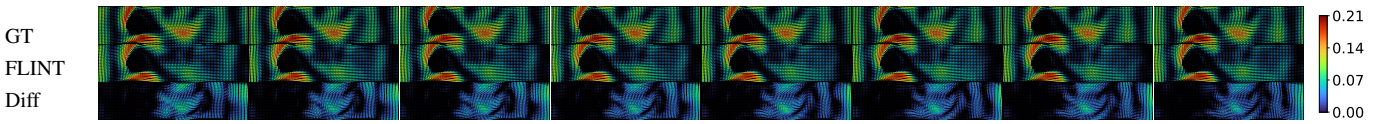
(a) Density interpolation 8×: comparing FLINT estimations against GT, from $t = 4606$. Rows from top to bottom: GT, interpolation, and difference.



(b) Flow estimation 8×: comparing FLINT estimations against GT, from $t = 4606$. Rows from top to bottom: GT, flow estimation, and difference.



(c) Density interpolation 32×: comparing FLINT estimations against GT, from $t = 4606$. Rows from top to bottom: GT, interpolation, and difference.



(d) Flow estimation 32×: comparing FLINT estimations against GT, from $t = 4606$. Rows from top to bottom: GT, flow estimation, and difference.

Fig. 12: LBS ensemble: Comparison of FLINT estimations at 8× and 32× interpolation. For both density interpolation and flow estimation, results are compared against the ground truth (GT). Each row shows: GT, FLINT estimations, and the difference.

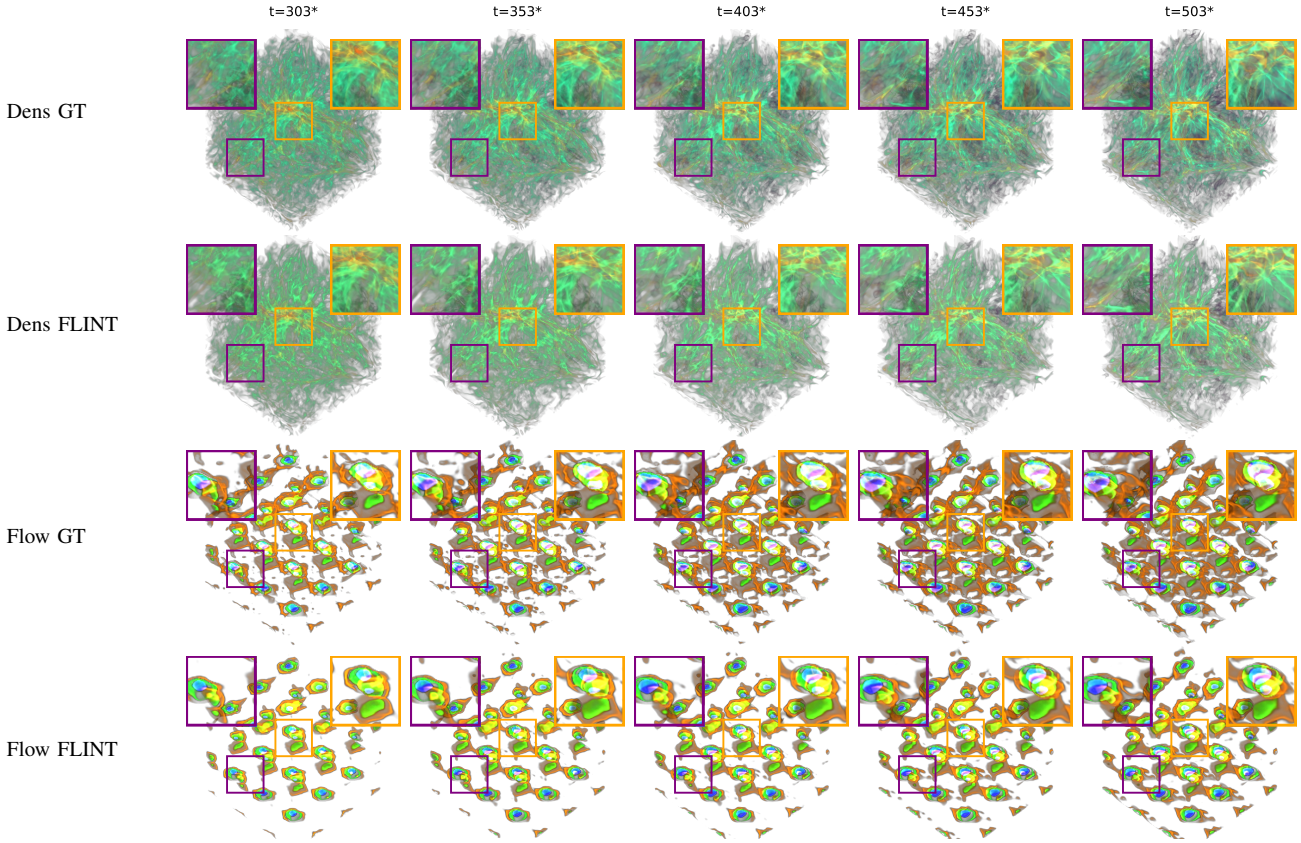
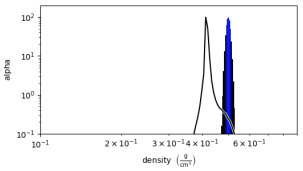


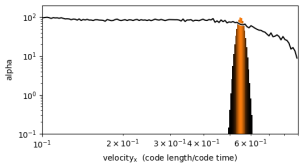
Fig. 13: Nyx: FLINT flow field estimation and temporal density interpolation during inference, 32×. From top to bottom, the rows show GT density, FLINT interpolated density, GT flow, and FLINT flow estimation. 3D rendering was used for the density and flow field visualization (● ● ● colors representing x, y, and z flow directions respectively).

E. Evaluation on Temperature and Flow Fields

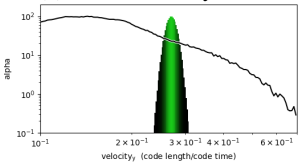
In Fig. 18, we present qualitative results for the Nyx simulation ensemble, 5×. The temperature interpolation demonstrates good performance, with the overall structure and spatial coherence of the field being well preserved across timesteps. For the flow fields estimated from the temperature field, some deviations are observed compared to the GT. These deviations are expected, as predicting flow from temperature is inherently more challenging due to the weaker relationship between temperature distributions and physical motion. Despite this, the results still show FLINT's capability to generate reasonable flow estimations even from scalar fields like temperature, which are less directly correlated with movement.



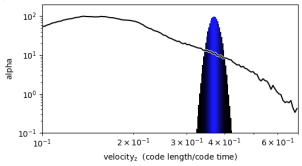
(a) TF for density field.



(b) TF for flow field in x direction.



(c) TF for flow field in y direction.

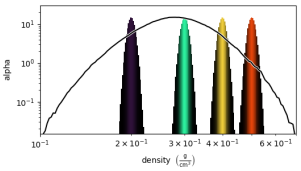


(d) TF for flow field in z direction.

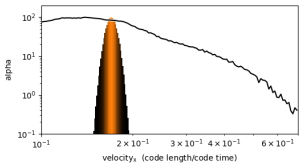
Fig. 19: Transfer functions for density and x, y, z components of the flow field, 5Jets.

F. 5Jets and Nyx Transfer Functions

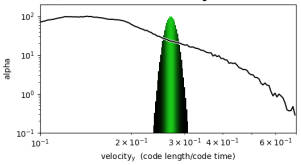
The transfer functions used for visualizing the density field and the x, y, z components of the flow field for the 5Jets and Nyx dataset are illustrated in Figs. 19 and 20. These transfer functions were selected experimentally to ensure that the rendered volumes are the most representative of the underlying data, effectively highlighting the key structures and dynamics.



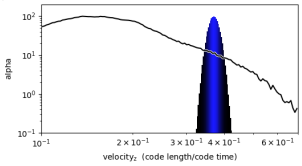
(a) TF for density field.



(b) TF for flow field in x direction.



(c) TF for flow field in y direction.



(d) TF for flow field in z direction.

Fig. 20: Transfer functions for density and x, y, z components of the flow field, Nyx.

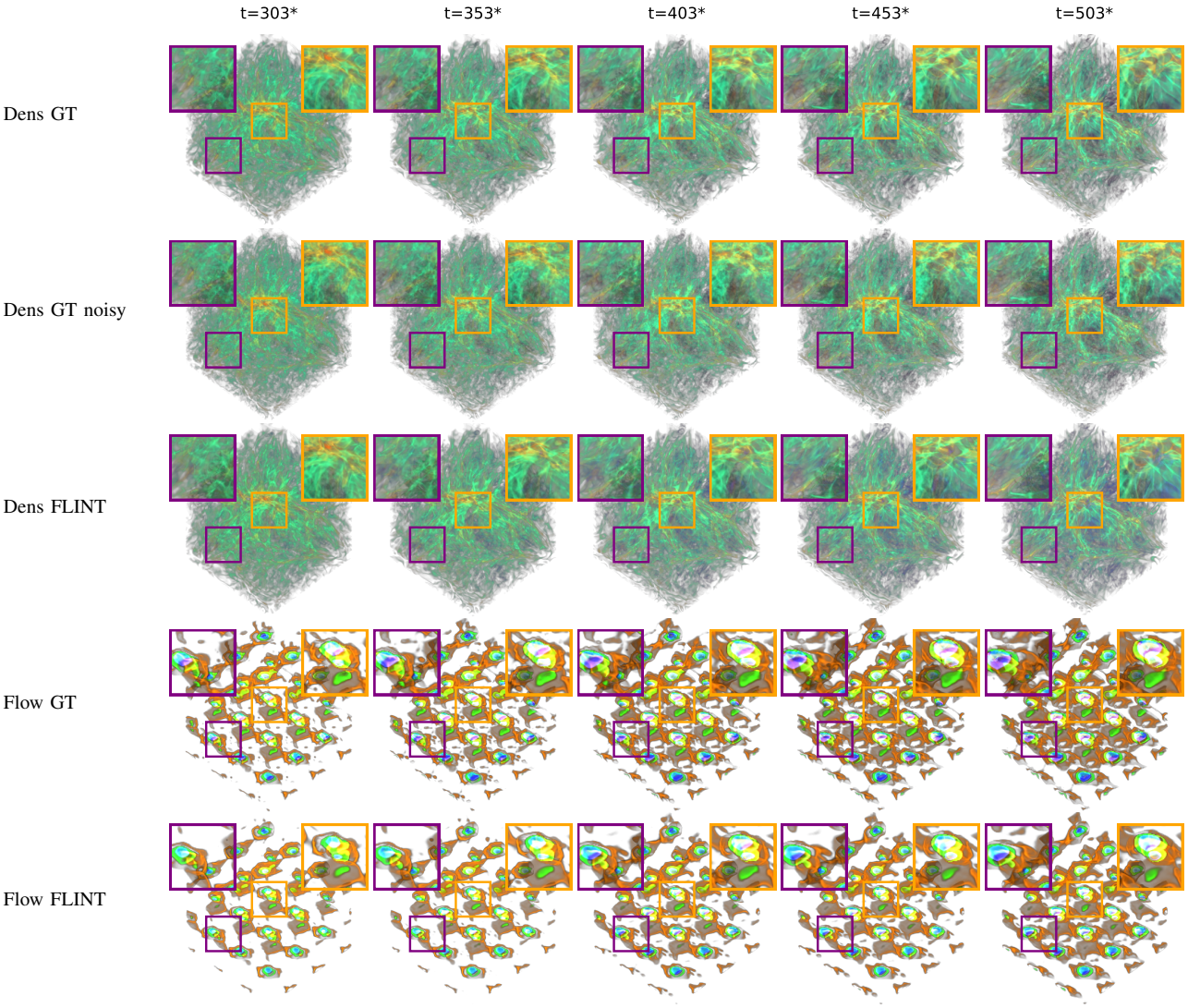


Fig. 17: Nyx: FLINT flow field estimation and temporal density interpolation during inference, 5×. From top to bottom, the rows show GT density, GT density with added noise, FLINT interpolated density, GT flow, and FLINT flow estimation. 3D rendering was used for the density and flow field visualization (● ● ● colors representing x, y, and z flow directions respectively).

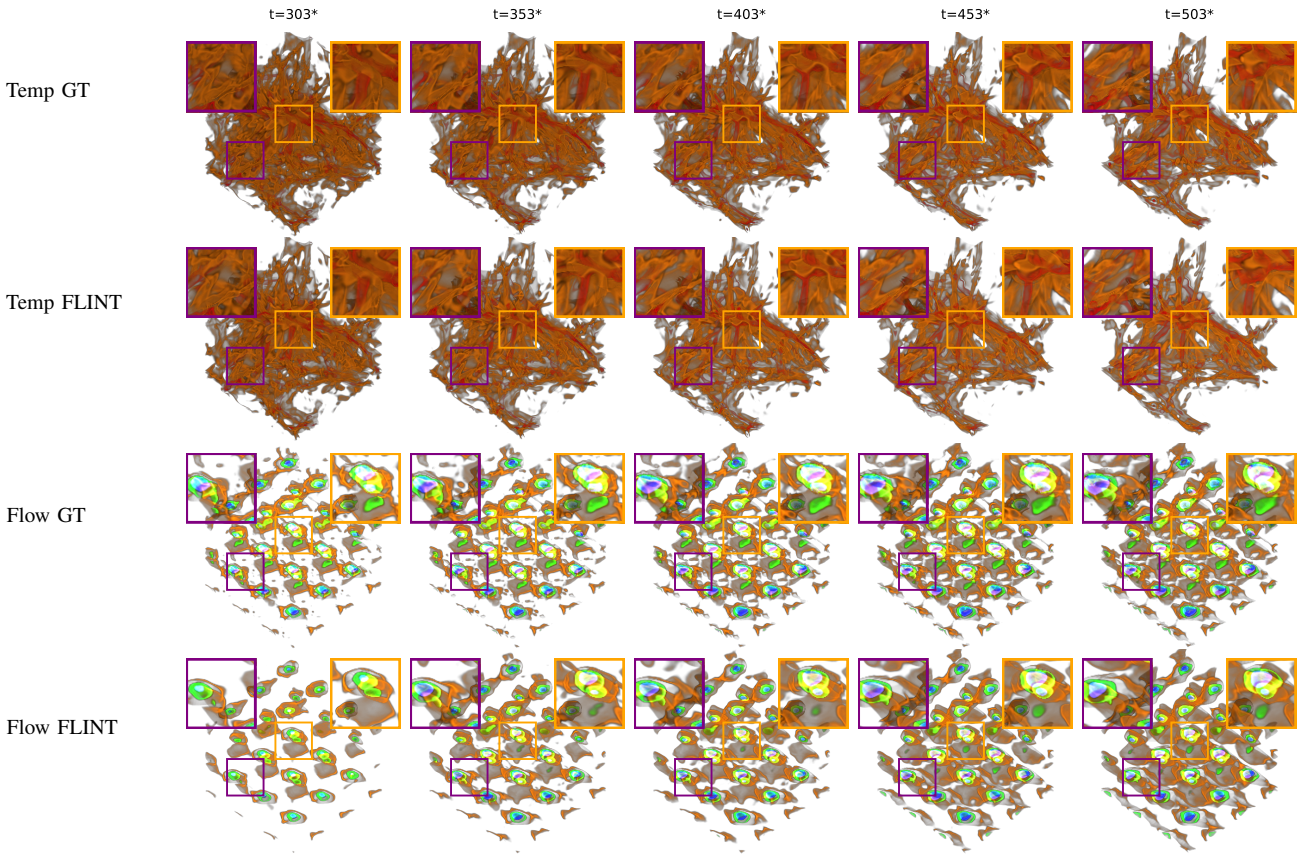


Fig. 18: Nyx: FLINT flow field estimation and temporal temperature field interpolation during inference, 5×. From top to bottom, the rows show GT temperature, FLINT interpolated temperature, GT flow, and FLINT flow estimation. 3D rendering was used for the temperature and flow field visualization (● ● ● colors representing x, y, and z flow directions respectively).