SSR-TVD: Spatial Super-Resolution for Time-Varying Data Analysis and Visualization

Jun Han and Chaoli Wang, *Senior Member, IEEE*

Abstract—We present SSR-TVD, a novel deep learning framework that produces coherent spatial super-resolution (SSR) of time-varying data (TVD) using adversarial learning. In scientific visualization, SSR-TVD is the first work that applies the generative adversarial network (GAN) to generate high-resolution volumes for three-dimensional time-varying data sets. The design of SSR-TVD includes a generator and two discriminators (spatial and temporal discriminators). The generator takes a low-resolution volume as input and outputs a synthesized high-resolution volume. To capture spatial and temporal coherence in the volume sequence, the two discriminators take the synthesized high-resolution volume(s) as input and produce a score indicating the realness of the volume(s). Our method can work in the in situ visualization setting by downscaling volumetric data from selected time steps as the simulation runs and upscaling downsampled volumes to their original resolution during postprocessing. To demonstrate the effectiveness of SSR-TVD, we show quantitative and qualitative results with several time-varying data sets of different characteristics and compare our method against volume upscaling using bicubic interpolation and a solution solely based on CNN.

Index Terms—Time-varying data visualization, deep learning, super-resolution, generative adversarial network.

1 Introduction

We present a novel deep learning framework for generating spatial super-resolution (SSR) of time-varying data (TVD). That is, given a low-resolution sequence with each volume, for example, of size $128 \times 128 \times 128$, we aim to generate the high-resolution sequence with each volume, for instance, of size $512 \times 512 \times 512$.

In scientific applications, upscaling low-resolution volumes to high-resolution ones is meaningful because of the following reasons. First, SSR can fit into the compression pipeline where data is compressed then decompressed in the scientific data analysis pipeline. Namely, scalar fields are downscaled at simulation time and upscaled to the original size during postprocessing. Moreover, SSR can achieve better visual quality and higher quantitative scores compared with state-of-the-art compression algorithms under the same setting (e.g., compression ratio, mean squared error). Second, with the SSR technique, the simulation can first run in high-resolution, generating high-resolution volumes for a certain number of time steps. It then runs in low-resolution, generating low-resolution volumes for subsequent time steps. During postprocessing, the high-resolution volumes are used for network training. Once trained, the network can then infer subsequent high-resolution volumes from the low-resolution ones. Such a scenario provides resource-saving at simulation time and quick verification/steering of the underlying simulation. Third, large-scale scientific simulations often produce a temporal sequence with thousands of time steps but could only save a small amount of data (e.g., every hundredth time step) due to limited disk space and I/O speed. If the simulation data can be downsampled individually and an effective upscaling approach is in place to recover the downsampled data,then scientists can afford to save more temporally refined volumes (e.g., every tenth time step) to enable more accurate investigation of dynamic features of the underlying time-varying data set.

Generating a high-resolution volume sequence from a low-resolution volume sequence poses two challenges. The first challenge is that the high-resolution sequence should maintain good temporal coherence. Unlike single volume super-resolution, we must consider a new solution that explicitly takes into account temporal coherence for volume sequence super-resolution. The second challenge is that human perception should be considered in the evaluation of upscaled volumes. Common methods usually design a deterministic interpolation kernel (e.g., trilinear or bicubic interpolation) for volume upscaling. These interpolations are performed only conditioned on a static interpolation kernel rather than a dynamic one to process different interpolated regions, and therefore, may blur features and result in artifacts in the rendered image. Meeting the desired visual quality is difficult if a deterministic kernel is applied to different local regions.

To address the above challenges, we advocate a deep learning approach as the neural network can learn features and their relationships from low-resolution volumes non-uniformly and non-locally, thus producing high-resolution volumes with high quality. Inspired by the sparse coding framework and the image super-resolution technique, we propose SSR-TVD that includes a generative model and two discriminators for producing spatiotemporally coherent SSR of a sequence of volumes using three different learning losses. As shown in Figure 1 (a), in our experiments, the *training* set consists of *high-resolution* volumes at *earlier* time steps of the sequence, and the *testing* set consists of *low-resolution* volumes at *later* time steps of the sequence. The volumes in the training set are first downsampled as the low-resolution volumes. We then train our SSR-TVD by minimizing the loss function that considers adversarial loss,

• *J. Han and C. Wang are with the Department of Computer Science and Engineering, University of Notre Dame, Notre Dame, IN 46556. E-mail: {jhan5, chaoli.wang}@nd.edu.*

content loss, and feature loss. To demonstrate the effectiveness of our approach, we show quantitative and qualitative results with several time-varying data sets of different characteristics. We compare our SSR-TVD against the widely-used BI and a solution solely based on CNN. We show that our method can achieve a better volume quality at the *data*, *image*, *feature*, and *perception* levels. The third row of Figure 4 shows our SSR-TVD results. The low-resolution volumes are obtained by downsampling the original volumes using the bicubic kernel with a downsampling factor of four in each dimension. During inference, SSR-TVD upscales the volumes back to the original resolution while achieving superior visual quality compared with other methods.

The contributions of this paper are as follows. First, our work applies *generative adversarial networks* (GANs) for generating SSR from scalar volume sequences, including multivariate cases. Second, we propose a novel generative adversarial architecture for upscaling volume data, which differs from the architectures commonly used in image and video super-resolution tasks. Third, we conduct an extensive comparison of SSR-TVD against BI and CNN using the data, image, feature, and perception quality metrics. Fourth, we investigate several hyperparameter settings and analyze how they could impact the performance of SSR-TVD.

2 Related Work

Deep learning for scientific visualization. Deep learning solutions have swept across the scientific visualization community to solve different volume and flow visualization tasks. For volume visualization, Zhou et al. [48] presented a deep learning solution for upscaling volume data, which preserves better local details and achieve higher quantitative values (e.g., SSIM and PSNR). Cheng et al. [3] presented a CNN-based system, which can discover and detect complex structures for better volume data understanding and visualization. Berger et al. [1] trained a generative model to simulate the volume rendering process using different viewpoints and transfer functions as input. Hong et al. [16] combined GAN and CNN to enable the exploration of volumetric data sets without explicit transfer function and rendering pipeline. He et al. [14] designed InSituNet that enables interactive posthoc exploration and analysis of ensemble simulations with parameter values never run before through the use of a network that trains pre-generated rendering images offline. Han and Wang [10] presented TSR-TVD, a deep learning solution for interpolating missing intermediate time steps for analyzing and visualizing time-varying data. Porter et al. [33] applied a CNN-like autoencoder to simultaneously learn features from multiple variables over time for selecting representative time steps. Wang et al. [40] proposed DeepOrganNet, an end-to-end deep neural net that generates fully high-fidelity 3D/4D organ geometric models from single-view medical images. Weiss et al. [43] designed an image-space solution that upscales a low-resolution isosurface to a high-resolution one.

For flow visualization, Hong et al. [17] used a recurrent neural network to optimize the process of data allocation data in particle tracing, which improves the I/O latency in distributed and parallel flow visualization. Han et al. [8] designed an autoencoder to extract latent features from streamlines and stream surfaces and applied dimensionality reduction and user-assisted grouping for representative selection from these features. Han et al. [9] proposed a two-stage machine learning solution for vector field reconstruction that given a set of streamlines, low-resolution vectors are first recovered and then upscaled to high resolution with high quality. Xie et al. [44] proposed tempoGAN that synthesizes SSR volume sequences where additional information (e.g., velocity and vorticity fields) is utilized during training to enhance temporal coherence. Guo et al. [7] designed SSR-VFD that produces coherent SSR of 3D vector field data with a scaling factor of 4 or 8 in each dimension.

Our work is similar to scalar field upscaling [48], vector field reconstruction [9], SSR-VFD [7], tempoGAN [44], and TSR-TVD [10]. Zhou et al. [48] and Han et al. [9] used CNNs to generate a high-resolution volume from a low-resolution one and only considered a single scalar or vector field. SSR-VFD only considers spatial coherence but not temporal coherence. As for tempoGAN, it uses additional information (velocity and/or vorticity) to ensure temporal coherence. In contrast, we use GAN in our network design and achieve spatiotemporally coherent volume sequence super-resolution for TVD without considering additional information. TSR-TVD and SSR-TVD both use a generative framework and follow similar loss function design. However, to ensure temporal coherence, TSR-TVD explicitly includes a temporal component with multiple ConvLSTM layers, while SSR-TVD uses a temporal discriminator similar to that presented in tempoGAN. Furthermore, TSR-TVD upscales the temporal dimension while SSR-TVD upscales the spatial dimensions.

Image and video super-resolution. Deep learning has achieved great success in image and video super-resolution. For images, Dong et al. [4] proposed an autoencoder that upscales single images with an upscaling factor of three (i.e., the scaled image is nine times of the original one in size). Ledig et al. [23] established a GAN with residual blocks to infer photo-realistic natural images for an upscaling factor of four. Zhang et al. [46] proposed a deep CNN with a channel attention mechanism to achieve better visual single image super-resolution results. For videos, Sajjadi et al. [37] utilized the early synthesized high-resolution frames to predict the subsequent frames through a recurrent video super-resolution solution. This treatment yields temporally-consistent results while reducing the computational cost. Jo et al. [20] proposed a DNN that generates dynamic upsampling filters and a residual image so that the low-resolution image can utilize the dynamic filter to generate a high-resolution image directly, and the computed residual can refine the structural details. Pérez-Pellitero et al. [32] established an adversarial recurrent network for video upscaling, where a temporally-consistent loss is computed through estimating the optical flow of two frames to guarantee the temporal coherence among different frames.

Our work is different from the above works. To estimate an accurate optical flow, researchers need to label a lot of optical flows to train a deep learning model. However, for volumetric data, this means that we need to label optical flow and train the corresponding model for *each* data set, which is time-consuming and impractical. Therefore, instead of utilizing optical flow, we leverage a temporal discriminator

to guarantee temporal coherence, which does not require an additional model to calculate extra information from the volumes. Besides, it also reduces training complexity. As for RNN, it is difficult and time-consuming to train since the gradient is calculated based on time and easy to vanish. Instead of using optical flow and RNNs to maintain temporal coherence, we propose a generative model that uses spatial and temporal discriminators to guarantee spatial and temporal coherence among different volumes.

Conditional GANs. GANs were introduced by Goodfellow et al. [5]. In a GAN model, two networks play a competitive game. Namely, one network (i.e., generator) generates data samples from observations to be undistinguishable with the real data. In contrast, the other network (i.e., discriminator) focus on judging the realness between the real and fake samples. As for conditional GANs (cGANs) [28], the generator synthesizes data from observations rather than noise. cGANs have been applied in video prediction [27], image synthesis [35], and photo generation [45]. Several works [19], [23], [31] have also used cGANs for image-to-image translations. But instead of only considering adversarial loss, other losses (such as L_2 regression, perceptual loss) are applied to require the output to be conditioned on the input. These works have achieved impressive results on image inpainting [31], image super-resolution [23], and future state prediction [47].

Our method differs from the above works in several aspects. First, we propose a novel module that combines residual block [13] and skip connection [36] to enhance the performance of SSR-TVD. Second, we leverage several techniques to stabilize SSR-TVD training, including a new feature loss and spectral normalization [29]. Third, we apply GAN to generate volume sequence super-resolution for 3D time-varying data sets.

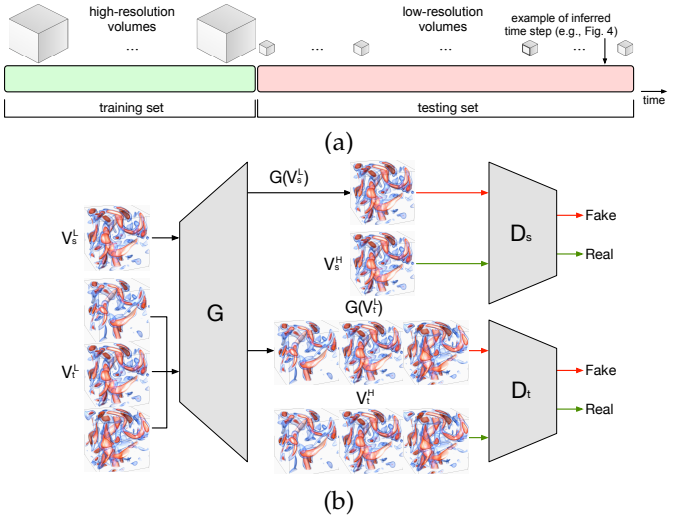


Fig. 1: (a) The training and testing sets from the volume sequence. (b) Overview of SSR-TVD. During training, a generator (G) is guided by two discriminator networks: one focuses on the spatial aspect (D_s) and the other focuses on the temporal aspect (D_t). During inference, only G is kept while D_s and D_t are discarded.

3 SSR-TVD

We aim to estimate a mapping function $\mathcal{F}$ from a low-resolution volume sequence $\mathbf{V}^{\mathrm{L}}$ to a high-resolution volume sequence $\mathbf{V}^{\mathrm{H}}$, while taking into account temporal coherence. Namely, $\mathbf{V}^{\mathrm{H}} = \mathcal{F}(\mathbf{V}^{\mathrm{L}})$. As shown in Figure 1 (b), SSR-TVD consists of three networks: a generator G, a spatial discriminator D_s, and a temporal discriminator D_t. G takes V^{L} as input, and output $G(V^{\mathrm{L}})$, a synthesized high-resolution volume. D_s takes V^{H} or $G(V^{\mathrm{L}})$ as input, and produces a score which indicates the realness of the input. Ideally, D_s will output 1 or 0 if the input is V^{H} or $G(V^{\mathrm{L}})$, respectively. As for D_t, the input is three consecutive volumes (i.e., a volume subsequence at three consecutive time steps), and the output is a score that describes the realness of these volumes. We choose three instead of five or more time steps to keep the computational cost low. We organize this section as follows. First, the design idea of SSR-TVD is offered, including the loss functions and the architectures of G, D_s, and D_t. Then, we will detail how to optimize SSR-TVD and stabilize the training process.

3.1 Loss Function

Notations. Let us denote $\mathbf{V}_s^{\mathrm{L}} = \{V_1^{\mathrm{L}}, \cdots, V_n^{\mathrm{L}}\}$ as a set of low-resolution volumes where V_i^{L} is the low-resolution volume at the ith time step, and $\mathbf{V}_s^{\mathrm{H}} = \{V_1^{\mathrm{H}}, \cdots, V_n^{\mathrm{H}}\}$ as a set of high-resolution volumes where V_i^{H} is the high-resolution counterpart of V_i^{L}. L, H, W are the dimensions of high-resolution volumes We also denote $\mathbf{V}_t^{\mathrm{L}} = \{(V_1^{\mathrm{L}}, V_2^{\mathrm{L}}, V_3^{\mathrm{L}}), \cdots, (V_{n-2}^{\mathrm{L}}, V_{n-1}^{\mathrm{L}}, V_n^{\mathrm{L}})\}$ and $\mathbf{V}_t^{\mathrm{H}} = \{(V_1^{\mathrm{H}}, V_2^{\mathrm{H}}, V_3^{\mathrm{H}}), \cdots, (V_{n-2}^{\mathrm{H}}, V_{n-1}^{\mathrm{H}}, V_n^{\mathrm{H}})\}$. Let θ_G, θ_{D_s} and θ_{D_t} be the learnable parameters in G, D_s, and D_t, respectively.

Adversarial loss. Following Mao et al. [26], we define the adversarial loss for G, D_s, and D_t as follows

$$\begin{aligned}\min_{\theta_G} \mathcal{L}_G =& \mathbb{E}_{V\in\mathbf{V}_s^{\mathrm{L}}}[(D_s(G(V))-1)^2]+ \\ & \mathbb{E}_{V\in\mathbf{V}_t^{\mathrm{L}}}[(D_t(G(V))-1)^2],\end{aligned} \tag{1}$$

$$\begin{aligned}\min_{\theta_{D_s}} \mathcal{L}_{D_s} =& (\mathbb{E}_{V\in\mathbf{V}_s^{\mathrm{H}}}[(D_s(V)-1)^2]+ \\ & \mathbb{E}_{V\in\mathbf{V}_s^{\mathrm{L}}}[(D_s(G(V)))^2])/2,\end{aligned} \tag{2}$$

$$\begin{aligned}\min_{\theta_{D_t}} \mathcal{L}_{D_t} =& (\mathbb{E}_{V\in\mathbf{V}_t^{\mathrm{H}}}[(D_t(V)-1)^2]+ \\ & \mathbb{E}_{V\in\mathbf{V}_t^{\mathrm{L}}}[(D_t(G(V)))^2])/2,\end{aligned} \tag{3}$$

where $\mathbb{E}[\cdot]$ denotes the expectation operation.

Following the above equations, a generative model G can learn to fool discriminators D_s and D_t. D_s is trained to distinguish super-resolved volumes from real volumes in the spatial aspect, and D_t is trained to distinguish super-resolved volumes in the temporal aspect. Through adversarial learning, G can learn a mapping from low-resolution volumes to high-resolution ones, and these synthesize high-resolution volumes are highly similar to real volumes and thus difficult to classify by D_s and D_t. This approach also encourages perceptual-driven quality in contrast to generating super-resolution volumes through only minimizing voxelwise error measurements, such as L_1 loss.

Content loss. Recent works [19], [31] have suggested that adding a traditional loss, such as L_2 distance, can improve the training stability compared against purely adversarial

learning. The role of the discriminator keeps the same, but the generator aims to not only generate indistinguishable volumes but also achieve a low mean squared error compared with real data. Therefore, we explore this option by using L_2 distance to optimize SSR-TVD.

$$\mathcal{L}_{L_2} = \mathbb{E}_{V' \in \mathbf{V}_s^{\mathrm{L}}, V \in \mathbf{V}_s^{\mathrm{H}}}[||G(V') - V||_2], \quad (4)$$

where $||\cdot||_2$ denotes L_2 norm.

Feature loss. We also incorporate a novel feature loss based on the discriminator D_t. This loss constrains G to produce similar features at different scales compared with real volumes. Specifically, features of synthesized volumes from different convolutional (Conv) layers of D_t are extracted and matched to these intermediate representations of real data at the same scale. Let us denote the feature representation extracted from the kth Conv layer as F^k. Then the feature loss is calculated as

$$\mathcal{L}_{\mathrm{F}} = \mathbb{E}_{V' \in \mathbf{V}_s^{\mathrm{L}}, V \in \mathbf{V}_s^{\mathrm{H}}} \sum_{k=1}^{T-1} \frac{1}{N_k}[||F^k(V) - F^k(G(V'))||_2], \quad (5)$$

where T is the total number of Conv layers in D_s and N_k denotes the number of elements in the kth Conv layer. This feature loss shares a similar purpose as the perceptual loss [21], [39], which has shown useful for improving GAN training and perceptual quality.

Taking all losses into consideration, the final loss function for G is defined as

$$\begin{aligned}\min_{\theta_G} \mathcal{L}_G = &\lambda_1(\mathbb{E}_{V \in \mathbf{V}_s^{\mathrm{L}}}[(D_s(G(V)) - 1)^2] + \\ &\mathbb{E}_{V \in \mathbf{V}_t^{\mathrm{L}}}[(D_t(G(V)) - 1)^2]) + \lambda_2 \mathcal{L}_{L_2} + \lambda_3 \mathcal{L}_{\mathrm{F}},\end{aligned} \quad (6)$$

where λ_1, λ_2, and λ_3 are hyperparameters, which control the relative importance of these three terms (adversarial, content, and feature loss).

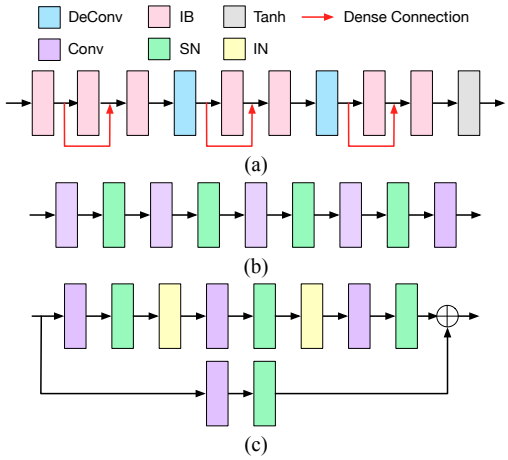


Fig. 2: Network architecture of SSR-TVD. (a) G, (b) D_s or D_t, and (c) IB. G has two DeConv layers and seven IBs. D_s and D_t share the same structure, while D_s takes one volume as input and D_t takes three consecutive volumes as input.

3.2 Network Architecture

Generator. As shown in Figure 2 (a), G takes V^{L} as input, and outputs V^{H}, a high-resolution version of V^{L}. To enhance the performance, we add two additional considerations in G: (1) combining low-level and high-level features through skip connection, (2) applying *spectral normalization* (SN) after each Conv layer. The skip connection connects feature maps from previous layers to the following layers so that the gradient can be backpropagated through multiple paths. In a residual block, the input is operated with three Conv layers without changing the spatial size, and the result of one Conv layer is added to the original input as a residual operation. These considerations effectively reduce gradient vanishing and allow us to build deeper networks.

The core of the generator G lies in the *identical blocks* (IBs). Each IB has two learning paths: one consists of four Conv layers appended by SN, *instance normalization* (IN) [38], and ReLU [30], and the other one contains one Conv layer appended by SN. These two paths are bridged by *skip connection*, as shown in Figure 2 (c). We set the kernel size to $3 \times 3 \times 3$, padding with 1 for all Conv layers in all IBs. Specifically, three IBs are applied to the low-resolution volume to extract semantic information, then two deconvolutional (DeConv) layers are utilized to upscale the low-resolution learned features. After each DeConv layer, we apply two IBs to refine the features. Besides, we use dense connection [18] to keep the information flowing smoothly in SSR-TVD. In the final IB layer, $\tanh(\cdot)$ is applied. In Table 1, we list the kernel size and the number of feature maps in each learning layer.

TABLE 1: Architecture parameter details of G.

type	kernel size	output channels	output size
input	N/A	1	$L/4 \times H/4 \times W/4$
IB+ReLU	3	16	$L/4 \times H/4 \times W/4$
IB+ReLU	3	64	$L/4 \times H/4 \times W/4$
IB+ReLU	3	128	$L/4 \times H/4 \times W/4$
DeConv+ReLU	4	128	$L/2 \times H/2 \times W/2$
IB+ReLU	3	128	$L/2 \times H/2 \times W/2$
IB+ReLU	3	64	$L/2 \times H/2 \times W/2$
DeConv+ReLU	4	32	$L \times H \times W$
IB+ReLU	3	8	$L \times H \times W$
IB+Tanh	3	1	$L \times H \times W$

Discriminators. To discriminate real volumes from synthesized ones in spatial and temporal aspects, we train two discriminator networks (D_s and D_t) to distinguish spatial difference and temporal difference. As sketched in Figure 2 (b), discriminators are composed of several Conv, SN layers and Leaky ReLU activation ($\alpha = 0.2$). In addition, following the guidelines in Radford et al. [34], we use the Conv operation to replace pooling layers throughout the network. Each D_s or D_t contains five Conv layers for downsampling the inputs, and the number of feature maps is set to 64, 128, 256, 512, and 1, respectively. We set the kernel size and stride to 4 and 2 to downscale the input for each dimension at each Conv layer except the final Conv layer. In the final Conv layer, it produces a $1 \times 1 \times 1$ dimensional output. We do not use the activation function in the last Conv layer. Note that D_s and D_t share the same structure but not the same weights, the only difference is that the input to D_s is one volume, while the input to D_t is three consecutive volumes (we concatenate V_{i-1}^{H}, V_i^{H}, and V_{i+1}^{H}).

Algorithm. As sketched in Algorithm 1, our SSR-TVD training algorithm consists of three parts: training D_S (spatial discriminator), training D_t (temporal discriminator), and training G (generator). The algorithm runs over a certain number of epochs T, and for each epoch, it trains the three networks successively. In order to generate high-

Algorithm 1 SSR-TVD training algorithm.

Require: initial generator parameters θ_G, initial spatial discriminator parameters θ_{D_s}, and initial temporal discriminator parameters θ_{D_t}
Require: number of D_s updates n_{D_s} per G iteration, number of D_t updates n_{D_t} per G iteration, number of G updates n_G per D_s or D_t iteration, number of training epochs T, learning rate α_G, α_{D_s}, and α_{D_t} for G, D_s, and D_t respectively
for $t = 1 \cdots T$ **do**
  for $i = 2 \cdots n-1$ **do**
    sample low-resolution data V_i^{L}, $(V_{i-1}^{\mathrm{L}}, V_i^{\mathrm{L}}, V_{i+1}^{\mathrm{L}})$ and high-resolution data V_i^{H}, $(V_{i-1}^{\mathrm{H}}, V_i^{\mathrm{H}}, V_{i+1}^{\mathrm{H}})$
    for $1 \cdots n_{D_s}$ **do**
      compute $\mathcal{L}_{D_s}$ based on Equation 2
      $\theta_{D_s} = \theta_{D_s} - \alpha_{D_s} \frac{\partial \mathcal{L}_{D_s}}{\partial \theta_{D_s}}$
    end for
    for $1 \cdots n_{D_t}$ **do**
      compute $\mathcal{L}_{D_t}$ based on Equation 3
      $\theta_{D_t} = \theta_{D_t} - \alpha_{D_t} \frac{\partial \mathcal{L}_{D_t}}{\partial \theta_{D_t}}$
    end for
    for $1 \cdots n_G$ **do**
      compute $\mathcal{L}_G$ based on Equation 6
      $\theta_G = \theta_G - \alpha_G \frac{\partial \mathcal{L}_G}{\partial \theta_G}$
    end for
  end for
end for

quality results, we should make the training process stable and avoid the model from collapsing, which means that the discriminators need to loop through the real data and synthesized data several times. Therefore, we usually set $n_{D_s} = n_{D_t} \geq n_G$.

3.3 Optimization

We study two techniques that stabilize the training of SSR-TVD. SN is applied in G as well as in D_s and D_t. Second, the *two time-scale update rule* (TTUR) [15] is leveraged since it can enhance the ability of discriminators in distinguishing real and fake volumes and reducing the computational cost.

SN. Miyato et al. [29] proposed SN, a weight normalization approach, that can stabilize the training of GANs. Unlike other normalization techniques, where additional hyperparameyers are required, SN only needs to set a spectral norm after Conv layers. In addition, SN can save computational cost during training. We also find that the generator can get benefits from adding SN because it can prevent model collapse and avoid unusual gradients.

TTUR. Regularization of discriminator [6] often slows down the process of GAN learning, because the generator can update once until regularized discriminators update multiple times (e.g., 5) during training. Heusel et al. [15] advocated using two different learning rates (i.e., TTUR) for generator and discriminator. The goal is to mitigate the problem of setting multiple updating times in a regularized discriminator per generator update, making it possible to use fewer discriminator updates per generator update. Thus, we utilize this technique in SSR-TVD optimization.

4 Results and Discussion

4.1 Data Sets and Network Training

We experimented with our approach using the data sets listed in Table 2. A single NVIDIA TESLA V100 GPU was used for training. We obtained the low-resolution volumes by downsampling the high-resolution volumes using the bicubic kernel with a downsampling factor of four and a constant padding. For each epoch, at each time step, four low-resolution sub-volumes and the corresponding high-resolution ones are cropped randomly. This cropping mechanism entails a low computational cost and requires less GPU memory. Note that we can apply the generative model to volumes of arbitrary size as it is fully convolutional. Because the value range for the output of the final activation function `tanh(·)` is $[-1, 1]$, we scaled the range of V^{L} input volumes to $[0, 1]$ and that of V^{H} volumes to $[-1, 1]$. For optimization, we initialized parameters in SSR-TVD using those suggested by He et al. [12] and applied the Adam optimizer [22] to update the parameters. We set one training sample per mini-batch. We set the learning rate with 4×10^{-4} for D_s and D_t, 10^{-4} for G, $\beta_1 = 0.5$, $\beta_2 = 0.999$. We set $\lambda_1 = 10^{-3}$, $\lambda_2 = 1$, and $\lambda_3 = 5 \times 10^{-2}$. We set ϵ to 10^{-4} in SN for numerical stability, and n_G, n_{D_s}, and n_{D_t} to 1, 2, and 2, respectively. All these hyperparameters are determined empirically.

TABLE 2: The dimensions, number of training epochs, and total training time of each data set.

data set (variable)	high-res dimension ($x \times y \times z \times t$)	epochs	training time (hours)
argon bubble	$640 \times 256 \times 256 \times 160$	500	30
combustion (HR)	$240 \times 360 \times 60 \times 100$	500	20
combustion (MF)	$240 \times 360 \times 60 \times 100$	500	20
combustion (YOH)	$240 \times 360 \times 60 \times 100$	500	20
five jets	$128 \times 128 \times 128 \times 100$	500	18
hurricane	$500 \times 500 \times 100 \times 48$	500	48
ionization (H)	$600 \times 248 \times 248 \times 100$	400	20
ionization (H+)	$600 \times 248 \times 248 \times 100$	400	20
ionization (He)	$600 \times 248 \times 248 \times 100$	400	20
ionization (He+)	$600 \times 248 \times 248 \times 100$	400	20
vortex	$128 \times 128 \times 128 \times 90$	400	16

TABLE 3: Average PSNR and SSIM values. The best ones are highlighted in bold.

	PSNR (dB)			SSIM		
data set (variable)	BI	CNN	SSR-TVD	BI	CNN	SSR-TVD
five jets	30.724	28.476	**40.145**	0.809	0.840	**0.867**
ionization (H+)	16.000	26.663	**38.584**	0.898	0.745	**0.902**
ionization (He+)	17.116	24.933	**34.738**	0.869	0.708	**0.870**
vortex	21.926	37.337	**40.795**	0.892	0.861	**0.923**

4.2 Results

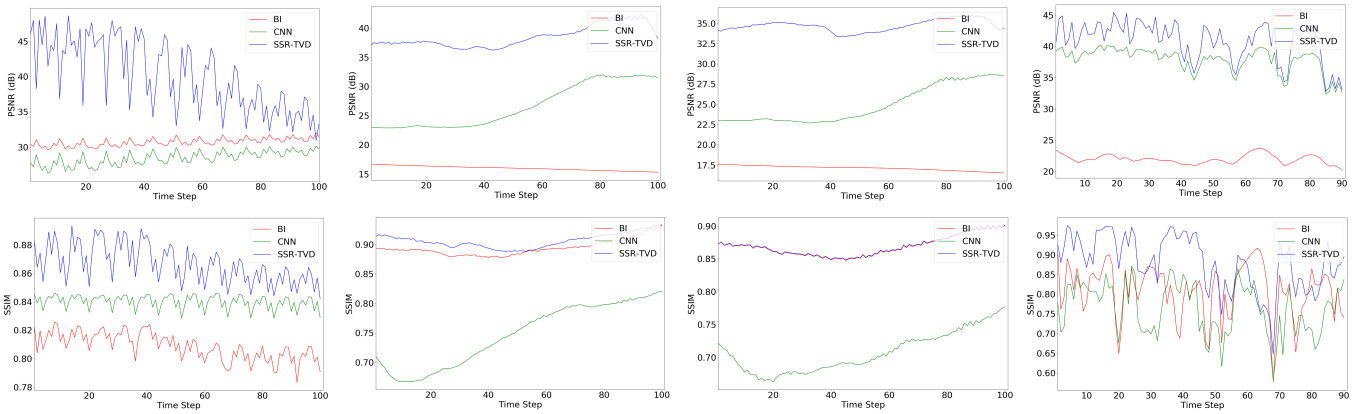
Due to the page limit, we could not always show SSR-TVD results for multiple time steps in the paper. These frame-to-frame comparison results are shown in the accompanying video. Unless otherwise stated, all visualization results presented in the paper for volumes synthesized by SSR-TVD are the inferred results (i.e., the network does not see these volumes during training). These inferred results

TABLE 4: Average IS values at selected isovalues. The best ones are highlighted in bold.

	BI		SSR-TVD	
data set (variable)	$v = -0.803$	$v = -0.608$	$v = -0.803$	$v = -0.608$
ionization (He+)	0.43	0.57	**0.76**	**0.80**
	$v = 0$	$v = 0.1$	$v = 0$	$v = 0.1$
vortex	0.82	0.85	**0.87**	**0.92**

TABLE 5: MOS values. The best ones are highlighted in bold.

	volume rendering			isosurface rendering		
data set (variable)	BI	CNN	SSR-TVD	BI	CNN	SSR-TVD
combustion (HR)	2.42	2.83	**3.57**	2.05	3.05	**3.90**
combustion (MF $\rightarrow$ YOH)	3.05	2.80	**3.38**	2.44	3.23	**3.73**
ionization (He+)	2.75	2.43	**4.08**	2.44	3.23	**3.73**
vortex	4.30	4.32	**4.38**	3.63	3.85	**4.18**



(a) five jets (b) ionization (H+) (c) ionization (He+) (d) vortex

Fig. 3: PSNR of synthesized volumes (top row) and SSIM of rendered images (bottom row) under BI, CNN, and SSR-TVD.

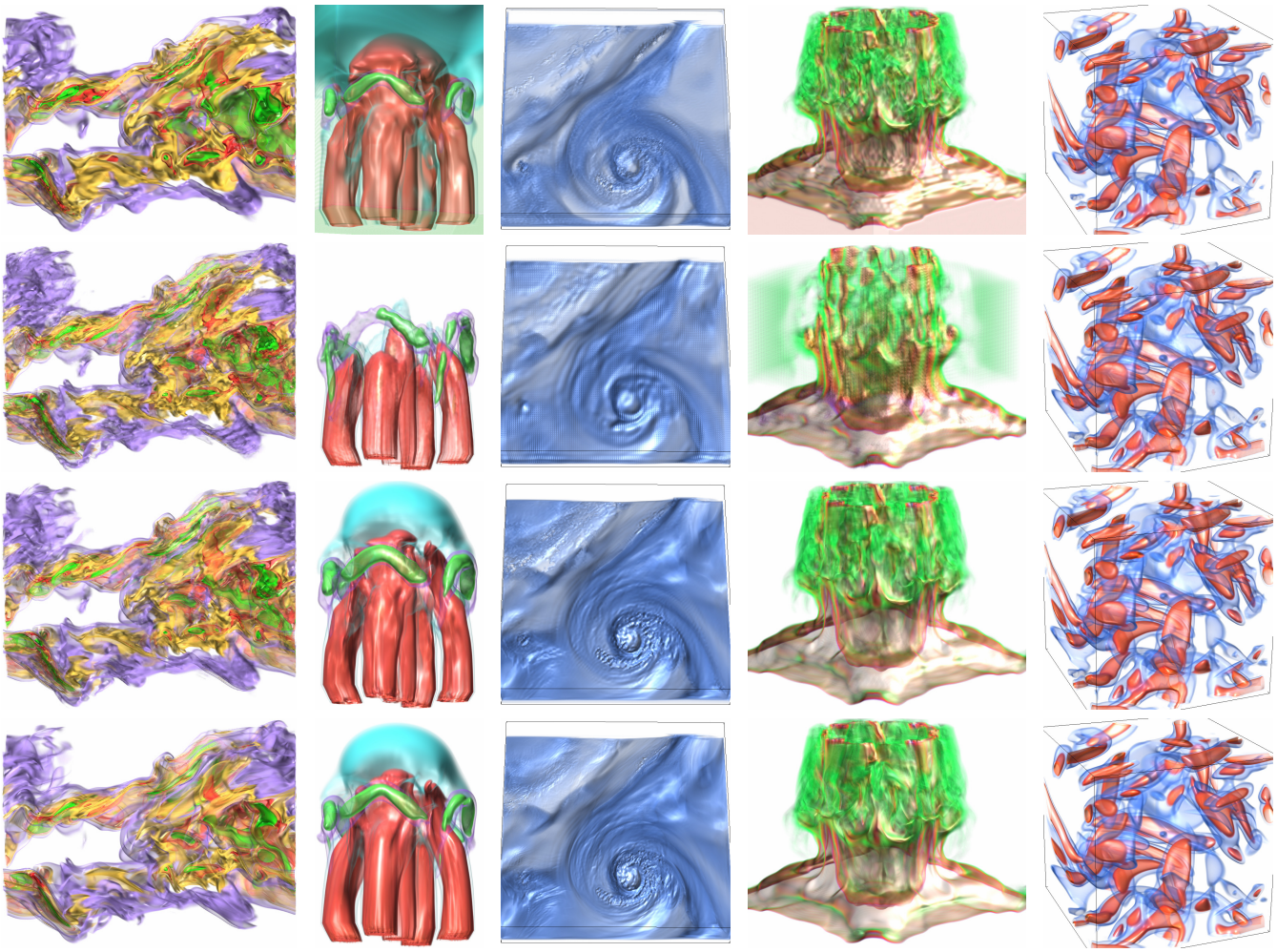
are not observed in the training data (see Figure 1 (a) for an illustration). We use the same setting for lighting, viewing, and transfer function (for volume rendering) for all rendering results of the same data set. In reference to the ground truth (GT), we compare our SSR-TVD results against BI and CNN. For network analysis, please refer to Section 1 in the Appendix.

Evaluation metrics. We use the *peak signal-to-noise ratio* (PSNR) to evaluate the quality of synthesized volumes at the data level. We apply the *structural similarity index* (SSIM) [41] to evaluate the quality of volume rendering images at the image level. Besides volume rendering, we also compare SSR-TVD against BI in terms of volumetric features, expressed in the form of isosurface. To quantify the similarity between two isosurfaces extracted, respectively, from the synthesized and GT volumes, we compute their *isosurface similarity* (IS) [2] at the feature level. The larger the IS, the more similar the two surfaces are. Finally, for each method being compared, we use the *mean opinion score* (MOS) to evaluate how close its rendering image is with respect to the GT image. We recruited ten students and asked them give a five-point score to each pair of images, where 1 is the lowest perceived quality, and 5 is the highest perceived quality. The mean of these scores is reported as MOS.

Quantitative and qualitative analysis. In Figure 3, we quantitatively compare SSR-TVD results against BI and CNN-based results at the data level (PSNR) and image level (SSIM). We utilize a post-upsampling framework [23], [42] as our CNN-based baseline model. We can see that in general, SSR-TVD achieves better performance compared with BI and CNN. At the data level, SSR-TVD produces the highest PSNR values across all four data sets. For the five jets and vortex data sets, PSNR values gradually decrease toward the end of the time sequence for SSR-TVD. This is because we only use 40% of the data (i.e., the early time steps) for training, which makes SSR-TVD produce a lower PSNR at the end of the time sequence. However, if we increase the training samples by using more time steps for training, SSR-TVD achieves higher PSNR for these later time steps. For the ionization (H+) and ionization (He+) data sets, PSNR values decrease at the first 40 time steps, then increase toward the end of the time sequence for CNN and SSR-TVD. This is because visual contents increase at the early time steps, then decrease at the later time steps. The more visual contents in a volume, the more difficult to predict. In Table 3, we report the average PSNR and SSIM values over the entire volume sequence for BI, CNN, and SSR-TVD. Again, SSR-TVD performs the best in terms of PSNR and SSIM. In terms of model storage, the model size of SSR-TVD is 60.6 MB, while that of CNN is 20.3 MB.

At the image level, SSR-TVD still generates higher SSIM compared with BI and CNN. It is the clear winner for the five jets and ionization (H+) data sets. SSR-TVD produces an average SSIM of 0.867 and 0.902, but BI only produces an average SSIM of 0.809 and 0.898, respectively. For the ionization (He+) data set, BI and SSR-TVD yield very close SSIM curves, and both are better than CNN. For the vortex data set, the SSIM curves exhibit more fluctuation for both methods. However, in general, SSR-TVD still outperforms BI and CNN by yielding higher SSIM values.

In Figure 4, we compare rendering results of the synthesized volumes generated by BI, CNN, and SSR-TVD. For the combustion (HR) data set, it is evident that SSR-TVD preserves more local details compared with BI and CNN. For instance, BI produces more orange and green content on the right side, and CNN generates more artifacts in the rendering images. It is clear that SSR-TVD produces a higher visual quality for the five jets data set. SSR-TVD produces more details in red (jets) and cyan (cap) parts, while BI cannot recover the red and cyan parts well and generates some artifacts and CNN fails to construct the cyan and green parts. For the ionization (He+) data set, it is clear that SSR-TVD produces a better visual result at the bottom layer compared with BI and CNN. Moreover, the rendering images generated by BI and CNN contain some noises. For the hurricane data set, SSR-TVD produces a better visual result at the center (eye) of the hurricane. It also produces finer details at the four corners of the volume. For the vortex data set, there is no significant difference among all three methods compared with GT, but upon closer examination, we can observe that SSR-TVD yields closer results at the bottom part. We also observe that all rendering images generated by CNN have artifacts. The possible reasons are (1) volumes are different from images in that a rendering process is involved, which is not captured by the post-upsampling framework; (2) applying deconvolution with-



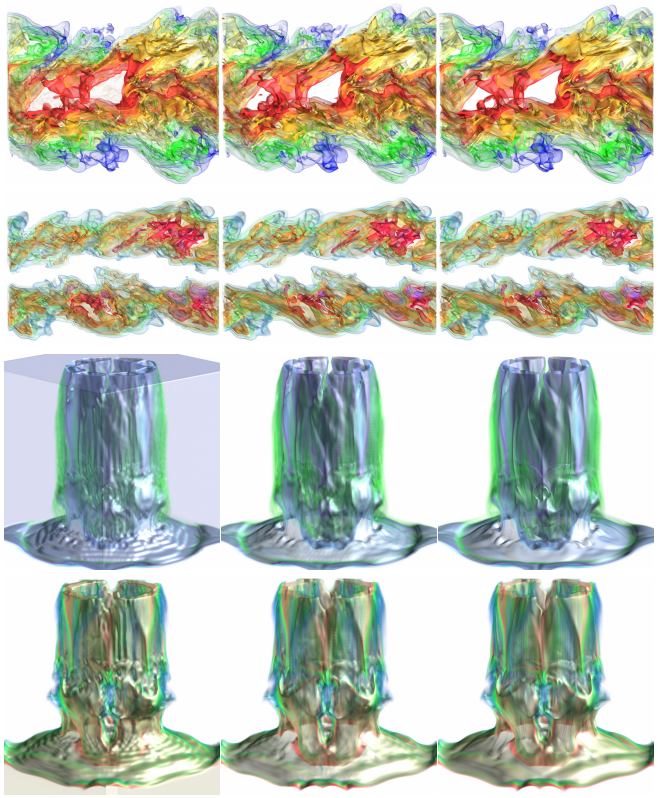
(a) combustion (HR) (b) five jets (c) hurricane (d) ionization (He+) (e) vortex

Fig. 4: Volume rendering results of same-variable inference. Top to bottom: BI, CNN, SSR-TVD, and GT.

out skip or dense connection cannot recover information (such as volume smoothness) which is not presented in the learned feature vector.

In Figure 5, we compare the rendering results of the synthesized volumes generated by BI and SSR-TVD through different-variable inference. Different-variable inference denotes that given one variable X of one data set used for training, another variable Y of the same data set is applied for inference (i.e., X $\rightarrow$ Y). For HR $\rightarrow$ MF of the combustion data set, SSR-TVD produces smoother visual results compared with BI. For example, the yellow part at the top-right corner and the green part at the bottom-right corner are closer to GT. For MF $\rightarrow$ YOH of the combustion data set, SSR-TVD generates more details compared with BI. For example, the orange part at the top-right corner and the red part at the bottom-middle corner. As for H $\rightarrow$ He of the ionization data set, it is clear that SSR-TVD gives high-quality visual results while BI leads to artifacts at the bottom layer and the surrounding region. For He+ $\rightarrow$ H+ of the ionization data set, BI still fails to recover the bottom layer and generates some artifacts at the middle layer, while SSR-TVD reconstructs these regions more faithfully.

In Figures 6 and 7, we compare isosurface rendering results of the synthesized volumes generated by BI and SSR-TVD using the combustion (MF), ionization (He+), five jets, and vortex data sets. For each data set, we choose two time steps and two isovalues to render the isosurfaces. For the combustion (MF) data set, it is obvious that SSR-TVD generates high-quality isosurfaces compared with BI, since BI fails to construct the isosurface close to GT at time steps 76 and 90 (we can see that the surface is filled with noises). For the ionization (He+) data set, it is clear that for time step 62, the isosurface generated by SSR-TVD contains more details (e.g., the bottom layer and the head). For time step 78, SSR-TVD produces high-quality isosurfaces since BI fails to construct the isosurface at the bottom layer. For the five jets data set, SSR-TVD still generates closer isosurfaces than BI, such as the head of the five jets for $v = -0.4$. For the vortex data set, SSR-TVD generates smoother isosurface than BI, such as the middle-left corner for time step 87. Moreover, the isosurfaces produced by SSR-TVD are similar to GT, for example, both SSR-TVD and GT generate some "open" features at the middle-right corner for time step 87, while these features synthesized by BI are closed. Furthermore, we report in Table 4 the average IS values over the entire volume sequence for these three data sets. The quantitative results also confirm that SSR-TVD leads to isosurfaces of better quality than BI.



(a) BI (b) SSR-TVD (c) GT

Fig. 5: Volume rendering results of different-variable inference. Top to bottom: combustion (YOH $\rightarrow$ MF), (b) combustion (MF $\rightarrow$ YOH), (c) ionization (H $\rightarrow$ He), and (d) ionization (He+ $\rightarrow$ H+). Displayed here are MF, YOH, He, and H+.

In Figure 8, we compare isosurface rendering results of the synthesized volumes generated by BI and SSR-TVD using different-variable inference. For the combustion (MF $\rightarrow$ HR) data set, it is clear that the isosurface generated by SSR-TVD preserves more structural details. For example, BI misses some details at the middle-left corner under $v = 0.85$ at time steps 64 and 80. For the ionization (H $\rightarrow$ H+) data set, the isosurfaces generated by SSR-VTD are smoother compared with those of BI. For example, BI fails to recover the bottom layer of the ionization well since it still produces many artifacts there.

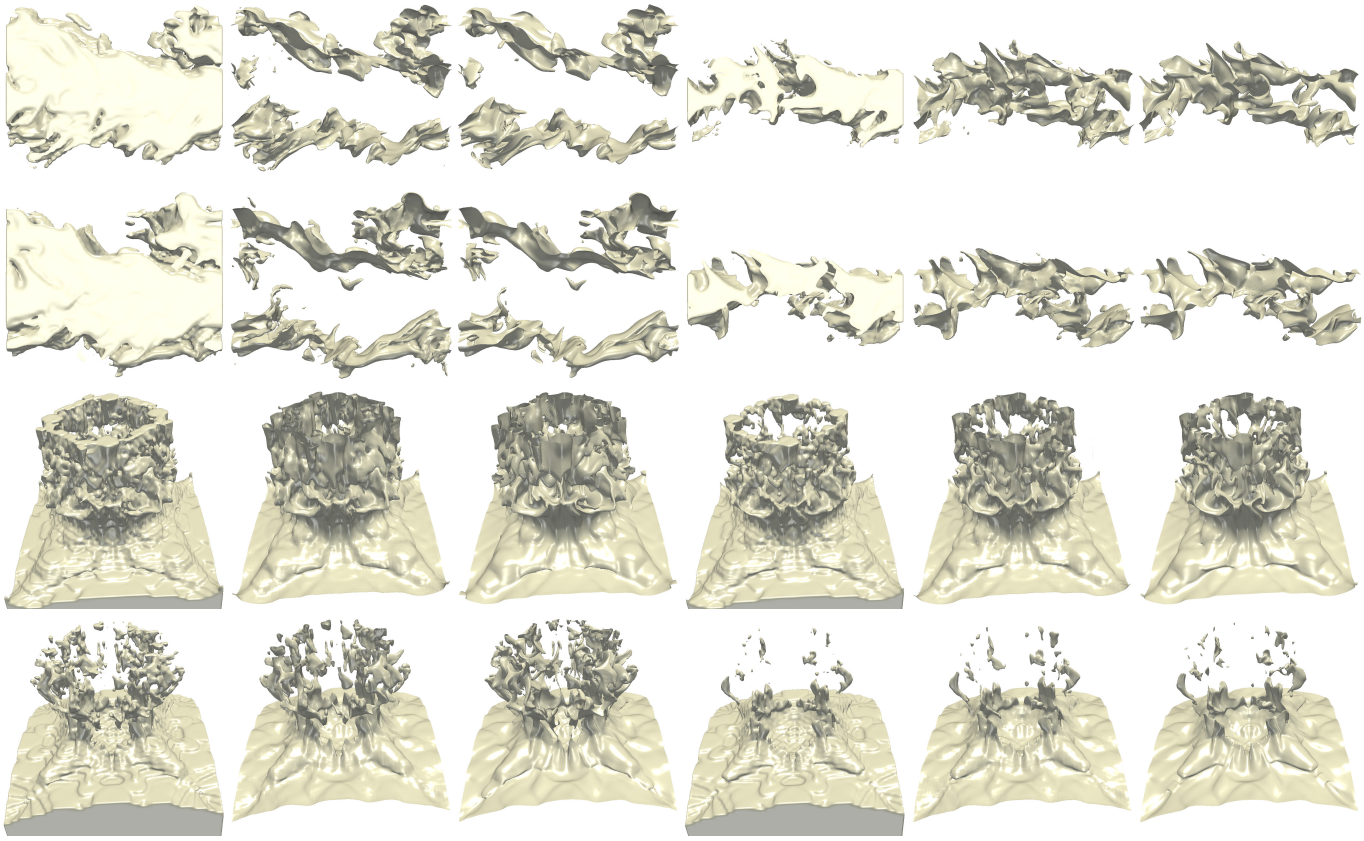
To evaluate the perceptual quality of images (i.e., volume rendering images or isosurface rendering images) rendered from volumetric data synthesized by BI, CNN, and SSR-TVD, compared with those rendered from the GT data, we conducted a user study. For each rendering option, we picked four data sets for comparison, and for each data set, we chose six different time steps. Hence, we collected a total of 144 image pairs (72 pairs for volume rendering and 72 pairs for isosurface rendering) for comparison. For each image pair, we set the left image as rendered from data synthesized by one of the three methods (BI, CNN, or SSR-TVD) and the right image as always rendered from the GT data. We recruited ten graduate students as participants in the study. The participants were asked to evaluate how close the left image is to the right image, by giving an integer score ranging from 1 (least similar) to 5 (most similar). Participants were allowed to go back and forth during their evaluation to adjust the scores accordingly, especially at the beginning, when they needed to calibrate the scores. Participants were informed that several factors, including the overall impression, content shift, local color/shape preservation, and noisy level, should be considered during the evaluation. After a brief introduction, each participant took about half an hour to complete the study and was paid $10. In Table 5, we report the average MOS values. As we can see, across these four data sets and two rendering options, SSR-TVD always achieves the best MOS.

In Figure 9, we compare isosurface rendering results of the synthesized volumes generated by SSR-TVD and a state-of-the-art lossy compression (LC) scheme [24], [25] which can effectively control data distortion while significantly reducing data size. We study two aspects (i.e., under the same compression ratio and under the same PSNR) for a fair comparison. Regardless of the training samples used for SSR-TVD, we set the compression ratio for the combustion data set to $14.98 = (60 \times 19.78)/(60 \times 0.31 + 60.6)$, where 60 is the total number of time steps for inference, 19.78 MB is the data size of the high-resolution volume for one time step, and 60.6 MB is the model size. As shown in the top row of Figure 9, we can see that the isosurfaces generated by LC are not as smooth at the top-left corner as those produced by SSR-TVD. Under the same PSNR (i.e., 30 dB), we can observe that LC fails to recover the structural details for the corresponding isosurfaces and produces more artifacts, as shown in the bottom row of Figure 9.

4.3 Discussion

Like other deep learning techniques, it takes a significant amount of time to train SSR-TVD. In Figure 3 (b) and (d) in the Appendix, we report the time curves for two data sets under different hyperparameter settings. With 40% of training samples and subvolume size of $16 \times 16 \times 16$, it takes sixteen hours to train the vortex data set (400 epochs) and two full days to train the hurricane data set (500 epochs). As the number of training samples or the subvolume size increases, the training time increases as well. However, because of the use of cropping, the training time actually depends on the (cropped) subvolume size rather than the volume size. Due to the limited GPU memory, for large volumes (e.g., hurricane, ionization), we infer individual subvolumes to form the entire volume. For small volumes (e.g., five jets, vortex), we infer the entire volumes directly. The inference time is between ten minutes (vortex) to eight hours (ionization) for the entire volume sequence.

Our current SSR-TVD framework has the following limitations. First, the transfer function is not considered as input for training SSR-TVD. We only treat the low-resolution 3D volumes as numerical data for synthesizing the corresponding super-resolution results. During postprocessing, we manually design 1D transfer functions that map high-resolution volumes to color and opacity and calculate lighting for visualization.Nevertheless, quantitative evaluation using PSNR at the data level and SSIM at the image level shows that SSR-TVD outperforms BI under both metrics. Although considering the transfer function as input may help improve the visual quality and SSIM, the training process



(a) BI (b) SSR-TVD (c) GT (d) BI (e) SSR-TVD (f) GT

Fig. 6: Isosurface rendering results of same-variable inference using the combustion (MF) data set for time steps 76 (1st row) and 90 (2nd row), and ionization (He+) data set for time steps 62 (3rd row) and 78 (4th row). The value range is normalized to $[-1, 1]$. For combustion (MF), (a-c): $v = -0.216$, (d-f): $v = 0.569$. For ionization (He+), (a-c): $v = -0.803$, (d-f): $v = -0.608$.

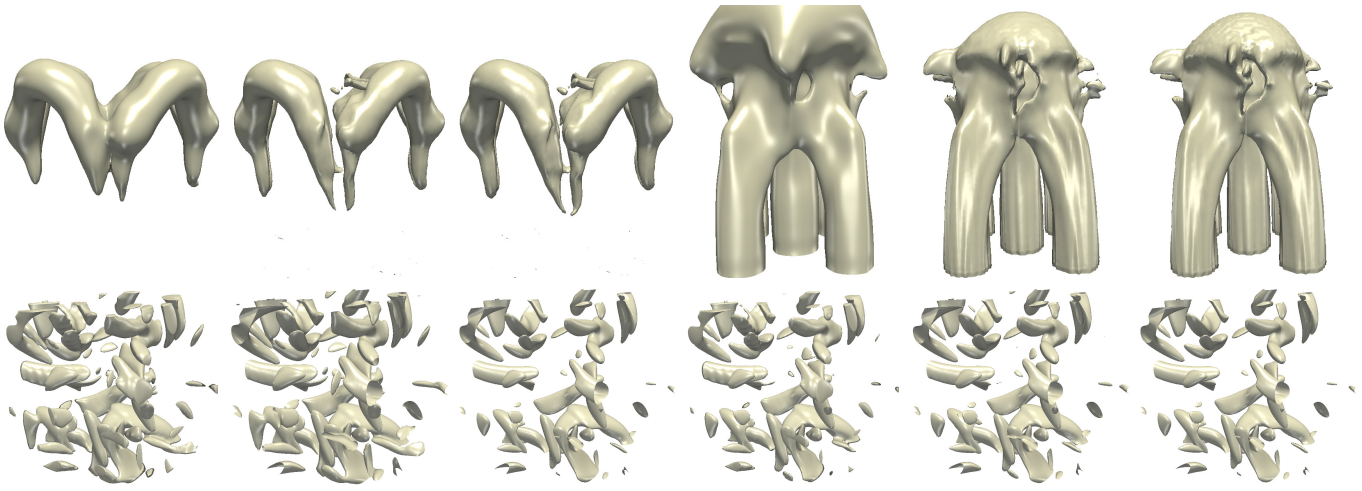
is highly dependent on the user-picked transfer function. In addition, it could make the data preprocessing more difficult. For example, how to choose a transfer function, how many transfer functions are enough for training, and how to incorporate the transfer function into optimizing the deep learning pipeline.

Second, we only experiment with using the first segment (i.e., starting from the first time step) of the time sequence as the training samples, and therefore, the PSNR curves may fall into the lower value range for the later time steps. We can further investigate which segment of the time sequence is the more characteristic and thus the best for training, which is likely to be data set dependent.

Third, while SSR-TVD is capable of generating high-resolution volumes on a scaling factor of four, it does not synthesize fine details on some data sets. An example is illustrated in Figure 10. As the results indicate, neither BI nor SSR-TVD could generate high-quality rendering results on the argon bubble data set. Both approaches fail to produce enough details on the rendering images. We suspect that this is because SSR-TVD has limited ability to estimate complex data distributions. For verification, we plot the histograms of four different data sets, as shown in Figure 11. We observe that the vortex and five jets data sets exhibit Gaussian distributions, and the hurricane data set follows a long-tail distribution. The argon bubble data set, however, does not follow any single data distribution but presents a mixture of Gaussian and long-tail distributions. The corresponding transfer function also crosses both distributions (refer to Figure 11 (a)). We note that GAN encounters a similar problem in image generation tasks. If the training pool contains images of various kinds of scenes, GAN will fail to generate high-quality results, since the data distribution is too complex for GAN to learn. As such, most works on GAN only focus on image collections of the same kind of scene (e.g., the indoor scene with furniture) rather than image collections of diverse scenes, such as ImageNet. In the future, we will design powerful modules to improve the ability of SSR-TVD in learning complex data distributions.

5 Conclusions and Future Work

We have presented SSR-TVD, a new deep learning solution for generating SSR of TVD. Using adversarial, content, and feature losses, SSR-TVD can upscale the volumes by a factor of four (making the size of each output volume 64 times the size of the input volume) while achieving better visual quality for volume upscaling compared with the de facto standard of BI and a solution solely based on CNN. Quantitative evaluation using PSNR (data-level), SSIM (image-level), IS (feature-level), and MOS (perception-level) also confirms the effectiveness of SSR-TVD.



(a) BI (b) SSR-TVD (c) GT (d) BI (e) SSR-TVD (f) GT

Fig. 7: Isosurface rendering results of same-variable inference using the five jets data set for time step 46 (top row) and vortex data set for time step 87 (bottom row). For five jets, (a-c): $v = -0.2$, (d-f): $v = 0.4$. For vortex, (a-c): $v = 0$, (d-f): $v = 0.1$.

SSR-TVD is an intermediate step of our research effort toward *data augmentation for scientific visualization*. Given a time-varying volume data set of size, for example, $128 \times 128 \times 128$ with 100 time steps, SSR-TVD generates the SSR volume sequence of size, for example, $512 \times 512 \times 512$ for each volume while keeping the temporal resolution the same. Previously, we presented TSR-TVD [10] which generates the temporal super-resolution (TSR) volume sequence of, for example, 500 time steps, while keeping the spatial resolution the same. Eventually, we would like to achieve spatiotemporal super-resolution (STSR) by producing the volume sequence of size, for example, $512 \times 512 \times 512$ with 500 time steps. Upscaling both spatial and temporal dimensions plays an important role in handling time-varying data produced from large-scale scientific simulations as scientists can only sparsely sample the simulation output because of the scarce storage space. Beyond spatiotemporal dimensions, our recent work on V2V [11] addresses variable selection and translation problems for multivariate time-varying data. Our research will provide a promising alternative for scientists to make better decisions depending on the nature of the simulations and the characteristics of the data.

ACKNOWLEDGEMENTS

This research was supported in part by the U.S. National Science Foundation through grants IIS-1455886, CNS-1629914, DUE-1833129, and IIS-1955395, and the NVIDIA GPU Grant Program. The authors would like to thank the anonymous reviewers for their insightful comments.

REFERENCES

[1] M. Berger, J. Li, and J. A. Levine. A generative model for volume rendering. *IEEE Transactions on Visualization and Computer Graphics*, 25(4):1636–1650, 2019.

[2] S. Bruckner and T. Möller. Isosurface similarity maps. *Computer Graphics Forum*, 29(3):773–782, 2010.

[3] H.-C. Cheng, A. Cardone, S. Jain, E. Krokos, K. Narayan, S. Subramaniam, and A. Varshney. Deep-learning-assisted volume visualization. *IEEE Transactions on Visualization and Computer Graphics*, 25(2):1378–1391, 2019.

[4] C. Dong, C. C. Loy, K. He, and X. Tang. Image super-resolution using deep convolutional networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 38(2):295–307, 2016.

[5] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In *Proceedings of Advances in Neural Information Processing Systems*, pages 2672–2680, 2014.

[6] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. C. Courville. Improved training of Wasserstein GANs. In *Proceedings of Advances in Neural Information Processing Systems*, pages 5767–5777, 2017.

[7] L. Guo, S. Ye, J. Han, H. Zheng, H. Gao, D. Z. Chen, J.-X. Wang, and C. Wang. SSR-VFD: Spatial super-resolution for vector field data analysis and visualization. In *Proceedings of IEEE Pacific Visualization Symposium*, pages 71–80, 2020.

[8] J. Han, J. Tao, and C. Wang. FlowNet: A deep learning framework for clustering and selection of streamlines and stream surfaces. *IEEE Transactions on Visualization and Computer Graphics*, 26(4):1732–1744, 2020.

[9] J. Han, J. Tao, H. Zheng, H. Guo, D. Z. Chen, and C. Wang. Flow field reduction via reconstructing vector data from 3D streamlines using deep learning. *IEEE Computer Graphics and Applications*, 39(4):54–67, 2019.

[10] J. Han and C. Wang. TSR-TVD: Temporal super-resolution for time-varying data analysis and visualization. *IEEE Transactions on Visualization and Computer Graphics*, 26(1):205–215, 2020.

[11] J. Han, H. Zheng, Y. Xing, D. Z. Chen, and C. Wang. V2V: A deep learning approach to variable-to-variable selection and translation for multivariate time-varying data. *IEEE Transactions on Visualization and Computer Graphics*, 27(2), 2021. In Press.

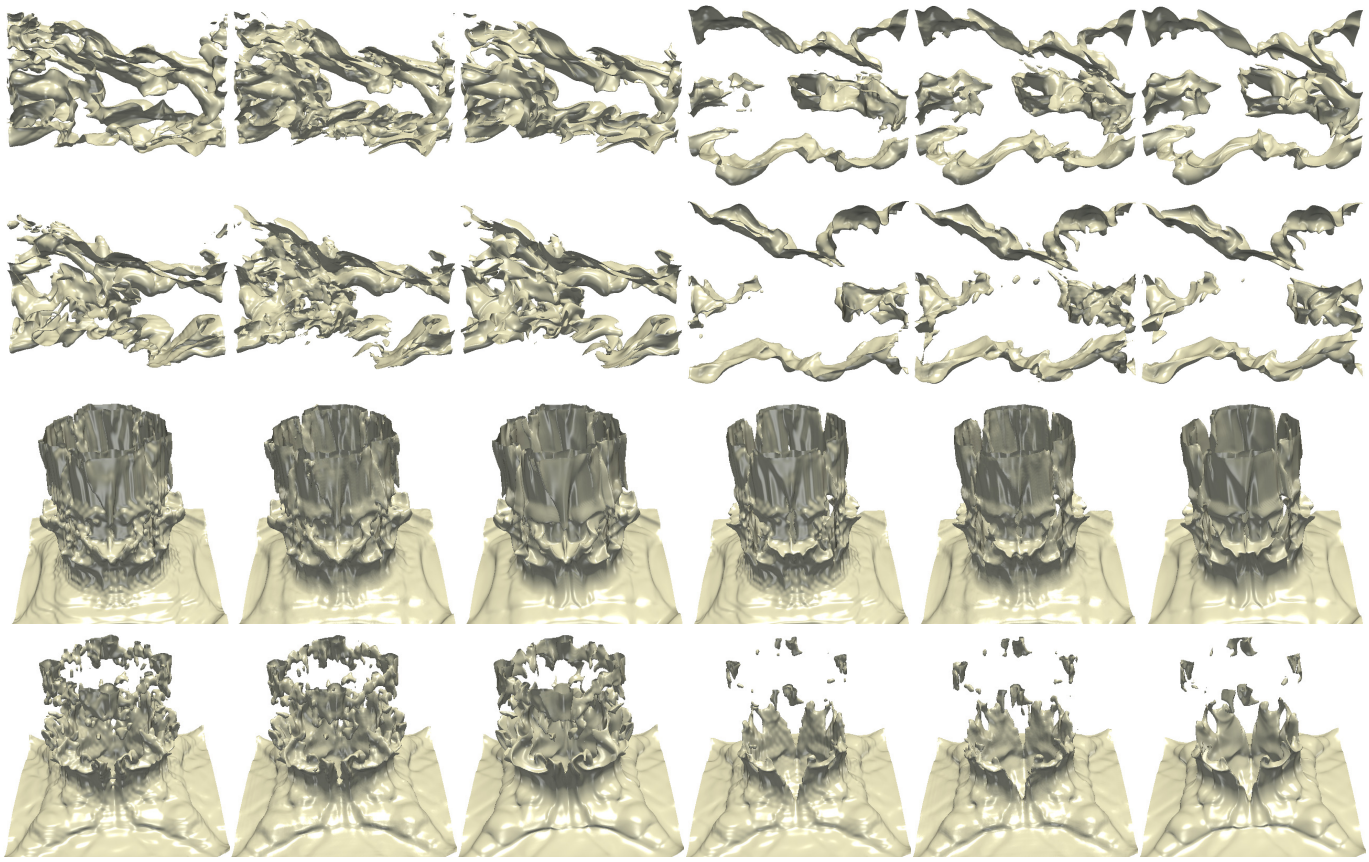
[12] K. He, X. Zhang, S. Ren, and J. Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of IEEE International Conference on Computer Vision*, pages 1026–1034, 2015.

[13] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016.

[14] W. He, J. Wang, H. Guo, K.-C. Wang, H.-W. Shen, M. Raj, Y. S. G. Nashed, and T. Peterka. InSituNet: Deep image synthesis for parameter space exploration of ensemble simulations. *IEEE Transactions on Visualization and Computer Graphics*, 26(1):23–33, 2020.

[15] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter. GANs trained by a two time-scale update rule converge to a local Nash equilibrium. In *Proceedings of Advances in Neural Information Processing Systems*, pages 6626–6637, 2017.

[16] F. Hong, C. Liu, and X. Yuan. DNN-VolVis: Interactive volume visualization supported by deep neural network. In *Proceedings of IEEE Pacific Visualization Symposium*, pages 282–291, 2019.



(a) BI (b) SSR-TVD (c) GT (d) BI (e) SSR-TVD (f) GT

Fig. 8: Isosurface rendering results of different-variable inference using the combustion (MF $\rightarrow$ HR) data set for time steps 64 (1st row) and 80 (2nd row) and the ionization (H $\rightarrow$ H+) data set for time steps 40 (3rd row) and 60 (4th row). For combustion (HR) , (a-c): $v = 0.2$, (d-f): $v = 0.85$. For ionization (H+), (a-c): $v = -0.6$, (d-f): $v = 0$.

[17] F. Hong, J. Zhang, and X. Yuan. Access pattern learning with long short-term memory for parallel particle tracing. In *Proceedings of IEEE Pacific Visualization Symposium*, pages 76–85, 2018.

[18] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger. Densely connected convolutional networks. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 4700–4708, 2017.

[19] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros. Image-to-image translation with conditional adversarial networks. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 1125–1134, 2017.

[20] Y. Jo, S. W. Oh, J. Kang, and S. J. Kim. Deep video super-resolution network using dynamic upsampling filters without explicit motion compensation. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 3224–3232, 2018.

[21] J. Johnson, A. Alahi, and F.-F. Li. Perceptual losses for real-time style transfer and super-resolution. In *Proceedings of European Conference on Computer Vision*, pages 694–711, 2016.

[22] D. Kingma and J. Ba. Adam: A method for stochastic optimization. In *Proceedings of International Conference for Learning Representations*, 2015.

[23] C. Ledig, L. Theis, F. Huszár, J. Caballero, A. Cunningham, A. Acosta, A. P. Aitken, A. Tejani, J. Totz, Z. Wang, and W. Shi. Photo-realistic single image super-resolution using a generative adversarial network. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 4681–4690, 2017.

[24] X. Liang, S. Di, D. Tao, Z. Chen, and F. Cappello. An efficient transformation scheme for lossy data compression with point-wise relative error bound. In *Proceedings of IEEE International Conference on Cluster Computing*, pages 179–189, 2018.

[25] X. Liang, S. Di, D. Tao, S. Li, S. Li, H. Guo, Z. Chen, and F. Cappello. Error-controlled lossy compression optimized for high compression ratios of scientific datasets. In *Proceedings of IEEE International Conference on Big Data*, pages 438–447, 2018.

[26] X. Mao, Q. Li, H. Xie, R. Y. Lau, Z. Wang, and S. P. Smolley. Least squares generative adversarial networks. In *Proceedings of International Conference on Computer Vision*, pages 2813–2821, 2017.

[27] M. Mathieu, C. Couprie, and Y. LeCun. Deep multi-scale video prediction beyond mean square error. In *Proceedings of International Conference on Machine Learning*, 2015.

[28] M. Mirza and S. Osindero. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*, 2014.

[29] T. Miyato, T. Kataoka, M. Koyama, and Y. Yoshida. Spectral normalization for generative adversarial networks. In *Proceedings of International Conference for Learning Representations*, 2018.

[30] V. Nair and G. E. Hinton. Rectified linear units improve restricted Boltzmann machines. In *Proceedings of International Conference on Machine Learning*, pages 807–814, 2010.

[31] D. Pathak, P. Krahenbuhl, J. Donahue, T. Darrell, and A. A. Efros. Context encoders: Feature learning by inpainting. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 2536–2544, 2016.

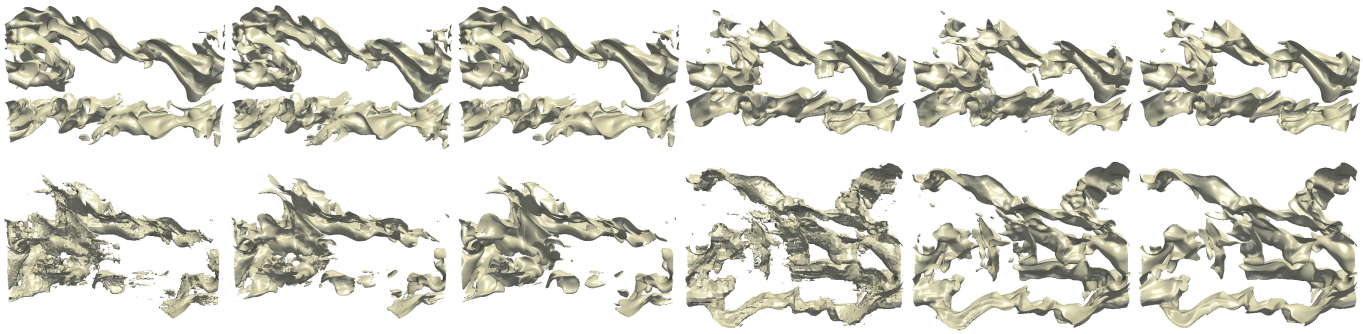
[32] E. Pérez-Pellitero, M. S. Sajjadi, M. Hirsch, and B. Schölkopf. Photorealistic video super resolution. *arXiv preprint arXiv:1807.07930*, 2018.

[33] W. P. Porter, Y. Xing, B. R. von Ohlen, J. Han, and C. Wang. A deep learning approach to selecting representative time steps for time-varying multivariate data. In *Proceedings of IEEE VIS Conference (Short Papers)*, pages 131–135, 2019.

[34] A. Radford, L. Metz, and S. Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015.

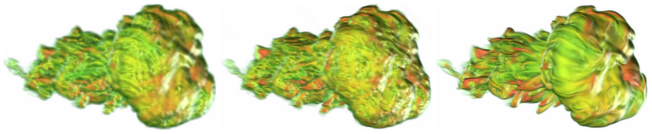
[35] S. E. Reed, Z. Akata, S. Mohan, S. Tenka, B. Schiele, and H. Lee. Learning what and where to draw. In *Proceedings of Advances in Neural Information Processing Systems*, pages 217–225, 2016.

[36] O. Ronneberger, P. Fischer, and T. Brox. U-Net: Convolutional



(a) LC (b) SSR-TVD (c) GT (d) LC (e) SSR-TVD (f) GT

Fig. 9: Isosurface rendering results using the combustion data set. Top row: MF for time step 59 under the same compression ratio of 14.98. Bottom row: HR for time step 72 under the same PSNR of 30 dB. For MF, (a-c): $v = 0$, (d-f): $v = 0.333$. For HR, (a-c): $v = 0$, (d-f): $v = 0.7$.



(a) BI (b) SSR-TVD (c) GT

Fig. 10: A failure case for the argon bubble data set.

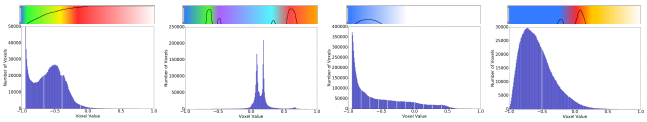


Fig. 11: Histograms and transfer functions of different data sets. The data value range shown in each histogram is normalized to $[-1, 1]$. The black curve in the transfer function denotes the opacity values. (a-d): argon bubble, five jets, hurricane, vortex.

networks for biomedical image segmentation. In *Proceedings of International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 234–241, 2015.

[37] M. S. Sajjadi, R. Vemulapalli, and M. Brown. Frame-recurrent video super-resolution. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 6626–6634, 2018.

[38] D. Ulyanov, A. Vedaldi, and V. Lempitsky. Instance normalization: The missing ingredient for fast stylization. *arXiv preprint arXiv:1607.08022*, 2016.

[39] T.-C. Wang, M.-Y. Liu, J.-Y. Zhu, A. Tao, J. Kautz, and B. Catanzaro. High-resolution image synthesis and semantic manipulation with conditional gans. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 8798–8807, 2018.

[40] Y. Wang, Z. Zhong, and J. Hua. DeepOrganNet: On-the-fly reconstruction and visualization of 3D/4D lung models from single-view projections by deep deformation network. *IEEE Transactions on Visualization and Computer Graphics*, 26(1):960–970, 2020.

[41] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli. Image quality assessment: From error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4):600–612, 2004.

[42] Z. Wang, J. Chen, and S. C. Hoi. Deep learning for image super-resolution: A survey. *arXiv preprint arXiv:1902.06068*, 2019.

[43] S. Weiss, M. Chu, N. Thuerey, and R. Westermann. Volumetric isosurface rendering with deep learning-based super-resolution. *IEEE Transactions on Visualization and Computer Graphics*, 2019. Accepted.

[44] Y. Xie, E. Franz, M. Chu, and N. Thuerey. tempoGAN: A temporally coherent, volumetric GAN for super-resolution fluid flow. *ACM Transactions on Graphics*, 37(4):95:1–95:15, 2018.

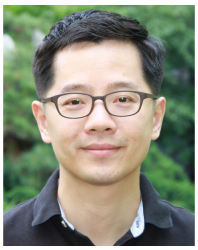
[45] D. Yoo, N. Kim, S. Park, A. S. Paek, and I. S. Kweon. Pixel-level domain transfer. In *Proceedings of European Conference on Computer Vision*, pages 517–532, 2016.

[46] Y. Zhang, K. Li, K. Li, L. Wang, B. Zhong, and Y. Fu. Image super-resolution using very deep residual channel attention networks. In *Proceedings of European Conference on Computer Vision*, pages 294–310, 2018.

[47] Y. Zhou and T. L. Berg. Learning temporal transformations from time-lapse videos. In *Proceedings of European Conference on Computer Vision*, pages 262–277, 2016.

[48] Z. Zhou, Y. Hou, Q. Wang, G. Chen, J. Lu, Y. Tao, and H. Lin. Volume upscaling with convolutional neural networks. In *Proceedings of Computer Graphics International*, pages 38:1–38:6, 2017.

Jun Han is a PhD student at the University of Notre Dame. He received a BS degree in software engineering and a MS degree in computer software and theory in 2014 and 2017, respectively. Both degrees are from Xidian University. His current research focuses on applying deep learning techniques to solve data visualization problems.

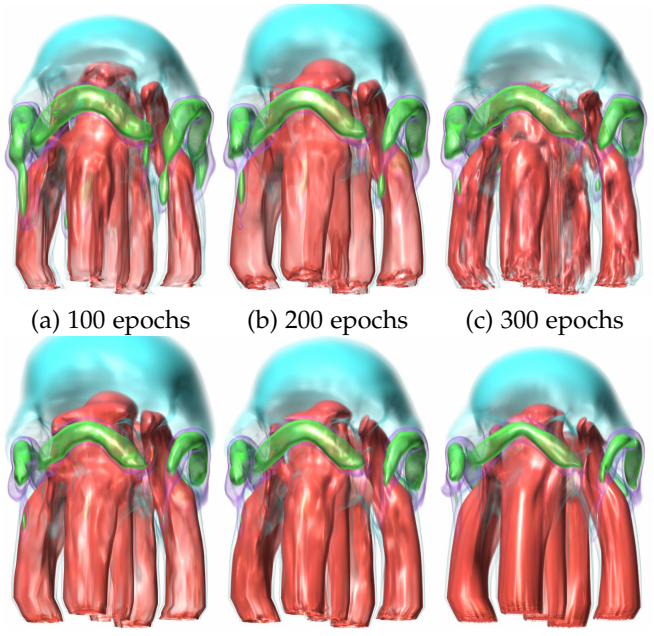


Chaoli Wang is an associate professor of computer science and engineering at the University of Notre Dame. He received a Ph.D. degree in computer and information science from The Ohio State University in 2006. Dr. Wang's main research interest is data visualization, in particular on the topics of time-varying multivariate data visualization, flow visualization, as well as information-theoretic algorithms, graph-based techniques, and deep learning solutions for big data analytics. He is an associate editor of IEEE Transactions on Visualization and Computer Graphics.

APPENDIX

1 NETWORK ANALYSIS

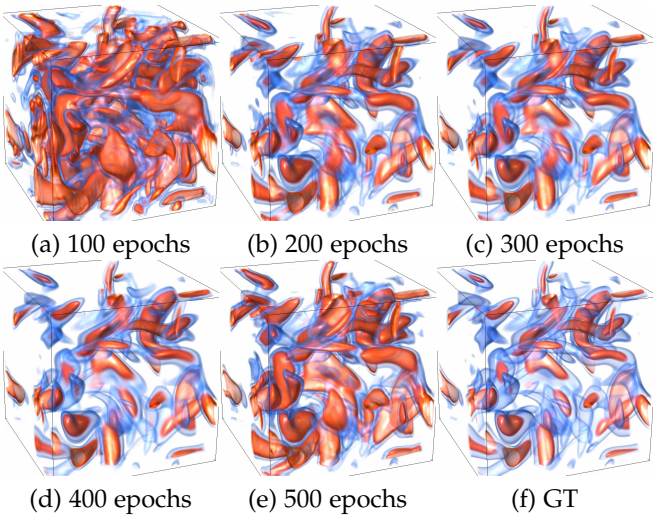
In order to evaluate SSR-TVD, we analyze the following hyperparameter settings: training epochs, training samples, subvolume size, temporal coherence, spatial coherence, and training stability. A detailed discussion is provided as follows.



(a) 100 epochs (b) 200 epochs (c) 300 epochs

(d) 400 epochs (e) 500 epochs (f) GT

Fig. 1: Volume rendering results under different training epochs using the five jets data set. The best match with GT is the result with 500 epochs.



(a) 100 epochs (b) 200 epochs (c) 300 epochs

(d) 400 epochs (e) 500 epochs (f) GT

Fig. 2: Volume rendering results under different training epochs using the vortex data set. The best match with GT is the result with 400 epochs.

Training epochs vs. visual quality. We investigate how the quality of the synthesized volume using SSR-TVD evolves with the increasing number of training epochs. Rendering of the volumes obtained after different numbers of training epochs is illustrated in Figures 1 and 2. For the five jets data set, during the whole training process, the change of the cyan part is unstable. For example, the rendering results from 300 and 500 epochs are better than the results from 200 and 400 epochs. However, the green part gets increasingly closer to GT as we apply more training epochs. For example, the green tail on the left part vanishes after 500 epochs, which matches GT. Moreover, we observe that after 500 epochs, there is no significant difference among synthesized results with different numbers of epochs, so we choose 500 epochs to train the five jets data set. For the vortex data set, we find that in the first 400 epochs, SSR-TVD generates better visual quality. For example, the red tail at the top-left corner becomes smaller as the training goes. However, after 400 epochs, the visual quality actually gets worse as SSR-TVD begins overfitting. Hence, we choose 400 epochs to train the vortex data set.

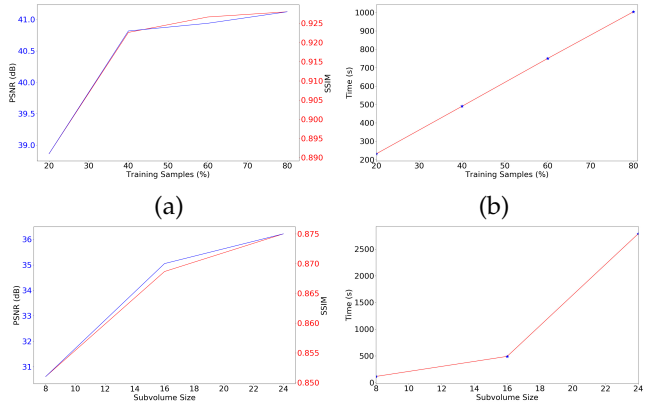
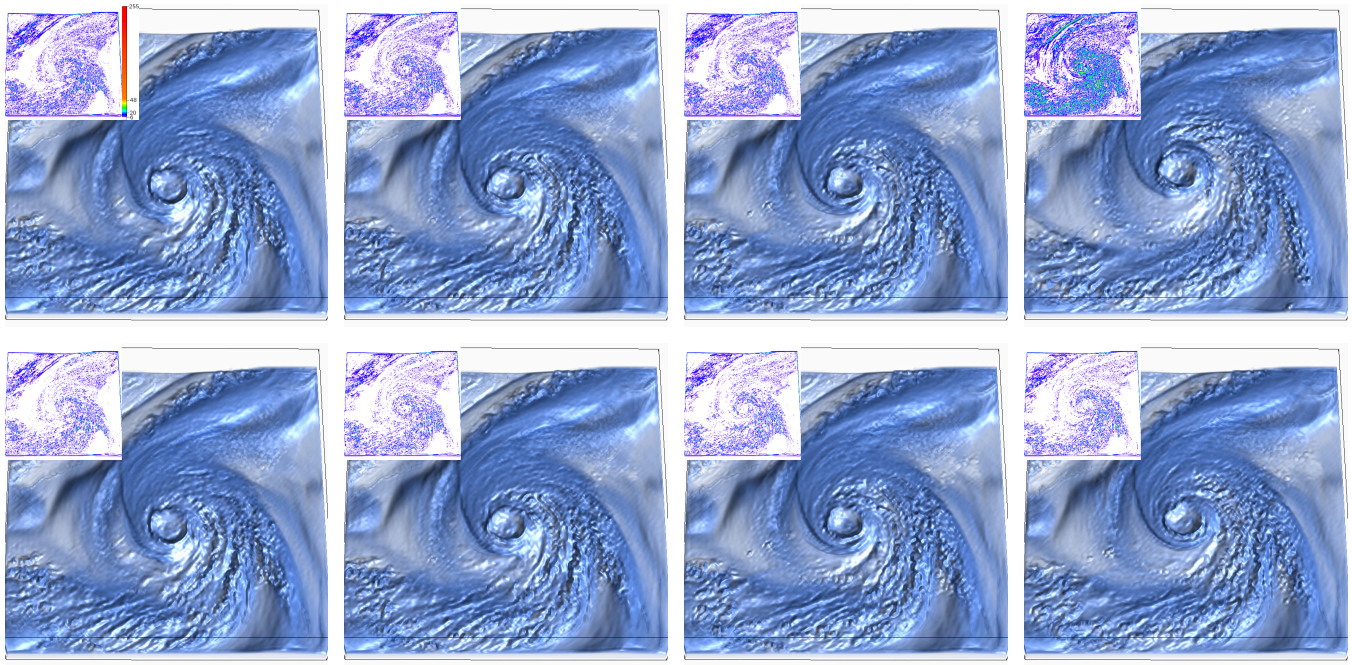


Fig. 3: Comparison of hyperparameter settings. (a) Average PSNR and SSIM and (b) average time (per epoch) under different training samples using the ionization (He+) data set. (c) Average PSNR and SSIM and (d) average time (per epoch) under different subvolume sizes using the vortex data set.

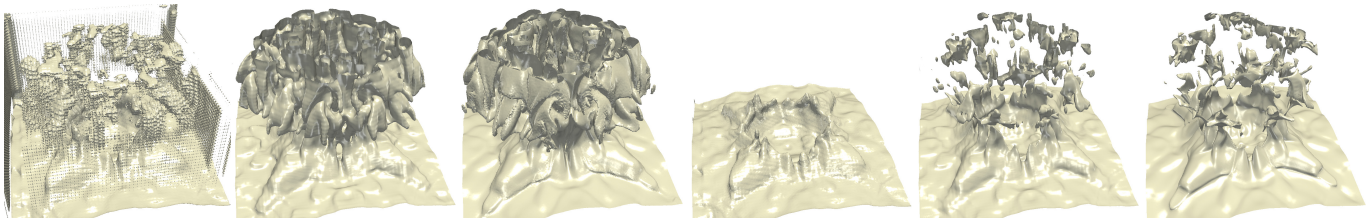
Training samples vs. PSNR and SSIM. For an upscaling factor of four, we study the influence of the number of training samples on PSNR. We use 20%, 40%, 60%, and 80% training samples to train SSR-TVD on the ionization (He+) data set. We plot the average PSNR and SSIM curves under different numbers of training samples, as shown in Figure 3 (a). We can see that PSNR and SSIM can be improved by using more training samples. However, this demands longer training time, as shown in Figure 3 (b). We observe that visual quality does not benefit from using more training samples. As a trade-off, we use 40% samples to train SSR-TVD.

Subvolume size vs. PSNR and SSIM. We perform training with subvolume sizes of $8 \times 8 \times 8$, $16 \times 16 \times 16$, and $24 \times 24 \times 24$ on the vortex data set. The average PSNR and SSIM curves are shown in Figure 3 (c). We can observe that training SSR-TVD benefits from a larger subvolume size since an enlarged receptive field helps the network capture more semantic information. However, a larger training subvolume size takes more time to train (as shown in Figure 3 (d)) and consumes more computing resources. As a trade-



(a) time step 21 (b) time step 22 (c) time step 23 (d) time step 24

Fig. 4: Volume rendering results using the hurricane data set without (top row) and with (bottom row) considering temporal coherence (D_t) during training. In each image, we also show on the side, pixelwise image differences with respect to GT.



(a) without D_s (b) with D_s (c) GT (d) without D_s (e) with D_s (f) GT

Fig. 5: Isosurface rendering results using the ionization (H+) data set for time step 74 without and with considering spatial coherence (D_s) during training. (a-c): $v = -0.961$, (d-f): $v = -0.843$.

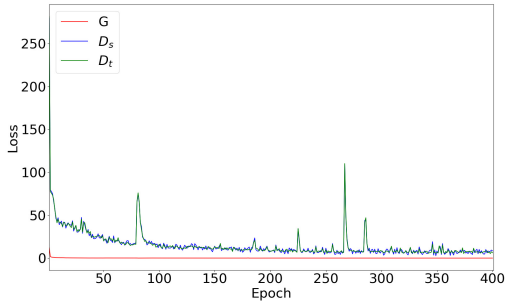


Fig. 6: Training loss curves of G, D_s, and D_t using the vortex data set.

off, we use the subvolume size of $16 \times 16 \times 16$ to train SSR-TVD.

Temporal coherence. To investigate the role of temporal coherence in SSR-TVD training, we conduct an experiment that trains SSR-TVD with and without considering D_t. We then render the resulting synthesized volumes and compare the visual difference, as shown in Figure 4. For objective comparison, we calculate pixel-wise differences as Han and Wang [10] to generate the difference image. The difference image is displayed at the top-left corner. The results indicate that adding D_t into SSR-TVD training can better preserve the hurricane's eye and its surroundings, especially for Figure 4 (d). We also compute the SSIM values of the corresponding time steps. Without considering D_t, the SSIM values are 0.851, 0.845, 0.842, and 0.826 for the four consecutive time steps, respectively, while with considering D_t, the SSIM values improve to 0.868, 0.864, 0.861, and 0.860, respectively.

Spatial coherence. To investigate the role of spatial coherence in SSR-TVD training, we conduct an experiment that trains SSR-TVD with and without considering D_s. We then render isosurfaces extracted from the resulting synthesized volumes and compare the visual difference, as shown in Figure 5. In reference to GT, the results without adding D_s reconstruct the isosurfaces far worse than the results with adding D_s. Therefore, adding D_s during training can yield better spatially coherent results.

Training stability. To study the stability of training SSR-TVD, we plot the loss curves of G, D_s, and D_t, as sketched in Figure 6. We can see that the loss of G decreases over

the training process, while the losses of D_s and D_t fluctuate four times during training. These fluctuations are caused by the SSR-TVD training mechanism: G needs to compete with D_s and D_t. The overall training process of G, D_s, and D_t are stable, which confirms the stability of SSR-TVD.