Department: Head
Editor: Name, xxxx@email

STSRNet: Deep Joint Space-Time Super-Resolution for Vector Field Visualization

Yifei An
Computer Information Network Center, Chinese Academy of Sciences
University of Chinese Academy of Sciences

Han-Wei Shen
The Ohio State University

Guihua Shan
Computer Information Network Center, Chinese Academy of Sciences
University of Chinese Academy of Sciences

Guan Li
Computer Information Network Center, Chinese Academy of Sciences

Jun Liu
Computer Information Network Center, Chinese Academy of Sciences

***Abstract*—We propose STSRNet, a joint space-time super-resolution deep learning based model for time-varying vector field data. Our method is designed to reconstruct high temporal resolution (HTR) and high spatial resolution (HSR) vector fields sequence from the corresponding low-resolution key frames. For large scale simulations, only data from a subset of time steps with reduced spatial resolution can be stored for post-hoc analysis. In this paper, we leverage a deep learning model to capture the non-linear complex changes of vector field data with a two-stage architecture: the first stage deforms a pair of low spatial resolution (LSR) key frames forward and backward to generate the intermediate LSR frames, and the second stage performs spatial super-resolution to output the high-resolution sequence. Our method is scalable and can handle different data sets. We demonstrate the effectiveness of our framework with several data sets through quantitative and qualitative evaluations.**

THE INTRODUCTION Effective analysis of vector fields plays a fundamental role in many scientific disciplines, such as aerodynamics, climate, and computational fluid dynamics. As the power of high performance computers increases rapidly in the recent years, physical simulations

 Published by the IEEE Computer Society

can produce vast amounts of data with finer spatial and temporal resolutions to ensure better reliability and reveal more detailed features. However, due to the bandwidth and storage constraints, scientists could only transmit or store very limited amount of data for post-hoc visualization and analysis. Quality of visualization can become problematic if the data are saved with a coarser grid resolution and a small number of time steps. Many recent efforts have been focused on in-situ visualization where data are directly visualized in real-time during the run of simulation. While alleviating the big data problem, the flexibility of post exploration is severely limited without high-resolution data.

This study focuses on improving researchers' ability in post-hoc analysis of time-varying vector data with more detailed spatial-temporal features. Instead of producing visualization results, we aim at performing super-resolution and generating high-resolution data, which presents two challenges: (1) Instabilities and complexity of unsteady flow fields. Linear interpolation can eliminate important details and hence suffers from its inability to reconstruct data with plausible evolution. It can be extremely hard to restore the large fluctuating dynamics between the key frames with long-range interpolation steps especially for unsteady flow. (2) Reconstruction of spatial features removed from the original simulation output to achieve high visual quality. The spatial upscale interpolation kernels should be data-dependent. For spatial or temporal super-resolution, several works have been proposed using deep learning methods to reconstruct data. For example, Guo *et al.* [6] focused on spatial super-resolution for vector field data. Han and Wang [7] generated temporally refined volumes. However, few works focus on achieving spatial-temporal super-resolution in vector fields to achieve better exploration.

In this work, we propose a joint deep learning framework to handle both spatial and temporal super-resolution for time-varying vector fields with neural networks. Our goal is to recover detailed time-varying patterns and features from reduced data to assist exploration of dynamic data in large size and long-term sequence without storing massive data sets in disk. Using the simulation output as the ground truth, we train a model that is capable of mapping data of lower spatio-temporal resolution to high-resolution data. The down-sampled datasets and trained model can be transmitted to local machines, where the high-resolution data are then reconstructed for post-hoc analysis. Our joint framework consists of two stages to address the super-resolution challenges in both spatial and temporal dimensions. The first stage takes a pair of low-resolution key frames as input and produce a sequence of spatial low-resolution intermediate frames to fill the gaps in time. Then the intermediate frames are input to the spatial super-resolution stage to infer high-resolution details in the vector field to generate fine-grained visualization result. Different from video enhancement task, our goal is to reconstruct vector field data rather than scalar fields. In this scenario, the directional features and high-order information of the data need to be taken into consideration.

We compare the accuracy and visualization effect of our predicted vector fields with ground truth and other alternative methods to demonstrate the efficacy of our framework when applied to different data sets. We also evaluate the effect of hyperparameters of the framework. Our main contributions are summarized as follows. First, we propose a novel pipeline to address the challenge for post-hoc visualizations when data are stored only at limited time steps and spatial resolutions. Second, we apply a sophisticate deep learning architecture for super-resolution tasks in vector field data. Third, we propose a physically-based loss function combined with temporal coherence for reconstructing vectors.

1. RELATED WORK

We review the topics related to our work in video restoration, vector field reconstruction and deep learning in scientific visualization.

1.1. Video restoration

Deep learning networks have proven to be effective in the field of computer vision in various tasks, including video restoration.

Video restoration aims at restoring the lost information from the inputs. For video frame interpolation, Niklaus *et al.* [13] used a CNN to learn a spatially-varying kernel to synthesize the intermediate frame. There are also some flow-

based methods to explicitly estimate motions, Jiang *et al.* [9] predicted forward and backward optical flow maps between two images and use another CNN to generate interpolated frames based on the predicted optical flow. Another video restoration task is video super-resolution aiming to reconstruct high-resolution video from the corresponding low-resolution one by exploiting temporal information. Several methods [16] [4] use optical flows to perform alignment between consecutive frames. There are also works focusing on both tasks of solving space-time super-resolution for video [19] [15] [12] [2]. However, it is difficult to obtain accurate flow with large motions. To address this problem, TDAN [17] introduces the defomable convolution for aligning the temporal features implicitly without estimating optical flow. And EDVR [18] improves it by performing it in a pyramid and cascading structures. We jointly achieve spatial and temporal super-resolution. Different from video, our focus is on vector field restoration, where we not only consider the magnitude, but also the direction of vectors.

1.2. Deep learning in scientific visualization

As the great advances and success of deep learning in many areas, there have been increasing efforts in incorporating these methods in scientific visualization and physical simulation fields. For simulations, Kim *et al.* [10] propose a generative model to construct fluid simulation velocities from a set of parameters. In scientific visualization, Han *et al.* [7] applied a recurrent generative model for generating temporal super-resolution of a volume sequence data. Then Han *et al.* [6] extends the work to spaital super-resolution, where they used the generative adversarial network to generate high-resolution time-varying volumes. He *et al.* [8] trained a deep learning model to learn the visualization images from the simulation and viewing parameters to support parameter exploration. To our knowledge, there are no work on performing spatial-temporal super-resolution on vector field to improve the data reduction rate and post-hoc visualization results, which is our focus.

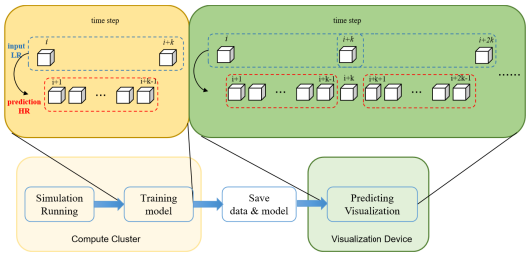


Figure 1. Overview of our pipeline.

2. Overview

We aim at performing super-resolution on vector fields in both spatial and temporal domains to address the problem that data are reduced at simulation time due to disk space constrains, and hence only a limited amount of data are stored for post-hoc analysis. We frame this problem as a supervised learning problem for training a neural network on a pair of low-resolution inputs for the purpose of high-resolution frames.

Figure 1 shows a high-level overview of the pipeline. First, given the high-resolution simulation data, we perform downsampling to generate training data pairs where we input low-resolution vector fields and produce a sequence of intermediate high-resolution vector fields to train our model, which is done by handling the super-resolution task both in spatial and temporal dimensions. We conduct simulation and train our model on high performance clusters. Second, we save the model and low-resolution data to portable visualization devices like laptops, then at inference time we can perform temporal and spatial super-resolution using the trained model to reconstruct high-resolution vector fields for post-hoc visualization.

3. STSRNet Architecture

In this section, we discuss our method for restoring the lost information due to I/O and storage constraints. The overall pipeline of our proposed joint framework is shown in Figure 2. We couple two different stages to tackle the problem. The first stage (red dashed box in Figure 2) is to estimate intermediate frames at a lower spatial resolution between the two end time steps. We use the U-Net [14] to estimate the motion between the frames at two end time steps

and then deform them forward and backward to synthesize the target frames. And the second stage (blue dashed box in Figure 2) handles the spatial super-resolution task to generate the final high-resolution outputs with temporal coherence maintained across these consecutive fields.

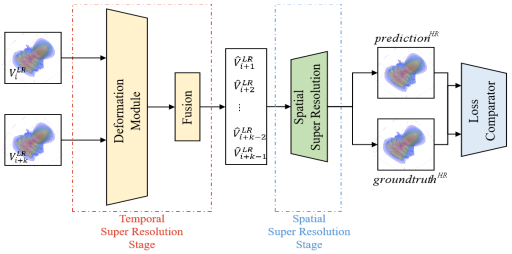


Figure 2. Framework of our method, having two separate stages to perform spatial-temporal super-resolution from low resolution inputs.

We denote the pair of input vector fields as $\mathbf{V}_i^{LR}(x)$ and $\mathbf{V}_{i+k}^{LR}(x)$, where $x \in \mathbb{R}^n$ ($n = 2$ for 2-D and 3 for 3-D), and LR represents the vector field V in spatial low-resolution (a coarse grid), and HR means the high-resolution field. Our ultimate goal is to use deep learning networks to learn a function f that models the vector fields between time step i and $i + k$, as

$$f(\mathbf{V}_i^{LR}, \mathbf{V}_{i+k}^{LR}) \approx \{\mathbf{V}_i^{HR}, \cdots, \mathbf{V}_{i+k}^{HR}\}, (k > 1) \quad (1)$$

3.1. Temporal Super-Resolution Stage

We first perform temporal super resolution with motion estimation. We generate the intermediate frames from both V_i and V_{i+k} bidirectionally. This stage comprises of two components: motion estimation and fusion. The architecture of each component is described as below.

Deformation with motion estimation Inspired by video frame interpolation works [3] [9], where optical flow $\mathbf{f}$ is estimated to determine the motion vector for each pixel in the frame. We focus on the dynamic changes along time between consecutive frames and aim to estimate the flow motion field.

Let $V_t(x)$ denotes a time-varying vector field, where x denotes the spatial coordinates (2 or 3 dimensions) and t denotes time. In this way, the intermediate vector field can be estimated by aligning the previous frame with the flow motion to the current step, and then adding the residual. Likewise it can also be generated from the later frame. Wan write the forward and backward alignments as:

$$\begin{aligned} \hat{V}_t^f(x) &= V_{t-1}(x - F_{t-1}(x)) + \Delta R_f \\ \hat{V}_t^b(x) &= V_{t+1}(x + F_t(x)) + \Delta R_f \end{aligned} \quad (2)$$

where $F_t(x)$ is the flow motion field.

Thus it is necessary to capture the forward and backward motion fied $F(x)$, and the residual ΔR. However, frame t is not available to compute the flow motion, so we estimate motion information between the the pair of key frames i and $i+k$, denoted as $F_{i\to i+k}$ and $F_{i+k\to i}$, to approximate the motion fields we need: $\hat{F}_{t\to i}$ and $\hat{F}_{t\to i+k}$. We employ the U-Net architecture [14] as our motion estimation network. Also, we use a rectify block to learn the residual ΔR between the deformed vector fields and the ground truth field.

Fusion After obtaining the forward and backward deformed frames $\hat{V}_t^f(x)$ and $\hat{V}_t^b(x)$, we introduce the visibility maps M_i and M_{i+k}, where $M_i(x) \in \{0, 1\}$ to denote the vector value from either forward and backward deformed frame contributing to the final result since generally averaging will produce blurry frame with artifacts, it will be trained in the model. Considering the temporal distance and the visibility map, the blended frame is

$$\hat{V}_t^{LR}(x) = (1 - \Delta t)M_0(x) \cdot \hat{V}_t^f(x) + \Delta t M_1(x) \cdot \hat{V}_t^b(x) \quad (3)$$

The synthesized low-resolution result $\hat{V}_t^{LR}(x)$ will be used in the following spatial super-resolution stage to generate the final high-resolution frames. In this way, the whole model will be trained jointly and enforces the estimated $\hat{V}_t^{LR}(x)$ to the real low-resolution frame.

3.2. Spatial Super-Resolution Stage

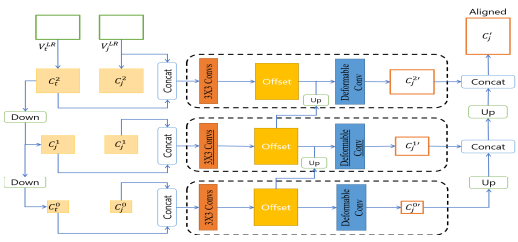


Figure 3. The architecture of multi-scale alignment.

To reconstruct spatial high-resolution frames, we align the context information from the neighboring frames instead of directly generating from the single low-resolution frame since the temporal information is indispensable. Deformable convolutional network has been proved successful in aligning frames to the reference without explicit motion estimation [18] [17]. The network will learn the offsets for convolution operation to fetch the information far away from the fixed kernel location in regular convolution. We also employ multi-scale structure with deformable alignment to handle the large displacement as in [18]. This kind of pyramid mechanism performs alignment from coarse to fine to improve accuracy as shown in Figure 3. Then, we take the fusion feature map C_t^f from the above module as input and outputs the corresponding final high-resolution frame utilizing a sub-pixel upscaling module with PixelShuffle.

3.3. Loss function

We design a novel loss function considering quality of reconstruction, directionality and structures of vector fields and temporal coherence. We not only use magnitude but the cosine angle loss to optimize the network. Moreover, the Jacobian matrix is also important for vector analysis, thus we introduce the jacobian loss to govern the training.

Magnitude loss

The most straightforward design to measure the difference between the generated field and the ground truth is the magnitude loss.

$$\mathcal{L}_v = \frac{1}{k-1}\sum_{t=i+1}^{i+k-1} ||\hat{V}_t - V_t^{GT}||_1 \tag{4}$$

Cosine distance loss

The directional loss function is defined as below.

$$\mathcal{L}_d = \frac{1}{k-1}\sum_{t=i+1}^{i+k-1} cosine\_distance(\hat{V}_t, V_t^{GT}) \tag{5}$$

where the $cosine\_distance$ is the mean value at every grid point.

Jacobian loss

For the velocity field, this high-order function needs multiple vector calculus operators such as divergence and curl. Thus, we introduce the jacobian loss defined as V, the gradient of it is the $n \times n$ jacobian matrix:

$$\mathcal{L}_j = \frac{1}{k-1}\sum_{t=i+1}^{i+k-1} ||\nabla\hat{V}_t - \nabla V_t^{GT}||_1 \tag{6}$$

where the ∇V is the jacobian function, and the size of jacobian matrix at every grid point is 2×2 for 2D and 3×3 for 3D. We calculate the mean difference between two vector fields for all grid points.

Temporal loss

To better maintain the temporal coherence and improve the accuracy of motion estimation, we introduce the temporal loss defined as

$$\begin{aligned}\mathcal{L}_T = &\frac{1}{k-1}\sum_{t=i+1}^{i+k-1} (||\hat{V}_t^f(x) - down(V_t^{GT})||_1 \\ &+ ||\hat{V}_t^b(x) - down(V_t^{GT})||_1)\end{aligned} \tag{7}$$

where $down$ denotes the downscaling operation.

Combining the above loss functions, we design our total loss as

$$min\mathcal{L} = \lambda_1\mathcal{L}_v + \lambda_2\mathcal{L}_d + +\lambda_3\mathcal{L}_j + \lambda_4\mathcal{L}_T \tag{8}$$

which is used for optimizing the whole model including temporal and spatial super-resolution stages.

4. Evaluations and Comparison

4.1. Data Sets and Implementation

Table 1. The dimension and time steps information of each simulation data set.

Data set	Dimension	Time steps	Size(GB)	Training time(h)
Solar Plume	$126 \times 126 \times 512$	28	2.7	15
Smoke	$112 \times 64 \times 32$	200	1.5	15
ESG ocean	3600×2400	45	32	21
Red sea	$500 \times 500 \times 50$	60	1500	40

We evaluated our framework using four data sets as shown in Table 1. In our implementation, the motion estimation module in the first stage using a U-Net architecture. At the second stage, we employed the Pyramid, Cascading and Deformable (PCD) structure for deformable alignment and residual blocks (RB) for the final reconstruction module. All the convolution filters have a 3×3 kernel size ($3 \times 3 \times 3$ in 3D). 75% of data is randomly sampled for training and the rest is used for testing, where we randomly

choose a time step and include its neighbors as a subsequence training sample until the selected time steps reaches 75% of the total data. For spatial down-sampling, we uniformly down-scale vector field using bicubic filter. We augmented the training data with affine transformation containing randomized rotations and flips. We optimized the model with the Adam optimizer. The learning rate was initialized as 4×10^{-4} and decayed as the number of epochs grew. The mini-batch size varied from 1 to 16 up to the data size. The scale factor for spatial resolution reduction was 4 by default along each dimension, and we synthesized the intermediate 5 frames from a pair of start and end frames. The model was implemented with Pytorch and conducted on 8 NVIDIA Tesla P100 GPUs.

4.2. Baselines and Metrics

Baselines. We compare our method with different baselines for evaluation.

- LERP + BI: Temporal linear interpolation for estimating intermediate frames and bicubic interpolation for upscaling the spatial field data.
- BiLSTM + CNN: Bidirectional LSTM is used for predicting intermediate frames, and we choose SRResNet [11], a post-upsampling method, as our CNN-based super-resolution model, which has several residual blocks followed by subpixel layers for upscaling results.

Evaluation Metrics. For quantitative evaluations, we used peak signal-to-noise ratio (PSNR), and mean of the closest point distancing (MCPD). We also use the λ_2 parameter to detect critical points to evaluate the differences in terms of features of vector fields.

PSNR (higher is better) measures the average difference among every grid point using the aggregated mean squared error between vector values.

$$\mathrm{PSNR}\left(\mathbf{V}, \hat{\mathbf{V}}\right) = 20 \log_{10} I(\mathbf{V}) - 10 \log_{10} \mathrm{MSE}\left(\mathbf{V}, \hat{\mathbf{V}}\right) \tag{9}$$

where $I(\mathbf{V})$ is the difference of the maximum and minimum value in $\mathbf{V}$.

A reconstructed vector field that yields high PSNR may still generate a wrong streamline due to the accumulation of errors during integration. Hence, we consider the similarity of particle trajectories(e.g., streamline, pathline) as another quantitative metric.

MCPD (lower is better) is used as the similarity measure for trajectories, which is the mean of the Euclidean distances between pairs of points formed by mapping each point of one trajectory to the closest point of the other. In this case, we use pathlines to evaluate the accumulated error considering temporal coherence.

$$\mathrm{MCPD}\left(S_i, S_j\right) = \mathrm{mean}\left(d_m\left(S_i, S_j\right), d_m\left(S_j, S_i\right)\right) \tag{10}$$

where $S_{i(j)}$ is a set of points $p_{k(l)}$, and

$$d_m(S_i, S_j) = \mathrm{mean}_{p_l \in F_i} \min_{p_k \in F_i} \|p_k - p_l\| \tag{11}$$

4.3. Qualitative and Quantitative Analysis

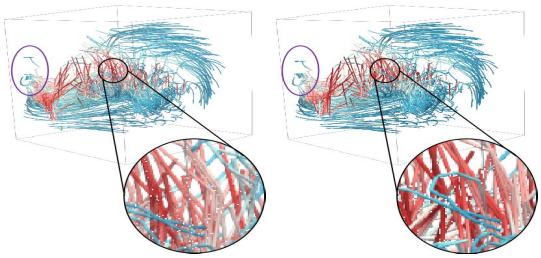
As stated in section 3, STSRNet has two stages that are trained as a whole. The first temporal super-resolution stage will generate intemediate results that are the input to the next spatial upscaling stage. Thus, to understand how each stage performs, it is necessary to compare the results separately.

4.3.1. Comparison of temporal super-resolution results To validate the effectiveness of our framework and the two modules in our network, we conduct a comprehensive study on the results. For evaluating the first temporal super-resolution module which generates low-resolution fields at a higher temporal rate, we compare the results against the baseline methods including linear interpolation (LERP) and BiLSTM.

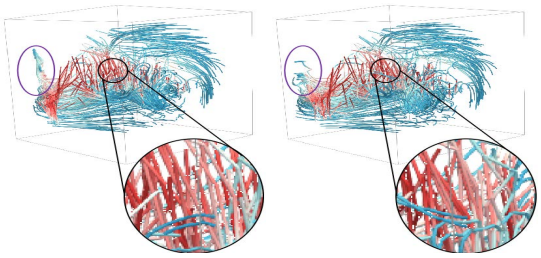
Table 2. Comparison of our first temporal super-resolution results against baselines with PSNR and MCPD.

Data set	Method	PSNR	MCPD
Solar plume	LERP	34.87	0.30
	BiLSTM	35.11	0.28
	ours	35.78	0.24
Smoke	LERP	35.31	0.29
	BiLSTM	39.50	0.25
	ours	39.56	0.23
ESG-Ocean	LERP	27.32	0.37
	BiLSTM	29.21	0.34
	ours	30.42	0.33
Red Sea	LERP	31.24	0.34
	BiLSTM	31.60	0.31
	ours	33.42	0.29

Table 2 reports the PSNR and MCPD results from using LERP, BiLSTM and first mod-



(a) Ground truth. (b) LERP.



(c) BiLSTM. (d) STSRNet (Only Temporal Stage).

Figure 4. Comparison of pathline results generated by only performing temporal super-resolution using plume data set under different methods.

ule in STSRNet with smoke data, plume data, global ocean data and red sea data. MCPD is computed from the pathlines we traced in the generated vector fields, where we fix the same seed points for all methods for a fair comparison as in Figure 4. We found that our temporal super-resolution method can generate results that achieve high PSNR and lower MCPD values.

In Figure 4, we visually compare the pathlines of high-resolution vector field generated by LERP, BiLSTM and our module using the plume dataset. We can see that in the purple circle, the LERP method produces inaccurate pathlines which is totally different from the ground truth. We can also observe that the same problems hold for BiLSTM method.

4.3.2. Comparison of final results To demonstrate the effectiveness of STSRNet, we compare the final estimated high-resolution vector fields with other baselines.

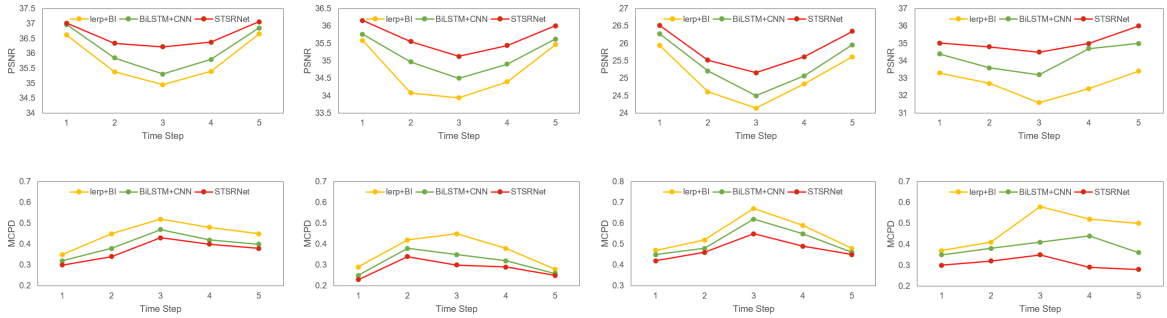
In Figure 5, we show the quantitative results at data level (PSNR) and feature level (MCPD) comparing our STSRNet against LERP+BI and BiLSTM+CNN. For PSNR, we can see that our method achieves better results at every time step. We also observe that for these three data sets, the closer the time step is to the ground truth time steps, i.e. towards the two ends, the higher PSNR value is. PSNR values fall when we move away from the two ends to the middle time step. For the solar plume data set, the curve is much steadier and smoother since data set itself changes relatively less. All three methods show low PSNR values in the global ocean data set because it exhibits fast-changing and complex behavior which is hard to predict. For MCPD, we can see that our method produces the lowest MCPD values which means more accurate results at the feature level. It is obvious that STSRNet is a winner in all data sets, which means STSRNet can preseve better temporal coherence when the vector fields have complex changing patterns. We can also see the similar trend that the results perform better in both ends and fall off in the middle.

Table 3. Comparison of our method against baselines with quantitative metrics.

Data set	Method	PSNR	MCPD
Solar plume	LERP+BI	35.72	0.35
	BiLSTM+CNN	36.15	0.32
	ours	36.60	0.30
Smoke	LERP+ BI	34.70	0.29
	BiLSTM+CNN	35.16	0.25
	ours	35.66	0.23
ESG-Ocean	LERP+ BI	25.0	0.37
	BiLSTM+CNN	25.40	0.35
	ours	25.83	0.32
Red sea	LERP+ BI	32.35	0.36
	BiLSTM+CNN	33.34	0.34
	ours	34.49	0.33

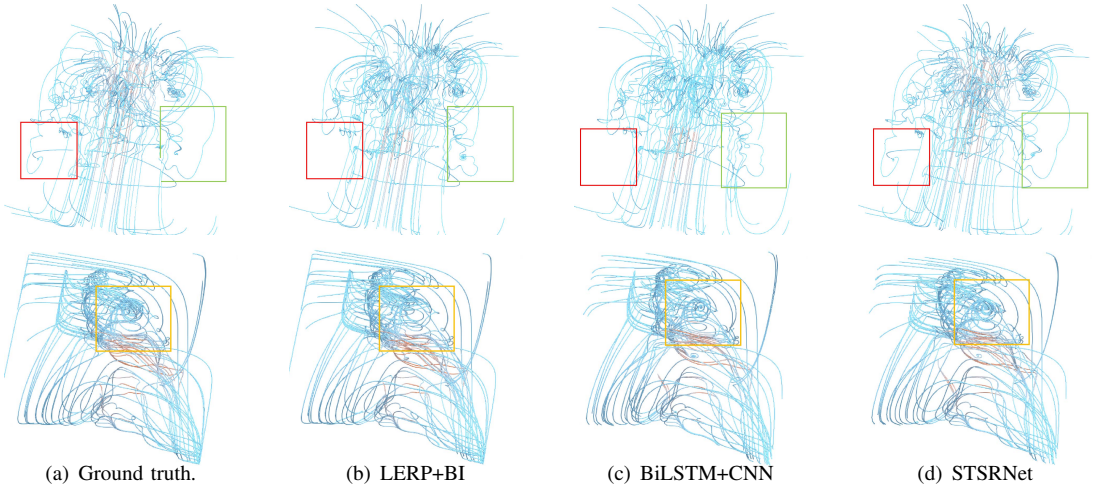
In Table 3, we report the quantitative comparison of our method with different datasets using PSNR and MCPD. The PSNR are the average value computed over the time sequence and the MCPD is the distance between the pathline generated from the whole sequence of vector fields. We can see that our method achieves higher PSNR and lower MCPD compared with the two baselines, which indicates our method can not only learn the data values but also capture the overall features.

In Figure 6, we compare the streamline rendering results of a single vector field generated by LERP+BI, BiLSTM+CNN and STSRNet. We can see that STSRNet can preserve better features such as critical points. For solar plume data set, LERP+BI and BiLSTM+CNN produce non-exist



(a) Solar Plume. (b) Smoke. (c) Global Ocean. (d) Red Sea.

Figure 5. Comparison of PSNR, and MCPD of generated streamline at each time step under LERP+BI and BiLSTM+CNN and STSRNet with different data sets.



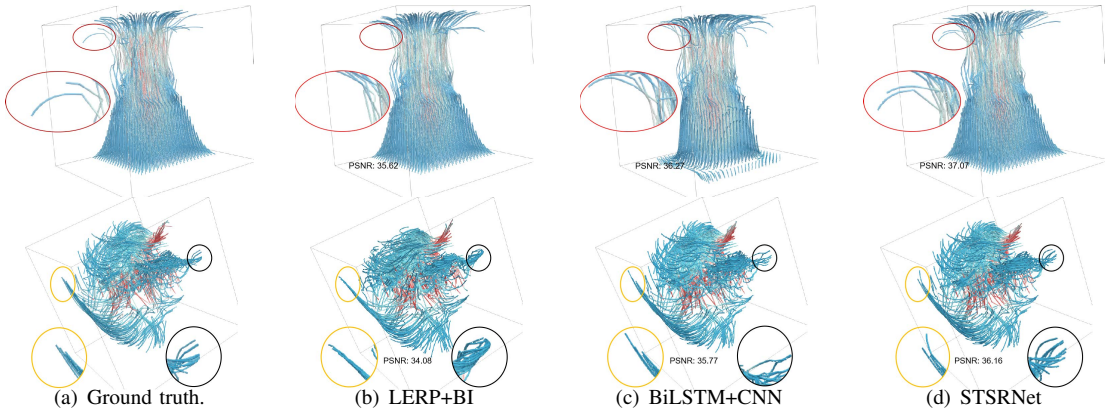
(a) Ground truth. (b) LERP+BI (c) BiLSTM+CNN (d) STSRNet

Figure 6. Comparison of steamline results. Top to bottom: plume data set, smoke data set.

vortexes and fail to generate streamline in right region, while STSRNet is closer to the ground truth. We highlight the obvious error regions made by LERP+BI and BiLSTM+CNN in red and green boxes. For smoke data set, STSRNet recover accurate streamlines around swirls at central region highlighted in yellow box, while other methods fail to recover the features.

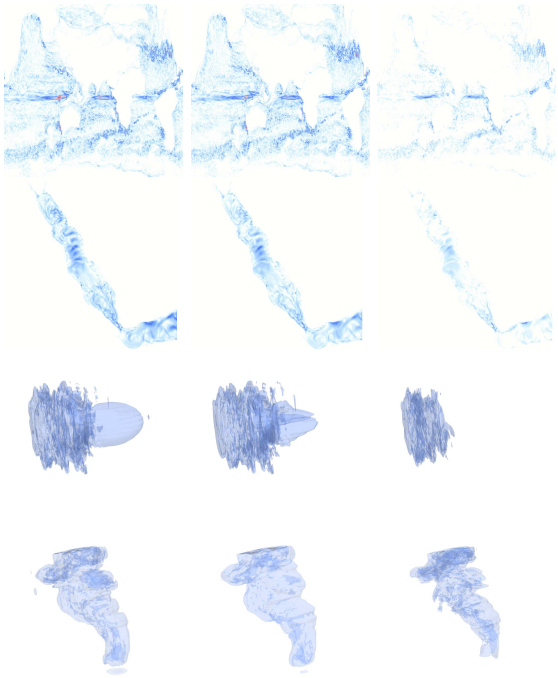
In Figure 7, we compare the pathlines rendering results of the vector field sequence generated by LERP+BI, BiLSTM+CNN and our STSRNet. For fairness, we use the same seed points to generate the pathlines for each data set. By comparing them with the ground truth, it is clear that STSRNet yields better results. For the smoke data set, we can see that in the red circles, our method is the only one that preserves the similar structure as in the ground truth. For solar plume data set, as in the yellow and green circles, we can observe that the linear interpolation method fails to reconstruct the right direction of the vector field. We can also see that in the black circles, our method leads to the most similar structure to the ground truth.

In Figure 8, we compare the volume rendering results of RMSE of vector fields generated by LERP+BI, BiLSTM+CNN, STSRNet compared with ground truth. We use the middle time step of the synthesized sequence of the vector fields since it has the most significant differences. For the global ocean data set, it is clear that our method introduces smaller errors across the global region especially around the vortices. For the red sea data set, we can see that STSRNet produces



(a) Ground truth. (b) LERP+BI (c) BiLSTM+CNN (d) STSRNet

Figure 7. Comparison of pathline results. Top to bottom: plume data set, smoke data set. We also reports the PSNR value for better comparison.



(a) LERP+BI (b) BiLSTM+CNN (c) STSRNet

Figure 8. Comparison of volume rendering of errors. Top to bottom: ocean data, red sea data, plume data, smoke data.

smaller errors compared with LERP+BI and BiLSTM+CNN. For the solar plume data set, both LERP+BI and BiLSTM+CNN generate the errors in the plume's tail region, while STSRNet only produces errors in the head region. For the smoke data set, STSRNet has smaller region of errors clearly.

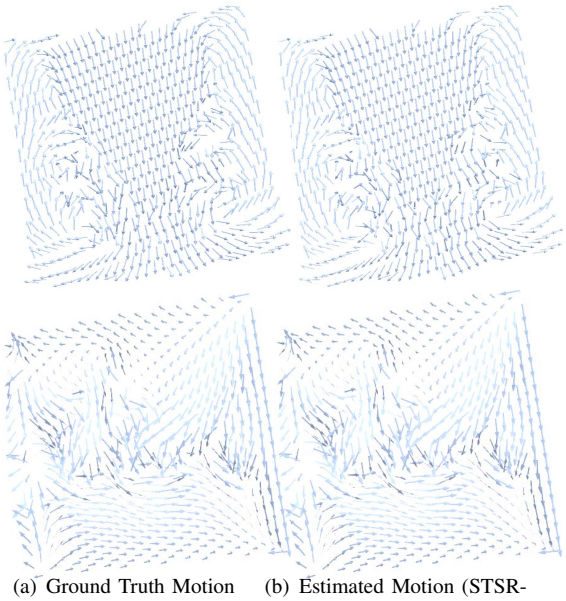
One advantage of super-resolution is that the size of data is reduced like compression methods but better quality can be achieved after the data are upscaled. Our method has no conflict with compression merthods since we can further compress our low-resolution data. To confirm this, we use lossless algorithm GZIP to compress our low-resolution data and quantitatively compare it against SZ compression method [5], a lossy compression, under same compression rate. The compression rate is the space for storing the original vector field data from the simulation divided by the space for storing the compressed data. From Table 4, we can see that our method achieves higher PSNR for all the data sets. Adding lossless compression can bring us an impressive compression rate (like 7680× for 3D solar plume data) without increasing error.

Table 4. Comparison of reconstruction PSNR for our method against SZ for each data set.

Data set	Compression Rate	Method	PSNR
Solar plume	7680×	SZ	14.67
		STSRNet+GZIP	36.60
Smoke	8697×	SZ	24.48
		STSRNet+GZIP	35.66
ESG-Ocean	396×	SZ	15.02
		STSRNet+GZIP	25.83
Red sea	614×	SZ	29.85
		STSRNet+GZIP	34.49

In our proposed framework, we first estimate the flow motion, which is a displacement field defined at the grid points, then we deform the previous frame to generate the next frame, or the other way round if the direction is backward. In Figure 9, we compare estimated motion field with the ground truth using plume and smoke data

set, where we visualize the motion with arrows to reveal its direction and magnitude. We slice the 3D data and sample the grid points for better clarity in the visualization. It is obvious that our estimated motion can recover the ground truth practically, apart from some arrows around the swirls.



(a) Ground Truth Motion (b) Estimated Motion (STSR-Net)

Figure 9. Comparison of estimated flow motion results with ground truth and our method. The left is ground truth and the right is our estimation. From top to bottom: plume data set, smoke data set.

To evaluate the effectiveness and accuracy of STSRNet in recovering flow specific featuresn. We use λ_2 parameter [1] for identifying the vortexes in the flow fields using the solar plume and smoke data sets. In Table 5, we calculate the average errors of the positions of critical points from the reconstructed data using our method and the ground truth. Clealy, STSRNet introduces smaller errors compared with the other two methods.

Table 5. Average errors of positions of critical points for LERP+BI, BiLSTM+CNN and STSRNet using plume data and smoke data set

Data set	LERP+BI	BiLSTM+CNN	STSRNet
Plume	0.377	0.373	0.017
Smoke	0.161	0.153	0.003

5. Conclusion and Future Work

We have presented a joint space-time super-resolution framework for vector fields. It can yield high-resolution intermediate frames with only a pair of low-resolution frames in a time interval as the input. Our framework consists of two modules. The first performs motion estimation and predicts the unavailable data frames at any time steps in between. And the second module reconstructs spatially high-resolution fields with deformable alignment and attention fusion. To achieve better performance, we introduce a novel loss function for vectors to capture the high-order feature of vector field. We explore the hyperparameters in our model including the loss function weights and network architecture. Through qualitative and quantitative comparisons, we validate the effectiveness of our module design for generating high quality results. In the future, we will study how to choose the best scale factors automatically, and our full model can be more than 100 MB in size, we would like to design lighter model without compromising the quality of the results.

6. ACKNOWLEDGMENT

The project was supported by National Key Research and Development Program of China (2019YFB1704201), and the National Key Scientific and Technological Infrastructure project “Earth System Science Numerical Simulator Facility” (EarthLab).

REFERENCES

1. On the identification of a vortex. *J Fluid Mech*, 1995.
2. Increasing space-time resolution in video. In *European Conference on Computer Vision*, 2002.
3. Wenbo Bao, Wei-Sheng Lai, Xiaoyun Zhang, Zhiyong Gao, and Ming-Hsuan Yang. MEMC-Net: Motion estimation and motion compensation driven neural network for video interpolation and enhancement. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, page 1–1, 2019.
4. Jose Caballero, Christian Ledig, Andrew Aitken, Alejandro Acosta, Johannes Totz, Zehan Wang, and Wenzhe Shi. Real-time video super-resolution with spatio-temporal networks and motion compensation. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jul 2017.

5. S. Di and F. Cappello. Fast error-bounded lossy hpc data compression with sz. In *2016 IEEE International Parallel and Distributed Processing Symposium (IPDPS)*, 2016.
6. Li Guo, Shaojie Ye, Jun Han, Hao Zheng, Han Gao, Danny Chen, Jian-Xun Wang, and Chaoli Wang. Ssr-vfd. 05 2020.
7. Jun Han and Chaoli Wang. TSR-TVD: Temporal super-resolution for time-varying data analysis and visualization. *IEEE Transactions on Visualization and Computer Graphics*, 26:205–215, 2019.
8. Wenbin He, Junpeng Wang, Hanqi Guo, Ko-Chih Wang, Han-Wei Shen, Mukund Raj, Youssef S. G. Nashed, and Tom Peterka. InSituNet: Deep image synthesis for parameter space exploration of ensemble simulations. *IEEE Transactions on Visualization and Computer Graphics*, page 1–1, 2019.
9. Huaizu Jiang, Deqing Sun, Varan Jampani, Ming-Hsuan Yang, Erik Learned-Miller, and Jan Kautz. Super SloMo: High quality estimation of multiple intermediate frames for video interpolation. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, Jun 2018.
10. Byungsoo Kim, Vinicius C. Azevedo, Nils Thuerey, Theodore Kim, Markus Gross, and Barbara Solenthaler. Deep Fluids: A generative network for parameterized fluid simulations. *Computer Graphics Forum*, 38(2):59–70, May 2019.
11. Christian Ledig, Lucas Theis, Ferenc Huszar, Jose Caballero, Andrew Cunningham, Alejandro Acosta, Andrew Aitken, Alykhan Tejani, Johannes Totz, Zehan Wang, and et al. Photo-realistic single image super-resolution using a generative adversarial network. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jul 2017.
12. U. Mudenagudi, S. Banerjee, and P. K. Kalra. Space-time super-resolution using graph-cut optimization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(5):995–1008, 2011.
13. Simon Niklaus, Long Mai, and Feng Liu. Video frame interpolation via adaptive separable convolution. *2017 IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.
14. Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-Net: Convolutional networks for biomedical image segmentation. *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*, page 234–241, 2015.
15. O. Shahar, A. Faktor, and M. Irani. Space-time super-resolution from a single video.
16. Xin Tao, Hongyun Gao, Renjie Liao, Jue Wang, and Jiaya Jia. Detail-revealing deep video super-resolution. *2017 IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.
17. Yapeng Tian, Yulun Zhang, Yun Fu, and Chenliang Xu. TDAN: Temporally deformable alignment network for video super-resolution, 2018.
18. Xintao Wang, Kelvin C. K. Chan, Ke Yu, Chao Dong, and Chen Change Loy. EDVR: Video restoration with enhanced deformable convolutional networks, 2019.
19. Xiaoyu Xiang, Yapeng Tian, Yulun Zhang, Yun Fu, Jan P. Allebach, and Chenliang Xu. Zooming Slow-Mo: Fast and accurate one-stage space-time video super-resolution, 2020.

Yifei An is currently pursuing M.S. degree in University of Chinese Academy of Sciences, Beijing, China. Her research interest is flow visualization. Contact her at yifei.an1996@gmail.com.

Han-Wei Shen is with the Ohio State University, Ohio, the United States. His primary research interests are scientific visualization and computer graphics. Contact him at shen.94@osu.edu.

Guihua Shan is with Computer Information Network Center, Chinese Academy of Sciences, Beijing, China. Her research interests are scientific visualization and interactive visualization. Contact her at sgh@sccas.cn. She is the corresponding author.

Guan Li is with Computer Information Network Center, Chinese Academy of Sciences, Beijing, China. His research interests are scientific visualization and in-situ visualization. Contact him at liguan@sccas.cn.

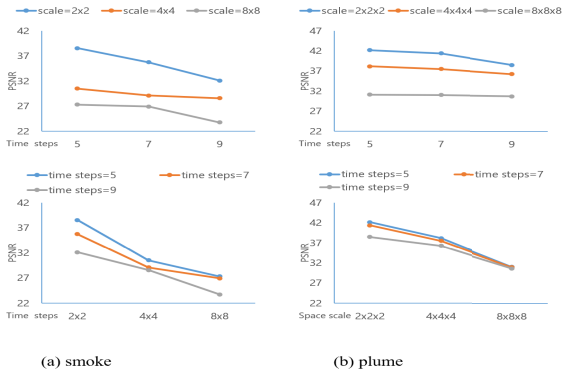
Jun Liu is with Computer Information Network Center, Chinese Academy of Sciences, Beijing, China. His research interests are scientific visualization and parallel computing. Contact him at liujun@sccas.cn.

Appendix

Evaluation for hyperparameters

In this section we validate the effectiveness of our default hyperparameters settings. We study the trade-off between different scaling factors in space and time and evaluate our method trained with different weighted combinations of loss items. We use validation data that were not used for training to inference the results.

Spatial and Temporal Scale Factors We analyze the trade-off between the scale factors and the results' quality. In Figure 10, we compare the results with different downscale factors in SSR and different time intervals in TSR independently. In the top row of the figure, each curve represents a fixed spatial downscale factor. When increasing the length of the time interval, we can see that the PSNR values change very little. In the bottom row, each curve represents a fixed time interval. It is shown that the quality degrades with higher scale factor. This means that the spatial super-resolution scale has more impact to the model performance, since it may lose more information.



(a) smoke (b) plume

Figure 10. Quantitative evaluation with of different scale factors and time steps we use for model training.(a) smoke data set (b) solar plume data set.

Loss Functions In subsection 3.3 we demonstrate that the overall loss is a linear combination of the different loss terms with different weights λ_i as in Equation 8. We tested the different weights used in the loss function to analyze the influence of the loss function to the reconstruction quality. We use the fixed default scale factor and time step data mentioned to validate the loss function independently.

Table 6. Evaluation the different weight λ_i settings for the combination of loss functions.

Network	Losses	PSNR	MCPD
M1	$\mathcal{L}_v + \mathcal{L}_d$	29.46	0.64
M2	$\mathcal{L}_v + \mathcal{L}_d + \mathcal{L}_T$	30.61	0.52
M3	$10\mathcal{L}_v + \mathcal{L}_d + 5\mathcal{L}_T + \mathcal{L}_j$	37.51	0.32

Table 6 shows the quantitative results with different combinations of loss items trained on the plume data set. It is shown that the M2 network gives the better PSNR results comparing to the M1 network since it involves temporal cohenrence. From the results in M2 and M3, we can see that providing the high-order loss $\mathcal{L}_j$ can improve the MCPD metric. It is crucial to capture the higher order differences so particle pathlines will not drift far away from the ground truth. Given the selected loss combinations, we find that the M3 network gives the most reliable performance. Therefore, we take it as our preferred model.