TSR-TVD: Temporal Super-Resolution for Time-Varying Data Analysis and Visualization

Jun Han and Chaoli Wang

University of Notre Dame

Motivation

- Background: Scientists Simulate long sequence, but Store limited

- Goal: Augment the temporal resolution
  - Given a low-resolution volume sequence, eg 100 time steps
  - Generate the high-resolution volume sequence, eg 500 time steps

Challenges & method

- C_1: Volume data could be non-linear
  - Standard linear interpolation (LERP) would fail
  - Could not capture complex evolution & non-linear changes
- C_2: Visual quality
  - Typical recurrence-based NN only measure voxelwise distance
  - High PSNR, no high-quality rendering results (blurry)
- Propose temporal super-resolution (TSR) from time-varying data (TVD), noted as TSR-TVD

Main contribution

- Apply RGN (a combination of RNN & GAN) to generate TSR
  - Natural Idea: RNN for temporal data, GAN for generating images (also consider GAN as part of loss function)

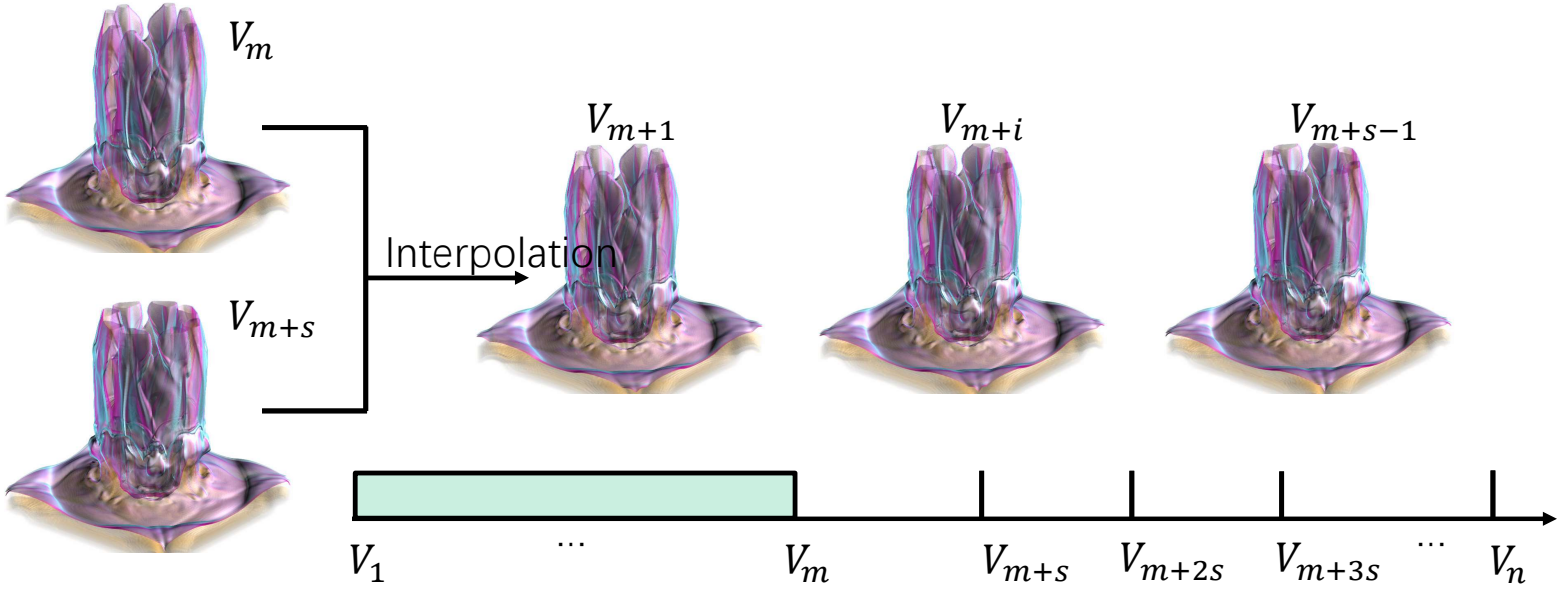
- Apply ConvLSTM layers & propose a new voxel shuffle layer
  - ConvLSTM layers: capture spatiotemporal relationship
  - voxel shuffle layer: accelerate the training process

- A new architecture for the TSR task
- Investigate hyperparameter settings & analyze the impact on performance

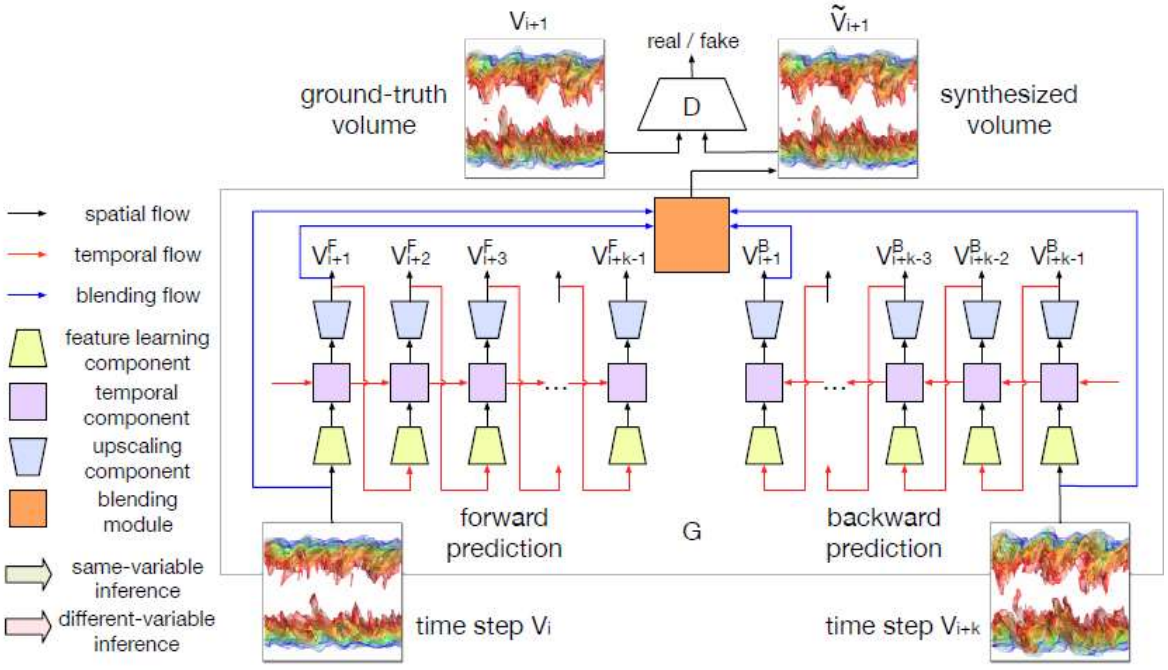
Problem definition

- Given a pair of volumes (V_i, V_{i+k})

- seek a function $\Phi(V_i, V_{i+k}) \approx V$, where $V = \{V_{i+1}, V_{i+2}, \cdots, V_{i+k-1}\}$, which are the intermediate volumes between V_i & V_{i+k}



recurrent generative approach



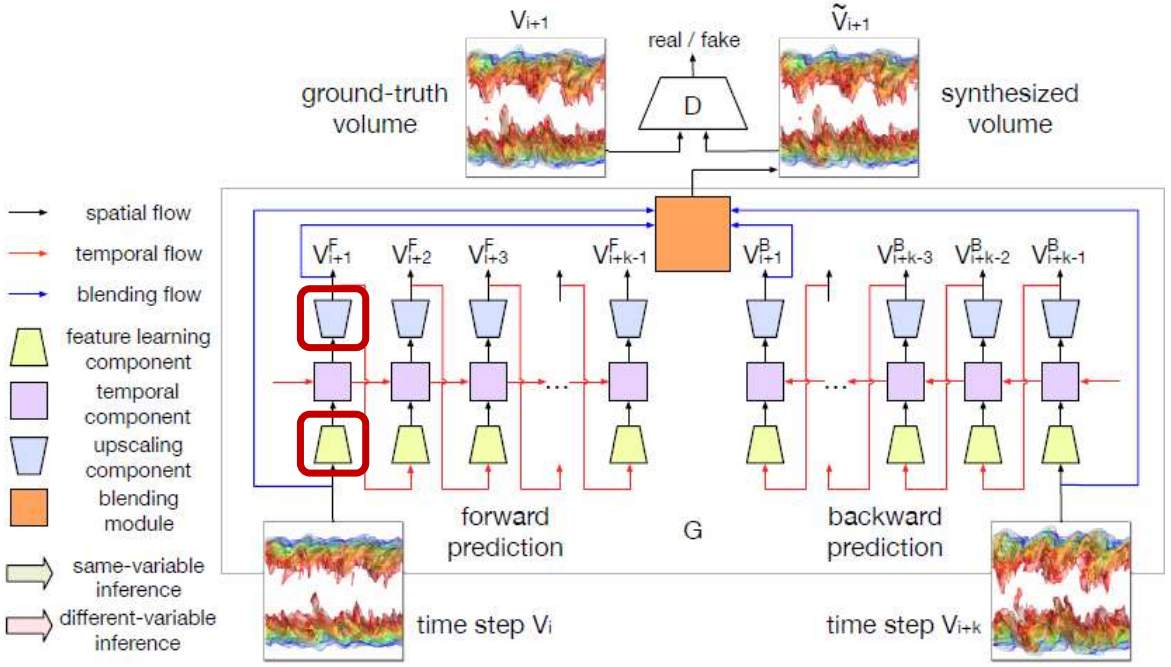
- A generator G & a discriminator D
- G —— two modules
  - Predicting module

$$\mathbf{V}^{\mathrm{F}} = \phi^{\mathrm{F}}_{\mathrm{PREDICT}}(V_i),$$
$$\mathbf{V}^{\mathrm{B}} = \phi^{\mathrm{B}}_{\mathrm{PREDICT}}(V_{i+k}),$$
$$\phi_{\mathrm{PREDICT}} = \{\phi^{\mathrm{F}}_{\mathrm{PREDICT}}, \phi^{\mathrm{B}}_{\mathrm{PREDICT}}\}.$$

  - Blending module

$$\tilde{\mathbf{V}} = \phi_{\mathrm{BLEND}}(V_i, V_{i+k}, \mathbf{V}^{\mathrm{F}}, \mathbf{V}^{\mathrm{B}}).$$

Reason for blend module



- In down-sample & up-sample network, detail features may be lost in down-sample & not be perfected recovered in up-sample
- Adding V_i, V_{i+k} help to eliminate noise

Loss function

- Notation
  - Input volume pairs $\mathbf{V}^T = \{(V_1, V_{k_1}), (V_{k_1}, V_{k_2}), \cdots, (V_{k_{n-1}}, V_{k_n})\}$
  - Ground truth intermediate volumes

$$\mathbf{V}^I = \{\{V_2, \cdots, V_{k_1-1}\}, \{V_{k_1+1}, \cdots, V_{k_2-1}\}, \cdots, \{V_{k_{n-1}+1}, \cdots, V_{k_n-1}\}\}$$

  - Learnable parameters in G and D: θ_G and θ_D
  - The maximal interpolation step K (i.e. k_{i+1}-$k_i \leqslant K$, $i \in [0, n-1]$)

Adversarial loss

$$\min_{\theta_G} \mathscr{L}_G = \mathbb{E}_{V \in \mathbf{V}^T} \left[\log D(G(V)) \right],$$

$$\min_{\theta_D} \mathscr{L}_D = \frac{1}{2} \mathbb{E}_{V \in \mathbf{V}^I} \left[\log D(V) \right] + \frac{1}{2} \mathbb{E}_{V \in \mathbf{V}^T} \left[\log \left(1 - D(G(V)) \right) \right],$$

- G and D compete with each other
- Intuition: encourage a **perceptual solution** rather than **PSNR or SSIM orientated solution**, such as L_1 loss (lead to blurry result)
- Limiation: using Adversarial loss alone lead to unstable training for G and D
- Solution: consider volumetric loss & feature loss

- (Althougt this equation seems wrong on the min/max)
  - the original GAN value function

$$\min_G \max_D V(D,G) = \mathbb{E}_{\boldsymbol{x} \sim p_{\text{data}}(\boldsymbol{x})} [\log D(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{z} \sim p_{\boldsymbol{z}}(\boldsymbol{z})} [\log(1 - D(G(\boldsymbol{z})))].$$

Feature loss

$$\mathscr{L}_F = \mathbb{E}_{V' \in \mathbf{V}^T, V \in \mathbf{V}^I} \sum_{k=1}^{N-1} \frac{1}{N_k} [||F^k(G(V')) - F^k(V)||_2],$$

- compute the feature difference between V^I and $G(V^T)$ based on D
- Features extracted from every convolution layer of D except the final layer
- N – the total number of Conv layers in D
- N_k – the number of elements in the kth Conv layer
- Fk - feature representation extracted from the kth Conv layer
- Feature loss is pretty similar with the VGG loss in style transfer

Volumetric loss

$$\mathscr{L}_V = \mathbb{E}_{V' \in \mathbf{V}^T, V \in \mathbf{V}^I} \left[||G(V') - V||_2 \right],$$

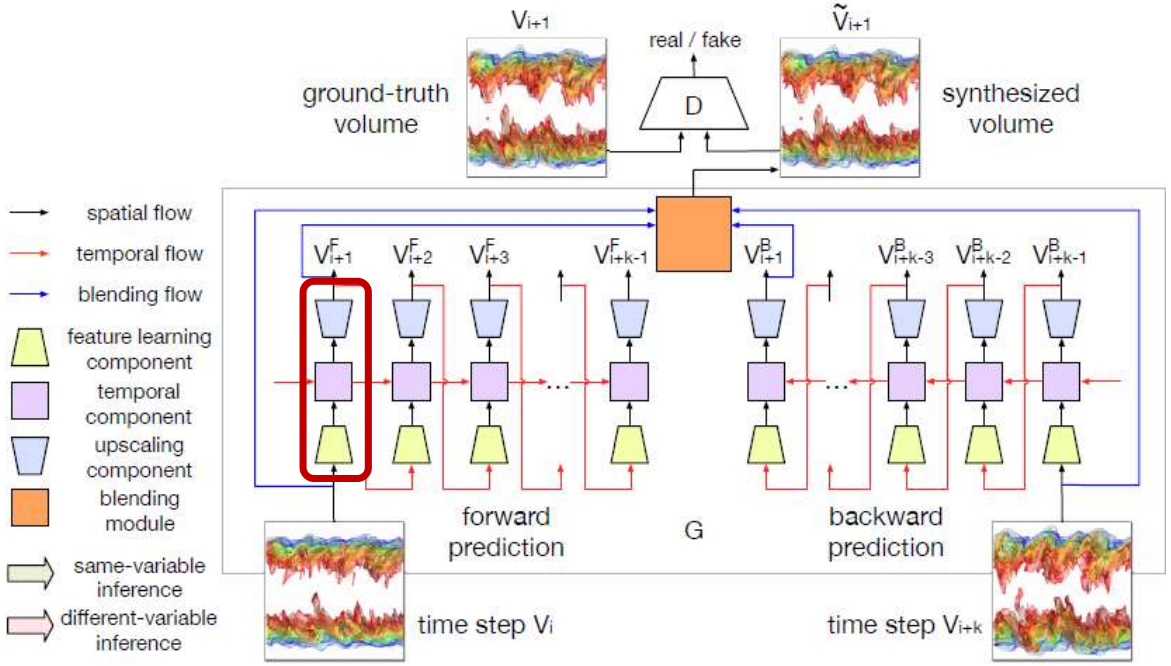
- Intuition: make the result of G not only fool D but also close to the GT in L_2 sense
- Also improve training stability

final loss

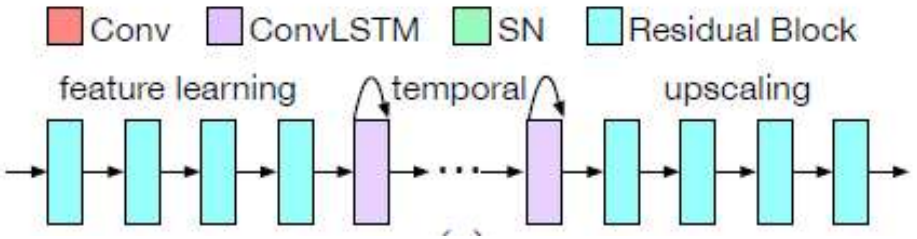
$$\min_{\theta_G} \mathscr{L}_G = \lambda_1 \left(\mathbb{E}_{V \in \mathbf{V}^T} \left[(D(G(V)) - 1)^2 \right] \right) + \lambda_2 \mathscr{L}_V + \lambda_3 \mathscr{L}_F,$$

- λ_1, λ_2, λ_3 are hyperparameters, which control the relative importance of three losses

Network architecture - G predicting module

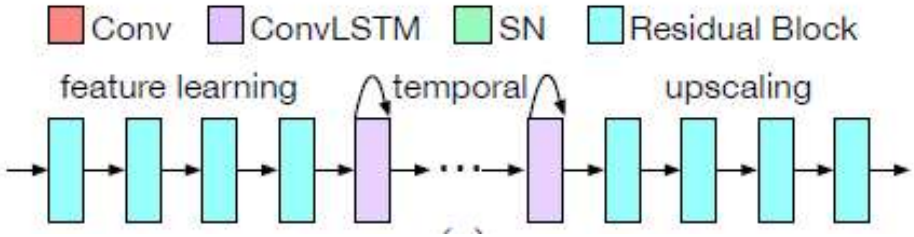


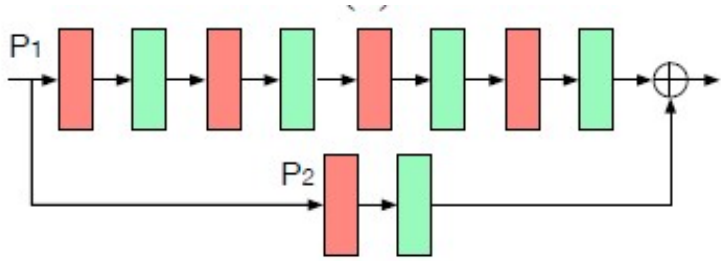
- Feature learning component
  - Downscale getting feature representation by redisual blocks
- Temporal component
  - with multiple ConvLSTM layers
- Upscaling component
  - Upscale by residual blocks



Feature learning component

- Extract feature representations from the volumes
- Core: four advanced residual blocks contains P_1 and P_2
- P_1: four Conv layers followed by spectral normalization and ReLU
- P_2: one Conv layer followed by SN and ReLU
- Bridged by skip connection



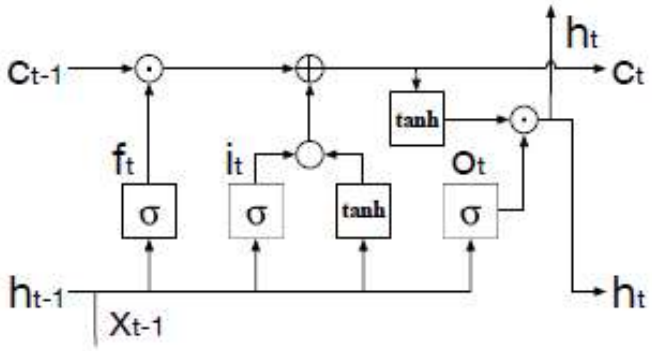


- Kernel size
  - first residual block 5x5x5
  - Last three residual blocks 3x3x3
- Stride 2
- Feature maps nums: 16, 32, 64, 64

Temporal component

- ConvLSTM
  - Predict the the next volumes via the previous volumes
- Difference between traditional LSTM
  - The matrix multiplication in trainditional LSTM is replaced by convolution operation

- (W_{xf}, W_{hf}, b_f), (W_{xi}, W_{hi}, b_i), (W_{xo}, W_{ho}, b_o), and (W_{xc}, W_{hc}, b_c) are learnable parameters
- ConvLSTM does not change the resolution



$$\mathbf{f}_t = \sigma(\mathbf{W}_{xf} \star \mathbf{x}_t + \mathbf{W}_{hf} \star \mathbf{h}_{t-1} + \mathbf{b}_f),$$
$$\mathbf{i}_t = \sigma(\mathbf{W}_{xi} \star \mathbf{x}_t + \mathbf{W}_{hi} \star \mathbf{h}_{t-1} + \mathbf{b}_i),$$
$$\mathbf{o}_t = \sigma(\mathbf{W}_{xo} \star \mathbf{x}_t + \mathbf{W}_{ho} \star \mathbf{h}_{t-1} + \mathbf{b}_o),$$
$$\mathbf{c}'_t = \tanh(\mathbf{W}_{xc} \star \mathbf{x}_t + \mathbf{W}_{hc} \star \mathbf{h}_{t-1} + \mathbf{b}_c),$$
$$\mathbf{c}_t = \mathbf{i}_t \odot \mathbf{c}'_t + \mathbf{f}_t \odot \mathbf{c}_{t-1},$$
$$\mathbf{h}_t = \mathbf{o}_t \odot \tanh(\mathbf{c}_t),$$

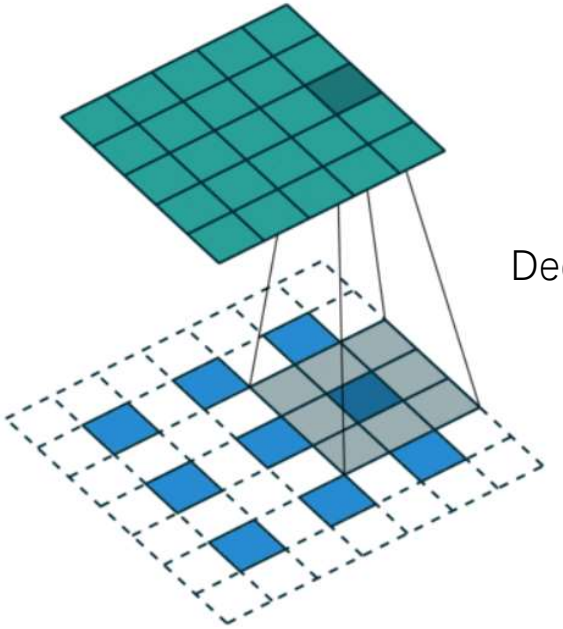
$\star$ is the convolution operation

Upscaling component

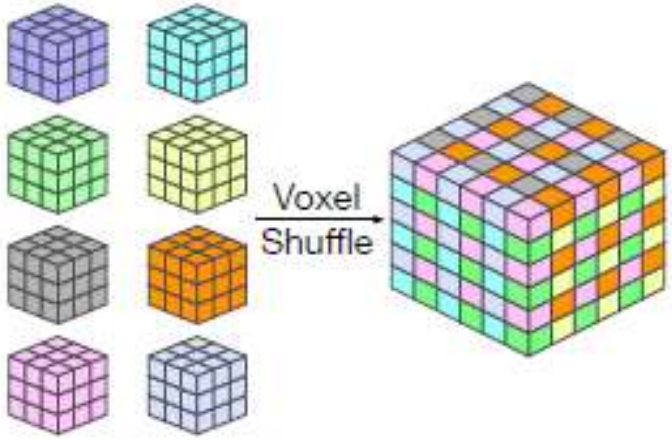
- Instead of using deconvolution layers because of high computational cost and unnecessary zero padding
- Propose a voxel shuffle layer

$$\mathbf{O} = \mathscr{S}(\mathbf{W} \star \mathbf{I} + \mathbf{b}),$$

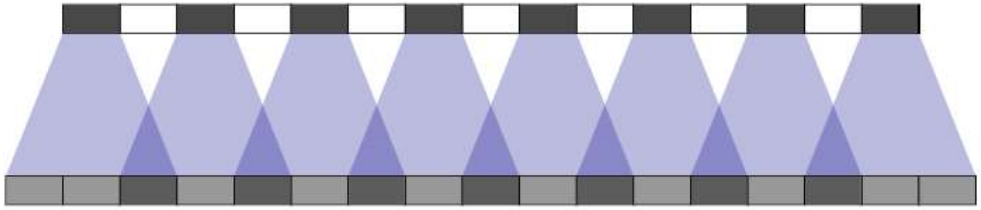
- I and O - the input and output
- W and b - the learnable parameters
- φ – a pereiodic shuffle operation that rearranges the elements of a [Cf^3, L, W, H] -> a tensor of size [C, fL, fW, fH]

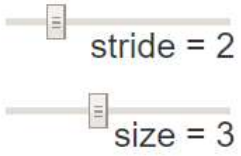


Deconvolution

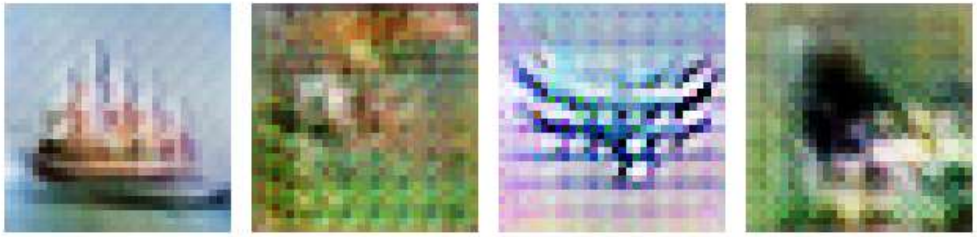


Upscaling component - Deconvolution - checkerboard effect



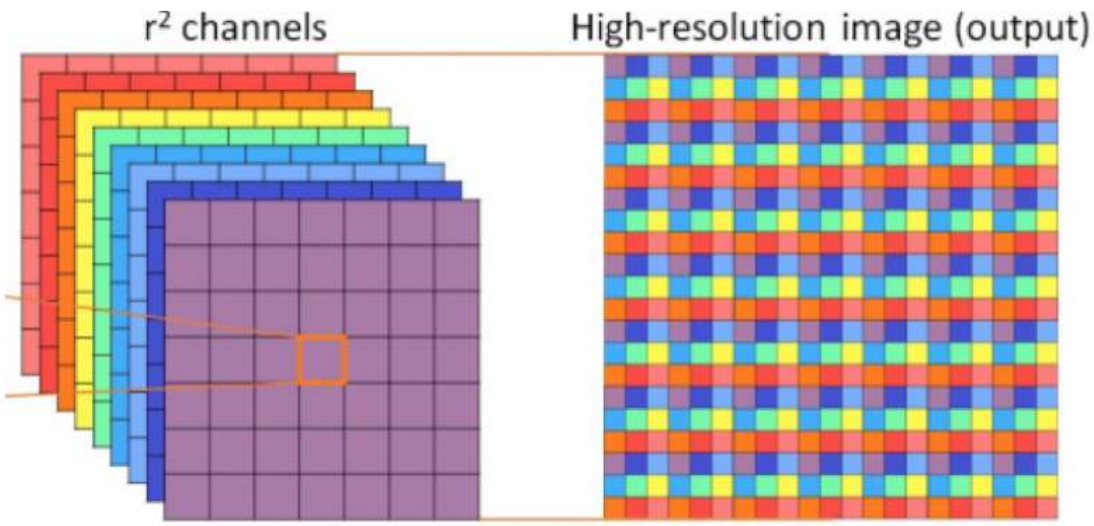


- uneven overlap
- when the kernel size (the output window size) is not divisible by the stride (the spacing between points on the top)



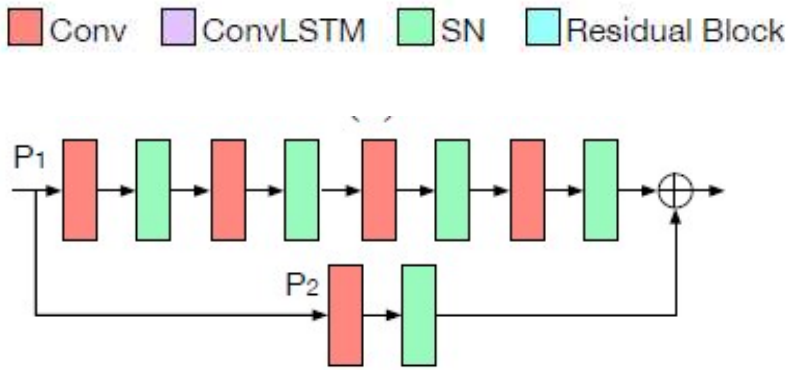
Upscaling component (cont)

- Pixel shuffle layer (2D version of voxel shuffle)



Upscaling component (cont)

- Architucture the same as the feature learning component
- Difference: add voxel shuffle layer after the last SN layer in P_1 and P_2



- Kernel size
  - first three residual block 3x3x3
  - Last residual block 5x5x5
- Up-scaling factor 2 for each voxel shuffle layer
  - means they would use 2^3C kernels
- Feature maps num: 64, 32, 16, 1

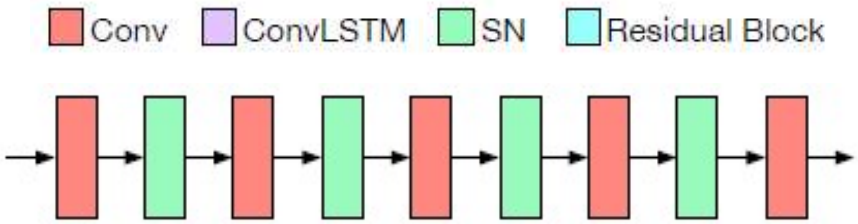
Blending module

- Inputs: forward prediction V_i^F and backward prediction V_i^B, the input volume pair V_{k_i} and $V_{k_{i+1}}$
- Output: final synthesized volume $\tilde{V}_i$

$$\tilde{\mathbf{V}}_i = w_i V_{k_i} + (1 - w_i) V_{k_{i+1}} + \frac{1}{2}(\mathbf{V}_i^{\mathrm{F}} + \mathbf{V}_i^{\mathrm{B}}),$$

- Where w_i is the weight control the importance of V_{k_i} and $V_{k_{i+1}}$.
- We can regard $w_i V_{k_i} + (1 - w_i) V_{k_{i+1}}$ as the interpolation result and $\frac{1}{2}(\mathbf{V}_i^{\mathrm{F}} + \mathbf{V}_i^{\mathrm{B}})$, as the residual

Discrinimator



- Several Conv and SN layers with leaky ReLU activation
- Avoid pooling layers
- Conv layer kernel size: 4 x 4 x 4 kernel size, feature maps 64, 128, 256, 512, 1
- First four layers: strided convolutions for reducing the volume resolusion.
  - Stride = 2
- Last layer: produce an output of size 1 x 1 x 1 & do not apply the activation function

Data Sets

Table 1. The dimensions and training epochs of each data set.

data set (variable)	dimension ($x \times y \times z \times t$)	epochs
climate (cam-fv)	$360 \times 181 \times 30 \times 31$	100
climate (fim)	$360 \times 181 \times 30 \times 31$	100
combustion (HR)	$480 \times 720 \times 120 \times 121$	100
combustion (MF)	$480 \times 720 \times 120 \times 121$	100
combustion (YOH)	$480 \times 720 \times 120 \times 121$	100
ionization (He)	$600 \times 248 \times 248 \times 100$	100
ionization (He+)	$600 \times 248 \times 248 \times 100$	100
solar plume	$126 \times 126 \times 512 \times 28$	200
supernova (E)	$256 \times 256 \times 256 \times 60$	200
supernova (VM)	$256 \times 256 \times 256 \times 60$	200
vortex	$128 \times 128 \times 128 \times 90$	200

Network training

- Random crop subvolumes with size 64 x 64 x 64
- Learning rates for G and D are different
  - individual learning rate for both the discriminator and the generator convergence to a stationary local Nash equilibrium
- For each epoch, the numbers of times they optimizes the discriminator and generator n_D and n_G are different, $n_D \geqslant n_G$
  - Discriminator must be powerful in order to get high quality images

Network training

- Hyperparameters in the loss function

$$\min_{\theta_G} \mathscr{L}_G = \lambda_1 \left(\mathbb{E}_{V \in \mathbf{V}^T}\left[\left(D(G(V)) - 1\right)^2\right]\right) + \lambda_2 \mathscr{L}_V + \lambda_3 \mathscr{L}_F,$$

- λ_1 is smaller than λ_2, λ_3
  - since they do not want to pay more attention to the adversarial loss and generate novel volumes rather volumes close to the GT

- The maximal interpolation step K
  - **Training phase:** K =3
    - Reason: large K would lead to gradient vanishing & prevent TSR-TVD from finding the globally optimal solution
  - **Inference phase:** K can be increased
    - Reason: The gradient computation is not required

Network training

Algorithm 1 TSR-TVD training algorithm.

Require: initial generator parameters θ_G and initial discriminator parameters θ_D

Require: number of D updates n_D per G iteration, number of G updates n_G per D iteration, number of training epochs T, learning rates α_G and α_D for G and D, respectively

for $t = 1 \cdots T$ **do**
  for $i = 1 \cdots n$ **do**
    sample $(V_{k_i}, V_{k_{i+1}})$ and $(V_{k_i+1}, \cdots, V_{k_{i+1}-1})$
    for $1 \cdots n_D$ **do**
      compute $\mathscr{L}_D$ according to Equation 6
      $\theta_D = \theta_D - \alpha_D \frac{\partial \mathscr{L}_D}{\partial \theta_D}$
    end for
    for $1 \cdots n_G$ **do**
      compute $\mathscr{L}_G$ according to Equation 9
      update θ_G according to Equation 21
    end for
  end for
end for

Evaluation metrics

- peak signal-to-noise ratio (PSNR) – data level

$$\mathrm{PSNR}(\mathbf{V},\mathbf{V}') = 20\log_{10} I(\mathbf{V}) - 10\log_{10}\mathrm{MSE}(\mathbf{V},\mathbf{V}'),$$

  - V and V' are the original and synthesized volumes
  - I(V) is the difference between the maximum and minimum values of V
  - MSE(V, V') is the mean squared error between V and V'

- Structural similarity index (SSIM) – image level

$$\mathrm{SSIM}(\mathbf{I},\mathbf{I}') = \frac{(2\mu_{\mathbf{I}}\mu_{\mathbf{I}'} + c_1)(2\sigma_{\mathbf{I},\mathbf{I}'} + c_2)}{(\mu_{\mathbf{I}}^2 + \mu_{\mathbf{I}'}^2 + c_1)(\sigma_{\mathbf{I}}^2 + \sigma_{\mathbf{I}'}^2 + c_2)},$$

  - I and I' are sub-images from the rendered images of V and V'
  - μ_I and $\mu_{I'}$ are the average values of I and I'
  - $\sigma_{\mathbf{I}}^2$ and $\sigma_{\mathbf{I}'}^2$ are the variances of I and I'
  - $\sigma_{\mathbf{I},\mathbf{I}'}$ is the covariance of I and I'
  - c_1 and c_2 are two small constants for stabilizing the division with the denominator.

Evaluation metrics

- Similarity between two isosurface extracted
  - Multual information between corresponding distance fields
    - Isosurface with isovalue h

$$L_h = \left\{ x \in \mathbb{R}^3 : f(x) = h \right\}$$

    - Distance field for a volume

$$D_h(x) = \min_{\forall y \in L_h} d(x,y)$$

    - Multual information

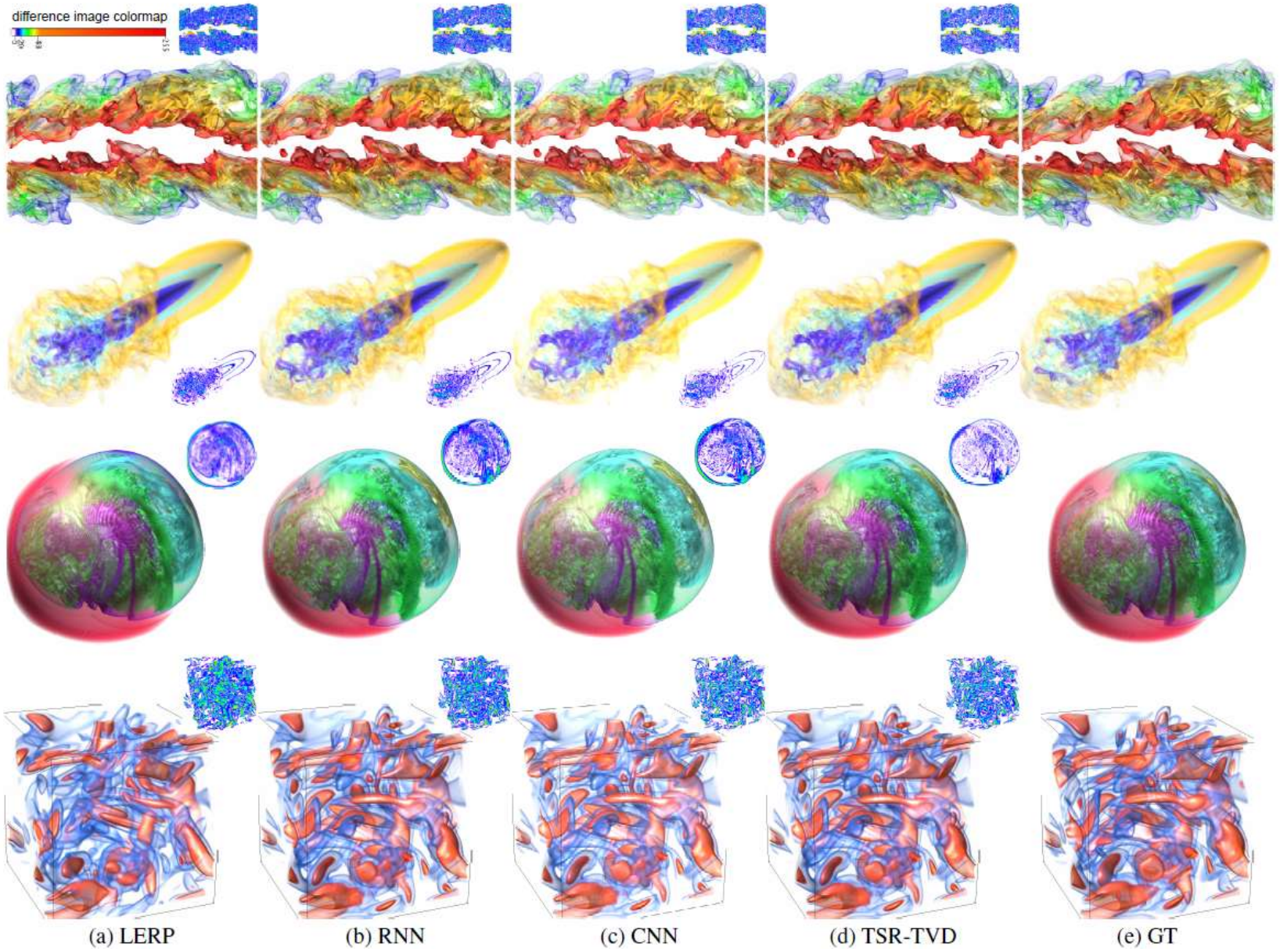
$$\hat{I}(X,Y) = \frac{2I(X,Y)}{H(X)+H(Y)}$$

  - The larger the isosurface value, the more similar two surfaces are.

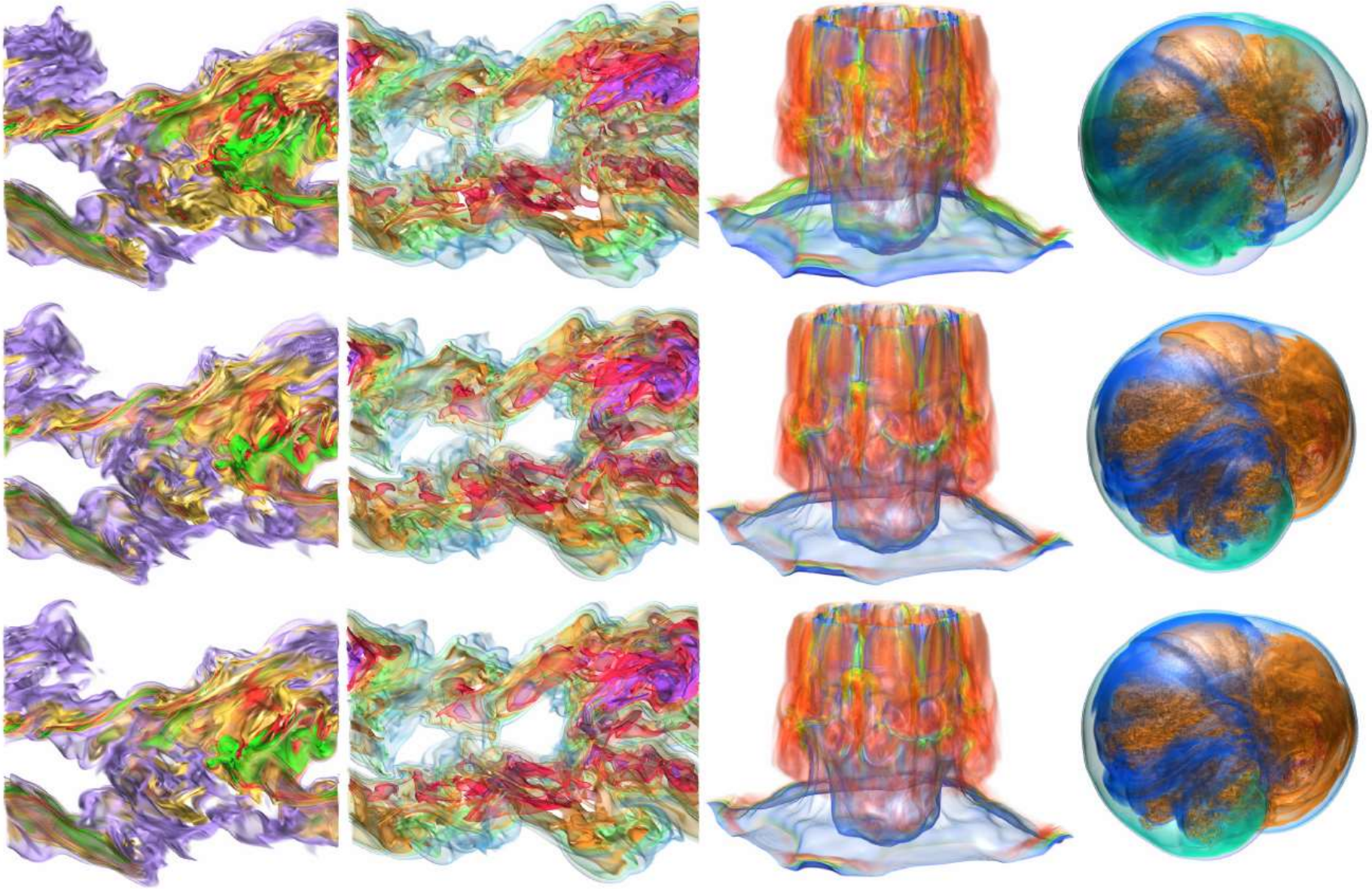
Analysis

- Baselines
  - LERP
  - RNN
    - The architecture of TSR-TVD but eliminate the discriminator
  - CNN
    - The architecture of TSR-TVD without ConvLSTM as the built-in temporal coherence predictor

- Calculate the pixelwise differences of images generated from generated and GT volumes
- Map noticeable pixel difference to nonwhite colors

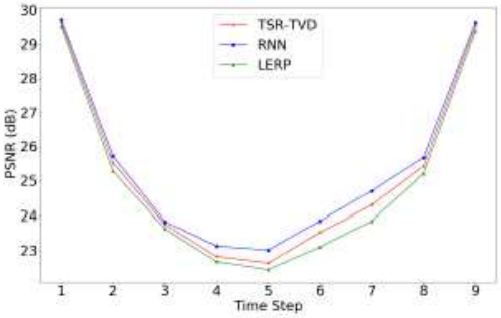


- Combustion:
  - RNN CNN TSR-TVD more detail
  - TSR-TVD less artifacts
- Solar plume:
  - Four close results
  - TSR-TVD closest
- Supernova:
  - TSR-TVD beter visual quatliy
  - LERP worst
  - RNN CNN artifacts on the right side
- vortex:
  - LERP worst
  - RNN CNN TSR-TVD similar results

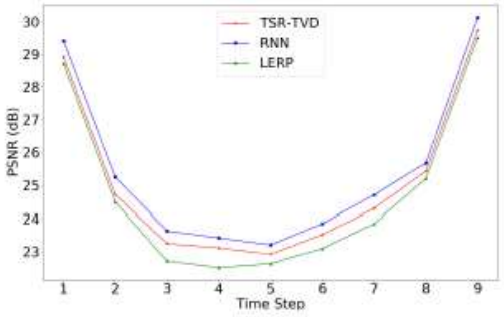


(a) combustion (MF → HR) (b) combustion (MF → YOH) (c) ionization (He → He+) (d) supernova (VM → E)

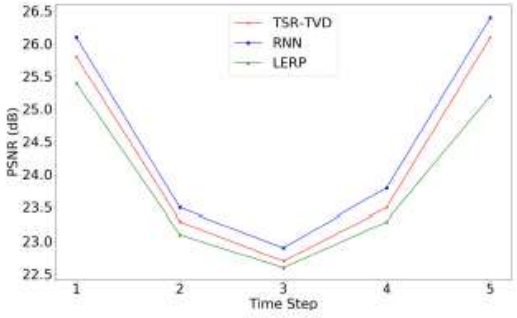
- Different variable inference
  - X for training
  - Y for inference
- Top to Bottom: LERP TSR-TVD GT



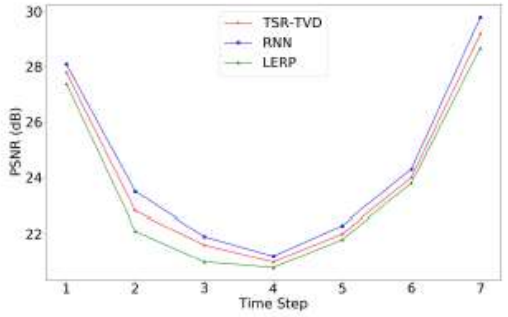
(a) combustion (HR)



(b) combustion (MF)



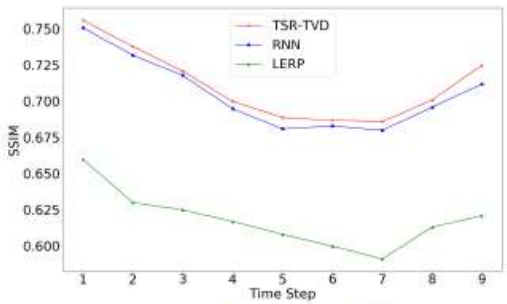
(c) supernova (E)



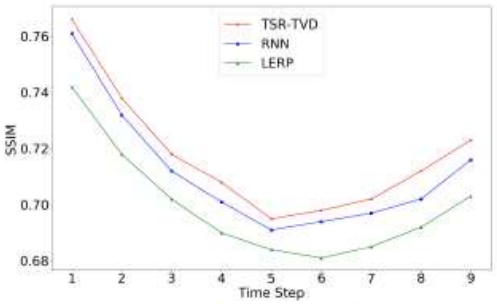
(d) vortex

- Comparison of PSNR of synthesized volumes using LERP, RNN, and TSR-TVD.
- Data level: RNN highest PSNR value
  - RNN is PSNR-oriented, PSNR would consider GAN-loss & perceptual loss
- PSNR cruve: peak at the end, fall as moving into center
- lower PSNR values for supernova
  - Reason: the supernova exhibits a fast-pacing rotational behavior

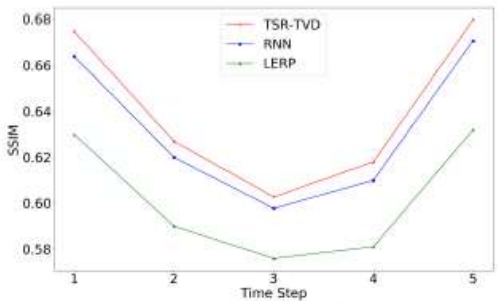
	PSNR (dB)			
data set (variable)	LERP	RNN	CNN	TSR
combustion (HR)	25.61	**26.13**	25.72	25.81
combustion (MF)	25.12	**25.86**	25.43	25.62
supernova (E)	22.34	**24.31**	23.81	23.74
vortex	26.62	**27.42**	26.85	26.90



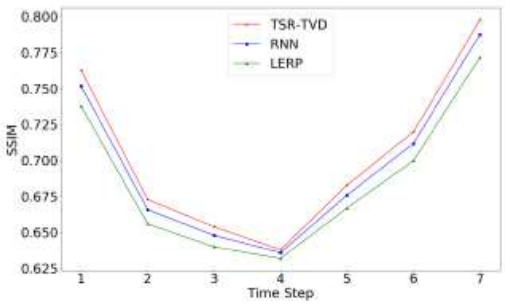
(a) combustion (HR)



(b) combustion (MF)



(c) supernova (E)

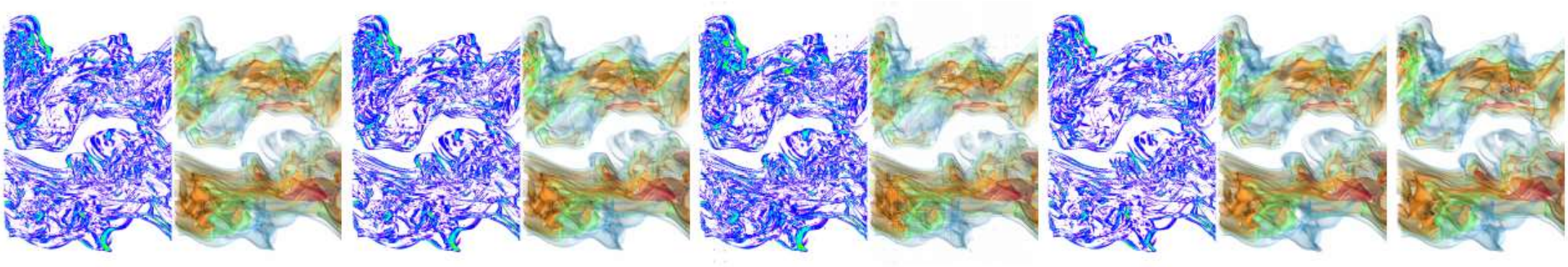


(d) vortex

- Comparison of PSNR of synthesized volumes (top row) and SSIM of rendered images (bottom row) using LERP, RNN, and TSR-TVD.
- Image level: TSR-TVD yields the highest SSIM values.

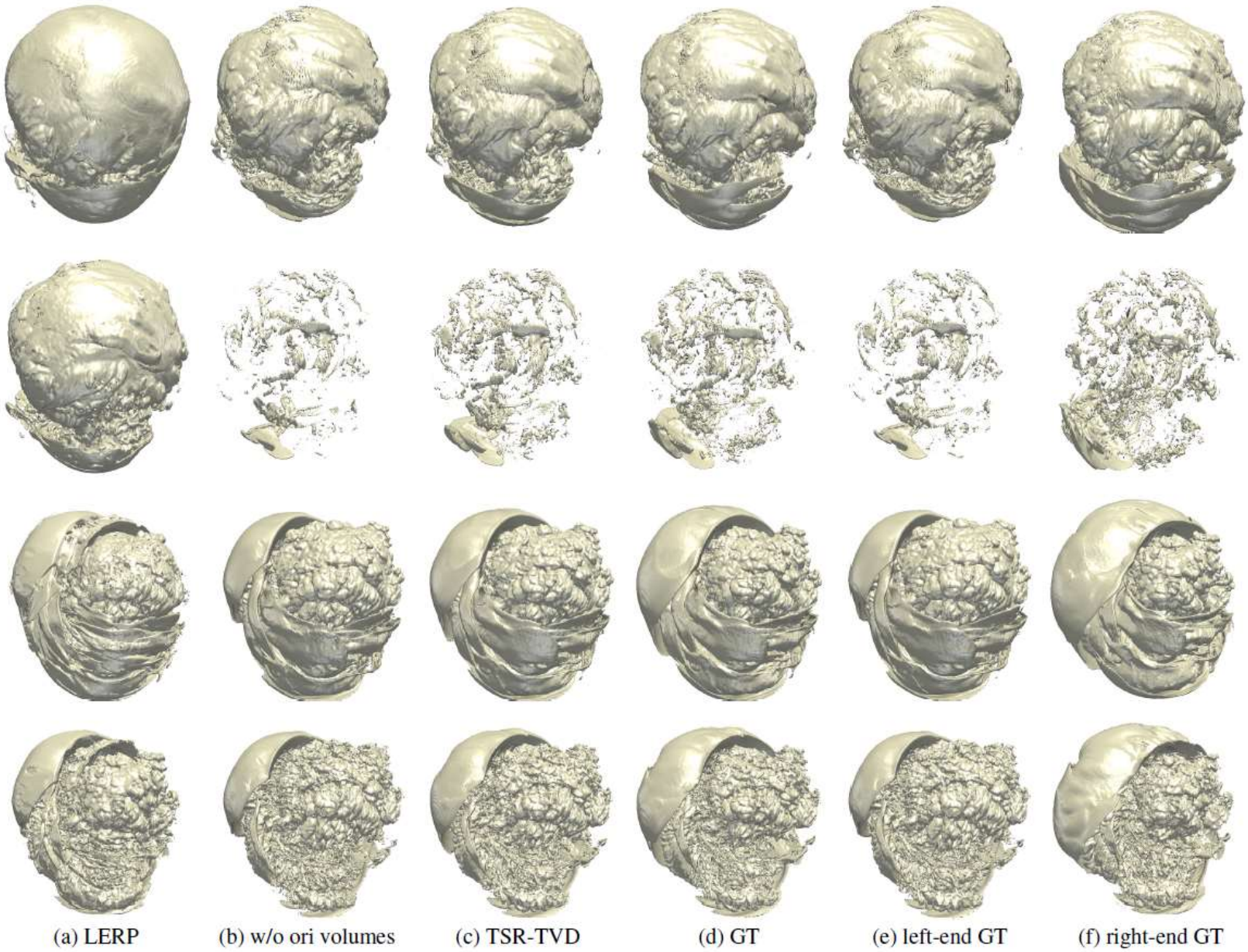
Table 2. Comparison of average PSNR and SSIM values.

	PSNR (dB)				SSIM			
data set (variable)	LERP	RNN	CNN	TSR	LERP	RNN	CNN	TSR
combustion (HR)	25.61	**26.13**	25.72	25.81	0.66	0.70	0.69	**0.72**
combustion (MF)	25.12	**25.86**	25.43	25.62	0.71	0.73	0.73	**0.74**
supernova (E)	22.34	**24.31**	23.81	23.74	0.61	0.64	0.63	**0.66**
vortex	26.62	**27.42**	26.85	26.90	0.73	0.75	0.75	**0.75**



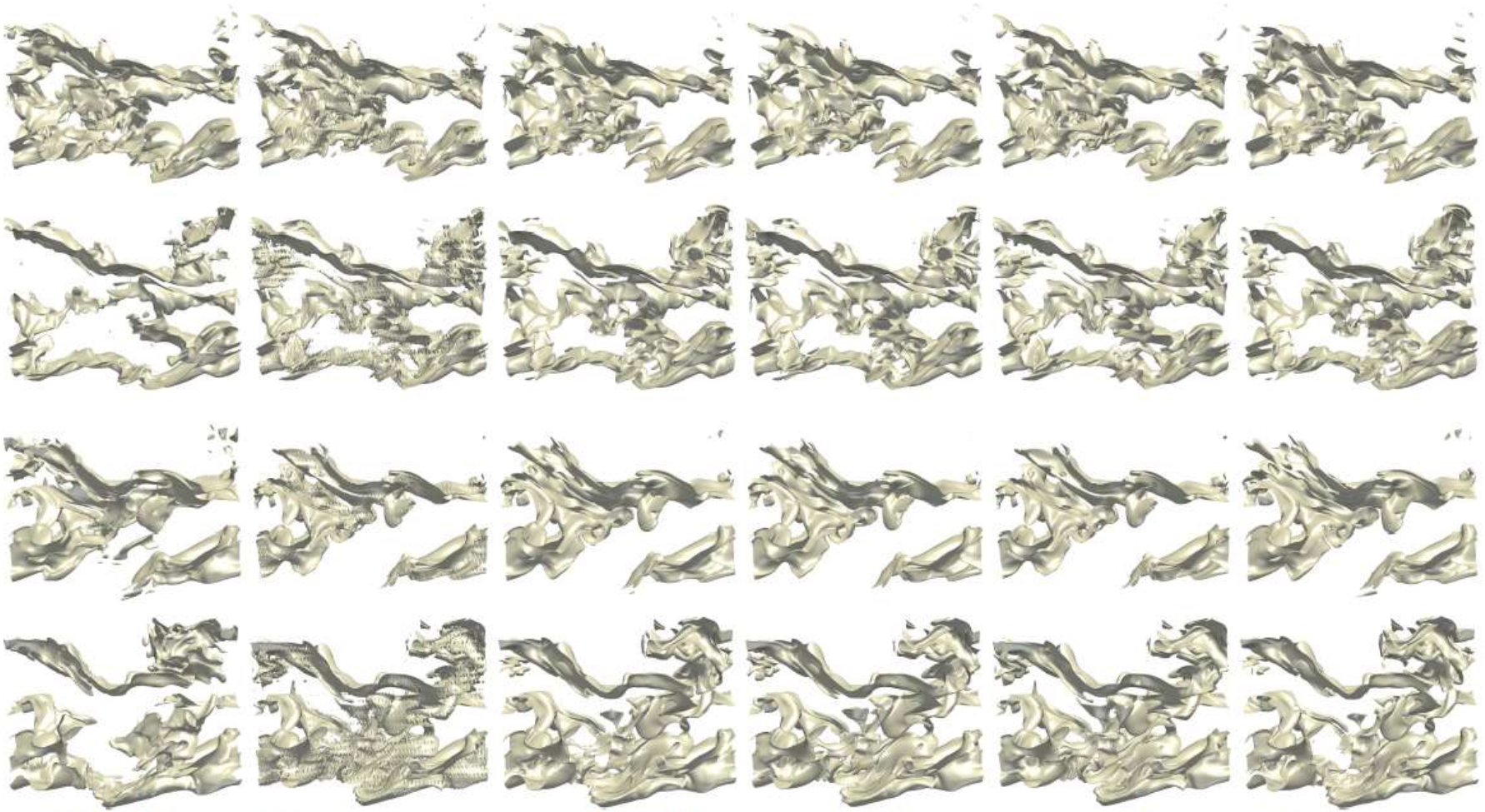
(a) w/o fwd prediction (b) w/o bwd prediction (c) w/o original volumes (d) full model (e) GT

- Variants
  - a, b only backward or forward prediction
    - Results well, but lack detail information
    - closed light cyan part at the middle-right corner
  - c without adding the original volumes into the blending module
    - Obvious noise
    - Reason: downsample & upsample module lose information at downsample phase but unable to recover at upsample phase



(a) LERP (b) w/o ori volumes (c) TSR-TVD (d) GT (e) left-end GT (f) right-end GT

- supernova (E)
  - Two timsteps and two isovalues
  - First row: time steps 39 with v = 0.
  - Second row: time steps 39 with v = 0.176.
  - Third row: time steps 55 with v = 0.
  - Last row: for time steps 55 with v = 0.176
- TSR-TVD is able to produces a surface very close to the GT



(a) LERP (b) w/o ori volumes (c) TSR-TVD (d) GT (e) left-end GT (f) right-end GT

- combustion (HR)
  - First row: time steps 97 with v = 0.255.
  - Second row: time steps 97 with v = 0.569.
  - Third row: time steps 114 with v = 0.255.
  - Last row: for time steps 114 with v = 0.569.

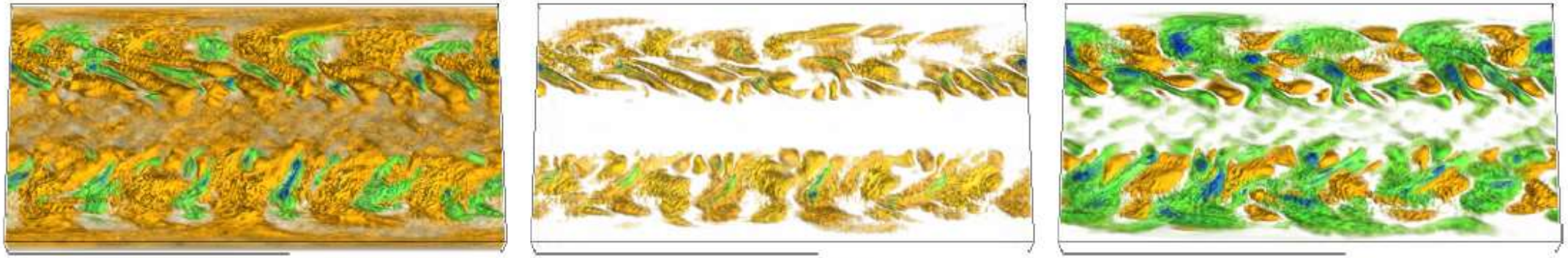
Comparision

Table 3. Comparison of average IS values at selected isovalues.

	LERP		TSR-TVD	
data set (variable)	$v = 0$	$v = 0.176$	$v = 0$	$v = 0.176$
supernova (E)	0.56	0.24	**0.71**	**0.68**
	$v = 0.255$	$v = 0.569$	$v = 0.255$	$v = 0.569$
combustion (HR)	0.65	0.59	**0.73**	**0.72**

- the average IS values over the entire volume sequence
- TSR-TVD leads to isosurfaces of better quality than LERP

Failure case



(a) LERP (b) TSR-TVD (c) GT

- same-variable inference using the climate (fim) data set
- Both TSR-TVD and LERP fail – green -> yellow, blue vanish
- Reason: the limitation of TSR-TVD in estimating the difference between data distributions

Failure case

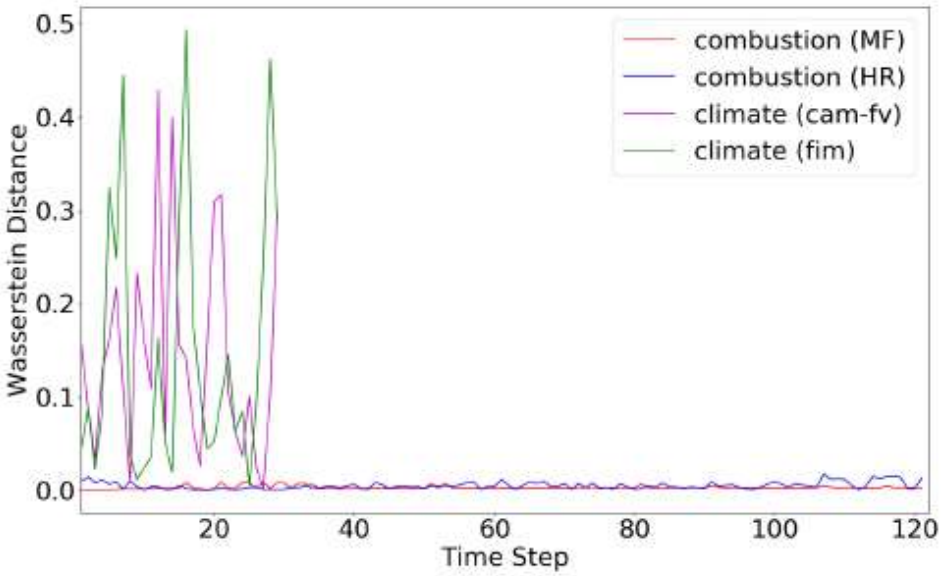


Fig. 5. Comparison of Wasserstein distances using the climate (cam-fv and fim) and combustion (HR and MF) data sets.

- Wasserstein distance to measure the similarity of two neighboring time steps from two different data sets
- combustion (HR and MF)
  - small distance
- climate (cam-fvand dim)
  - Distance fluctuate drasitically
  - Average distance is high
- TSR-TVD limitation of capturing the fluctuation between data distributions.

Discussion

- 40% training samples & subvolume size of 64x64x64
  - six hours training time on combustion data (100 epoch)
    - Crop 19200 subvoluems
    - 100 epochs, 4 subvolumes for each volume pair, 48 volume pairs (1 ~ 5, 2 ~ 6, ⋯, 48 ~ 52)
  - Half a day on supernova (200 epoch)

- Inference time:
  - Main constrain: GPU memory limitation
  - Small data such as vortex, whole volume as input
  - Big data such as combustion, still crop
    - 170,100 subvolumes
    - 18 volume pairs (52 ~ 56, 56 ~ 60, ⋯, 118 ~ 122), 30 times along x dimension, 45 times along y dimension, 7 times along z dimension
    - Overlap in space to avoid spatial discontinuity
    - Concatenate through a weighted concatenation algorithm

Limitation

- Did not consider transfer function or the visualization process
  - Incoporating transfer function in the traing phase, boost the performance at the image level
  - But may demand training from scratch whenever the transfer function changes

- only infer the intermediate volumes at any integer time steps rather than arbitrary time steps

- Only considers temporal coherence through the recurrent module.
  - better way: incorporate temporal coherence into loss function design

Conclusion & future work

- TSR-TVD a deep learning solution for generating temporal super-solution of time-varying data
- Resolve volume sequences for the rest of time series via same-variable inference or different-variable inference
- Compared with LERP, RNN, CNN better quality

- Besides temporal super-resolution, consider spatial super-resolution
- Eventual goal: achieve spatiotemporal super-resolution