Computer Vision and Image Understanding 168 (2018) 79–92

Contents lists available at ScienceDirect

Computer Vision and Image Understanding

journal homepage: www.elsevier.com/locate/cviu

Video super-resolution based on spatial-temporal recurrent residual networks

Wenhan Yang[a], Jiashi Feng[b], Guosen Xie[c], Jiaying Liu[*,1,a], Zongming Guo[a], Shuicheng Yan[d]

[a] Institute of Computer Science and Technology, Peking University, Beijing 100871, PR China
[b] Department of Electrical and Computer Engineering, National University of Singapore, 117583, Singapore
[c] NLPR, Institute of Automation, Chinese Academy of Sciences, Beijing 100190, PR China
[d] Artificial Intelligence Institute, Qihoo 360 Technology Company, Ltd., Beijing 100015, PR China

ARTICLE INFO

Keywords:
Spatial residue
Temporal residue
Video super-resolution
Inter-frame motion context
Intra-frame redundancy

ABSTRACT

In this paper, we propose a new video Super-Resolution (SR) method by jointly modeling intra-frame redundancy and inter-frame motion context in a unified deep network. Different from conventional methods, the proposed Spatial-Temporal Recurrent Residual Network (STR-ResNet) investigates both spatial and temporal residues, which are represented by the difference between a high resolution (HR) frame and its corresponding low resolution (LR) frame and the difference between adjacent HR frames, respectively. This spatial-temporal residual learning model is then utilized to connect the intra-frame and inter-frame redundancies within video sequences in a recurrent convolutional network and to predict HR temporal residues in the penultimate layer as guidance to benefit estimating the spatial residue for video SR. Extensive experiments have demonstrated that the proposed STR-ResNet is able to efficiently reconstruct videos with diversified contents and complex motions, which outperforms the existing video SR approaches and offers new state-of-the-art performances on benchmark datasets.

1. Introduction

Video super-resolution (SR) aims to produce high-resolution (HR) video frames from a sequence of low-resolution (LR) inputs. In recent years, video super-resolution has been drawing increasing interest from both academia and industry. Although various HR video devices have been developed constantly, it is still highly expensive to produce, store and transmit HR videos. Thus, there is a great demand for modern SR techniques to generate HR videos from LR ones.

The video SR problem, as well as other signal super-resolution problems, can be summarized as restoring the original scene $\mathbf{x}_t$ from its several quality-degraded observations $\{\mathbf{y}_t\}$. Typically, the observation can be modeled as

$$\mathbf{y}_t = \mathbf{D}_t\mathbf{x}_t + \mathbf{v}_t,\ t = 1,\ ...,T. \tag{1}$$

Here $\mathbf{D}_t$ encapsulates various signal quality degradation factors at the time instance *t, e.g.*, motion blur, defocus blur and down-sampling. Additive noise during observation at that time is denoted as $\mathbf{v}_t$. Generally, the SR problem, i.e., solving out $\mathbf{x}_t$ in Eq. (1), is an ill-posed linear inverse problem that is rather challenging. Thus, accurately estimating $\mathbf{x}_t$ demands either sufficient observations $\mathbf{y}_t$ or proper priors on $\mathbf{x}_t$.

All video SR methods can be divided into two classes: reconstruction-based and learning-based. Reconstructed-based methods (Baker and Kanade, 1999; Farsiu et al., 2004; He and Kondi, 2006; Kanaev and Miller, 2013; Liu and Sun, 2014; Omer and Tanaka, 2009; Rudin et al., 1992) craft a video SR process to solve the inverse estimation problem of (1). They usually perform motion compensation at first, then perform deblurring by estimating blur functions in $\mathbf{D}_t$ of (1), and finally recover details by local correspondences. The hand-crafted video SR process cannot be applicable for every practical scenario of different properties and perform not well to some unexpected cases.

In contrast, learning-based methods handle the ill-posed inverse estimation by learning useful priors for video SR from a large collection of videos. Typical methods include recently developed deep learning-based video SR methods (Huang et al., 2015; 2017; Liao et al., 2015a) and give some examples of non-deep learning approaches. In Liao et al. (2015a), a funnel shape convolutional neural network (CNN) was developed to predict HR frames from LR frames that are aligned by optical flow in advance. It shows superior performance on recovering HR video

* Corresponding author.
E-mail address: liujiaying@pku.edu.cn (J. Liu).
[1] This work was supported by National Natural Science Foundation of China under contract no. 61772043, with additional support by the State Scholarship Fund from the China Scholarship Council.

https://doi.org/10.1016/j.cviu.2017.09.002
Received 24 January 2017; Received in revised form 25 June 2017; Accepted 16 September 2017
Available online 09 November 2017
1077-3142/

frames captured in still scenes. However, this CNN model suffers from high computational cost (as it relies on time-consuming regularized optical flow methods) as well as visual artifacts caused by complex motions in the video frames. In Huang et al. (2015); 2017), a bidirectional recurrent convolutional network (BRCN) was employed to model the temporal correlation among multiple frames and further boost the performance for video SR over previous methods.

However, previous learning-based video SR methods that learn to predict HR frames directly based on LR frames, suffer from following limitations. First, these methods concentrate on exploiting between-frame correlations and does not *jointly* consider the intra- and inter-frame correlations that are both critical for the quality of video SR. This unfavorably limits the capacity of the network for recovering HR frames with complex contents. Second, the successive input LR frames are usually highly correlated with the whole signal of the HR frames, but are not correlated with the high frequency details of these HR images. In the case where dominant training frames present slow motion, the learned priors hardly capture hard cases, such as large movements and shot changes, where neighboring frames distinguished-contributed operations are needed. Third, it is desirable for the joint estimation of video SR to impose priors on missing high frequency signals. However, in previous methods, the potential constraints are directly enforced on the estimated HR frames.

To solve the above-mentioned issues, in this work, we propose a unified deep neural network architecture to *jointly* model the intra-frame and the inter-frame correlation in an end-to-end trainable manner. Compared with previous (deep) video SR methods (Huang et al., 2015; 2017; Liao et al., 2015a), our proposed deep network model does not require explicit computation of optical flow or motion compensation. In addition, our proposed model unifies the convolutional neural networks (CNNs) and recurrent neural networks (RNNs) which are known to be powerful in modeling sequential data. Combining the spatial convolutional and temporal recurrent architectures enables our model to capture spatial and temporal correlations jointly. Specially, it models spatial and temporal correlations among multiple video frames jointly. The temporal residues of HR frames are predicted based on input LR frames along with their temporal residues to further regularize estimation of the spatial residues.

This architectural choice enables the network to handle the videos containing complex motions in a moving scene, offering pleasant video SR results with few artifacts in a time-efficient way.

More concretely, we propose a **S**patial **T**emporal **R**ecurrent **Res**idual **Net**work (STR-ResNet) for video SR as show in Fig. 1. As aforementioned, SRT-ResNet models spatial and temporal correlations among multiple video frames jointly. In STR-ResNet, one basic component is

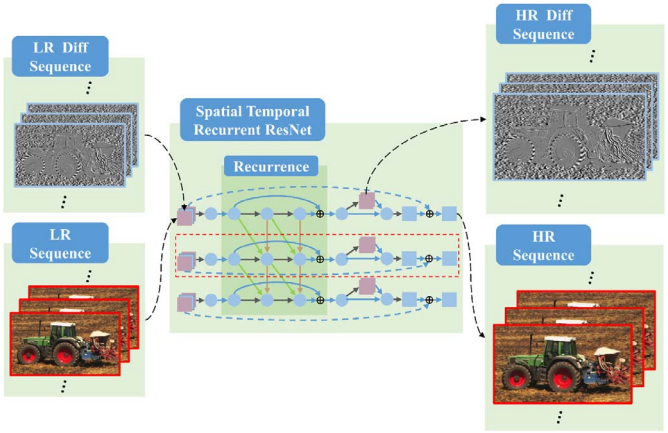


Fig. 1. The architecture of our proposed spatial-temporal recurrent residual network (STR-ResNet) for video SR. It takes not only the LR frames but also the differences of these adjacent LR frames as the input. Some reconstructed features are constrained to predict the differences of adjacent HR frames in the penultimate layer.

the spatial residual CNN (SRes-CNN) for single frame SR, which has a bypass connection for learning the residue between LR and HR feature maps. SRes-CNN is able to capture the correlation information among pixels within a single frame, and tries to recover an HR frame based on its corresponding LR frame through utilizing such correlations. Then, STR-ResNet stacks multiple SRes-CNNs together with recurrent connections between them. The global recurrent architecture captures the temporal contextual correlation and recovers the HR frame using both its corresponding LR frame and its adjacent frames. To better model inter-frame motions, STR-ResNet takes not only multiple LR frames but also the residue of these adjacent LR frames as inputs and tries to predict the temporal residues of HR frames in the penultimate layer. An HR frame is thus recovered by STR-ResNet by summing up its corresponding LR frame and the predicted spatial residue via the SRes-CNN component, under the guidance of the predicted temporal residue from adjacent frames via recurrent residual learning.

By separating the video frames into LR observations and the spatial residue within a single frame, the low frequency parts of HR frames and LR frames are untangled. Thus, the models can only focus on describing high-frequency details. By considering the temporal residues, in both their prediction path from LR temporal residues to HR temporal residues and their connection to spatial residues, the proposed STR-ResNet models both the spatial and temporal correlations jointly and achieves outstanding video SR performance with relatively low computational complexity.

In summary, we make the following contributions in this work to solving the challenging video SR problem:

- We propose a novel deep convolutional neural network architecture specifically for video SR. It follows a joint spatial-temporal residual learning and aims to predict the HR temporal residues which further facilitate the predictions of spatial residues and HR frames. By embedding the temporal residue prediction, the proposed architecture is capable of implicitly modeling the motion context among multiple video frames for video SR. It provides high-quality video SR results on benchmark datasets with relatively low computational complexity.
- To the best of our knowledge, the proposed STR-ResNet is the first research attempt to incorporate the bypass connection in a deep network to embed the joint spatial-temporal residue prediction and model temporal correlations in video frame sequences for video processing. The incorporated residual architecture implicitly models inter-frame motion context and is demonstrated to be beneficial for video SR.
- We are also among the first to investigate and unify the spatial convolutional, temporal recurrent and residual architectures into a single deep neural network to solve video SR problems. Extensive experiments on video SR benchmark datasets clearly demonstrate the contribution of each component to the overall performance.

The rest of this paper is organized as follows. Related work is briefly reviewed in Section 2. In Section 3, we introduce our spatial-temporal residual learning. Then, we construct a deep network to model it step-by-step and present the details of the proposed STR-ResNet, which models both spatial and temporal redundancies jointly in a unified network, as well as its constituent SRes-CNN in Section 4. Experimental results are presented in Section 5. More analysis and discussion on our method are provided in Section 6. Concluding remarks are given in Section 7.

2. Related work

Single image super-resolution was first investigated by Irani and Peleg (1991). By now, it can be divided into two categories: reconstruction-based and learning-based. Reconstruction-based methods adopt regularizations, such as gradient histogram (Sun et al., 2011),

nonlocal filter (Huhle et al., 2010; Mairal et al., 2009; Zhang et al., 2013) and total variation (Marquina and Osher, 2008), to guide the SR. Learning-based methods learn the mapping function from the training data to model the spatial correlation of single images. These methods include neighbor embedding (Chang et al., 2004), sparse representation (Yang et al., 2010), anchor regression (Timofte et al., 2013), random forest (Salvador and Prez-Pellitero, 2015), tensor regression (Yin et al., 2015), ramp transformation (Singh and Ahuja, 2015) and deep learning (Cui et al., 2014; Dong et al., 2014; Zeng et al., 2016). Some recent works focus on super-resolution on a specific kind of images, such as depth image (Ismaeil et al., 2016; Joshi and Chaudhuri, 2006), multi-spectral image (Aguena and Mascarenhas, 2006) and multi-resolution (Lu and Li, 2014). There are also some recent works on the SR performance evaluation (Ma et al., 2017) or bridging the image SR to high-level computer vision tasks (Nguyen et al., 2013; Timofte et al., 2016).

Compared with the images where SR mainly relies on utilizing the intrinsic spatial correlation (Freeman et al., 2002; Irani and Peleg, 1991; Sun et al., 2011; Yuan et al., 2013), the videos additionally present the temporal correlation among adjacent frames that is valuable for their SR in particular. Thus the attempts to effectively exploit such temporal correlation motivate several recent video SR approaches (Baker and Kanade, 1999; Farsiu et al., 2004; Huang et al., 2015; 2017; Protter et al., 2009; Zhao and Sawhney, 2002). Although it is conceptually straightforward, exploiting the temporal correlation immediately proposes several important challenges to modern video SR techniques: *e.g.*, how to estimate and model motion across frames properly for video SR and how to establish the correspondence between pixels from adjacent frames based on the motion estimation.

Most of the existing video SR methods exploit motion information in the following two ways: *explicitly* aligning multiple frames according to estimated motion and *implicitly* embedding motion estimation to regularize the process of recovering HR frames. Accordingly these video SR methods can be divided into two categories: the *explicit motion-based methods* that align LR frames according to either optical flow (Fransens et al., 2007; Liu and Sun, 2014) or motion compensation (Baker and Kanade, 1999) and the *implicit motion-based methods* that embed motion as a weighting term (Farsiu et al., 2004; He and Kondi, 2006; Kanaev and Miller, 2013; Omer and Tanaka, 2009) or a regularization term (Liu and Sun, 2014; Rudin et al., 1992; Yuan et al., 2013; Zhang et al., 2015) for tuning the HR estimation.

Explicit motion-based methods generally suffer from heavy computational cost for motion compensation, and artifacts caused by inaccurate registration of local irregular motions. To overcome these deficiencies and get rid of explicit motion estimation, implicit motion-based methods embed motion context into the HR estimation. For example, the nonlocal similarity and kernel regression among multiple frames can be employed to model the temporal and spatial correlations implicitly (Protter et al., 2009; Takeda et al., 2009a). Benefited from implicit motion estimation, these methods avoid visual artifacts due to inaccurate motion estimation and are able to handle local motions effectively. However, they may fail in dealing with large motions.

Recently, several deep learning methods (Huang et al., 2015; Liao et al., 2015a) have been proposed to address the video SR problem in both explicit and implicit ways. Compared with conventional methods, in these works, CNNs and RNNs are used to model some parts of the video SR pipeline, i.e. feature extraction, motion compensation, and multi-frame fusions, achieving superior video SR performance.

Besides the video SR, many deep learning-based low level processing applications raised, with promising performance. These applications include denoising (Agostinelli et al., 2013; Burger et al., 2012a,b; Jain and Seung, 2009; Vincent et al., 2010), completion (Xie et al., 2012), super-resolution (201, 2016; Dong et al., 2014; Osendorfer et al., 2014), deblurring (Schuler et al., 2016), deconvolution (Xu et al., 2014) and style transfer (Gatys et al., 2016; Yan et al., 2016). They focus on exploiting a deep network to learn a mapping between the source / degraded signal and the target / high-quality signal for a single image, by capturing the spatial correlation.

3. Spatial-temporal residual learning

In this section, we illustrate our spatial-temporal residual learning. Let $\mathbf{x}$ and $\mathbf{y}$ be HR and up-sampled LR sequences which have the same size to the HR sequences, respectively. Then, the inverse mapping function for video SR can be represented by

$$\mathbf{x} = f^n(\mathbf{y}), \tag{2}$$

where $f^n(\cdot)$ is the process to predict the HR image $\mathbf{X}$ directly based on the LR image $\mathbf{Y}$.

An end-to-end learning following Eq. (2) will be trapped into the difficulties mentioned in Section 1: (i) separation of intra- and inter-correlations modeling, (ii) contamination from frames due to treating neighboring frames equally and lack of constraints for adjacent predicted frames.

To address the issues (i) and (ii), we change to solve video SR by learning to predict the spatial residue $\mathbf{r}_t = \mathbf{x}_t - \mathbf{y}_t$ instead of the whole $\mathbf{x}_t$ as

$$\mathbf{X} = f^s(\mathbf{Y}) + \mathbf{Y}, \tag{3}$$

where $f^s(\cdot)$ is the process to predict the difference between the HR image $\mathbf{x}$ and the LR image $\mathbf{Y}$ based on $\mathbf{Y}$.

This change makes a learning-based method capable to fit the high-frequency mapping between LR frames $\{\mathbf{y}_t\}$ and the spatial residue $\{\mathbf{r}_t\}$ instead of the mapping between LR frames $\{\mathbf{y}_t\}$ and HR frames $\{\mathbf{x}_t\}$, where the low frequency mapping plays a dominant role. In the degradation, the down-sampling operation usually brings in aliasing effect that the local high-frequency patterns change after the down-sampling operation. Following the simplification in Liu and Sun (2014), we regard the aliasing signal as structural noise. Thus, we do not model it explicitly and expect the recovery process, i.e. $f^s(\cdot)$, can automatically model its removal.

Besides, from the perspective of patch statistic, Eq. (3) provides more structural correspondences than Eq. (2) as shown in Fig. 2. We calculate "patch repetitiveness" across frames to observe the structural correspondences via the average MSE between a local 5×5 patch and it most similar patches. We first search the top-10 similar patches for each patch among three successive frames and calculate the MSE between the patch and its similar patches. Then, the average MSE of each patch is converted into a probability based on Gaussian function. We calculate this statistic in two domains – the original signal domain and spatial residual domain, and visualize the results. The subfigure (d) in Fig. 2 is the heat map for the patch repetitiveness of (b) across frames in

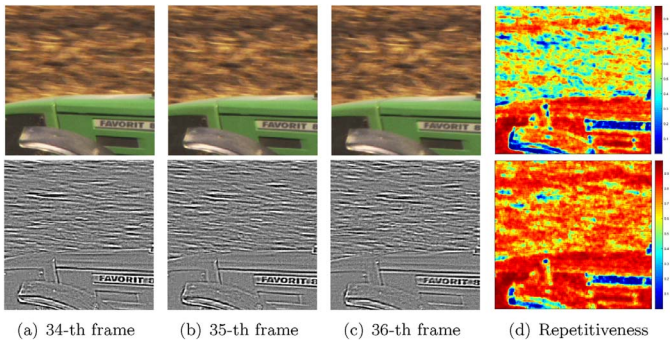


Fig. 2. Top panel: (a)–(c) local regions in the *Tractor* sequence and (d) the patch repetitiveness of the 35th frame. Bottom panel: (a)–(c) local regions in the temporal residues of the *Tractor* sequence and (d) the patch repetitiveness of the spatial residue in the 35th frame. Red signifies high values, Blue signifies low values. It is clearly demonstrated that the spatial residue domain across frames provides more patch repetitiveness than the original image space.

the top panel – an normal 2D sub-image in Eq. (2), that for the patch repetitiveness of (b) across frames in the bottom panel – the difference image – in the spatial residue space in Eq. (3), respectively. In these heat maps, red signifies high values and blue signifies low values. From the result, the third issue (iii) is clearly demonstrated that the spatial residue domain across frames provides more patch repetitiveness than the original image space across frames. This property is significant for us to design a learning-based image SR approach, especially when many previous works (Dong et al., 2013; Wu and Zheng, 2013) have proved that the structural correspondence in the target domain provides useful information to infer and locate the manifold where the HR signal locate.

Then, to address the issue (iii) and impose effective constraints on the predicted HR frames, we build the connection between spatial and temporal residues. We define HR frames t and $t+1$ as

$$\mathbf{x}_t = \mathbf{r}_t + \mathbf{y}_t, \tag{4}$$

$$\mathbf{x}_{t+1} = \mathbf{r}_{t+1} + \mathbf{y}_{t+1}, \tag{5}$$

Let Eq. (5) subtract Eq. (4), we have

$$\mathbf{x}_{t+1} - \mathbf{x}_t = (\mathbf{r}_{t+1} - \mathbf{r}_t) + \mathbf{y}_{t+1} - \mathbf{y}_t. \tag{6}$$

Define the temporal residue for $\mathbf{x}_t$ and $\mathbf{y}_t$. $\delta_t^{\mathbf{x}} = \mathbf{x}_{t+1} - \mathbf{x}_t$ and $\delta_t^{\mathbf{y}} = \mathbf{y}_{t+1} - \mathbf{y}_t$, then

$$\delta_t^{\mathbf{x}} - \delta_t^{\mathbf{y}} = \mathbf{r}_{t+1} - \mathbf{r}_t. \tag{7}$$

Although the derivation of Eq. (7) is straightforward, it bridges the predictions of the spatial and temporal residues. This connection is beneficial to video SR from two aspects. First, it provides a more effective learning strategy – the prediction for temporal residues $\delta_t^{\mathbf{x}}$ is first learned and then precise predictions for $\delta_t^{\mathbf{x}}$ will naturally lead to more precise spatial residue estimations and finally more precise $\hat{\mathbf{x}}_t$. Second, with Eq. (7), the predictions for spatial residues $\mathbf{r}_{t+1}$ and $\mathbf{r}_t$ are regularized explicitly by that their differences are equal to the differences of the temporal residue $\delta_t^{\mathbf{x}} - \delta_t^{\mathbf{y}}$. This provides effective side information to regularize a learning-based model, leading to both fast convergence rate and higher accuracy.

4. Spatial-temporal recurrent residual networks for multi-frame SR

In this section, a basic network structure – SRes-CNN for spatial residual learning for single image SR is presented in formulation. Then, motivated by our spatial-temporal residual learning, we construct our proposed STR-ResNet step by step. Finally, a new proposed STR-ResNet by stacking and connecting the basic component – SRes-CNN for joint temporal learning is elaborated.

4.1. Architecture of SRes-CNN

Single frame SR aims to reconstruct an HR frame from a single LR frame. Some recent deep learning based SR methods (Dong et al., 2014; Wang et al., 2015a; Yang et al., 2017) propose to use a CNN model to extract features from LR frames and then map them to HR ones. A typical CNN architecture for single frame SR consists of three convolutional layers as proposed in Dong et al. (2014) which jointly performs sparse coding and reconstruction over the LR frames as shown in Fig. 4 (a). However, striving for directly recovering the complete HR frames may cause the CNN models to miss some important high frequency details. In contrast, separately modeling LR signals and their residues with high-frequency details, as shown in Fig. 4 (b)–(e), could recover high frequency details better. Besides, we hope to construct an easy training network and expect its training to converge fast and to a good state even without advanced training skills.

Keeping such an idea in mind, we propose a new CNN architecture – Spatial Residual CNN (SRes-CNN) – to learn spatial residue between HR and LR frames as shown in Fig. 4 (c). Specifically, SRes-CNN contains

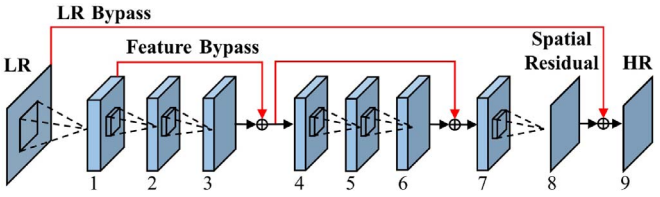


Fig. 3. The bypass structure and spatial residual learning in the proposed SRes-CNN. The *feature bypass* connection forwards the feature maps output from a previous layer (1st / 4th) to a later one (4th / 7th). The *LR bypass* from the LR frame to the last layer (9th) makes the network focus on predicting the residue, the high frequency component of a frame.

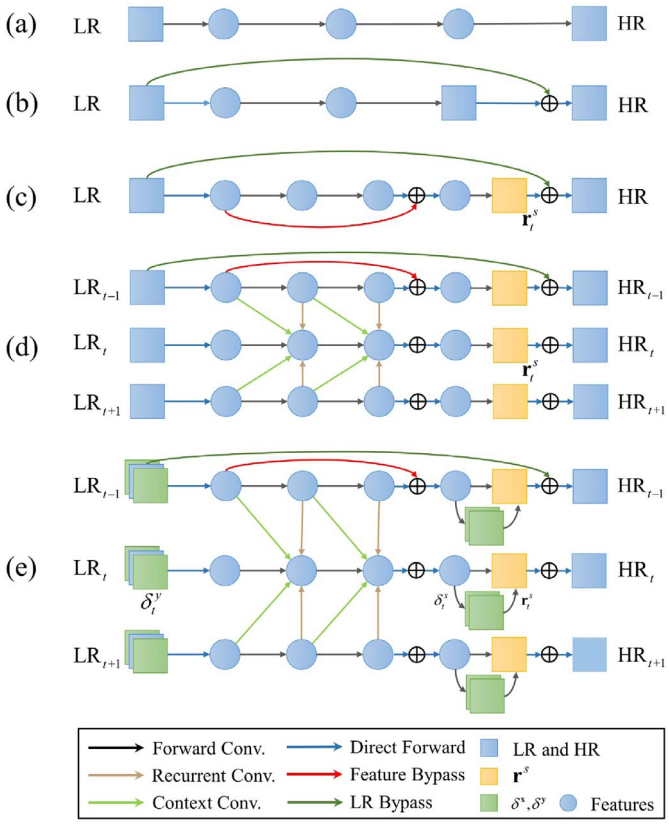


Fig. 4. Network architectures from vanilla SRCNN to our proposed spatial-temporal residual network. (a) SRCNN. (b) SRCNN with LR bypass connections. (c) SRes-CNN has both LR and feature bypass connections. (d) Multiple SRes-CNNs connected by context and recurrent convolutions to model inter-frame motion context. (e) In STR-ResNet, the differences of LR images $\delta_t^{\mathbf{y}}$ are inputed into the network and parts of features in the penultimate layer aim to predict the differences of HR images $\delta_t^{\mathbf{x}}$, which further regularize and benefit the joint estimation of $\{\hat{\mathbf{x}}_t\}$. (Best viewed in color.).

nice layers, including six convolutional layers, three bypass connections and three element-wise summations, as shown in Fig. 3. The bypass connections forward the feature maps output from the ith layer ($i = 1, 4$ for the SRes-CNN we use in the experiments) to the $(i+2)$th layer directly. Then, the feature maps output from the $(i+2)$th and ith layers are fused as input to the next $(i+3)$th convolution layer. To focus on predicting the high-frequency components, SRes-CNN also establishes a bypass connection from the input LR frame to the penultimate layer. Note that, these two kinds of bypass connections play different roles in STR-ResNet. The first "long-range"one directly forwards an input LR frame to its penultimate layer (the 7th one). The other bypass connections provide a coarse-to-fine refinement. For example, the feature maps of the 1st layer correspond to the low-level features directly extracted from the LR image, and then the feature maps of the 3rd and 5th layers therefore concentrate on capturing the enhanced details of HR

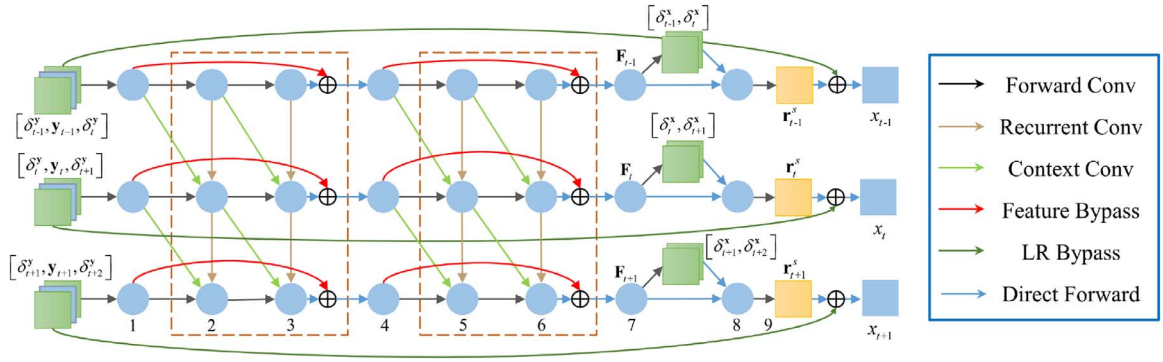


Fig. 5. The architecture of the STR-ResNet. It has a two-layer structure, which includes spatial and temporal residuals jointly in a unified deep framework. To model the inter-frame correlation, we construct a temporal residual RNN by piling up and connecting spatial residual CNNs. It takes not only the LR frames but also the differences of these adjacent LR frames as the input. Some reconstructed features are constrained to reconstruct the differences of adjacent HR frames in the penultimate layer. (Best viewed in color) (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.

features. Besides, the bypass connections also make constructing a deeper network possible and speed up the training process (Dong et al., 2014).

We here provide more formal description corresponding to Fig. 3 on the operations of each layer in SRes-CNN. The output of each layer, denoted as $\mathbf{C}_i$ for $i = 1, ...,9$, is calculated as

$$\begin{aligned}
\mathbf{C}_1 &= \max(0, \mathbf{W}_1 * \mathbf{y}_t + \mathbf{B}_1),\\
\mathbf{C}_2 &= \max(0, \mathbf{W}_2 * \mathbf{C}_1 + \mathbf{B}_2),\\
\mathbf{C}_3 &= \mathbf{W}_3 * \mathbf{C}_2 + \mathbf{B}_3\\
\mathbf{C}_4 &= \max(0, \mathbf{C}_3 + \mathbf{C}_1),\\
\mathbf{C}_5 &= \max(0, \mathbf{W}_5 * \mathbf{C}_4 + \mathbf{B}_5),\\
\mathbf{C}_6 &= \mathbf{W}_6 * \mathbf{C}_5 + \mathbf{B}_6,\\
\mathbf{C}_7 &= \max(0, \mathbf{C}_6 + \mathbf{C}_4),\\
\mathbf{C}_8 &= \mathbf{W}_h * \mathbf{C}_7 + \mathbf{B}_h,\\
\mathbf{C}_9 &= \mathbf{y}_t + \hat{\mathbf{x}}_{h,t},\\
\hat{\mathbf{x}}_{h,t} &= \mathbf{C}_8,\\
\hat{\mathbf{x}}_t &= \mathbf{C}_9,
\end{aligned} \tag{8}$$

where $\mathbf{W}_i$ and $\mathbf{B}_i$ are the filters and biases associated with the ith layer respectively. Here we use the subscript h to indicate the parameters and outputs that are related to high frequency predictions. Regarding the network size, $\mathbf{W}_i$ consists of n_i filters with a size of $n_{(i-1)} \times f_i \times f_i$, and n_{i-1} is the number of input feature maps of the ith layer which also counts the output feature maps of the $(i-1)$st layer. We use f_i to denote the kernel size of convolution filters of the ith layer. The bias $\mathbf{B}_i$ is an n_i dimensional vector. The outputs of each convolution layer (besides the last convolution layer $\mathbf{C}_8$) also go through a Rectified Linear Unit (ReLU). Particularly, n_0 is the channel number of an input frame, where $n_0 = 1$ for the gray frame and $n_0 = 3$ for the color frame, respectively. The last convolutional layer is not connected with a ReLU unit. It is noted that, $\mathbf{C}_4$, $\mathbf{C}_7$ and $\hat{\mathbf{x}}_t$ have bypass connections from $\mathbf{C}_1$, $\mathbf{C}_4$ and $\mathbf{y}_t$ respectively. Taking the single LR and HR frames as the input and output of the network respectively, SRes-CNN is capable of predicting the HR frame according to a single LR frame by utilizing the spatial correlation. However, it does not capture the temporal correlation among adjacent frames in videos.

4.2. Modeling spatial-temporal residues: step by step

We now proceed to illustrate how to construct SRes-CNN and STR-ResNet step by step in details. The vanilla SRCNN (Dong et al., 2014) models the learning paradigm of Eq. (2) shown in Fig. 4 (a). The network learns to predict $\mathbf{x}_t$ based on $\mathbf{y}_t$ directly. To model the learning paradigm of Eq. (3), a bypass connection that forwards $\mathbf{y}_t$ to the final prediction $\hat{\mathbf{x}}_t$ is added as shown in Fig. 4 (b). Training this network needs additional training constraints and a deliberate crafted hyper-parameter tuning, i.e. adjustable gradient clipping (Kim et al., 2016b), thus, an improved version – the recurrent residue learning (Yang et al., 2017) with not only the LR forward path but also the feature forward path is proposed as shown in Fig. 4 (c). With bypass connections, the network converges faster and to a better solution.

To utilize the temporal redundancy among different frames, the relationship between different frames is modeled. We follow a similar method as Huang et al. (2015) by adding recurrent and context convolutions in each recurrence of the recurrent residual learning, as shown in Fig. 4 (d). To utilize temporal residues to facilitate the SR network training and HR image predictions, motivated by Eq. (7), we further propose a network structure as shown in Fig. 4 (e), where the differences of LR images δ_t^y are inputed into the network and parts of features in the penultimate layer aim to predict the differences of HR images δ_t^x, which further regularize and benefit the joint estimation of $\{\hat{\mathbf{x}}_t\}$. We then focus on illustrating Fig. 4 (c) and (e) in formulation.

4.3. Architecture of STR-ResNet

We now elaborate how the STR-ResNet exploits inter-frame correlation by connecting multiple SRes-CNNs with convolutions and how it incorporates temporal residual information for multi-frame SR. The intuition of choosing the architecture is to propagate information across multiple frames recurrently in order to capture the temporal context. STR-ResNet uses recurrent units to connect several SRes-CNNs to embed the temporal correlation. The STR-ResNet takes not only the LR frames but also the differences of adjacent LR frames as inputs. It reconstructs an HR frame through fusing its corresponding LR frame and the predicted spatial residue, under the guidance of the predicted temporal residues among adjacent frames. As shown in Fig. 5, STR-ResNet performs following six types of operations:

1. **Forward convolution**. The convolution operations in each SRes-CNN component for single frame SR.
2. **Recurrent convolution**. To propagate information across adjacent frames and restore lost information from the adjacent frames, STR-ResNet performs recurrent convolutions (the gray arrows between frames as shown in Fig. 5) to propagate the features of the ith layer of the adjacent $(t-1)$st and $(t+1)$st frames (defined as $\mathbf{C}^a_{(t-1,i)}$ and $\mathbf{C}^a_{(t+1,i)}$) to the ith layer of the tth frame (defined as $\mathbf{C}^{r,p}_{(t,i)}$ and $\mathbf{C}^{r,n}_{(t,i)}$.)
3. **Context convolution**. With the similar intuition of transmitting complementary information among frames, the context convolution (the light-green arrows between frames as shown in Fig. 5) propagates the features of the $(i-1)$th layer of the adjacent $(t-1)$st and $(t+1)$st frames (defined as $\mathbf{C}^a_{(t-1,i-1)}$ and $\mathbf{C}^a_{(t+1,i-1)}$) to the ith layer of the tth frame (defined as $\mathbf{C}^{c,p}_{(t,i)}$ and $\mathbf{C}^{c,n}_{(t,i)}$.)
4. **Temporal residue embedding**. In the 8th layer, we first predict the

temporal residues (the green rectangles between the 7th and 8th layers as shown in Fig. 5). In the training, these outputs are constrained by the loss function to regress the ground-truth temporal residues, which will be presented more clearly in the next subsection. Then, we concatenate the predicted temporal residues with the output feature maps from the 7th layer to generate the output feature maps of the 8th layer.

5. **Feature bypass**. The operation to transmit the features output from the 1st/4th layers and combine them with the output of the 3rd/6th layers respectively.
6. **LR bypass**. It bypasses the LR frames to the output of the 8th layer, which generates the estimated HR details of frame *t*.
7. **Feed forward**. The operation to propagate the feature maps to the subsequent unit.

Among these operations, the recurrent and context convolutions are only deployed in the 2nd, 3rd, 5th and 6th layers of SRes-CNNs as shown in Fig. 5 (b). All the recurrent connections transmit outputs of layers (2nd, 3rd, 5th and 6th) on the tth frame to their corresponding layers (2nd, 3rd, 5th and 6th) of the adjacent $(t-1)$th and $(t+1)$th frames. All the context connections transmit from a previous layer (1st, 2nd, 4th and 5th) of the tth frame to their corresponding next layer (2nd, 3rd, 5th and 6th) of the adjacent $(t-1)$th and $(t+1)$th frames. After all these convolutions, an element-wise summation operation is employed to fuse these convolution outputs and produce a new feature map. The outputs of the five convolutional operations and the fusion operation are formulated as follows,

$$\begin{aligned}
\mathbf{C}^{f}_{(t,i)} &= \mathbf{W}^{f}_{i} * \mathbf{C}^{a}_{(t,i-1)} + \mathbf{B}^{f}_{i},\\
\mathbf{C}^{c,p}_{(t,i)} &= \mathbf{W}^{c,p}_{i} * \mathbf{C}^{a}_{(t-1,i-1)} + \mathbf{B}^{c,p}_{i},\\
\mathbf{C}^{c,n}_{(t,i)} &= \mathbf{W}^{c,n}_{i} * \mathbf{C}^{a}_{(t+1,i-1)} + \mathbf{B}^{c,n}_{i},\\
\mathbf{C}^{r,p}_{(t,i)} &= \mathbf{W}^{r,p}_{i} * \mathbf{C}^{a}_{(t-1,i)} + \mathbf{B}^{r,p}_{i},\\
\mathbf{C}^{r,n}_{(t,i)} &= \mathbf{W}^{r,n}_{i} * \mathbf{C}^{a}_{(t+1,i)} + \mathbf{B}^{r,n}_{i},\\
\mathbf{C}^{a}_{(t,i)} &= \max(0,\ \mathbf{C}^{f}_{(t,i)} + \mathbf{C}^{c,p}_{(t,i)} + \mathbf{C}^{c,n}_{(t,i)} + \mathbf{C}^{r,p}_{(t,i)} + \mathbf{C}^{r,n}_{(t,i)}),
\end{aligned} \tag{9}$$

where $i = 2, 3, \ldots, 6$, and $\mathbf{W}$ and $\mathbf{B}$ are filters and biases, respectively. The superscripts *f, c, r* and *a* denote the unit type – forward convolution, context convolution, recurrent convolution and element-wise summation aggregation. The superscripts *p, n* denote the direction of the convolution, from the previous frame or the next frame. The subscript (t, i) denotes that the operation is performed on the ith layer of the tth frame. Consequently, $\mathbf{C}^{f}_{(t,i)}$,$\mathbf{C}^{c,p}_{(t,i)}$,$\mathbf{C}^{c,n}_{(t,i)}$,$\mathbf{C}^{r,p}_{(t,i)}$ and $\mathbf{C}^{r,n}_{(t,i)}$ are the outputs of the forward convolution, context convolution from the previous frame, context convolution from the next frame, recurrent convolution from the previous frame and recurrent convolution from the next frame in the ith layer of the tth frame respectively. $\mathbf{C}^{a}_{(t,i)}$ performs an element-wise summation overall all the five outputs, for combining the predictions from the current frame and adjacent frames. A ReLU unit is then connected subsequently. The responses of previous layers are as follows,

$$\mathbf{C}^{a}_{(t,i)} = \mathbf{C}^{f}_{(t,i)}, \quad \text{for } i = 1, 4, 7, 9. \tag{10}$$

For the 8th layer, we try to predict the temporal residues of HR frames and utilize them as parts of the features to estimate the spatial residues,

$$\delta^{\mathbf{x}}_{t} = \mathbf{W}_{\delta} * \mathbf{C}^{a}_{(t,7)} + \mathbf{b}_{\delta},\ \mathbf{C}^{a}_{(t,8)} = [\mathbf{C}^{a}_{(t,7)},\ \delta^{\mathbf{x}}_{t}]. \tag{11}$$

With the help of context and recurrent convolutions as well as the temporal residue constraints, the STR-ResNet captures the inter-frame motion context propagated from adjacent frames for video SR.

4.4. Training STR-ResNet

To learn meaningful features and capture some consistent motion contexts between frames, STR-ResNet shares its parameters among different frames. That is, for all $\mathbf{C}^{f}_{(t,i)}$, $\mathbf{C}^{c,p}_{(t,i)}$, $\mathbf{C}^{c,n}_{(t,i)}$, $\mathbf{C}^{r,p}_{(t,i)}$, and $\mathbf{C}^{r,n}_{(t,i)}$, their parameters $\{\mathbf{W}^{f}_{i}, \mathbf{B}^{f}_{i}\}$,$\{\mathbf{W}^{c,p}_{i}, \mathbf{B}^{c,p}_{i}\}$,$\{\mathbf{W}^{c,n}_{i}, \mathbf{B}^{c,n}_{i}\}$,$\{\mathbf{W}^{r,p}_{i}, \mathbf{B}^{r,p}_{i}\}$ and $\{\mathbf{W}^{r,n}_{i}, \mathbf{B}^{r,n}_{i}\}$ are decided by the unit type, denoted by superscript, and layer depth, and have nothing to do with the frame number.

For training STR-ResNet, provided with LR video frames $\{\mathbf{y}^{g}_{t}\}$ and HR frames $\{\mathbf{x}^{g}_{t}\}$, we minimize the Mean Square Error (MSE) between the predicted frames and the ground truth HR frames:

$$\begin{aligned}
\min_{\mathbf{\Theta}} \sum_{t=1}^{9} \lambda_t \left\| \hat{\mathbf{x}}_t(\mathbf{y}^{g}_{t}, \mathbf{\Theta}) + \mathbf{y}^{g}_{t} - \mathbf{x}^{g}_{t} \right\|_F^2 \\
+ c \sum_{t=1}^{9} \left\| \hat{\delta}^{\mathbf{x}}_{t}(\mathbf{y}^{g}_{t}, \mathbf{\Theta}) + \mathbf{x}^{g}_{t} - \mathbf{x}^{g}_{t-1} \right\|_F^2,
\end{aligned} \tag{12}$$

where

$$\mathbf{\Theta} = (\mathbf{W}^{f}, \mathbf{B}^{f}, \mathbf{W}^{c,p}, \mathbf{B}^{c,p}, \mathbf{W}^{c,n}, \mathbf{B}^{c,n}, \mathbf{W}^{r,p}, \mathbf{B}^{r,p}, \mathbf{W}^{r,n}, \mathbf{B}^{r,n}) \tag{13}$$

$\mathbf{x}^{g}_{0} = \mathbf{x}^{g}_{1}$ and $\{\lambda_i, i = 1, 2, \ldots, 8, 9\}$ are the weighting parameters that control the relative importance of these terms. c is set to 0.1 to play a role but not the dominant one. We set $n_T = 9$ as the step/recurrence number because it is the maximum value of temporal recurrences that can be affordable for the GPU memory when using a mini-batch of 6 samples for training. Besides, it is also the default setting in the previous RNN-based video SR method (Huang et al., 2015). For the setting of λ_t, we propose to use a coarse-to-fine strategy: (1) *"scattered" pre-training*, i.e., setting $\lambda_t = 1$ for all t forces the network to capture general motion trends and to learn the features that are good at reconstructing a whole video clip instead of a single frame; (2) *"focused" fine-tune*, namely associating the values of λ_t with an exponential decayed weight from the center frame to other frames to focus on the prediction of the center frame. The value configuration of λ_t is illustrated as Table 1.

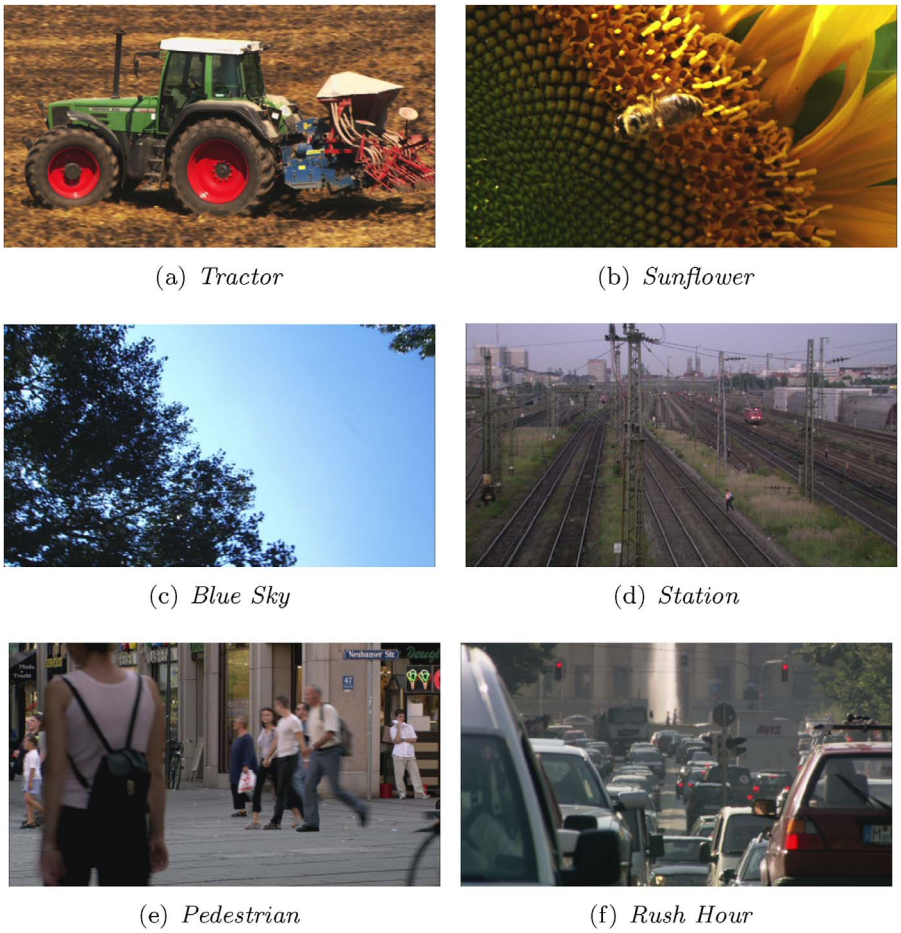
4.5. Image degradation and aliasing

In our work, we follow previous works (Huang et al., 2015; Liao et al., 2015b) and only evaluate on 4 × enlargement, because it is considered as the most difficult case among the commonly used experimental settings usually with 2, 3 and 4 as scaling factors. For the blur kernel, a comprehensive study has been conducted in Bayesian video SR (Liu and Sun, 2014). We follow its conclusion that, a point spread function kernel for upscaling factor of 4 can be approximated by a Gaussian with standard deviation from 1.2 to 2.4, and use a Gaussian kernel with standard deviation 1.6 in our degradation setting.

Besides, as discussed in Liu and Sun (2014), this blurring operation brings in the aliasing effect. High frequency components in the original signal present different local patterns after down-sampling. This effect may lead to inaccurate motion estimations and raise the problem of inter-frame inconsistency. However, in our STR-ResNet, we do not explicitly model it because of two reasons: 1) STR-ResNet does not rely on an explicit motion estimation; 2) in such a degradation, the aliasing can be modeled as random noise (Liu and Sun, 2014), because the magnitude of the aliasing signal is relatively small compared to the whole signal. Thus, the video SR with aliasing can be regarded as a problem of joint denoising and SR, and is expected to be addressed through an end-to-end learning.

Table 1
The adopted values of λ_t in "focused" fine-tune.

#Frame	1	2	3	4	5	6	7	8	9
3	—	—	—	0.7	1	0.7	—	—	—
5	—	—	0.5	0.7	1	0.7	0.5	—	—
7	—	0.35	0.5	0.7	1	0.7	0.5	0.35	—
9	0.25	0.35	0.5	0.7	1	0.7	0.5	0.35	0.25



(a) *Tractor* (b) *Sunflower*

(c) *Blue Sky* (d) *Station*

(e) *Pedestrian* (f) *Rush Hour*

Fig. 6. The test HDTV sequences with their names as the captions.

5. Experiments

In this section, we evaluate the performance of the proposed SR method and compare it with state-of-the-art single image SR and video SR methods.

5.1. Comparison methods

The compared single image SR baselines include Bicubic interpolation, A+ (Timofte et al., 2014) and super-resolution convolution neural network (SRCNN) (Dong et al., 2014). The compared video SR baselines include a commercial software video enhancer (VE)[2], 3DSKR (Takeda et al., 2009b), Draft SR (Liao et al., 2015a) and BRCN (Huang et al., 2015). We implement BRCN using Caffe (Jia et al., 2014). Other baseline methods are tested by the released executable or source codes provided by the authors. For learning based methods, including (Timofte et al., 2014) and (Dong et al., 2014), we retrain or fine-tune the models on the training set in the given experimental setting. For the pixel shifting case, such as VE, we first adjust the input LR image by Bicubic interpolation before SR.

5.2. Parameter setting

To evaluate the effectiveness of our method, we simulate the degradation process and enlarge the generated LR images to their original scales. Peak Signal to Noise Ratio (PSNR) is chosen as the metric. The testing scaling factor is chosen as 4. In the simulation of degradation, the LR frames are generated by blurring HR frames with a 9×9 Gaussian filters with blur level 1.6.

[2] http://www.infognition.com/videoenhancer/.

5.3. Datasets

For training our STR-ResNet, we use 300 collected video sequences, sampled uniformly from 30 high-quality 1080 p HD video clips as our training set[3,4]. We use 6 HDTV sequences downloaded from the Xiph.org Video Test Media[2] as the testing set, which are commonly used high quality video sequences for video coding testing. The name and content of the six sequences are shown in Fig. 6. To reduce the memory storage needed in the training phrase, we crop these frame groups into 75,000 overlapped patch groups as the input of training. Each patch group contains 9 adjacent patches in the temporal domain with the same location in the spatial domain. Similar to Dong et al. (2014), the size of the spatial window of each patch group is set to 33×33 and the spatial stride is set to 11.

5.4. Network training

The proposed STR-ResNet uses the following parameters: all convolutions have a kernel size of 3×3 and a padding size 1; the layer type and number are set as mentioned above; the channel size of the intermediate layers is set to 64. We employ stochastic gradient descent[5] to train the whole network. The training strategy is standard: learning rates of weights and biases of these filters are set to 0.0001 initially and decrease to 0.00001 after 2.5×10^5 iterations (about 37 epochs). We stop the training in 3×10^5 iterations (about 44 epochs). In the first step, we set $\lambda_t = 1$ and in the second step, we set λ_t as mentioned in Table 1. The batch size is set to 6.

[3] https://media.xiph.org/video/derf/. (**Xiph.org Video Test Media [derf's collection]**).

[4] http://www.harmonicinc.com/resources/videos/4k-video-clip-center. (**Dataset from Harmonic Inc.**).

[5] http://caffe.berkeleyvision.org/tutorial/solver.html.

Table 2
PSNR results among different methods for Video SR (scaling factor: 4). The bold numbers denote the best performance.

Video	Bicubic	NE + LLE	A +	SRCNN	VE
Tractor	31.10	32.04	32.07	32.13	31.27
Sunflower	37.85	38.75	38.87	38.69	37.55
Blue ky	28.77	30.02	30.02	30.16	29.19
Station	33.35	34.20	34.26	34.38	33.36
Pedestrian	33.51	34.28	34.43	34.55	33.60
Rush Hour	38.17	39.17	39.15	38.90	37.96
Average	33.79	34.74	34.80	34.80	33.82
Video	3DSKR	Draft5	Draft31	BRCN	STR-ResNet
Tractor	32.27	31.73	30.34	33.23	**33.85**
Sunflower	37.57	35.62	36.43	39.28	**40.02**
Blue Sky	29.74	30.34	30.92	31.40	**32.23**
Station	34.80	32.99	33.22	35.20	**35.63**
Pedestrian	33.91	33.40	31.78	34.95	**35.22**
Rush Hour	37.49	36.93	36.22	39.86	**40.30**
Average	34.30	33.50	33.15	35.65	**36.21**

Table 3
SSIM results among different methods for Video SR (scaling factor: 4). The bold numbers denote the best performance.

Video	Bicubic	NE + LLE	A +	SRCNN	VE
Tractor	0.8315	0.8465	0.8496	0.8514	0.8307
Sunflower	0.9626	0.9650	0.9662	0.9658	0.9600
Blue Sky	0.8957	0.9092	0.9128	0.9150	0.9019
Station	0.8738	0.8716	0.8756	0.8781	0.8636
Pedestrian	0.8836	0.8918	0.8955	0.8980	0.8810
Rush Hour	0.9471	0.9499	0.9505	0.9496	0.9428
Average	0.8990	0.9057	0.9083	0.9097	0.8967
Video	3DSKR	Draft5	Draft31	BRCN	STR-ResNet
Tractor	0.8742	0.8221	0.8074	0.8745	**0.8867**
Sunflower	0.9653	0.9432	0.9566	0.9631	**0.9673**
Blue Sky	0.9197	0.9229	0.9332	0.9277	**0.9349**
Station	0.8953	0.8911	0.9045	0.8877	**0.8952**
Pedestrian	0.8971	0.8572	0.8309	0.8990	**0.9024**
Rush Hour	0.9471	0.9318	0.9266	0.9487	**0.9521**
Average	0.9165	0.8947	0.8932	0.9168	**0.9231**

5.5. Objective evaluation

We evaluate these methods with PSNR. Tables 2 and 3 show PSNR results of compared video super-resolution methods on the testing image set. The proposed method and the BRCN method are evaluated with 9 adjacent frames as inputs. For Draft Learn, we report its results in two cases: (1) taking 31 adjacent LR frames (Draft31) as its input; (2) taking 5 adjacent LR frames (Draft5) as its input. For 3DSKR, the HR estimation is generated based on adjacent 5 LR frames. From the result, one can observe that even compared with the recent Draft Learning and BRCN, our proposed STR-ResNet achieves a significant performance gain over them. In particular, the average gain over the second best BRCN is as high as 0.56 dB. VE and 3DKR achieve better reconstructed results than Bicubic. However, their PSNR results are lower than very recent single image SR methods, such as SRCNN and A+, which only make use of the intra-frame spatial correlation. Due to exploiting the spatial and temporal correlations jointly, the proposed STR-ResNet achieves the best objective result. Draft learning shows inferior performance to many methods in the objective evaluation because it suffers from the artifacts caused by inaccurate optical flow estimation. However, it is worth noting that the subjective evaluation hereafter will demonstrate its superiority on visual quality for reconstructing salient features of images, where optical flow estimation is reliable.

5.6. Subjective evaluation

Figs. 7–9 visualize the SR results of different methods. Bicubic generates blurred results. A+ and SRCNN generate sharper results. However, without exploiting the temporal correlation, some visually important features are blurred, such as the brand text in Fig. 7 and the long edge of the tractor surface in Fig. 8. In contrast, video SR methods, such as 3DSKR and Draft Learning, generate results with richer details. But 3DSKR may suffer from inaccurate motion estimation and generate block artifacts, and Draft Learning produces granular artifacts in smooth regions, where optical flow estimation is unreliable. Due to RNN's strong capacity of modeling complex motions, BRCN and our method present rather sharp results. Especially, the proposed STR-ResNet recovers details with a very natural look, such as the long edge of the tractor in Fig. 8 and the wings of the bee in Fig. 9.

We also visualize the SR results on a group of adjacent frames from four real-time SR methods (A+, VE, BRCN and STR-ResNet) in Fig. 10. A+ and VE generate over-smooth reconstructed results, *e.g.*, a wider railtrack. The result of BRCN contains obvious ring artifacts. The proposed STR-ResNet produces a clean estimation, with a long direct railtrack and the sharpest Letter B.

5.7. Time cost

We report the time cost of our STR-ResNet and compare its efficiency with other state-of-the-art methods. Table 4 presents their running time (in secs.) in 4 × enlargement on input images with two resolution input settings (50 × 50 and 495 × 270, *Sunflower*). The BRCN is implemented by ourselves. Other compared methods are tested based on the public available codes from the authors. We implement BRCN and our STR-ResNet using Caffe with its Matlab wrapper. We evaluate the running time of all the algorithms with the following machine configuration: Intel X5675 3.07 GHz and 24 GB memory. For A+, NE + LLE, NE + NNLS, 3DKR, and Draft, their publicly available CPU

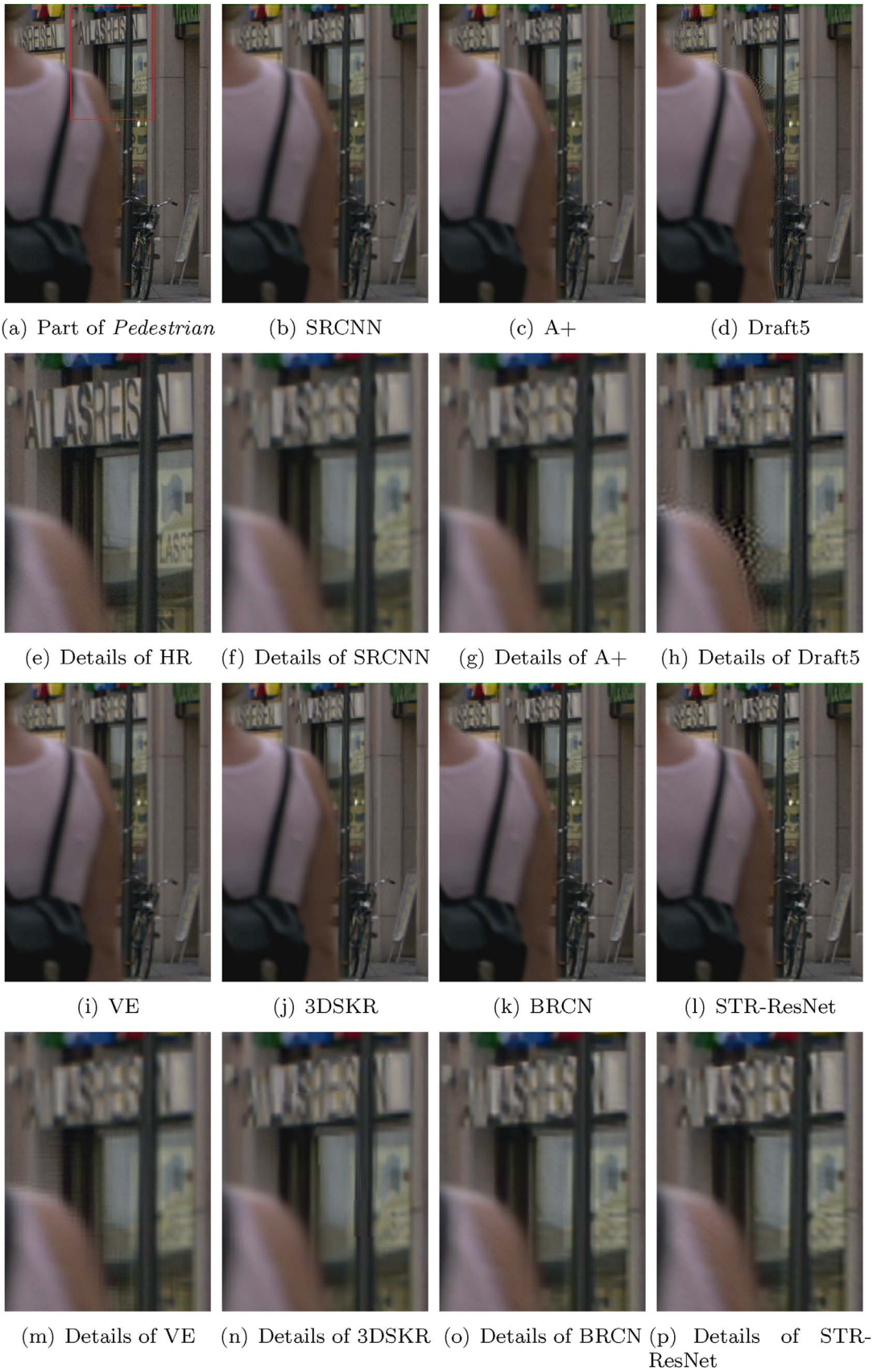


Fig. 7. The reconstruction results of *Pedestrian* with different methods (enlarge factor: 4 ×).

Fig. 8. The reconstruction results of *Tractor* with different methods (enlarge factor: 4 ×).

versions are tested. For BRCN and STR-ResNet, their GPU versions are tested. For SRCNN, both versions are evaluated.

We present the running time in Table 4 and for better visualizing the trade-off between the effectiveness and efficiency of these methods, we also present Fig. 10. As shown in Table 4, two single image SR methods, A+ and NE+LLE, are the most time-efficient. Our method, with GPU support, costs 2.797 and 124.945 s for performing SR on an input image with input sizes of 50 × 50 and 480 × 270 and the corresponding output sizes of 200 × 200 and 1920 × 1080. BRCN is faster than SR-ResNet because it owns a lighter framework. SR-ResNet and BRCN keep the same in orders as the CPU version of SRCNN, NE+NNLS in running time. Comparatively, two effective video SR methods, 3DKR and DraftLearn, suffer from high computational complexity, with more than 3 h to reconstruct one HR frame. That is because they suffer from the

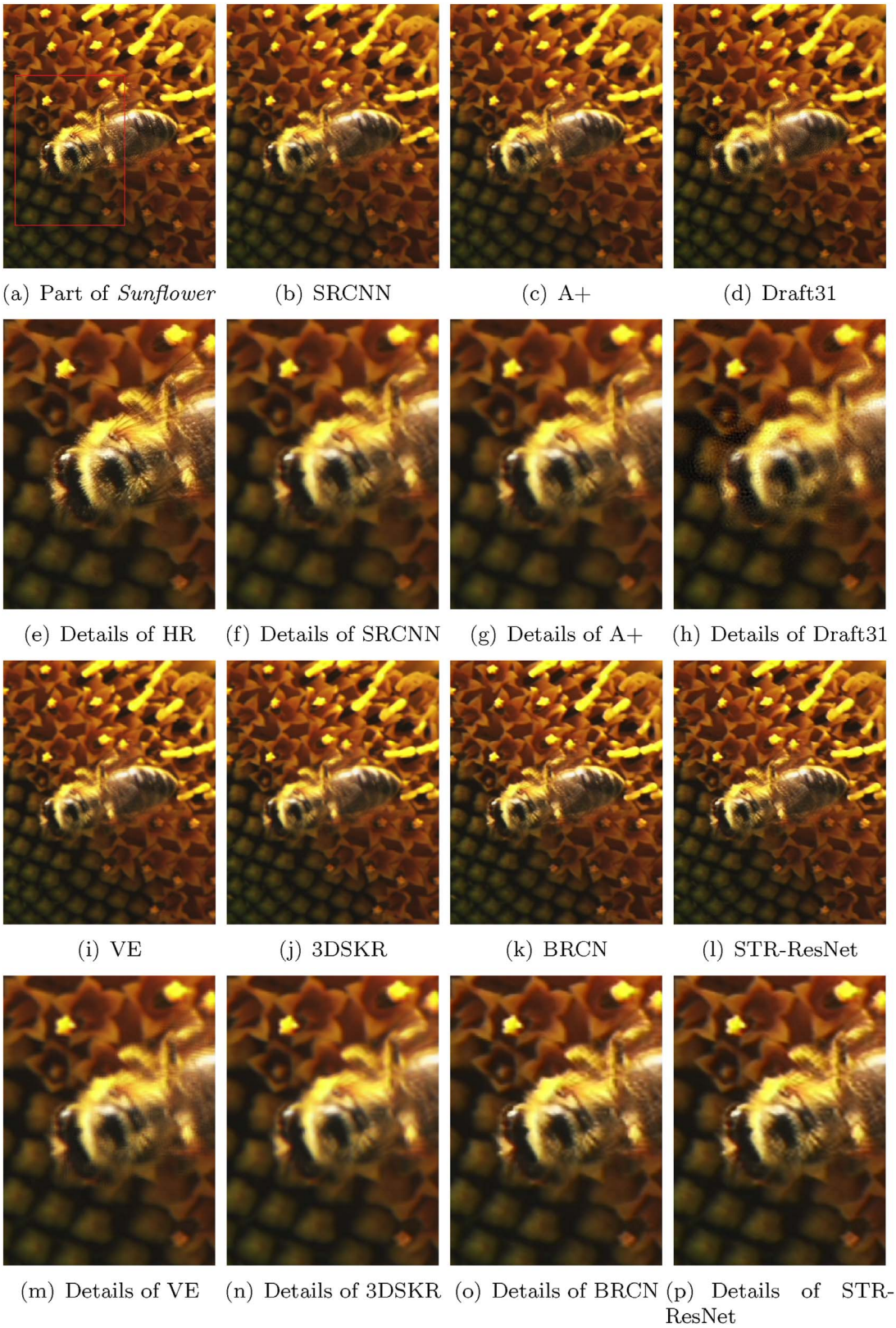


Fig. 9. The reconstruction results of *Sunflower* with different methods (enlarge factor: 4 ×).

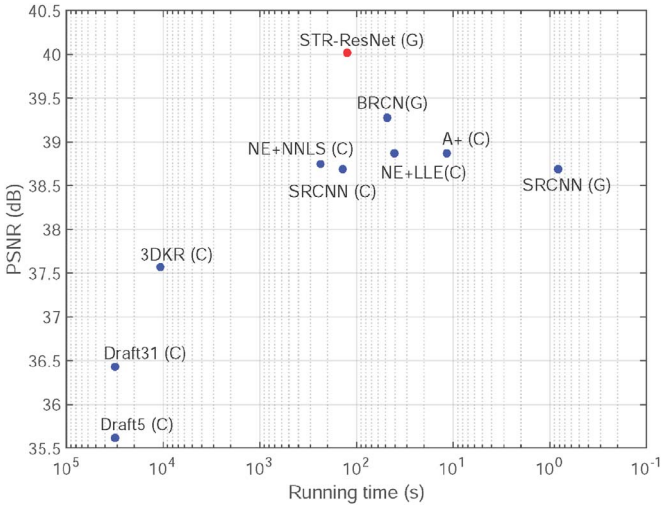


Fig. 10. The performance of our STR-ResNet compared with state-of-the-art methods, including the effectiveness and time complexity, in 4 × enlargement on *sunflower* (the input spatial resolution: 480 × 270). (C) and (G) denote the speeds of the CPU and GPU version, respectively.

Table 4
The time complexity of STR-ResNet compared with state-of-the-art methods.

Input Resolution	SRCNN (C)	SRCNN (G)	A+ (C)	NE+LLE (C)
50 × 50	2.465	0.005	0.141	0.662
480 × 270	137.950	0.816	11.760	40.696
Input resolution	3DKR (C)	Draft (C)	BRCN (G)	STR-ResNet (G)
50 × 50	134.169	625.222	1.206	2.797
480 × 270	10693.080	31431.126	48.497	124.945

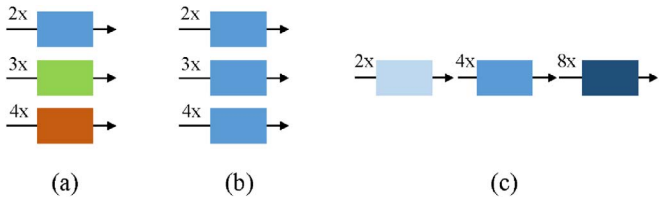


Fig. 11. Different ways to deal with various super-resolution factors. (a) One model for one degradation condition. (b) One model for all cases. (c) A cascaded or recurrent model to handle different cases at different stages.

computational burden of steering kernel computation and regularized optical flow estimation, respectively.

6. More discussions

STR-ResNet addresses the problems of video super-resolution implicitly. Now, we explain the configuration about them and how the network handles the related factors.

6.1. Degradation factors

The capacity of STR-ResNet to handle the restoration is decided by the paired training data synthesized from a certain kind of degradation conditions. It is also flexible for STR-ResNet to extend to deal with different degradation conditions in commonly used ways: (1) one model for one degradation condition as shown in Fig. 11 (a), such as SRCNN (201, 2016), A+ (Timofte et al., 2014) and BRCN (Huang et al., 2015); (2) one model for all cases as shown in Fig. 11 (b), such as VDSR (Kim et al., 2016b) and DRCN (Kim et al., 2016a); (3) a cascaded or recurrent model to handle different cases at different stages as shown in Fig. 11 (c), such as CSCN(Wang et al., 2015b) and deep Laplacian pyramid network (Lai et al., 2017).

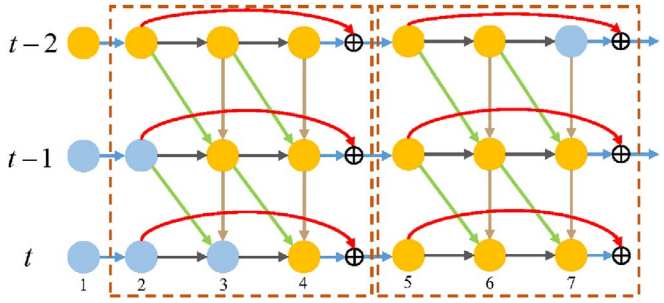


Fig. 12. The convolutional paths that propagate the information of the $(t-2)$th frame to the tth one, denoted in orange color.(For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article).

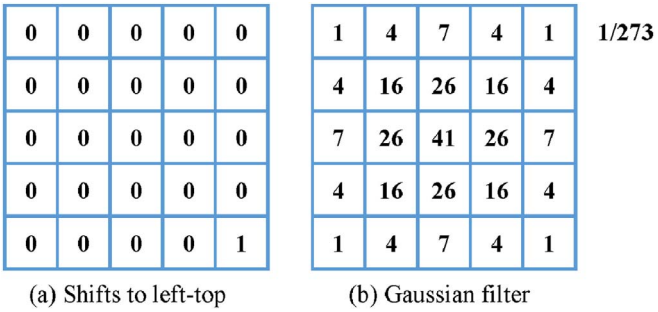


Fig. 13. The convolutional layers for (a) shifting the pixel locations and (b) carrying on Gaussian filter, respectively.

6.2. Modeling motion context

We use RNNs to model the temporal dependency between adjacent frames. The information is propagated by inter-frame connections, i.e. context convolution and recurrent convolution, through every direct adjacent frame pairs gradually. For example, as shown in Fig. 12, to predict the tth HR frame, the information of the $(t-2)$th frame is first propagated to the $(t-1)$th sub-network which aims to estimate the $(t-1)$th HR frame. Then, the information in the $(t-1)$th sub-network passes to the tth sub-network. In this process, the feature transformation, alignment and fusion between the adjacent frames are modeled end-to-end.

Some works (Kim et al., 2016a; 2016b) on related fields have proved that, a convolutional network has the capacity to automatically estimate motions. For a single convolutional layer, it can both model the geometric transformation and carry on filter processing. Using the kernels in Fig. 13(a) and (b), the convolutions could shift pixels in the left-top direction and carry on Gaussian filter, respectively.

By cascading several inter-frame convolutional connections among adjacent frames, STR-ResNet embeds motions implicitly and equally

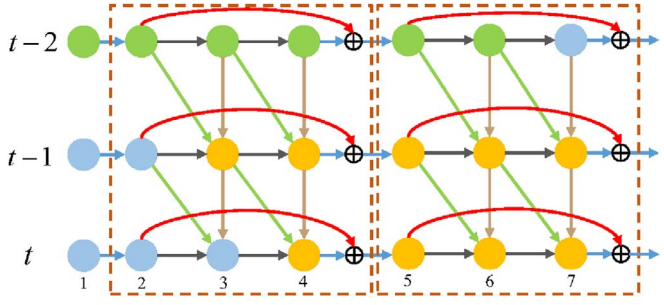


Fig. 14. The large displacement makes it hard for the information of the $(t-2)$th frame to be transported to the $(t-1)$th sub-network.

Table 5
The PSNR results of STR-ResNet with and without motion compensations.

Video	No-compensation	Compensation
Tractor	33.85	**33.92**
Sunflower	**40.02**	39.99
Blue Sky	**32.23**	32.12
Station	35.63	**36.01**
Pedestrian	**35.22**	35.31
Rush Hour	**40.30**	40.11
Average	36.21	**36.24**

carries on filters on the motion trajectories. Besides, to better model inter-frame motions, STR-ResNet further takes not only multiple LR frames but also the residues of these adjacent LR frames as inputs and tries to predict the temporal residues of HR frames in the penultimate layer. It is clearly shown that, in the ablation analysis of the supplementary material, adding temporal residue prediction boosts the SR performance.

6.3. Occlusions and large displacements

Because the tth sub-network is constrained to estimate the tth HR frame, only useful information from adjacent frames for that purpose is aggregated to the tth sub-network. The information flow between the adjacent frames where occlusions and large displacements happen will be cut off.

For example, as shown in Fig. 14, if the $(t-2)$th frame has a large displacement to the $(t-1)$th one, the $(t-1)$th sub-network may fuse less information from the $(t-2)$th sub-network. Therefore, when estimating the tth HR frame, the information propagated from the $(t-1)$th sub-network contains less information from the $(t-2)$th sub-network. Then, the tth frame prediction is influenced little by the large displacement.

We provide a video supplementary material[6] to show the robustness of STR-ResNet when occlusions and large displacements exist. In the supplementary material, the SR results of *Pedestrian* and *Tractor* sequences clearly demonstrate that our method presents naturally looking results.

6.4. Motion compensations

We compare the versions with/without motion compensations. The results are shown in Table 5. The limits and advantages of STR-ResNet in an implicit way to model motions are observed. The STR-ResNet without motion compensations is capable to handle complex motions robustly and achieves superior performance in Sequences Sunflower, Blue Sky and Rush Hour. Comparatively, when the motions in the sequences are consistent and there are available salient geometric features, motion compensation significantly boosts the SR performance.

6.5. Temporal consistency

Most learning-based SR approaches, including STR-ResNet, by nature are good at keeping temporal consistency. These methods, trained solely with MSE loss, they usually “regression to mean” (Timofte et al., 2016). Namely, the network tends to predict the mean of several HR signals. Thus, based on the similar LR inputs, the network will reconstruct similar HR results. Our video supplementary material demonstrates that, compared with Video Enhancer, the implicit methods BRCN and STR-ResNet provide more temporally consistent SR results.

[6] http://www.icst.pku.edu.cn/struct/att/STR-Video-SR.mp4.

7. Conclusion and future work

In this paper, we proposed a novel Spatial-Temporal Recurrent Residual Network (STR-ResNet) for video super-resolution. This network simultaneously models high frequency details of single frames, the differences between high resolution (HR) and low resolution (LR) frames, as well as the changes of these adjacent detail frames. To model intra-frame correlation, a CNN structure with bypass connections is constructed to learn spatial residual of a single frame. To model inter-frame correlation, STR-ResNet estimates the temporal residue implicitly. Extensive experiments have demonstrated the effectiveness and efficiency of our method for video SR. However, the recurrence step of STR-ResNet to model frames is limited by available GPU memory and the convolutional recurrent connections cannot have a long-term memory. In the future work, we aim to overcome such limitations and implement a longer-term memorized video SR method, which makes use of longer LR video clips to reconstruct one HR frame.

Supplementary material

Supplementary material associated with this article can be found, in the online version, at 10.1016/j.cviu.2017.09.002.

References

Anon, 2016. Image super-resolution using deep convolutional networks. IEEE Trans. Pattern Anal. Mach. Intell. 38 (2), 295–307.

Agostinelli, F., Anderson, M.R., Lee, H., 2013. Adaptive multi-column deep neural networks with application to robust image denoising. Proc. Annual Conference on Neural Information Processing Systems. pp. 1493–1501.

Aguena, M.L., Mascarenhas, N.D., 2006. Multispectral image data fusion using {POCS} and super-resolution. Comput. Vision Image Understand. 102 (2), 178–187.

Baker, S., Kanade, T., 1999. Super-resolution optical flow. CMU-RI-TR-99-36.

Burger, H. C., Schuler, C. J., Harmeling, S., 2012a. Image denoising with multi-layer perceptrons, part 1: comparison with existing algorithms and with bounds. ArXiv preprint, arXiv:1211.1544.

Burger, H. C., Schuler, C. J., Harmeling, S., 2012b. Image denoising with multi-layer perceptrons, part 2: training trade-offs and analysis of their mechanisms. ArXiv preprint, arXiv:1211.1552.

Chang, H., Yeung, D.-Y., Xiong, Y., 2004. Super-resolution through neighbor embedding. Proc. IEEE Int'l Conf. Computer Vision and Pattern Recognition. vol. 1. I–I

Cui, Z., Chang, H., Shan, S., Zhong, B., Chen, X., 2014. Deep network cascade for image super-resolution. Proc. IEEE European Conf. Computer Vision.

Dong, C., Loy, C.C., He, K., Tang, X., 2014. Image super-resolution using deep convolutional networks. Proc. IEEE European Conf. Computer Vision.

Dong, W., Zhang, L., Shi, G., 2013. Centralized sparse representation for image restoration. Proc. IEEE Int'l Conf. Computer Vision. pp. 1259–1266.

Farsiu, S., Robinson, M.D., Elad, M., Milanfar, P., 2004. Fast and robust multiframe super resolution. IEEE Trans. Image Process. 13 (10), 1327–1344.

Fransens, R., Strecha, C., Gool, L.V., 2007. Optical flow based super-resolution: a probabilistic approach. Comput. Vision Image Understand. 106 (1), 106–115.

Freeman, W.T., Jones, T.R., Pasztor, E.C., 2002. Example-based super-resolution. IEEE Comput. Graph. Appl. 22, 56–65.

Gatys, L.A., Ecker, A.S., Bethge, M., 2016. Image style transfer using convolutional neural networks. Proc. IEEE Int'l Conf. Computer Vision and Pattern Recognition. pp. 2414–2423.

He, H., Kondi, L.P., 2006. An image super-resolution algorithm for different error levels per frame. IEEE Trans. Image Process. 15 (3), 592–603.

Huang, Y., Wang, W., Wang, L., 2015. Bidirectional recurrent convolutional networks for multi-frame super-resolution. Proc. Annual Conference on Neural Information Processing Systems. pp. 235–243.

Huang, Y., Wang, W., Wang, L., 2017. Video super-resolution via bidirectional recurrent convolutional networks. IEEE Trans. Pattern Anal. Mach. Intell. PP (99). 1–1

Huhle, B., Schairer, T., Jenke, P., Straer, W., 2010. Fusion of range and color images for denoising and resolution enhancement with a non-local filter. Comput. Vision Image Understand. 114 (12), 1336–1345.

Irani, M., Peleg, S., 1991. Improving resolution by image registration. CVGIP: Graph. Models Image Process. 53 (3), 231–239.

Ismaeil, K.A., Aouada, D., Mirbach, B., Ottersten, B., 2016. Enhancement of dynamic depth scenes by upsampling for precise super-resolution (up-sr). Comput. Vision Image Understand. 147, 38–49. Spontaneous Facial Behaviour Analysis.

Jain, V., Seung, S., 2009. Natural image denoising with convolutional networks. In: Koller, D., Schuurmans, D., Bengio, Y., Bottou, L. (Eds.), Proc. Annual Conference on Neural Information Processing Systems, pp. 769–776.

Jia, Y., Shelhamer, E., Donahue, J., et al., 2014. Caffe:convolutional architecture for fast feature embedding. ACM International Conference on Multimedia. pp. 675–678.

Joshi, M.V., Chaudhuri, S., 2006. Simultaneous estimation of super-resolved depth map and intensity field using photometric cue. Comput. Vision Image Understand. 101

(1), 31–44.
Kanaev, A.V., Miller, C.W., 2013. Multi-frame super-resolution algorithm for complex motion patterns. Opt. Express 21 (17), 19850–19866.
Kim, J., Lee, J.K., Lee, K.M., 2016a. Accurate image super-resolution using very deep convolutional networks. Proc. IEEE Int'l Conf. Computer Vision and Pattern Recognition. pp. 1646–1654.
Kim, J., Lee, J.K., Lee, K.M., 2016b. Deeply-recursive convolutional network for image super-resolution. Proc. IEEE Int'l Conf. Computer Vision and Pattern Recognition. pp. 1637–1645. http://dx.doi.org/10.1109/CVPR.2016.181.
Lai, W.-S., Huang, J.-B., Ahuja, N., Yang, M.-H., 2017. Deep laplacian pyramid networks for fast and accurate super-resolution. IEEE Conf. Comput. Vis. Pattern Recognit.
Liao, R., Tao, X., Li, R., Ma, Z., Jia, J., 2015a. Video super-resolution via deep draft-ensemble learning. Proc. IEEE Int'l Conf. Computer Vision. pp. 531–539.
Liao, R., Tao, X., Li, R., Ma, Z., Jia, J., 2015b. Video super-resolution via deep draft-ensemble learning. Proc. IEEE Int'l Conf. Computer Vision. pp. 531–539.
Liu, C., Sun, D., 2014. On bayesian adaptive video super resolution. IEEE Trans. Pattern Anal. Mach. Intell. 36 (2), 346–360.
Lu, X., Li, X., 2014. Multiresolution imaging. IEEE Trans. Cybern. 44 (1), 149–160.
Ma, C., Yang, C.-Y., Yang, X., Yang, M.-H., 2017. Learning a no-reference quality metric for single-image super-resolution. Comput. Vision Image Understand.
Mairal, J., Bach, F., Ponce, J., Sapiro, G., Zisserman, A., 2009. Non-local sparse models for image restoration. Proc. IEEE Int'l Conf. Computer Vision. pp. 2272–2279.
Marquina, A., Osher, S., 2008. Image super-resolution by TV-regularization and bregman iteration. J. Sci. Comput. 37 (3), 367–382.
Nguyen, K., Fookes, C., Sridharan, S., Denman, S., 2013. Feature-domain super-resolution for iris recognition. Comput. Vision Image Understand. 117 (10), 1526–1535.
Omer, O.A., Tanaka, T., 2009. Region-based weighted-norm approach to video super-resolution with adaptive regularization. Proc. IEEE Int'l Conf. Acoustics, Speech, and Signal Processing. pp. 833–836.
Osendorfer, C., Soyer, H., van der Smagt, P., 2014. Image super-resolution with fast approximate convolutional sparse coding. Neural Information Processing.
Protter, M., Elad, M., Takeda, H., Milanfar, P., 2009. Generalizing the nonlocal-means to super-resolution reconstruction. IEEE Trans. Image Process. 18 (1), 36–51.
Rudin, L.I., Osher, S., Fatemi, E., 1992. Nonlinear total variation based noise removal algorithms. Physica D 60, 259–268.
Salvador, J., Prez-Pellitero, E., 2015. Naive bayes super-resolution forest. Proc. IEEE Int'l Conf. Computer Vision. pp. 325–333.
Schuler, C.J., Hirsch, M., Harmeling, S., Schölkopf, B., 2016. Learning to Deblur. IEEE Trans. Pattern Anal. Mach. Intell. 38 (7), 1439–1451.
Singh, A., Ahuja, N., 2015. Learning ramp transformation for single image super-resolution. Comput. Vision Image Understand. 135, 109–125.
Sun, J., Sun, J., Xu, Z., Shum, H.Y., 2011. Gradient profile prior and its applications in image super-resolution and enhancement. IEEE Trans. Image Process. 20 (6), 1529–1542.
Takeda, H., Milanfar, P., Protter, M., Elad, M., 2009. Super-resolution without explicit subpixel motion estimation. IEEE Trans. Image Process. 18 (9), 1958–1975.
Takeda, H., Milanfar, P., Protter, M., Elad, M., 2009. Super-resolution without explicit subpixel motion estimation. IEEE Trans. Image Process. 18 (9), 1958–1975.
Timofte, R., De, V., Van Gool, L., 2013. Anchored neighborhood regression for fast example-based super-resolution. Proc. IEEE Int'l Conf. Computer Vision.
Timofte, R., De Smet, V., Van Gool, L., 2014. A+: Adjusted anchored neighborhood regression for fast super-resolution. Proc. IEEE Asia Conf. Computer Vision.
Timofte, R., Smet, V.D., Gool, L.V., 2016. Semantic super-resolution: when and where is it useful? Comput. Vision Image Understand. 142, 1–12.
Vincent, P., Larochelle, H., Lajoie, I., Bengio, Y., Manzagol, P.-A., 2010. Stacked denoising autoencoders: learning useful representations in a deep network with a local denoising criterion. J. Mach. Learn. Res.
Wang, Z., Liu, D., Yang, J., Han, W., Huang, T., 2015. Deep networks for image super-resolution with sparse prior. Proc. IEEE Int'l Conf. Computer Vision. pp. 370–378.
Wang, Z., Liu, D., Yang, J., Han, W., Huang, T., 2015. Deep networks for image super-resolution with sparse prior. Proc. IEEE Int'l Conf. Computer Vision. pp. 370–378. http://dx.doi.org/10.1109/ICCV.2015.50.
Wu, W., Zheng, C., 2013. Single image super-resolution using self-similarity and generalized nonlocal mean. IEEE International Conference of IEEE Region. pp. 1–4.
Xie, J., Xu, L., Chen, E., 2012. Image denoising and inpainting with deep neural networks. In: Pereira, F., Burges, C.J.C., Bottou, L., Weinberger, K.Q. (Eds.), Proc. Annual Conference on Neural Information Processing Systems, pp. 341–349.
Xu, L., Ren, J.S., Liu, C., Jia, J., 2014. Deep convolutional neural network for image deconvolution. Proc. Annual Conference on Neural Information Processing Systems. pp. 1790–1798.
Yan, Z., Zhang, H., Wang, B., Paris, S., Yu, Y., 2016. Automatic photo adjustment using deep neural networks. ACM Trans. Graph. 35 (2), 11:1–11:15.
Yang, J.C., Wright, J., Huang, T.S., Ma, Y., 2010. Image super-resolution via sparse representation. IEEE Trans. Image Process. 19 (11), 2861–2873.
Yang, W., Feng, J., Yang, J., et al., 2017. Deep edge guided recurrent residual learning for image super-resolution. IEEE Trans. Image Process. 26 (12), 5895–5907.
Yin, M., Gao, J., Cai, S., 2015. Image super-resolution via 2d tensor regression learning. Comput. Vision Image Understand. 132, 12–23.
Yuan, Q., Zhang, L., Shen, H., 2013. Regional spatially adaptive total variation super-resolution with spatial information filtering and clustering. IEEE Trans. Image Process. 22 (6), 2327–2342.
Zeng, K., Yu, J., Wang, R., Li, C., Tao, D., 2016. Coupled deep autoencoder for single image super-resolution. IEEE Trans.Cybern. PP (99), 1–11.
Zhang, H., Yang, J., Zhang, Y., Huang, T.S., 2013. Image and video restorations via nonlocal kernel regression. IEEE Trans. Cybern. 43 (3), 1035–1046.
Zhang, X., Xiong, R., Ma, S., Li, G., Gao, W., 2015. Video super-resolution with registration-reliability regulation and adaptive total variation. J. Vis. Commun. Image Represent.
Zhao, W., Sawhney, H., 2002. Is super-resolution with optical flow feasible? Proc. IEEE European Conf. Computer Vision. Berlin, Heidelberg.