MultiBoot VSR: Multi-Stage Multi-Reference Bootstrapping for Video Super-Resolution

Ratheesh Kalarot and Fatih Porikli
San Diego CBG Device & Hardware CC, Huawei
ratheesh.kalarot@huawei.com, fatih.porikli@huawei.com

Abstract

To make the best use of the previous estimations and shared redundancy across the consecutive video frames, here we propose a scene and class agnostic, fully convolutional neural network model for 4× video super-resolution. One stage of our network is composed of a motion compensation based input subnetwork, a blending backbone, and a spatial upsampling subnetwork. We recurrently apply this network to reconstruct high-resolution frames and then reuse them as additional reference frames after reshuffling them into multiple low-resolution images. This allows us to bootstrap and enhance image quality progressively. Our experiments show that our method generates temporally consistent and high-quality results without artifacts. Our method is ranked as the second best based on the SSIM scores on the NTIRE2019 VSR Challenge, Clean Track.

Figure 1. **Left:** Bicubic. **Right:** Sample results of our video super-resolution method. Test images are from the REDS [15] validation dataset of the NTIRE 2019 VSR Challenge Clean Track.

1. Introduction

Improving the spatial resolution of visual data using its original low-resolution version by non-optical means has been one of the ultimate goals of image enhancement for many years. Nowadays, video super-resolution (VSR) has become even more critical as video accounted for 73 percent of all internet traffic in 2016, and it will make up 82 percent in 2021 [19], reaching an astounding bandwidth of 187 exabytes (187 billion gigabytes) in a few years. The demand is coming from all types of internet video, including on-demand content, webcam viewing, traditional TV options available over the internet as well as live video thanks to new video offerings of social media, broadcast of live sports, video surveillance and live over-the-top bundles from content providers. VSR promises not only a reduced bandwidth but also reduced latency for all these applications.

Nevertheless, the reconstruction of high-resolution (HR) version from a low-resolution (LR) input is an ill-posed problem since the original HR information is lost. To manage this, additional constraints that mainly rely on data redundancy or multiple observations are imposed. The redundancy is often enforced in the form of local correlations by imposing sparsity constraints [27] or assuming constancy of various image statistics such as multi-scale patch recurrence [6] to recover lost high-frequency details. In case different observations of the same scene are available, the shared redundancy across the observations are used to regulate the problem to invert the downscaling process [17]. Video data naturally has a high degree of correlation across consecutive frames within the same shot, which can be exploited to reconstruct its HR version.

As the demand amplifies, the new trends in super-resolution standing on the recent success of the convolutional neural networks have also revolutionized the field of VSR. Deep learning based techniques have quickly achieved better high-resolution estimations. In spite of the great strides, the existing VSR schemes still have several issues. For instance, patch based neural networks, in particular the ones with shallower structures such as [1], [10],

[20] lack of global context and focus on local features. On the other hand, deeper network with a larger number of parameters are harder to train as at the initial training stages, the choice of the optimal hyperparameters such as the learning rate becomes crucial yet more difficult to estimate. Because of the GPU memory limitations, VSR networks typically trained in a patch based fashion. However, patch size (even for big 128×128 patches) may not allow covering large object motion (as in classical aperture problem) and learning benefiting aspects from larger receptive fields. A direct adaptation of single image super-resolution frameworks, e.g. [11], for video, e.g. [18], often requires upscaling at the end of the pipeline at once, which causes a learning bottleneck when the target upsampling factor is 4× or more.

In this paper, we propose a fully convolutional neural network model for 4× video super-resolution that is capable of generating sharp video frames with high-resolution details by taking advantage of motion compensated reference frames and reusing the high-resolution versions of the reference frames estimated in the previous stages for a bootstrapped (in other words, recurrent) resolution enhancement process. Sample results of our method can be seen in Fig. 1 and 5. We do not make any assumption about the objects in the images; thus our method is class-agnostic.

Our intuition is that the super-resolution performance of neural networks improves when the number of low-resolution references that it can build evidence on high-resolution details increases. For this reason, we employ multiple motion-compensated reference frames of the current frame. As noted in previous works, it is not straightforward to optimize the neural networks for temporally consistent results since no information of the previously super-resolved frame is directly included in the current step. To encourage temporally consistent results, we use a bootstrapped frame-recurrent approach where the reconstructed high-resolution frame of the previous step is dispatched into the network after rearranging its pixels into multiple low-resolution images.

Our model consists of three main components as shown in Fig 2; an input subnetwork that shuffles and combines multiple motion-compensated reference frames, a blending backbone that applies fully convolutional blocks on low-resolution feature maps, and a spatial upsampling subnetwork that reconstructs the high-resolution image.

The blending backbone is made up of fully convolutional residual units. After a long series of residual units, we embed a direct skip connection from the first feature layer to the last one to maintain the influence of the original reference frames on the feature map of the last layer. Thus, our backbone is conditioned on reconstructing the residual info, which includes the missing high-resolution patterns in visual data. The residual blocks and direct skip connection also allow us to deepen the blending backbone, which boosts the overall representation capacity of the network and to increase the areas of the receptive fields for the higher level convolutional layers, which enables better contextual feedback. The representation capacity is supposedly proportional to the number of network parameters. The blending backbone utilizes different combinations of motion compensated reference frames including the shuffled and low-resolution mapped versions of the estimated high-resolution output of a previous stage. This permits transferring the initial model into progressively more complex networks in the following stages. Note that, each state is an independent network.

Following the blending backbone, we apply a spatial upsampling subnetwork to reconstruct a higher-resolution image from the feature map. This subnetwork uses pixel shuffling with learned weights, thus it does not require deconvolutions. One can consider that the blending backbone prepares the best possible feature maps, which have a large number of channels, and the spatial upsampling subnetwork layers for rearranging the feature maps into the high-resolution image using the learned weights of the filters of these layers.

We then bootstrap the VSR process by space-to-depth rearranging the estimated high-resolution image into multiple low-resolution channels, updating the motion compensation, combining the estimated and original frames, and applying a similar network again as the next stage. This frame-recurrent approach bootstraps on the estimated high-resolution results and provides additional performance improvements.

To summarize, the contributions of this paper are:

- We introduce a general-purpose, class-agnostic, and fully convolutional network for video super-resolution that processes multiple-reference frames in their original low-resolution format throughout its blending backbone and then reconstructs the high-resolution output from rearranged feature maps.
- We recurrently apply the network to leverage on the reconstructed high-resolution outputs from the previous stages to bootstrap and enhance image quality progressively.
- As our experiments demonstrate, our method generates temporally consistent results and handles complex real-world scenes depicting moving objects, fast camera motion, uncontrolled lighting conditions, shadows without inducing perceptual artifacts.
- Our method is ranked officially as the second best based on the SSIM scores on the NTIRE 2019 VSR Challenge Clean Track with our previously submitted results. Our latest model further improves our submitted results by 0.3 dB in terms of the PSNR scores.

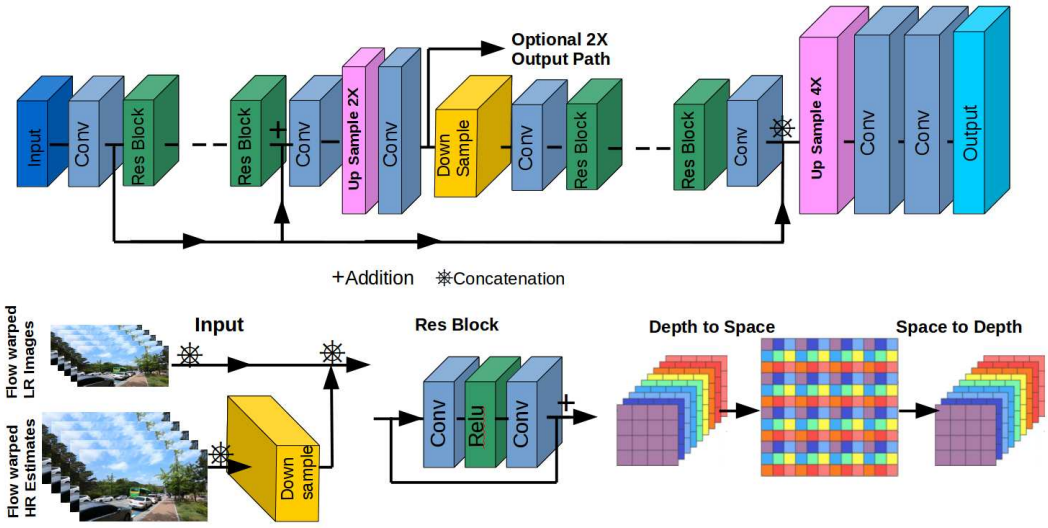

Figure 2. Network architecture. Our network has three parts; an input subnetwork (with optical flow based motion compensation), a blending backbone, and a spatial upsampling subnetwork.

2. Related Work

VSR methods have mainly emerged as adaptations of image super-resolution techniques, for which a survey of conventional methods can be found in [11, 26].

It is possible to categorize the past approaches in single image-resolution in terms of the reference examples they utilize. The internal reference based methods attempt to identify internal redundancies using patch recurrence to obtain essential information in upscaling of the patches [6]. The recurrence assumes patches in a single image encode multi-scale relationships, which may allow inferring the missing high-frequency content at a given scale from the coarser scale patches. Dictionary learning approaches, which define the input image as a sparse linear combination of dictionary atoms coupled to an HR dictionary [13], may employ both internal and external reference examples since such dictionaries are learned in a data-driven manner using internal and external training pairs [24, 25, 6, 27].

Deep learning based methods build models on mainly external references while approximating complex nonlinear functions inherent in super-resolution task. One of the pioneering convolutional neural networks, SRCNN, is made up of a simple 3-layer model [2]. Since then, neural networks have been attaining superior performance, often leveraging on deep structures and residual learning [9]. These methods take a bicubic enlarged LR image as an input to their first convolution layer, which leads to high computational burden. To address this problem, the work in [21] keeps the spatial resolution of the feature maps in the network as in LR image and obtains the HR image through a simple reordering of the multi-channel output feature map. Similar solutions to improve efficiency have also been proposed based on transposed convolutions [3, 7]. Removing batch normalization layers, reusing the saved memory (which can be up to 40 percent of the original network) to employ a much larger model, and postponing the upsampling to the very end of the network, [11] further improved the performance. They argue that batch normalization causes the loss of the scale information and hinders range flexibility of the network.

Extending super-resolution task from images to videos, VSR methods expose and exploit the temporal correlations to access multiple references of the same scene through aggregating spatial information across consecutive frames while compensating for interframe motion. Focusing on deep learning based solutions, the main idea is to align and warp neighboring frames to the current frame before all images are fed into a super-resolution network [8, 1, 23]. It is therefore quite common for VSR to explicitly compute motion, for instance, using off-the-shelf optical flow algorithms [22]. Computing motion, on the other hand, is usually expensive. Alternatively, motion estimation layers can be integrated into the super-resolution [12] as part of the network. Similarly, [14] uses a sliding window approach and combines the frame alignment and super-resolution steps. Joint motion compensation and super-resolution can also be performed using comparably shallow recurrent bidirectional networks [4] without explicit motion compensation. However, training an integrated and often low-capacity motion estimation layer is challenging, which might distort accuracy. Rather than compensating for motion, [13, 8] operate on a stack of frames at the same time to generate different high-resolution images and then condense the results into a single image. However, such methods are sensitive to the degree of motion in

the videos. Another noteworthy VSR method [18] applies perceptual loss in their adversarial recurrent network that uses optical flow in order to exploit temporal cues across frames and a temporal-consistency loss term to reinforce coherency across frames. This network postpones upscaling to the very end of the generative network, which arguably makes the learning more difficult than stage-wise learning. The work in [1] introduced pixel shuffle based upsampling. Their network also has a dedicated optical flow subnetwork. However, a patch-based training for motion compensation may not generalize and account for large motion. Later, [10] provided minor improvements by switching from explicit motion compensation to 3D filters. A frame-recurrent approach which passes the previously estimated high-resolution frame as an input for the following iteration is recently proposed by [20]. Unlike our method, this work uses only the previous frame as a reference and attempts to learn an individual optical flow network, which can be considered as its weaknesses.

3. Proposed Method: MultiBoot

Our bootstrapped VSR solution is composed of multiple stages. In this paper, we report the results for two consecutive stages, yet our methodology can be applied recurrently more than that.

Each stage has a small input subnetwork, a deep blending backbone, and a spatial upsampling subnetwork as illustrated in Fig 2. See Table 1 for the network parameters.

3.1. Input Subnetwork

The input subnetwork takes multiple motion-compensated (warped) reference frames as the input tensor and applies convolutional filter layers on it. Each reference frame is an RGB image, which is normalized it to [-1,1] for efficient backpropagation. We select these reference frames within a temporal window centered around the current frame. We empirically set the temporal window size to $[t-2, t+2]$, five frames including the current frame, as a trade-off between the speed and accuracy. In our experiments, a temporal window $[t-3, t+3]$ generated slightly better estimations. The GPU inference time for adding two more reference frames is minor (1 millisecond); however, the main issue is the cost of the additional optical computations, which take around 400 milliseconds.

These frames can be sampled uniformly, which is what we report in this paper. They can be selected based on some motion or frame difference measure as well. For instance, the sampling frequency can decrease when the motion is small and vice versa.

We warp each reference frame to the current frame using the interframe motion from each reference image to the current frame as shown in Fig. 3. We estimate dense optical flow by the pre-trained model of FlowNet 2.0 [5], which is reported to be among the best neural network models. FlowNet 2.0 relies on an arrangement of stacked networks that capture large displacements in coarse flow estimates, which are then refined by the following networks. In a final step, these multi-scale estimates are fused by a shallow fusion network. The input subnetwork arranges the warped images of the reference frames and current image in a 15-channels tensor. It then applies 256 filters, $3\times3\times15$ each.

The above description is for the first stage. For the following stages, we have additional references. After the first stage, we estimated 5 HR images; thus in addition to LR reference frames, we also use these 5 estimated HR images. We first rearrange (space-to-depth) the pixels of an estimated HR image into 16 LR images. We then combine all LR images from the 5 estimated HR images into a 240-

Table 1. MultiBoot: Network Parameters

Subnetwork	Type	Shape	Params
Input Stage1	Kernel	3x3x15x256	34560
	Bias	256	256
Blending 16× ResBlocks Stage1	Kernel	3x3x256x256	9437184
	Bias	256	4096
	Kernel	3x3x256x256	9437184
	Bias	256	4096
Upsampling Stage1	Kernel	3x3x256x256	589824
	Bias	256	256
	Kernel	3x3x256x1024	2359296
	Bias	1024	1024
	Kernel	3x3x256x3	6912
	Bias	3	3
	Kernel	3x3x12x256	27648
	Bias	256	256
Input Stage2	Kernel	3x3x255x256	587520
	Bias	256	256
Blending 16× ResBlocks Stage2	Kernel	3x3x256x256	9437184
	Bias	256	4096
	Kernel	3x3x256x256	9437184
	Bias	256	4096
Upsampling Stage2	Kernel	3x3x256x256	589824
	Bias	256	256
	Kernel	3x3x512x2048	9437184
	Bias	2048	2048
	Kernel	3x3x512x2048	9437184
	Bias	2048	2048
	Kernel	3x3x512x3	13824
	Bias	3	3
	Kernel	3x3x3x3	81
	Bias	3	3
Total trainable parameters in Stage1			21,902,595
Total trainable parameters in Stage2			38,952,791

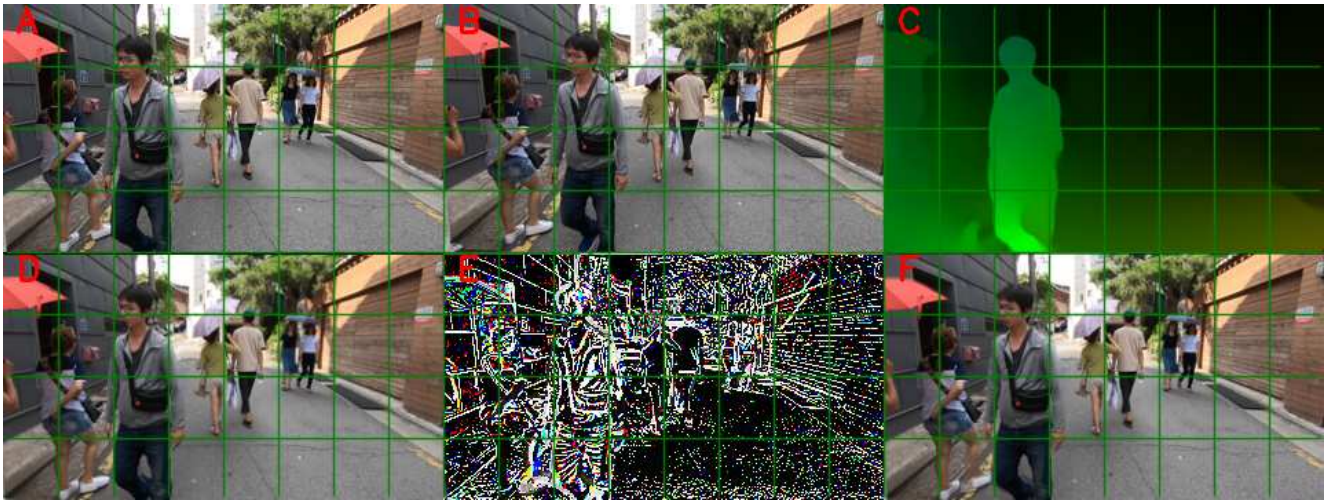

Figure 3. An example of motion compensation: **A)** Input image I_t. **B)** Image I_{t+2}. The grid is added to make the apparent motion across the frames more clear. **C)** Visualization of optical flow from I_t to I_{t+2} computed using Flownet2. **D)** Image $I_{t+2\to t}$ that corresponds I_{t+2} warped to I_t using the computed flow. **E)** Visualization of the difference $|I_t - I_{t+2\to t}|$. **F)** Weighted average image $0.5(I_t + I_{t+2\to t})$, where sharper areas suggest accurate motion compensation and warping.

channels tensor, and concatenate the LR reference frames of the first stage to obtain a 255-channels tensor that has the original LR spatial resolution. For $2\times$ super-resolution, the number of LR images after the space-to-depth rearrangement would be 4.

Please notice that our network uses the original LR resolution frames in all its subnetworks and stages. Since we use the same image size and RGB modality for all references (from the original LR frames as well as from the rearranged images of the estimated HR frames), the learning becomes more efficient. Multiple references provide spatially vibrant local patterns for super-resolution task.

3.2. Blending Backbone

The blending backbone applies fully convolutional blocks on low-resolution feature maps generated by the input subnetwork layers. It is made up of 16 fully convolutional residual units. Each residual unit has a front convolutional layer followed by a ReLU and a second convolutional layer with a skip connection from the first one. Similarly, the blending backbone has also a direct skip connection from the input to the last residual block. This skip connection allows our network to learn the missing high-resolution details by reconstructing the residual info. The structure of the blending backbone of each stage is identical.

The residual blocks and the direct skip connection also permits deepening the blending backbone for each stage. This boosts the overall capacity and increases the receptive field sizes. Thus, our blending backbone feature maps have better access to contextual information in the reference images.

3.3. Spatial Upsampling

We apply a spatial upsampling subnetwork to reconstruct a higher-resolution image from the feature map of the blending backbone. Since we shuffle pixels and we apply a set of convolutional filters, our upsampling does not require deconvolution operations. We rearrange the comparably large number of feature map channels per pixel into a high-resolution image using the learned weights of the filters of the upsampling subnetwork layers. We set the number of layers for the first stage and the second stage to 4 and 5 as the second stage feature map has more channels. Each stage provides $2\times$ super-resolution, yet it is possible to set the upsampling factor to larger ratios since there the feature maps are sufficiently deep.

For a goal of improving the PSNR measure, a straightforward loss function would be the mean squared error (MSE), which measures the average of the squares of the errors, i.e., the average squared difference between the estimated values and what is estimated. However, MSE heavily penalizes the outliers. Recently, the work in [11] empirically demonstrated that the mean absolute error (MAE) works better than the MSE. In our experiments, we also made a similar observation. In particular, at the initial stages of the training, using the MSE based loss functions caused instability. Nevertheless, it is slow to learn by an MAE-based loss at the later epochs. Therefore, we opted to impose the Huber loss function, which is differentiable and combines the benefits of the MAE and MSE. It is defined as

$$L_\delta(d) = \begin{cases} \frac{1}{2}d^2 & \text{for } |d| \le \delta, \\ \delta|d| - \frac{\delta^2}{2} & \text{otherwise} \end{cases} \tag{1}$$

Table 2. NTIRE 2019 VSR Clean Track - ordered by SSIM

Team	PSNR	SSIM
CUHK	31.79	0.8962
Ours - MultiBoot	31.00	**0.8822**
Baidu	31.13	0.8811
TTI	30.97	0.8804
NERCMS	30.91	0.8782
UIUC-IFP	30.81	0.8748
BMIPL UNIST	30.43	0.8666
IPCV IITM	29.99	0.8570
NEU SMILE Lab	29.39	0.8419
MVGL	28.81	0.8249
Team India	28.81	0.8241
Withdrawn team	28.54	0.8170
Bicubic (baseline)	26.48	0.7799

where

$$d = I_{HR}(x, y) - \hat{I}_{HR}(x, y) \quad (2)$$

is the pixel-wise difference between the target (ground-truth) HR image I_{HR} and the estimated HR image $\hat{I}_{HR}$. Above, we set $\delta = 1$, which is the point where the Huber loss function changes from a quadratic to linear.

We trained the first stage network and then the second stage network by using the first stage parameters for initialization. Figure 4 shows the 32×32 input reference patches to the first stage and the estimated 128×128 output patches.

4. Experiments & Discussion

As shown in Table 2, our method is ranked as the second best based on the SSIM scores on the NTIRE2019 VSR Challenge Clean Track with our previously submitted results [16]. When we fine-tuned the hyperparameters of our method after the challenge deadline, the PSNR further improved 0.3 dB over the submitted results. Sample results can be seen in Fig. 5.

Dataset/Augmentation: For training and testing our method, we only used the REDS datasets [15] provided by the NTIRE 2019 benchmark. These datasets include 1280×720 HR images with and the corresponding 320×180 LR versions. There are 240 videos for training, 30 videos for validation, and 30 videos for testing. Each video has 100 frames. The HR versions of the test videos are not released.

We sampled randomly around 1 million (24000∗48) patches of size 128×128 from the 240 training HR videos, 48 patches per image, for our training. We used 3000∗16 patches from the 30 validation videos, 16 per image, for validation.

We augmented the data at the training time by applying one of these transformations; rotating 90°, 180°, 270°, flipping vertically, and flipping horizontally. It is possible to use more sophisticated data augmentation schemes with better computational resources.

Compared to other VSR benchmarks, NTIRE 2019 benchmark is an order of magnitude larger. Our network size (with 32 layers, 256-channels feature maps, 60M parameters) is not at the diminishing return of PSNR, and there is room for further improvements, e.g., using wider and deeper models computational resources allowing.

Despite the variety of videos scenes in the training set, data-driven models may not fully generalize to arbitrary videos. We found that even with short durations videos for fine-tuning, the VSR performance improves.

Training: We trained our network on Nvidia 2080 ti GPU with a batch size of 32 LR-HR image pairs using TensorFlow python library. The training takes around a week, where most of the reduction of the loss happens within the first 48 hours. We used a learning rate of 1e-5 for warm starting, 5e-5 for mid-epoch training, and a lowered rate of 1e-6 towards the later steps. We observed that learning rates of 1e-4 and higher (as reported in the literature) were not stable.

Number of Residual Blocks: Our network executes convolutions all in LR spatial size, thus it allows a high number of residual blocks than some of the existing methods that first bicubic upsample the given LR image then apply convolutions to images and feature maps in HR spatial size. We tested different numbers of residual blocks; 12, 22, 32, and 42. Increasing the number of residual blocks, as expected, improved performance. Using more residual blocks in the blending backbone provides better feature maps. However, the GPU memory sets a limit, in particular, for larger patch sizes and upscaling factors.

Size of Feature Maps: Many existing super-resolution models that use LR convolutions postpone to upsampling to the very end of the network with deconvolutional layers. This makes it hard for the model to converge in training. For speed, we observed that pixel rearranging (i.e. depth-to-space conversion) and pixel tiling are computationally less complex than learning deconvolution filters. In our experiments, using there was no significant performance difference between using deconvolutional filters and pixel arranging. Furthermore, we employ 2× super-resolving after the first stage. This gradual upscaling is easier to learn than a single-shot 4× upscaling. We also tested different feature map depth (256 vs. 384) per layer; however, PSNR improvements were not substantial.

Temporal Window: In our current architecture, we tried different numbers of reference frames. We observed 0.10 dB PSNR improvement from $[t-1, t+1]$ to $[t-2, t+2]$ and 0.05 dB improvement from $[t-2, t+2]$ to $[t-3, t+3]$.

Model Variants: As a part of our ablation study, we tested different models. We resized the input to 2×, applied 32 residual blocks, and rearranged pixels 2× at the end for

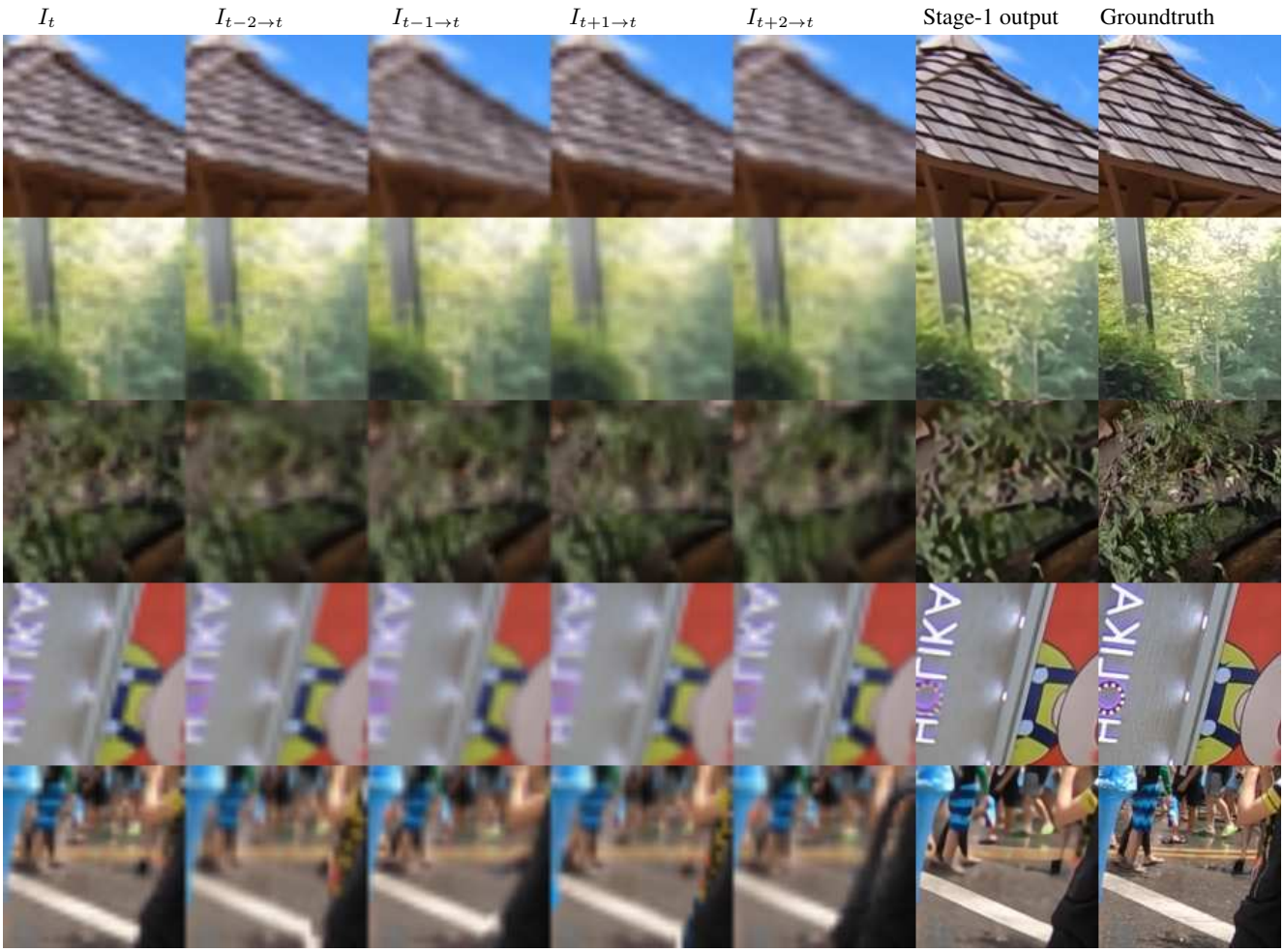

Figure 4. Input and output of the first stage. **Columns 1-5:** Sample 32×32 patches of the motion compensated input frames I_t, I_{t-2}, I_{t-1}, I_{t+1}, and I_{t+2}. **Column 6:** Estimated 128×128 HR patch after the first stage. **Last column:** Groundtruth HR patch.

a total of 4× super-resolution while keeping the number of references and other parameters same. This model has a comparable number of parameters to our proposed model, yet twice as slow (650 milliseconds vs. 1170 milliseconds). Since the tensors become large, the maximum number of images in training batches drops to 8 from 32, which also negatively affects the convergence behavior.

Run-time: The inference time computational performance depends on the number of residual blocks. Our model runs at 647 milliseconds (490, 570, 730) for 32 residual blocks (12, 22, 42). The optical flow computation takes 200 milliseconds per image. However, these numbers can be much smaller since neither our model nor the optical flow is optimized for the GPU.

5. Conclusions

We presented a fully convolutional, multi-stage neural network for 4× video super-resolution. To this end, we used multiple warped reference frames within a temporal window around the current frame as input to the first stage. In the second stage, we concatenated the LR versions of the estimated and rearranged HR outputs that are computed in the previous stage to the reference frames. Our network processes the input frames in low spatial resolution, thus it has low memory requirements.

For future work, we aim to investigate joint architectures that share part of the weights and combine them into a single model with different processing branches, which would allow training part of the network with image super-resolution datasets and then fine-tune with video super-resolution datasets.

References

[1] Jose Caballero, Christian Ledig, Andrew Aitken, Alejandro Acosta, Johannes Totz, Zehan Wang, and Wenzhe Shi. Real-time video super-resolution with spatio-temporal networks and motion compensation. In *Proceedings of the IEEE Con-*

Bicubic Single image SR **Ours** Groundtruth HR

Figure 5. Results show that our MultiBoot method can recover missing patterns and generate sharp images without significant artifacts. Single Image SR results are from the first stage of our network without any reference frames.

ference on Computer Vision and Pattern Recognition, pages 4778–4787, 2017. 1, 3, 4

[2] Chao Dong, Chen Change Loy, Kaiming He, and Xiaoou Tang. Image super-resolution using deep convolutional networks. *IEEE transactions on pattern analysis and machine intelligence*, 38(2):295–307, 2016. 3

[3] Chao Dong, Chen Change Loy, and Xiaoou Tang. Accelerating the super-resolution convolutional neural network. In *European conference on computer vision*, pages 391–407. Springer, 2016. 3

[4] Yan Huang, Wei Wang, and Liang Wang. Bidirectional recurrent convolutional networks for multi-frame super-resolution. In *Advances in Neural Information Processing Systems*, pages 235–243, 2015. 3

[5] Eddy Ilg, Nikolaus Mayer, Tonmoy Saikia, Margret Keuper, Alexey Dosovitskiy, and Thomas Brox. Flownet 2.0: Evolution of optical flow estimation with deep networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2462–2470, 2017. 4

[6] Daniel Glasner Shai Bagon Michal Irani. Super-resolution from a single image. In *Proceedings of the IEEE International Conference on Computer Vision, Kyoto, Japan*, pages 349–356, 2009. 1, 3

[7] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In *European conference on computer vision*, pages 694–711. Springer, 2016. 3

[8] Armin Kappeler, Seunghwan Yoo, Qiqin Dai, and Aggelos K Katsaggelos. Video super-resolution with convolutional neural networks. *IEEE Transactions on Computational Imaging*, 2(2):109–122, 2016. 3

[9] Jiwon Kim, Jung Kwon Lee, and Kyoung Mu Lee. Accurate image super-resolution using very deep convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1646–1654, 2016. 3

[10] Soo Ye Kim, Jeongyeon Lim, Taeyoung Na, and Munchurl Kim. 3dsrnet: Video super-resolution using 3d convolutional neural networks. *arXiv preprint arXiv:1812.09079*, 2018. 1, 4

[11] Bee Lim, Sanghyun Son, Heewon Kim, Seungjun Nah, and Kyoung Mu Lee. Enhanced deep residual networks for single image super-resolution. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 136–144, 2017. 2, 3, 5

[12] Ce Liu and Deqing Sun. A bayesian approach to adaptive video super resolution. In *CVPR 2011*, pages 209–216. IEEE, 2011. 3

[13] Ding Liu, Zhaowen Wang, Yuchen Fan, Xianming Liu, Zhangyang Wang, Shiyu Chang, and Thomas Huang. Robust video super-resolution with learned temporal dynamics. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2507–2515, 2017. 3

[14] Osama Makansi, Eddy Ilg, and Thomas Brox. End-to-end learning of video super-resolution with motion compensation. In *German conference on pattern recognition*, pages 203–214. Springer, 2017. 3

[15] Seungjun Nah, Sungyong Baik, Seokil Hong, Gyeongsik Moon, Sanghyun Son, Radu Timofte, and Kyoung Mu Lee. Ntire 2019 challenges on video deblurring and super-resolution: Dataset and study. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, June 2019. 1, 6

[16] Seungjun Nah, Radu Timofte, Shuhang Gu, Sungyong Baik, Seokil Hong, Gyeongsik Moon, Sanghyun Son, Kyoung Mu Lee, Xintao Wang, Kelvin C.K. Chan, Ke Yu, Chao Dong, Chen Change Loy, Yuchen Fan, Jiahui Yu, Ding Liu, Thomas S. Huang, Xiao Liu, Chao Li, Dongliang He, Yukang Ding, Shilei Wen, Fatih Porikli, Ratheesh Kalarot, et al. Ntire 2019 challenge on video super-resolution: Methods and results. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, June 2019. 6

[17] Sung Cheol Park, Min Kyu Park, and Moon Gi Kang. Super-resolution image reconstruction: a technical overview. *IEEE signal processing magazine*, 20(3):21–36, 2003. 1

[18] Eduardo Pérez-Pellitero, Mehdi Sajjadi, Michael Hirsch, and Bernhard Schölkopf. Photorealistic video super resolution. *arXiv preprint arXiv:1807.07930*, 2018. 2, 4

[19] Cisco Tech Report. VNI global fixed and mobile internet traffic forecasts. *www.cisco.com/c/en/us/solutions/service-provider/visual-networking-index-vni/*, 2017. 1

[20] Mehdi SM Sajjadi, Raviteja Vemulapalli, and Matthew Brown. Frame-recurrent video super-resolution. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6626–6634, 2018. 2, 4

[21] Wenzhe Shi, Jose Caballero, Ferenc Huszár, Johannes Totz, Andrew P Aitken, Rob Bishop, Daniel Rueckert, and Zehan Wang. Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1874–1883, 2016. 3

[22] Hiroyuki Takeda, Peyman Milanfar, Matan Protter, and Michael Elad. Super-resolution without explicit subpixel motion estimation. *IEEE Transactions on Image Processing*, 18(9):1958–1975, 2009. 3

[23] Xin Tao, Hongyun Gao, Renjie Liao, Jue Wang, and Jiaya Jia. Detail-revealing deep video super-resolution. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4472–4480, 2017. 3

[24] Radu Timofte, Vincent De Smet, and Luc Van Gool. Anchored neighborhood regression for fast example-based super-resolution. In *Proceedings of the IEEE international conference on computer vision*, pages 1920–1927, 2013. 3

[25] Radu Timofte, Vincent De Smet, and Luc Van Gool. A+: Adjusted anchored neighborhood regression for fast super-resolution. In *Asian conference on computer vision*, pages 111–126. Springer, 2014. 3

[26] Zhihao Wang, Jian Chen, and Steven CH Hoi. Deep learning for image super-resolution: A survey. *arXiv preprint arXiv:1902.06068*, 2019. 3

[27] Jianchao Yang, John Wright, Thomas S Huang, and Yi Ma. Image super-resolution via sparse representation. *IEEE transactions on image processing*, 19(11):2861–2873, 2010. 1, 3