Online Video Super-Resolution with Convolutional Kernel Bypass Grafts

Jun Xiao, Xinyang Jiang, Ningxin Zheng, Huan Yang, Yifan Yang, Yuqing Yang, Dongsheng Li, Kin-Man Lam

***Abstract*—Deep learning-based models have achieved remarkable performance in video super-resolution (VSR) in recent years, but most of these models are less applicable to online video applications. These methods solely consider the distortion quality and ignore crucial requirements for online applications, e.g., low latency and low model complexity. In this paper, we focus on online video transmission in which VSR algorithms are required to generate high-resolution video sequences frame by frame in real time. To address such challenges, we propose an extremely low-latency VSR algorithm based on a novel kernel knowledge transfer method, named the convolutional kernel bypass graft (CKBG). First, we design a lightweight network structure that does not require future frames as inputs and saves extra time for caching these frames. Then, our proposed CKBG method enhances this lightweight base model by bypassing the original network with "kernel grafts", which are extra convolutional kernels containing the prior knowledge of the external pretrained image SR models. During the testing phase, we further accelerate the grafted multibranch network by converting it into a simple single-path structure. The experimental results show that our proposed method can process online video sequences up to 110 FPS with very low model complexity and competitive SR performance.**

***Index Terms*—Video Super-resolution, deep lightweight model, video restoration**

I. Introduction

VIDEO super-resolution (VSR) is a fundamental task in computer vision that aims to generate high-resolution (HR) video sequences given the corresponding low-resolution (LR) counterparts. In general, VSR is a challenging problem because of its ill-posed nature, which means that an LR frame can be generated from an infinite number of possible HR frames. With the rapid development of video applications in the last decade, VSR has demonstrated substantial industrial value and thus attracted researchers' attention.

Online video applications (e.g., cloud gaming, live broadcasting, and online video conferences) have become increasingly popular, especially during COVID-19. Although existing deep learning-based models have achieved unprecedented success in VSR tasks, most of them are less applicable in online situations because developers solely pursue performance improvement and are seldomly concerned with latency and model complexity [1]–[3]. Therefore, super-resolving online video sequences is still a challenging and important problem. In this paper, we focus on the online VSR setting, where users need to receive super-resolved video sequences frame-by-frame in real time.

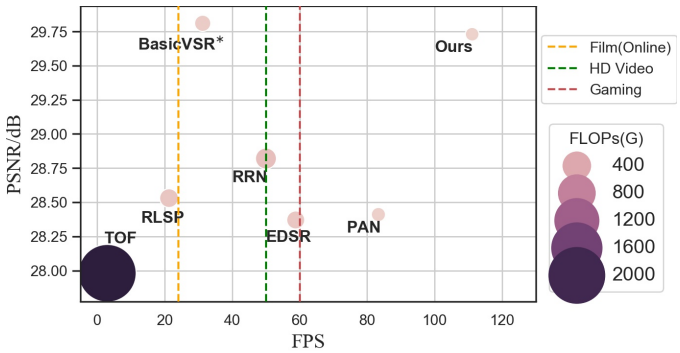


Fig. 1. Average PSNR, FPS and FLOPs (G) of different methods evaluated on the REDS4 dataset. All methods are deployed in a Tesla V-100 GPU, and the size of each input image is 180×320. The dotted lines indicate the recommended FPS of different online applications, e.g., for "Gaming", the recommended FPS for cloud games is 60 FPS. "BasicVSR*" is the modified version of BasicVSR and meets online VSR requirements.

In contrast to offline VSR, where there is no restriction on model complexity and latency, online VSR poses two key challenges to existing VSR methods. The first challenge is that online applications have extremely strict requirements in terms of *low latency* and *buffering lag* because they involve real-time user interaction. In some online applications (e.g., online video conferences), slight latency will significantly affect the user's experience and the conference quality, which is undesirable. However, most state-of-the-art VSR methods need to cache future frames to super-resolve the current frame, which will unavoidably introduce large latency. For example, TOF [4] and RLSP [5] use optical flows to align future frames and propagate the information contained in those future frames to the current frame for feature aggregation. Second, to reduce the transmission bandwidth, online VSR methods are usually deployed on client devices rather than cloud servers, which require the models to have low model complexity and real-time speed. However, the majority of deep VSR models adopt complicated network modules with high computational complexity in exchange for better performance, such as progressive fusion blocks [6]–[8] and nonlocal attention blocks [9], [10]. The client devices (e.g., personal computers, mobile phones, etc.) are usually resource-constrained and cannot support models with such high computational complexity. It should be noted that the distortion quality of VSR methods is highly related to

Jun Xiao and Kin-Man Lam are with the Department of Electronic and Information Engineering, the Hong Kong Polytechnic University. E-mail: jun.xiao@connect.polyu.hk.

Xinyang Jiang, Ningxin Zheng, Huan Yang, Yifan Yang, Yuqing Yang, and Dongsheng Li are with Microsoft Research Asia. E-mail: xinyangj@hotmail.com

Most work in this paper was finished when Jun Xiao interned at Microsoft Research Asia.

their model complexity. Reducing the model complexity will inevitably deteriorate the distortion quality of the generated images. Therefore, it is essential to achieve a good trade-off among the distortion quality of the generated videos, the processing latency, and the model complexity for online VSR algorithms.

In this paper, we propose a low-latency online VSR solution based on a novel knowledge transfer-based method called the convolutional kernel bypass graft (CKBG). To maximize the performance with no cached future frames and limited model capacity/latency, we propose leveraging the prior information from large pretrained image SR models. Borrowing the concept of a heart bypass graft, CKBG enhances a VSR base model by bypassing the original network with "kernel grafts", which are extra convolutional kernels containing the prior knowledge of external pretrained image SR models. Specifically, given the number of kernels extracted from large pretrained image SR models, CKBG learns a set of kernel bases with the K-means algorithm in the Wasserstein space, and the "kernel grafts" are obtained by learning a linear representation under the space spanned by the learned bases. The grafted multibranch structure of our network can be converted into a single-path structure with kernel reparameterization [11], [12], which can effectively reduce the model complexity and latency.

The main contributions of this paper are as follows:

- In this paper, we mainly focus on online video super-resolution, where no future frames are accessible, and propose an extremely low-latency and effective online VSR method.
- To further improve the performance, we propose the CKBG scheme, which incorporates the prior information learned from large pretrained image SR models into a VSR base model.
- The experimental results show that our proposed method can process video sequences with up to 110 FPS and achieve promising performance compared with other state-of-the-art VSR methods, as summarized in Fig. 1.

II. Related Works

A. Deep Lightweight Image SR Methods

In recent years, increasing efforts have been invested into exploring deep lightweight models for image SR because most promising deep SR methods [13], [14] require high computational complexity, which significantly limits their applications in resource-constrained devices. The common methods [15]–[18] for reducing model parameters adopt recurrent structures, which share the parameters and enhance the features with multiple cycles for image reconstruction. However, the recursive strategy increases the processing time, and the performance gain is limited. Considering the effectiveness of the group convolution in [19], Ahn *et al.* [20] proposed a cascaded convolutional network for image super-resolution. They combined the modified residual block with group convolution, significantly reducing the number of model parameters. The methods in [21], [22] extend the group convolution and propose an information-distillation block that splits the input feature into several groups for further processing and then concatenates the output feature of each group for feature fusion. These methods have shown remarkable trade-off performance in reducing model complexity and maintaining distortion quality. With the split-and-concatenate strategy, Zhao *et al.* [23] proposed the pixel-attention mechanism to enhance useful information at each pixel location. Considering the characteristics of local regions, Xiao *et al.* [24] proposed an efficient method for generating dynamic convolution kernels that can adaptively extract local features for image super-resolution. Unlike the above methods, which design efficient modules by hand, the methods in [25], [26] utilize neural network architecture search techniques to automatically find efficient model designs. Inspired by the structure reparameterization techniques [11], [12], Zhang *et al.* [27] proposed the edge convolutional block to accelerate the processing speed of deep models. Even though these deep lightweight image SR models have demonstrated their effectiveness, in terms of performance and real-time speed , these methods do not consider the long-range temporal dependency when applied to video sequences.

B. Deep Video Super-resolution Methods

With the rapid development of video applications, deep learning-based video processing algorithms [28]–[30] have become increasingly popular, especially for deep VSR methods [4]–[6], [8], [31]–[40]. Unlike image SR, VSR needs to consider object motions and the temporal correspondence between successive frames. The method in [8] adopted the efficient spatial transformer for motion compensation and then combined it with ESPCN [32] to synthesize HR video sequences. As object motions have the property of being spatially variant, DUF [33] used a dual-path residual dense network to predict the residue between the ground-truth frames and the input LR frames in one path and dynamically upsample the input LR frames in another path. In [4], it was found that incorporating an optical flow estimation network into a task-specific network for joint training, named TOF, is beneficial to the overall performance. However, its estimated motion field is different from the ground-truth optical flow, and the accuracy of the estimated optical flow is very sensitive to local illumination changes. Instead of using optical flow for the alignment of video frames, EDVR [6] adopted deformable convolution [34] to align the features from multiscale levels. According to [35], deformable convolution cannot effectively capture long-range dependencies and suffers from unstable training. In practice, the runtime of deformable convolution is very slow and cannot satisfy the real-time requirements. To achieve better efficiency, Fuoli D *et al.* [5] proposed an efficient recurrent network to investigate the information from adjacent frames only for practical VSR tasks. This method directly inputs the extracted features of the hidden state from the previous step into the current step for feature fusion so that information can temporally propagate along the video sequences. Since VSR is an ill-posed problem, RSDN [36] introduced structural and detailed information to regularize the process for generating HR frames. Recently, bidirectional recurrent methods [37]–[41], such as BasicVSR and its variant

[37], [40], have shown their effectiveness in VSR, because these models can fully exploit the information from the forward and backward directions of the input video sequences. However, these methods need to acquire whole video sequences beforehand, and therefore, they are impractical in online scenarios. A recent work [42] proposed a unidirectional method for video denoising that mimics backward behaviors with the look-ahead mechanism, leading to promising results. However, this method requires caching future frames for restoration, so it inevitably introduces undesirable latency in online situations. It should be noted that video sequences are transmitted in streaming format in online applications. It is impractical to capture the information contained in future frames unless latency is introduced. In this paper, we focus on online VSR scenarios in which future frames are inaccessible and the deployed devices are resource-constrained, such as for low-configuration devices. Therefore, the latency and model complexity of online VSR algorithms are significantly important in online situations.

C. Knowledge Transfer for Image and Video Super-resolution

Knowledge distillation is a well-known technique that transfers knowledge from a large deep model to a smaller one. In knowledge distillation, the large model is called the teacher network, while the smaller model is called the student network. Recently, Gao *et al.* [43] proposed distilling knowledge from a teacher image super-resolution model by minimizing the distance between the statistical properties (i.e., maximum values, mean values, etc.) of the teacher and student networks' feature maps . He *et al.* [44] proposed a feature-affinity distillation (FAKD) method for image super-resolution that transfers knowledge by using the correlation matrices of feature maps, and Lee *et al.* [45] proposed leveraging the privileged information from ground-truth images and distilling knowledge by minimizing the distance between the features of the teacher network and those of the student network. Xiao *et al.* [46] proposed an effective knowledge-distillation method for video super-resolution that forces the spatial and temporal characteristics of the teacher network and student network to be consistent. All these distillation-based super-resolution models require that the network topology of the teacher network and the student network are consistent. In contrast, our proposed CKBG does not restrict the network structures, which provides more flexibility than distillation-based methods. In addition, the large teacher models used in our CKBG-based model are not involved in the training process, which is significantly different from the distillation-based methods.

III. The Proposed Method

The overall structure of our proposed method is illustrated in Fig. 2. Our method adopts the recurrent structure, which only utilizes the information in the current and previous frames. Therefore, it does not need to cache future frames during the super-resolution process and saves the time cost. In particular, the proposed network first uses optical flow to align the features extracted from the previous frame and then temporally

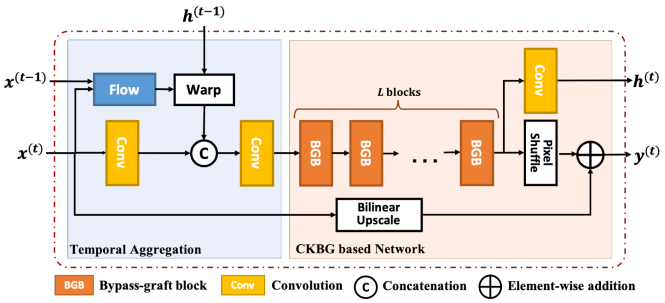


Fig. 2. Overall pipeline of our proposed method. The "Flow" represents the flow estimation function. In our proposed method, we employ SPyNet to compute the optical flow between the current and previous frames. The "Warp" represents the warping function, which aligns the features extracted from the previous frame according to the computed flow.

fuses the aligned features with the features extracted from the current frame. Then, the temporally aggregated features are forwarded to the cascaded bypass-graft blocks (BGBs) for feature extraction, where they are constructed based on our proposed CKBG method. At the output of our model, we adopt the PixelShuffle operator to upsample the extracted features and combine them with the bilinearly upscaled LR inputs to generate the final HR frames.

A. Temporal Aggregation

For temporal aggregation, we first utilize SPyNet [47], denoted as $S(\cdot)$, to estimate the optical flow $f^{(t-1)}$ between the current LR frame $x^{(t)}$ and the previous frame $x^{(t-1)}$, which is computed as follows:

$$f^{(t-1)} = S(x^{(t)}, x^{(t-1)}). \tag{1}$$

Then, we use the estimated optical flow to perform alignment in the feature space. The warped feature of the previous frame is computed as follows:

$$\hat{h}^{(t-1)} = \text{Warp}(h^{(t-1)}, f^{(t-1)}), \tag{2}$$

where $h^{(t-1)}$ represents the feature extracted from the previous frame, $\hat{h}^{(t-1)}$ is the corresponding warped feature, and $\text{Warp}(\cdot)$ represents the warp operator. To avoid significantly increasing the model complexity, we simply concatenate the feature extracted from the current frame with the aligned feature along the channel dimension. Then, a convolutional layer is used to aggregate the features temporally. The aggregated feature $F^{(t)}$ is calculated as follows:

$$F^{(t)} = \text{conv}(\text{Cat}(\text{conv}(x^{(t)}), \hat{h}^{(t-1)})), \tag{3}$$

where $\text{Cat}(\cdot)$ is the concatenation operator, and $\text{conv}(\cdot)$ represents a 3×3 convolutional operator.

B. Convolutional Kernel Bypass Graft

1) Push-Forward Mapping: Assume μ and ν are two probability measures. Given a mapping function $T : X \to Y$, if for any measurable set $B \subset Y$, we have the following:

$$\mu(T^{-1}(B)) = \nu(B), \quad T^{-1}(B) \subset X \tag{4}$$

Then, ν is said to be the push-forward of μ by T, and we have $\nu = T_{\#}\mu$. It is worth noting that push-forward mapping is a measure-preserving mapping.

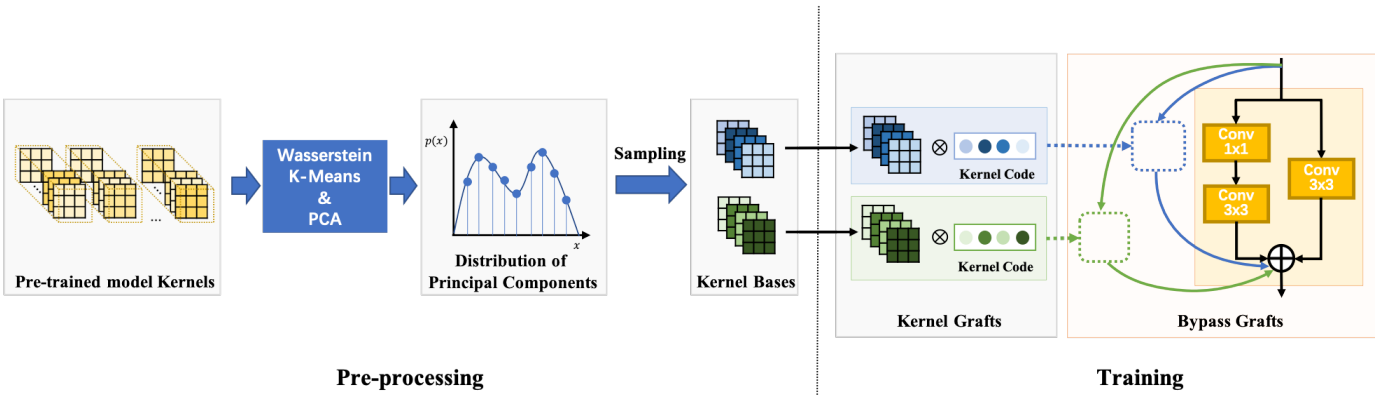


Fig. 3. Overall pipeline of the kernel bypass graft in the preprocessing and training stages. During the preprocessing stage, the external kernels are extracted from pretrained image SR models. Then, the kernel bases are learned by utilizing the K-means algorithm in the Wasserstein space and principal component analysis (PCA). During the training phase, the kernel grafts are bypassed to a lightweight video super-resolution model, forming a multibranch structure. The model parameters, except the grafted kernels, are updated by using the gradient-based optimization method, e.g., SGD. Note: all the parallel convolutional kernels are merged into a single convolutional kernel in the testing phase.

2) Wasserstein Distance: Suppose μ and ν are probability measures defined on the Polish spaces $\mathcal{X}$ and $\mathcal{Y}$, respectively. Let $\prod(\mu,\nu)$ denote the set of joint distributions of μ and ν. For each element $\pi \in \prod(\mu,\nu)$, we have the following: $\int_{\mathcal{X}} \mathrm{d}\pi(x,y) = \mathrm{d}\nu(y)$ and $\int_{\mathcal{Y}} \mathrm{d}\pi(x,y) = \mathrm{d}\mu(x)$, where $x \in \mathcal{X}$ and $y \in \mathcal{Y}$. The optimal transportation problem aims to find an optimal transport mapping with the minimum transportation cost between two locations x and y, which is defined as follows:

$$\min_{\pi \in \prod(\mu,\nu)} \int_{\mathcal{X}\times\mathcal{Y}} c(x,y)\,\mathrm{d}\pi(x,y), \tag{5}$$

where $c(x,y) : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}^+ \cup \{0\}$ denotes the cost function from x to y. This is the well-known Kantorovich's formulation. In this case, π is also called transport mapping, and $\mathrm{d}\pi(x,y)$ specifies the transported mass between x and y. Based on this, the Wasserstein distance [48] is defined as follows:

$$\mathcal{W}_p^p(x,y) = \left(\min_{\pi \in \prod(\mu,\nu)} \int_{\mathcal{X}\times\mathcal{Y}} |x-y|^p \,\mathrm{d}\pi(x,y) \right)^{1/p}, \tag{6}$$

where $c(x,y) = |x-y|^p$. If $p=2$, it is called the 2-Wasserstein distance, which is denoted by $\mathcal{W}_2^2$.

Theorem 1: (Bernier's Theorem). Let μ and ν be two probability measures on $\mathbb{R}^n$, where μ is continuous and does not give a probability mass to negligible sets. Then, there is exactly one measurable map T such that $\nu = T_{\#}\mu$ and $T = \nabla\phi$ for some convex function ϕ in that any two such maps coincide with $d\mu$ almost everywhere.

The proof of Bernier's theorem can be found in [48]. This theorem reveals that the existence of the solution to the 2-Wasserstein problem is the gradient of a convex function ϕ. In addition, this theorem guarantees that the obtained optimal transport mapping function has the measure-preserving property. We consider the 2-Wasserstein metric space in this paper because it has good properties for computation.

3) Wasserstein Barycenter: Assuming that there are N_1 probability distributions $\{\mu_i\}_{i=1}^{N_1}$ defined in the 2-Wasserstein metric space, the Fréchet mean of these distributions is defined as follows:

$$\nu = \arg\min_{\nu \in \mathcal{P}(\mathcal{Y})} \sum_{i=1}^{N_1} \lambda_i \mathcal{W}_2^2(\mu_i,\nu), \tag{7}$$

where λ_i is the weight associated with the i-th probability distribution, and $\sum_{i=1}^{N_1}\lambda_i = 1$. The solution ν is also called the Wasserstein barycenter, which achieves the minimum weighted 2-Wasserstein distance for every μ_i. It is worth noting that the Wasserstein distance is based on the optimal transportation problem, as shown in Eqn. (6). Therefore, the calculation of the Wasserstein barycenter requires solving the optimal transportation problem and obtaining the optimal transport mapping function. Bernier's theorem shows that the solution to the optimal transportation problem exists, which is the gradient of a convex function. In addition, this theorem reveals that the optimal transport mapping is a push-forward mapping and has the measure-preserving property. As a result, the Wasserstein barycenter, based on the optimal transportation, has similar geometric properties to that of the sample distributions.

4) Kernel Prior Learning and Grafting: As shown in Fig. 3, the proposed CKBG is based on a two-stage learning framework. During the preprocessing stage, CKBG extracts convolutional kernels from the pretrained image SR models, which are called external kernels. Then, it learns a set of kernel bases with the K-means algorithm in the Wasserstein space and principal component analysis (PCA). During the training stage, the "kernel grafts" are obtained by learning linear representations under the space spanned by the kernel bases, and the corresponding coefficients are called kernel codes. The learned "kernel grafts" are bypassed to a branch in the base VSR network, forming a bypass-graft block (BGB). It should be noted that the obtained BGB has a multibranch structure in the training stage, but this structure can be further converted into a single-path structure for acceleration during the testing phase. In this section, we elaborate on the learning, grafting, and testing of the proposed CKBG.

Kernel Prior Learning. A previous study [49] shows that a small subset of the convolutional filters in a deep SISR net-

work can significantly contribute to the function of a specific degradation removal when the LR images contain multiple degradations. This phenomenon means that the convolutional kernels of the pretrained image SR model have prior knowledge for reconstruction. In this paper, we propose a method for extracting the prior knowledge of convolutional kernels from pretrained image SR models, which is then transferred to a base VSR model for performance improvement. Specifically, we focus on a single-degradation setting: LR videos that are generated by bicubic downsampling kernels without considering additional blurring and noise. The pretrained image SR model used in this paper is EDSR [13], which is a classic image SR model that has achieved promising results. EDSR is trained on the synthesized dataset, where the LR images are corrupted by the bicubic downsampling operator. In our method, all EDSR convolutional kernels are extracted because they contain considerable prior information for restoring this degradation. However, the number of extracted kernels is more than two thousand, and these kernels might have large redundancy.

To remove redundancy and obtain more representative kernels, we cluster similar kernels by performing the K-means algorithm in the Wasserstein space, resulting in a set of cluster centroids. Assuming that K convolutional kernels are obtained from a pretrained image SR model, denoted as $\{\mathbf{k}_i\}_{i=1}^{K}$, the cluster centroids are obtained by solving the following optimization problem:

$$\arg\min_{\mathbf{S}} \sum_{i=1}^{M} \sum_{\mathbf{k}\in S_i} \mathcal{W}_2^2(\mathbf{k}, \mathbf{c}_i), \tag{8}$$

where $\mathbf{S} = \{S_1, \cdots, S_M\}$ denotes the partition of the extracted kernels, and M is the number of clusters. $\mathbf{c}_i$ represents the i-th cluster centroid in the Wasserstein space. From Eqn. (7), we can find that the obtained kernel centroid of each kernel cluster is equivalent to the Wasserstein barycenter of the associated cluster. In this case, the weights associated with the Wasserstein distance between the kernels and the corresponding centroids are all the same. We set the number of cluster centroids to 256 in this paper and use the variational method provided in [50] to solve the above optimization problem. It is worth noting that we perform K-means clustering in the Wasserstein space rather than in the Euclidean space because the optimal transport mapping has the measure-preserving property and can better preserve the geometric properties (e.g., shape) of the extracted kernels. As a result, the learned cluster centroids in the Wasserstein space have a similar geometric structure to the extracted kernels, which makes them better able to inherit the prior information from the extracted kernels. Fig. 4(a) illustrates a 1-D example of six unimodal distributions. As shown in Fig. 4(b), we find that the centroid distribution computed in the Euclidean space is severely distorted. In contrast, the geometric properties of the cluster centroids obtained in the Wasserstein space are more similar to the original distributions. Since the output responses are highly related to the geometric properties of the kernels, the kernel centroids obtained in the Wasserstein space can avoid generating distortion, leading to similar output responses. As shown in Fig. 5, the barycenter filter "BM" is the Wasserstein barycenter of two given kernels K_1 and K_2, and it inherits the shape of these two given kernels K_1 and K_2. Therefore, given an input signal, the output response of the barycenter filter is more similar to that of the original convolutional kernels K_1 and K_2. In contrast, "AM" represents the aggregated filter obtained in the Euclidean space, which cannot preserve the shape of the two given kernels K_1 and K_2, so the output response of the "AM" filter is distorted, which means that this filter cannot preserve the prior information of the given convolutional kernels.

The proposed CKBG enhances the base VSR model by learning a set of new kernels and grafting the kernels in the original network. We called the learned kernels "kernel grafts". In this paper, the "kernel grafts" are learned based on a kernel space constructed from the cluster centroids, which contain prior knowledge extracted from the pretrained image SR model. Specifically, this kernel space is represented by the kernel bases obtained by performing PCA on the cluster centroids as follows:

$$CC^T = U\Sigma U^T, \tag{9}$$

where $C = [c_1, \cdots, c_M]$ is the cluster centroid matrix, C^T is its transpose matrix, $U = [\mathbf{u}_1, \cdots, \mathbf{u}_M]$ is the eigenvector matrix of C, and $\Sigma = \text{Diag}(\sigma_1, \cdots, \sigma_M)$ contains the corresponding eigenvalues sorted in descending order. The eigenvectors in U are the principal components of the cluster centroids, which can be used as the potential bases of the kernel prior subspace. Larger singular values indicate that the corresponding eigenvectors are more significant.

Kernel Graft. As shown in Fig. 3, several kernel grafts are generated to bypass the original kernels of a convolutional layer in the base network. For each kernel graft, we first select a set of bases to form the kernel space, and the selected bases are the principal components of the cluster centroids sampled from the following categorical distribution:

$$\mu_i \sim p(\mathbf{\Theta}), \tag{10}$$

where $\mathbf{\Theta} = [\theta_1, \cdots, \theta_M]$, $\theta_i = \sigma_i / \sum_{j=1}^{M} \sigma_j, \forall i = 1, \cdots, M$.

This equation implies that the bases are sampled according to their significance, which corresponds to their singular values. In other words, those leading principal components are more likely to be selected as the bases for the kernel space. Meanwhile, the sampling process introduces the randomness and increases diversity of kernel grafts.

Finally, the convolutional kernel graft is obtained by learning a linear combination of the sampled kernel bases. Fig. 6 illustrates the network structure of the proposed BGB during the training and testing stages. In our implementation, learning a linear combination of the kernel bases (i.e., 3×3 convolution) is equivalent to adding a 1×1 convolution (i.e., the patterned boxes) before the corresponding kernel bases. During the training stage, the parameters of the 1×1 convolutions in the bypassed branches are updated simultaneously with the parameters of the base network, while the kernel bases are kept fixed.

BGB in Testing. During the testing phase, the multibranch structure of the BGB can be converted into a single convo-

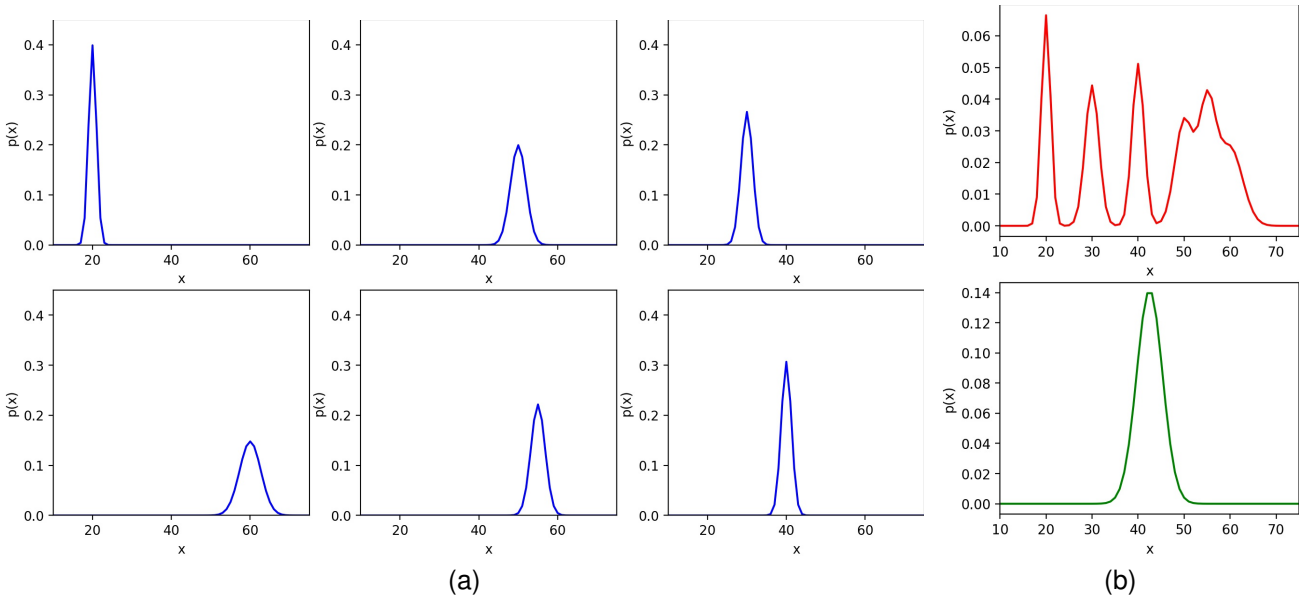


Fig. 4. (a). Samples of different unimodal distributions. (b). The centroids are computed in the Euclidean space (red) and the Wasserstein space (green). The horizontal axis denotes the support of the distributions, and the vertical axis is the value of the distributions.

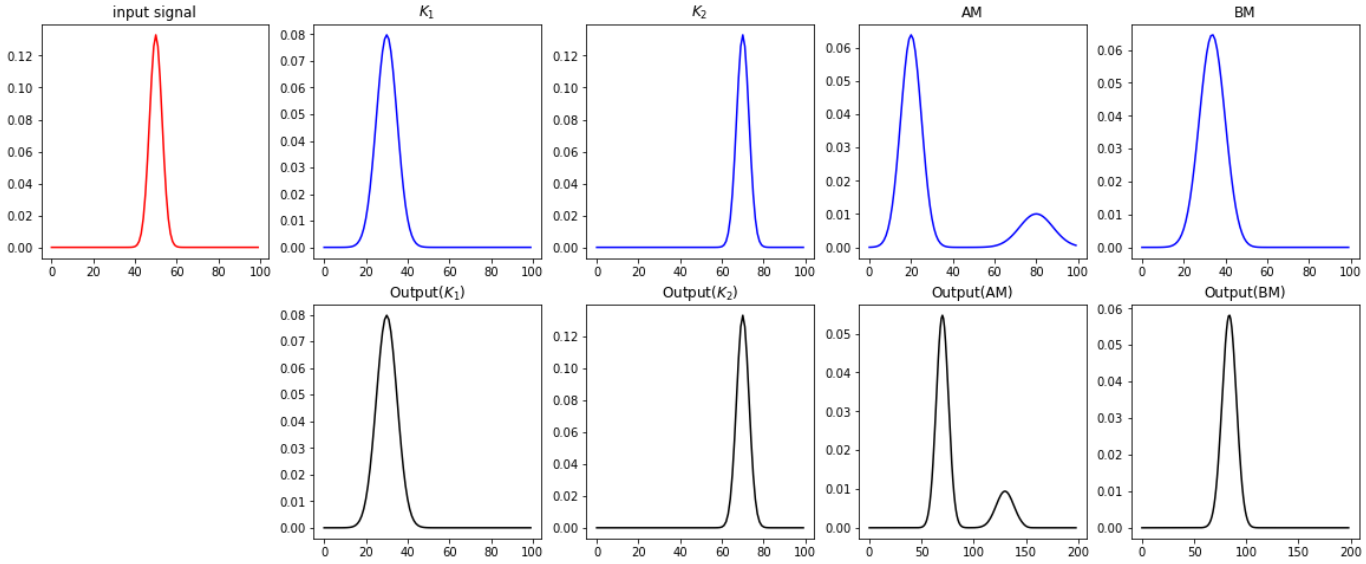


Fig. 5. Illustration of a 1-D input signal passing through different convolutional filters. The red curve represents the input signal. The blue curves represent four convolutional filters. The "AM" is the mean filter of the convolutional filters K_1 and K_2 in the Euclidean space, and the "BM" is the barycenter filter of the convolutional filters K_1 and K_2 in the Wasserstein space. The black curves are the corresponding output responses.

lution according to the linear property of the convolutional operation. This conversion is called kernel reparameterization [11], and it involves two types of kernel reparameterizations: sequential convolutions and parallel convolutions. Specifically, the consecutive 1×1 and 3×3 convolutional operators are first merged to form a single convolution, leading to a parallel structure. Then, all 3×3 convolutional kernels in the parallel branches are merged into a single convolutional operator, resulting in a highly efficient single-path structure [12]. It is worth noting that the re-parameterized kernels at the testing stage are equivalent to the original multibranch structure in the training stage. Therefore, the the processing speed can be significantly accelerated without sacrificing any performance. More details about the kernel reparameterization in our proposed method can be found in Appendix A.

C. Loss Function

We adopt the Charbonnier loss function to measure the distance between the generated frames and the ground-truth (GT) frames; it is defined as follows:

$$L = \frac{1}{T}\sum_{t=1}^{T}\sqrt{\left\| I_{SR}^{(t)} - I_{GT}^{(t)} \right\|_2^2 + \epsilon}, \tag{11}$$

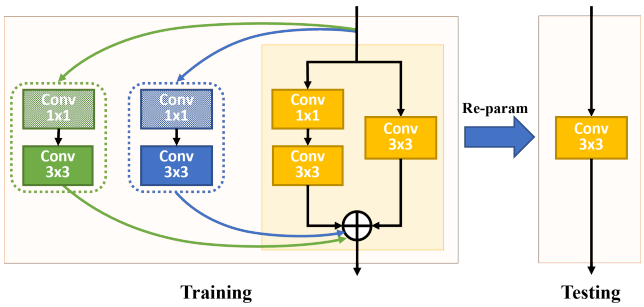


Fig. 6. Overall structure of the proposed bypass-graft block (BGB) in the training stage and testing stage. The convolutional layers marked by orange are in the base VSR model. The green and blue arrows denote the grafted branches. "Re-param" denotes kernel reparameterization, which converts the multibranch structure into a single-path structure. Note: the parameters of the grafted kernels are not updated during training.

where $I_{SR}^{(t)}$ and $I_{GT}^{(t)}$ denote the generated SR frame and the ground-truth frame at the t-th time step, respectively, T is the number of input LR frames, and ϵ is a hyperparameter.

IV. Experiments and Analysis

Datasets. The REDS dataset [51] and Vimeo-90K dataset [4], which are two widely used datasets in VSR, are adopted for training. During the testing stage, the REDS4 dataset [40] and Vid4 dataset [52] are used for evaluation. In these two datasets, the upscaling factor is 4, and the bicubic downsampling kernel is used to generate LR video sequences.

Evaluation Metrics. The performance of online VSR algorithms should include three perspectives: model complexity, latency, and the distortion quality of the generated videos. We measure model complexity by considering the number of model parameters, floating point operations per second (FLOPs), and the number of activations. For distortion quality assessment, the peak signal-to-noise ratio (PSNR) and structural similarity index measure (SSIM) are adopted for evaluation. Since our proposed method focuses on online scenarios, the latency t is very important and is defined as follows:

$$t = t_{\text{cache}} + t_{\text{run}}, \tag{12}$$

where t_{cache} denotes the time needed to cache the frames required by the VSR methods, and t_{run} denotes the runtime required to super-resolve an input frame. If the frame rate is 24 FPS, then the cache time for one frame is 40 ms. It is worth noting that distortion and latency are well-known trade-offs. The trade-off score function [53] is used to objectively measure the performance of the algorithms in online situations and is defined as follows:

$$\text{score} = \frac{2^{2\times\text{PSNR}}}{C \times t}, \tag{13}$$

where C is a constant hyperparameter set to $2^{50.0}$ in the experiment, so the trade-off scores of the compared methods are normalized to the same scale for better comparison. A model with a higher score can achieve a better trade-off between distortion quality and latency, so the model is more effective and applicable in online scenarios.

Implementation Details. In the experiments, we trained our proposed model using patches of size 80×80 randomly cropped from the input video sequences. For data augmentation, we randomly flipped and rotated the input video sequences. The number of channels in our model is set to 64, and the batch size is 8 in the training process. We use Adam [54] with $\beta_1 = 0.9$ and $\beta_2 = 0.999$ to update the weights of the model. The initial learning rate is 2×10^{-4}, and the cosine annealing strategy is utilized to adaptively adjust the learning rate during the training process. The total number of iterations used in training is 6×10^5. We implemented the proposed method with PyTorch in Tesla V-100 GPUs, and it took approximately eight days to complete the training.

A. Experiment on the REDS4 and Vid4 Datasets

In this experiment, we compared our proposed method with EDSR-M [13], PAN [23], RLSP [5], RRN [55], TOF [4], EDVR [6], BasicVSR [40] and BasicVSR++ [37]. Among these methods, EDSR-M and PAN are single-image SR models with lower model complexity. Since online video SR algorithms have a high requirement in terms of latency, these two methods are appropriate since they do not need to cache frames and have the ability to process videos in real time . RLSP and RRN are recurrent-based video SR methods that have achieved promising results. We adopted RLSP-7-128 in the experiments; it has seven convolutional blocks, and the number of channels in the intermediate layers is 128 because the model complexity of RLSP-7-128 is close to our proposed CKBG model. RRN only incorporates the information from the previous frame into the current state for generating HR frames, so this method is also suitable for online scenarios. Both methods were originally trained on videos blurred by Gaussian kernels, but the LR videos are generated by only the bicubic downsampling operator in our experiments. Therefore, we need to retrain these two models using publicly available codes following the default settings for a fair comparison. The original BasicVSR and BasicVSR++ are bidirectional VSR models, so they require whole video sequences for restoration. Therefore, these two methods are not suitable in online scenarios. We modified BasicVSR and BasicVSR++ by removing the backward part and reducing its model size for online application, and they are denoted as BasicVSR* and BasicVSR++*, respectively. The implementation details of BasicVSR* and BasicVSR++* can be found in Appendix B. We used the open-source codes provided by the authors to implement the other compared methods. All of the evaluation results, including the number of model parameters, the number of activations, FLOPs, the average PSNR and SSIM, the required runtime , and the trade-off score, are reported in Table I. The best results of the online methods are highlighted in bold, and the second-best results of the online methods are underlined.

It can be observed that EDVR can achieve promising performance in terms of the PSNR and SSIM on the REDS4 and the Vid4 datasets, but this model has high computational complexity and requires caching future frames for restoration. In addition, the runtime of this model is as high as 378 ms, which

TABLE I

COMPARISON OF THE AVERAGE PSNR, SSIM, MODEL COMPLEXITY, LATENCY, AND TRADE-OFF SCORE OF DIFFERENT VSR METHODS. NOTE THAT " # PARAM. " AND " # ACT. " REPRESENT THE NUMBER OF MODEL PARAMETERS AND THE NUMBER OF ACTIVATIONS, RESPECTIVELY. " ONLINE " REFERS TO WHETHER A METHOD CAN BE APPLIED ONLINE. " TIME " REPRESENTS THE LATENCY MEASURED ON THE REDS4 DATASET IN A TESLA-V100 GPU. THE AVERAGE SIZE OF THE INPUT FRAME IS 180×320. " RGB " AND " Y " MEAN THAT THE EVALUATION METRICS ARE MEASURED IN THE RGB SPACE AND Y CHANNEL, RESPECTIVELY. THE BEST RESULTS OF THE ONLINE METHODS ARE HIGHLIGHTED IN BOLD. THE SECOND-BEST RESULTS OF ONLINE METHODS ARE UNDERLINED.

Methods		# Param.	FLOPs	# Act.	Online	Time	REDS4 (RGB)			Vid4 (Y)		
							PSNR	SSIM	Score	PSNR	SSIM	Score
	Bicubic	-	-	-	-	-	26.13	0.7388	-	23.78	0.6347	-
SISR	EDSR-M [13]	1,571 K	114.28 G	201.83 M	✓	17 ms	28.37	0.8078	6.28	25.31	0.6608	0.19
	PAN [23]	**272 K**	28.29 G	237.88 M	✓	12 ms	28.41	0.8089	9.41	25.35	0.7306	0.14
VSR	TOF [4]	1,405 K	2,175.25 G	1,251.93 M	✗	334 ms	27.98	0.7990	0.18	25.89	0.7651	0.01
	RLSP [5]	1,154 K	132.94 G	108.74 M	✗	47 ms	28.53	0.8136	2.83	25.69	0.7530	0.06
	EDVR [6]	20,633 K	463.56 G	743.36 M	✗	378 ms	31.08	0.8800	12.10	27.29	0.8246	0.10
	RRN [55]	3,364 K	193.62 G	164.96 M	✓	20 ms	28.82	0.8234	9.97	25.85	0.7660	0.40
	BasicVSR++ [37]	7,320 K	940.43 G	515.62 M	✗	77 ms	32.29	0.9069	318.10	27.79	0.9500	0.85
	BasicVSR++* [37]	2,489 K	632.89 G	369.09 M	✗	41 ms	30.02	0.8594	25.67	26.83	0.8154	0.50
	BasicVSR* [40]	1,887 K	71.33 G	185.24 M	✓	32 ms	**29.81**	**0.8535**	24.59	**26.47**	**0.7986**	0.40
	CKBG (ours)	1,750 K	**17.85 G**	**34.09 M**	✓	**9 ms**	29.73	0.8514	**78.25**	26.34	0.7857	**1.28**

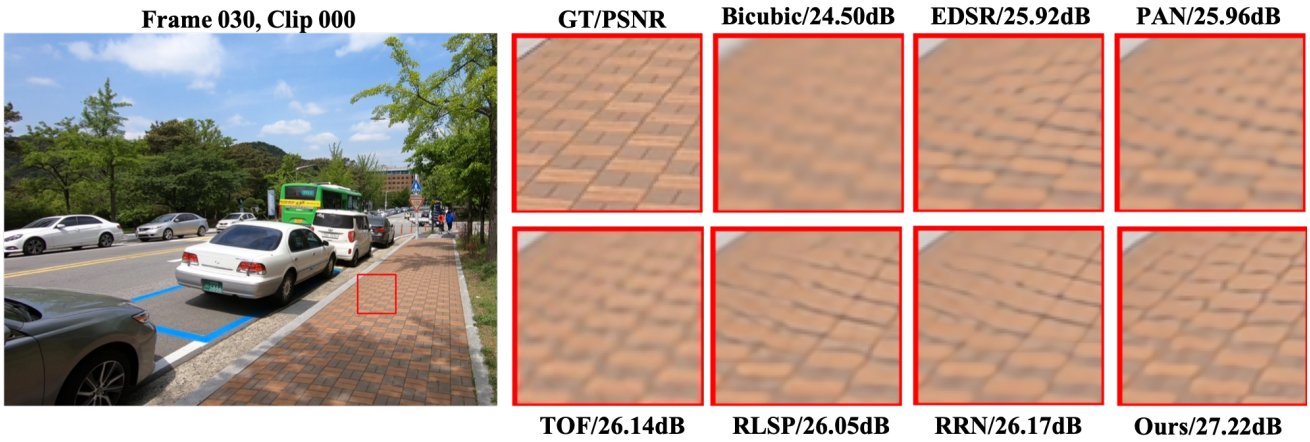


Fig. 7. The illustrated image is selected from the REDS4 dataset. The region marked by the red box is generated by different VSR methods for visual comparison.

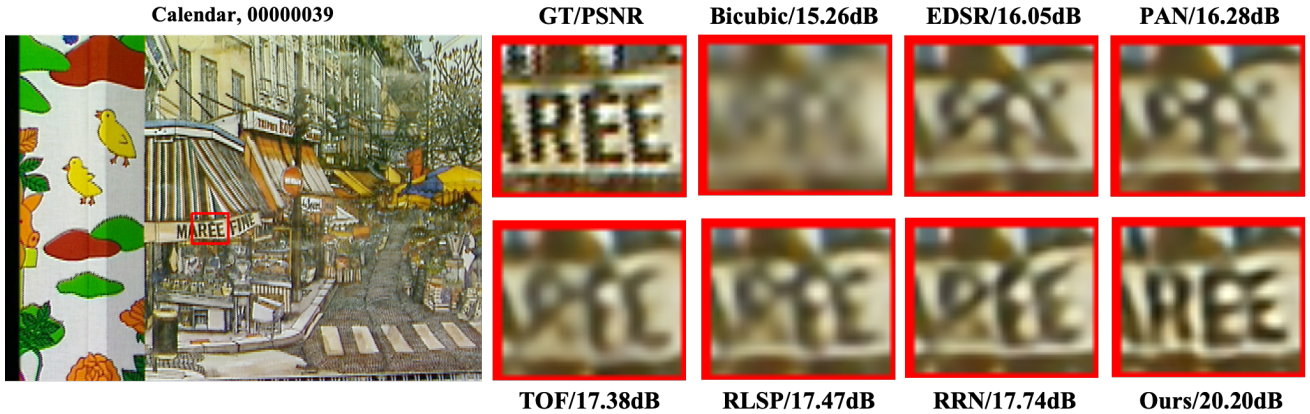


Fig. 8. The illustrated image is selected from the Vid4 dataset. The region marked by the red box is generated by different VSR methods for visual comparison.

does not satisfy the real-time requirements. Therefore, EDVR is not suitable for online applications. BasicVSR++ achieves the best performance in terms of the PSNR, the SSIM, and the trade-off score, but BasicVSR++ adopts a bidirectional structure and requires access to the whole video sequence for restoration, which is impractical in online situations. Compared with the original BasicVSR++, the modified model BasicVSR++* has lower model complexity and processing time, but the performance is unavoidably degraded. Compared with BasicVSR*, BasicVSR++* introduces complex modules based on deformable convolution, which unavoidably increases the model complexity and runtime. As observed in Table I, the runtime of BasicVSR++* still barely satisfies the real-time requirement in online situations. It is worth noting that processing speed is very important in online applications, and more effort in terms of reducing the runtime of the online VSR models is needed so that the models can be widely used in different online scenarios.

As observed, our proposed model has the lowest model computational complexity because our proposed method requires fewer FLOPs than the other compared methods, including the two image SR methods, EDSR-M and PAN. In terms of distortion quality, our proposed method significantly outperforms EDSR-M, PAN, TOF, RLSP, and RRN in terms of the PSNR and SSIM. In addition, we found that the latency of our proposed method as measured on a Tesla V-100 GPU is less than 10 ms on the REDS4 dataset, which is much lower than that of the most efficient image SR model (i.e., PAN). The average size of the video sequences in the RED4 dataset is 180×320, and the upscaling factor used in our experiments is 4, which means that our proposed method takes less than 10 ms to produce an HR video frame. This result means that our proposed method is significantly efficient and runs much faster than the other compared VSR methods. Overall, compared to other efficient image SR and VSR methods, our proposed method requires extremely low latency and achieves competitive performance in terms of the distortion quality. Therefore, our proposed method has a higher trade-off score, which shows that our proposed method outperforms other compared methods in online situations.

Figs. 7 and 8 illustrate the visual results of different VSR methods. For better visualization, a region in each image marked by a red rectangular box is cropped and enlarged. In Fig. 7, we can find that the video frames generated by EDSR, PAN, RLSP, TOF, and RRN suffer from severe distortion and lack of sharp texture information in the marked dense grids, while our method can effectively preserve object details (e.g., edges and textures). Similarly, in Fig. 8, we can easily observe that our proposed method has a better ability to generate clear image content (e.g., the word "REE" in the marked region) than the other compared methods. Overall, the illustrated results show that our proposed method has a better ability to produce images with low distortion, leading to high visual quality. Please find more visual results in Appendix C.

B. Experiment on Different Hardware Devices

Online VSR methods are usually deployed on client devices, and the configurations of personal devices vary. In this experiment, we evaluated the latency of different VSR methods on two devices, an NVIDIA 2080 Ti and 1080 Ti. These two devices have been widely used in gaming applications, which require video algorithms to process up to 60 FPS[1]. The REDS4 dataset is used in this experiment, where the size of the generated video frame is 720×1280. The evaluation results on the 2080 Ti and 1080 Ti are shown in Table II. In addition, the relationship between the PSNR and the FPS achieved by different methods is illustrated in Fig. 1. We find that the latency of our method is significantly lower than that of the other methods. In particular, for the 2080 Ti, the processing speed of our proposed method is up to 110 FPS. In addition, we find that none of the three compared methods can meet the latency requirements of online gaming on devices equipped with the 1080 Ti, but our method can effectively process the video sequences at a rate of up to 80 FPS. It is worth noting that personal laptops equipped with the 1080 Ti are not highly configured for online games. This result shows that our method is less affected by hardware devices and more applicable to different industrial products.

TABLE II
LATENCY REQUIRED BY DIFFERENT METHODS RUNNING ON VARIOUS DEVICES. ALL MODELS ARE EVALUATED ON THE REDS4 DATASET, AND THE AVERAGE SIZE OF THE INPUT FRAME IS 180×320. THE BEST RESULTS ARE HIGHLIGHTED IN BOLD.

		Methods			
		EDSR-M	PAN	RRN	CKBG (ours)
#Param		1,571 K	**272 K**	3,364 K	1,750 K
PSNR		28.37	28.41	28.82	**29.73**
2080Ti	Time	22 ms	16 ms	27 ms	**9 ms**
	FPS	45.45	62.5	37.03	**111.11**
	Gaming	✗	✓	✗	✓
1080Ti	Time	40 ms	36 ms	44 ms	**12 ms**
	FPS	25.00	27.77	22.72	**83.33**
	Gaming	✗	✗	✗	✓

C. Experiments on the Real-world VideoLQ Dataset

We also conducted an experiment on the VideoLQ dataset, which contains video frames captured in real-world situations. Since this dataset does not provide the ground-truth videos, we compared our proposed CKBG with EDSR, RRN, RLSP, and BasicVSR* in terms of the nonreference quality measures, i.e., BRISQUE [56] and NIQE [57]. The original BasicVSR and its variants are not included in the comparison because these models have a bidirectional structure and are not suitable for online situations. Table III shows the average BRISQUE and NIQE scores of different models. We find that our proposed CKBG model achieves better performance than RRN and BasicVSR*. Although RLSP can achieve the best performance, in terms of BRISQUE, it needs to cache a future frame, which increases the latency during the restoration process. Fig. 9 illustrates the images generated by different VSR methods. Since no ground-truth image is provided in the VideoLQ

[1]https://www.gamingscan.com/best-fps-gaming

dataset, we only visually compare the quality of the generated images. We cropped the region marked by the red rectangle and enlarged it for better comparison. As observed, EDSR, RRN, RLSP, and BasicVSR* suffer from the over-smoothing issue and struggle to generate the details of dense regions. In contrast, our proposed CKBG method has a better ability to generate sharp and detailed information in dense grids, leading to higher visual quality. It is worth noting that all of the models are trained on the synthesized data, where LR videos are generated by the bicubic downsampling kernels. Due to the degradation mismatch between the training data and the testing data, the performance of all of the models unavoidably degrades. Many previous works have extensively studied this issue [58], [59]. However, in this paper, we focus on reducing the latency of video super-resolution methods without caching future frames while maintaining the reconstruction quality in terms of the PSNR and SSIM in online situations. For simplicity, we adopted the synthesized datasets to train our model. Multiple degradations and perceptual quality enhancement are other challenging issues in real-world situations. We will consider these two issues in our future works.

TABLE III
THE AVERAGE BRISQUE AND NIQE OF DIFFERENT METHODS ON THE VIDEOLQ DATASET. THE BEST RESULTS OF ONLINE METHODS ARE HIGHLIGHTED IN BOLD.

	EDSR	RRN	RLSP	BasicVSR*	CKBG
Online	✓	✓	✗	✓	✓
BRISQUE ↓	62.035	62.133	59.183	61.587	**60.568**
NIQE ↓	**6.567**	6.786	6.804	6.868	6.834

D. Ablation Study on the Kernel Graft

To evaluate the performance of the kernel bypass graft, we conducted three experiments, which are described in this section. In the first experiment, we evaluated the performance of the kernel bypass graft. Specifically, we trained our proposed method in two different settings for comparison. One setting is that the model does not adopt any bypass-graft kernel, denoted as w/o Graft. The second is that the model adopts the bypass-graft kernel obtained in the Euclidean space, denoted as E-Kmeans. Our proposed method learns the kernel graft in the Wasserstein space and is denoted as W-Kmeans. The average PSNR and SSIM for the different settings on the REDS4 dataset are shown in Table IV. As observed, the model with the graft learned in the Euclidean space achieves comparable results with the model without using grafts because the grafted kernels learned in the Euclidean space cannot effectively inherit prior information from the extracted kernels. In contrast, our model using the kernel graft learned in the Wasserstein space significantly outperforms other compared settings because the optimal mapping can help the learned kernels better preserve the geometric properties.

Then, we conducted experiments to investigate the effect of different numbers of cluster centroids K on our proposed method. Specifically, we set K = 64, 128, 256, 512 and ∞, and the number of channels in the intermediate layers is set to 64. All models with different settings are evaluated on the REDS4 dataset in terms of the PSNR and SSIM. The experimental results are shown in Table V. When K = 64, the number of kernel centroids equals the number of channels in the intermediate layers. In this case, no sampling technique is used. When K is larger than 64, the sampling technique is introduced to generate different grafted kernels each time. When the number of centroids is increased from 64 to 256, the performance improves correspondingly because the different grafted kernels produce different features for reconstruction. However, when the number of centroids is increased from 256 to 512, the model achieves similar performance because the cluster centroids are overcomplete. One extreme case involves setting the number of cluster centroids equal to the number of extracted kernels, i.e., $K = \infty$. We find that the corresponding model cannot produce satisfactory performance because the kernel information is not effectively aggregated.

TABLE IV
AVERAGE PSNR AND SSIM OF OUR PROPOSED MODEL WITH DIFFERENT SETTINGS ON THE REDS4 DATASET. ↑ △ DENOTES THE PERFORMANCE INCREMENT. THE BEST RESULTS ARE HIGHLIGHTED IN BOLD.

	w/o Graft	E-Kmeans	W-Kmeans
PSNR	29.58	29.59	**29.73**
↑ △	0.00	0.01	**0.15**
SSIM	0.8480	0.8484	**0.8514**
↑ △	0.0000	0.0004	**0.0034**

TABLE V
AVERAGE PSNR AND SSIM OF THE PROPOSED MODEL WITH DIFFERENT NUMBERS OF CLUSTER CENTROIDS ON THE REDS4 DATASET. NOTE: $K = \infty$ MEANS THAT EACH OBSERVED SAMPLE IS A CLUSTER CENTROID. THE BEST RESULTS ARE HIGHLIGHTED IN BOLD.

	K=64	K=128	K=256	K=512	K=∞
PSNR	29.60	29.64	**29.73**	29.70	29.56
SSIM	0.8473	0.8471	**0.8514**	0.8512	0.8474

Furthermore, we conducted experiments to evaluate the effect of the proposed CKBG model with different numbers of bypassed branches. Specifically, we set the number of branches of our CKBG model B to 0, 1, 2, and 3. The other configurations remain the same. All models are evaluated on the REDS4 dataset. The average PSNR and SSIM of the models with different settings are shown in Table VI. As observed, when the number of bypassed branches is increased from 0 to 2, the performance of our proposed model improves significantly because extra information is incorporated for feature extraction and reconstruction. However, when the number of bypass branches is increased to 3, the performance drops slightly because the number of bypassed branches is greater than the number of main branches in the base model. The information from the bypassed branches dominates and suppresses the information from the main base model during the reconstruction process. We also found that the proposed CKBG model with three bypassed branches needs ten days to complete the training process. We will further investigate this

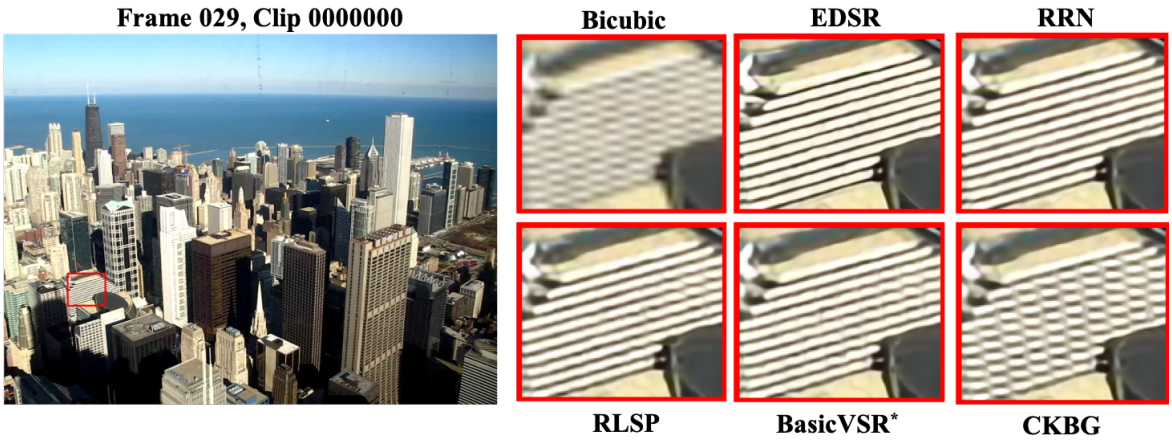


Fig. 9. The illustrated image is selected from the VideoLQ dataset. The region marked by the red box is generated by different VSR methods for visual comparison.

issue and accelerate the training speed in our future works.

TABLE VI
AVERAGE PSNR AND SSIM OF THE MODELS WITH DIFFERENT NUMBERS OF BYPASSED BRANCHES ON THE REDS4 DATASET. ALL RESULTS ARE EVALUATED ON THE RGB CHANNEL. THE BEST RESULTS ARE HIGHLIGHTED IN BOLD.

	$B=0$	$B=1$	$B=2$	$B=3$
PSNR	29.58	29.67	**29.73**	29.70
SSIM	0.8480	0.8497	**0.8514**	0.8514

E. Ablation Study on Different Kernels

In this experiment, we compared our proposed bypass-grafted kernels with other re-parameterized kernels, such as the ACB kernel [60], RepVGG kernel [12], and ECB kernel [27]. Specifically, we retrained the model by replacing our proposed grafted kernel with other re-parameterized kernels. The average PSNR and SSIM of the different kernel settings on the REDS4 dataset are shown in Table VII. We find that our method achieves the best performance compared with other kernel settings. Fig. 10 illustrates the feature maps generated from the last convolutional block of the models with different kernels. We highlighted regions of dense grids with red and blue rectangular boxes for better visual comparison. As observed, the generated result of our method looks much sharper, and the marked region of dense grids has less distorted content.

TABLE VII
AVERAGE PSNR AND SSIM RESULTS OF DIFFERENT RE-PARAMETERIZATION CONVOLUTIONAL BLOCKS ON THE REDS4 DATASET. THE BEST RESULTS ARE HIGHLIGHTED IN BOLD.

	ACB [60]	RepVGG [12]	ECB [27]	CKBG (ours)
PSNR	29.53	29.51	29.58	**29.73**
SSIM	0.8471	0.8464	0.8478	**0.8514**

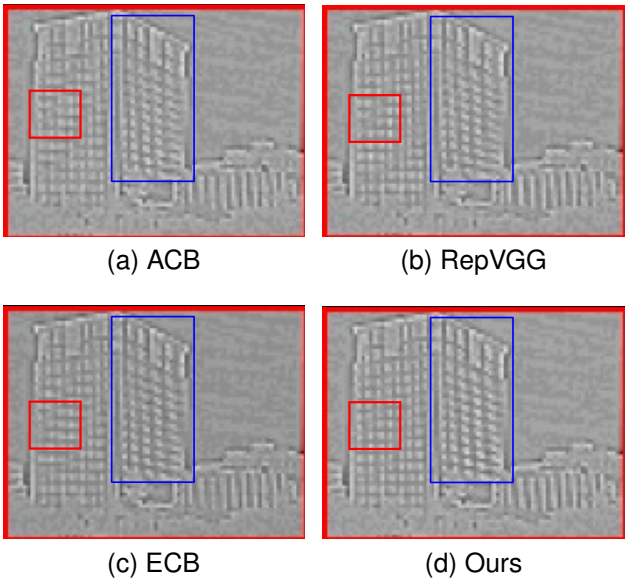


Fig. 10. Resulting features generated by ACB, RepVGG, ECB, and our proposed method, i.e., CKBG.

F. Ablation Study on Knowledge Transfer

We further compare our method with two deep super-resolution models, i.e., FAKD [44] and FDVDNet [46], which are based on knowledge distillation. The compared methods distill prior knowledge from the teacher network based on the feature map generated from the intermediate layers. In addition, the teacher networks provide supervised signals to guide the student networks for training. In contrast, our proposed CKBG method extracts convolutional kernels from a pretrained image super-resolution model and grafts the extracted kernels to a lightweight model before training. The average PSNR and runtime of the different methods on the Vid4 dataset are tabulated in Table VIII, where the best results are highlighted in bold. The runtimes of the different methods are measured on a device equipped with an NVIDIA GeForce GTX 1080 Ti. As seen in Table VIII, our method can

achieve the best performance with the lowest runtime compared with FAKD and FDVDNet. These results demonstrate that our method is more effective and efficient for video super-resolution.

TABLE VIII
AVERAGE PSNR AND RUNTIME OF SR MODELS WITH DIFFERENT KNOWLEDGE TRANSFER METHODS ON THE VID4 DATASET. THE BEST RESULTS ARE HIGHLIGHTED IN BOLD. THE RUNTIME IS MEASURED IN AN NVIDIA 1080 TI.

	Bicubic [60]	FAKD [44]	FDVDNet [46]	CKBG (ours)
PSNR	23.78	25.42	26.14	**26.34**
Time	-	28.31 ms	17.50 ms	**12.26 ms**

V. CONCLUSION

Online video applications have high requirements for VSR algorithms in terms of processing latency, model complexity, and distortion quality, but few of the existing VSR methods can handle such challenging issues simultaneously. In this paper, we propose an extremely low-latency VSR method for online applications. A novel knowledge transfer method, called the convolutional kernel bypass graft, is proposed. The proposed CKBG aims to improve the performance of a base VSR network by bypassing a set of extra kernels containing rich prior knowledge from external, pretrained SR models (i.e., kernel grafts). The experimental results show that our proposed method can process a video sequence at a rate of up to 110 FPS and achieve the best trade-off between the distortion quality and the processing latency compared with other competitive VSR methods.

ACKNOWLEDGMENTS

This work was supported by the Hong Kong Research Grants Council (RGC) Research Impact Fund (RIF) under Grant R5001-18.

APPENDIX A
KERNEL REPARAMETERIZATION OF THE CKBG

In this paper, we propose a novel kernel knowledge-transfer method called convolutional kernel graft bypass that enhances a base VSR model by bypassing the learned "kernel graft" from a large pretrained model. The resulting network module is called a bypass graft block (BGB), which has a multibranch structure for feature extraction. In our method, the proposed BGB is trained in the form of a multibranch structure, but the multiple branches can be converted into a convolutional operator during the testing stage because of the linear property of the convolutional operation [12]. This conversion is called kernel reparameterization, which involves two types of reparameterizations: sequential convolutions and parallel convolution, as shown in Fig. 11. Our proposed method involves two types of kernel reparameterization: sequential convolution and parallel convolution. During the testing stage, the 1×1 convolution and 3×3 convolution sequences are first merged into 3×3 convolutions to form a parallel structure. Then, all 3×3 convolutional kernels in the parallel branches are merged into a single convolution kernel. The resulting structure is a single-path topology. Next, we elaborate on how to perform kernel reparameterization for sequential and parallel convolutions.

Type I: Sequential convolution. Suppose we have a sequential convolution of a 1×1 kernel and a $K\times K$ kernel, denoted as $F_1\in\mathbb{R}^{\mathrm{C_{in}}\times\mathrm{C_{mid}}\times1\times1}$ and $F_2\in\mathbb{R}^{\mathrm{C_{mid}}\times\mathrm{C_{out}}\times K\times K}$, respectively, where $\mathrm{C_{in}}$, $\mathrm{C_{mid}}$, and $\mathrm{C_{out}}$ represent the number of channels of the input features, intermediate features, and output features, respectively. Given the input features F_{in}, the output F_{out} of the sequential convolution is computed as follows:

$$F_{\text{out}} = (F_{\text{in}} * F_1 + \mathbf{b}_1) * F_2 + \mathbf{b}_2, \tag{14}$$

$$= F_{\text{in}} * F_1 * F_2 + \mathbf{b}_1 * F_2 + \mathbf{b}_2, \tag{15}$$

where $*$ denotes the convolutional operation, and $\mathbf{b}_1\in\mathbb{R}^{1\times C_{\text{mid}}}$ and $\mathbf{b}_2\in\mathbb{R}^{1\times C_{\text{out}}}$ are the corresponding bias terms. According to the linear property of the convolutional operator, the convolution kernels F_1 and F_2 in the first term can be merged as follows:

$$\hat{F} = F_1 * F_2, \tag{16}$$

where $\hat{F}$ denotes the resulting convolution kernel with a size of $\mathrm{C_{in}}\times\mathrm{C_{out}}\times K\times K$. For the second term $\mathbf{b}_1 * F_2$, the corresponding re-parameterized result $\hat{\mathbf{b}}_1=[\hat{b}_{1,1},\cdots,\hat{b}_{1,C_{\text{mid}}}]$ is computed as follows:

$$\hat{b}_{1,c_{\text{out}}} = \sum_{c=1}^{C_{\text{mid}}}\sum_{m=1}^{K}\sum_{n=1}^{K} b_{1,c}F^{(2)}_{c_{\text{out}},c,m,n}, \tag{17}$$

where $c_{\text{out}}=1,\cdots,C_{\text{out}}$, and $F^{(2)}_{c_{\text{out}},c,m,n}$ represents the kernel value at position (c_{out},c,m,n) of convolution kernel F_2. Then, the final re-parameterized bias term is obtained as follows:

$$\hat{\mathbf{b}} = \hat{\mathbf{b}}_1 + \mathbf{b}_2. \tag{18}$$

As a result, the sequential convolution of a 1×1 convolution and a $K\times K$ convolution is merged into a single convolutional kernel after performing kernel reparameterization. Eqn. (15) can be rewritten as follows:

$$F_{\text{out}} = F_{\text{in}} * \hat{F} + \hat{\mathbf{b}}. \tag{19}$$

Type II: Parallel convolution. Assume that the parameter set of the i-th branch in the ℓ-th BGB is denoted as $\Theta^\ell_i=\{F^\ell_i,\mathbf{b}^\ell_i\}$, where $F^\ell_i\in\mathbb{R}^{C_{\text{in}}\times C_{\text{out}}\times K\times K}$ and $\mathbf{b}\in\mathbb{R}^{1\times C_{\text{out}}}$ represent the convolution kernel and the bias term, respectively. Without loss of generality, suppose a BGB has B branches. Given an input feature map F^ℓ_{in}, the output of the multibranch block is computed as follows:

$$F^\ell_{\text{out}} = \sum_{i=1}^{B}(F^\ell_{\text{in}} * F^\ell_i + \mathbf{b}^\ell_i). \tag{20}$$

Then, we can perform kernel reparameterization based on the linear property of convolution again. As a result, Eqn. (20) can be rewritten as follows:

$$F^\ell_{\text{out}} = F^\ell_{\text{in}} * \left(\sum_{b=1}^{B}F^\ell_b\right) + \left(\sum_{b=1}^{B}\mathbf{b}^\ell_b\right), \tag{21}$$

$$= F^\ell_{\text{in}}\hat{F}^\ell + \hat{\mathbf{b}}^\ell, \tag{22}$$

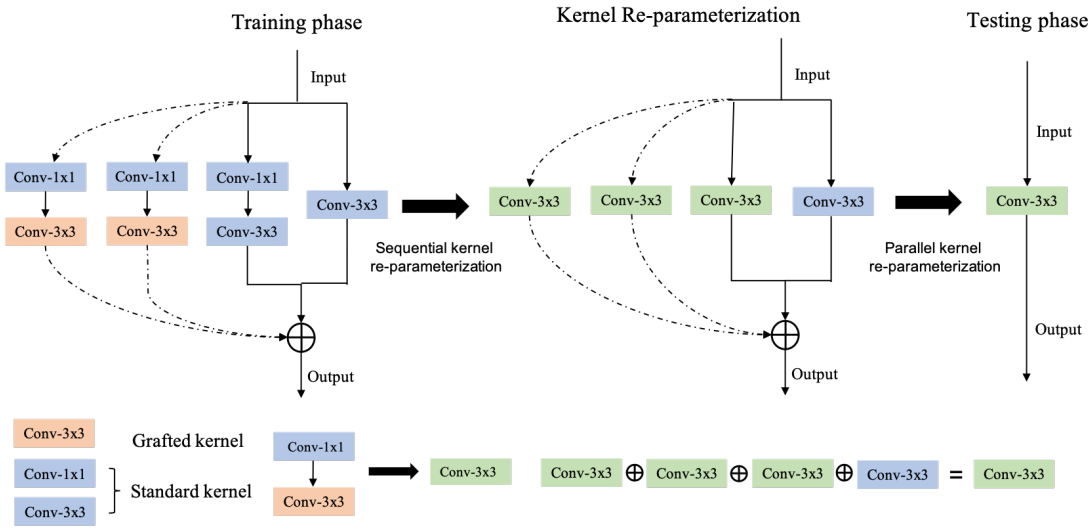


Fig. 11. Illustration of the kernel reparameterization of sequential convolutions and parallel convolutions during the testing stage. The dotted lines represent the grafted branches. The grafted kernels marked by orange are fixed during the training stage. The re-parameterized kernels are marked in green.

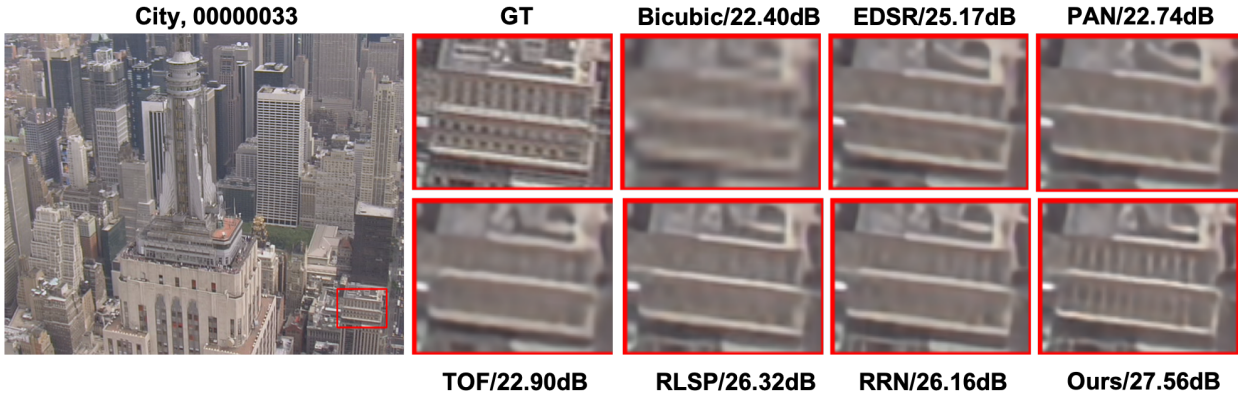


Fig. 12. The illustrated image "City 00000033" is selected from the Vid4 dataset. The region marked by the red box is generated by different VSR methods for visual comparison.

where $\hat{F}^\ell = \sum_{b=1}^{B} F_b^\ell$ and $\hat{\mathbf{b}}^\ell = \sum_{b=1}^{B} \mathbf{b}_j^\ell$ are the corresponding re-parameterized convolutional kernel and the bias term, respectively. As a result, the multibranch structure is converted into a single-path structure during the testing stage.

Since kernel reparameterization relies on the linear property of the convolution operation, the re-parameterized kernel in the testing stage is equivalent to the multibranch structure in the training stage without sacrificing any performance. The resulting single-path structure is very efficient, so the processing speed can be significantly accelerated.

Appendix B
Implementation Details of BasicVSR* and BasicVSR++*

BasicVSR [40] and BasicVSR++ [37] are bidirectional models that have shown their effectiveness for VSR. However, the original BasicVSR and BasicVSR++ cannot be applied to online video applications because they require access to the whole video sequence, which is impractical in online scenarios. To make BasicVSR and BasicVSR++ meet the online requirements, we modify BasicVSR and BasicVSR++, which are denoted as BasicVSR* and BasicVSR++*. Specifically, we first remove the backward part of the model so that it does not need to cache future frames. To meet the requirement of low model complexity, we reduce the number of residual blocks from 60 to 15, and the number of feature channels is reduced from 64 to 32. The other configurations remain the same as the original model for training.

Appendix C
More Visual Results

In Fig. 12 and 13, we provide additional visual results generated by different image/video SR methods for visual comparison. As illustrated, our proposed method has a better ability to preserve the shapes and textures of objects in the images than the other compared methods. In addition, the images generated by our proposed method contain less distorted content than those produced by the other compared methods, leading to the best visual quality.

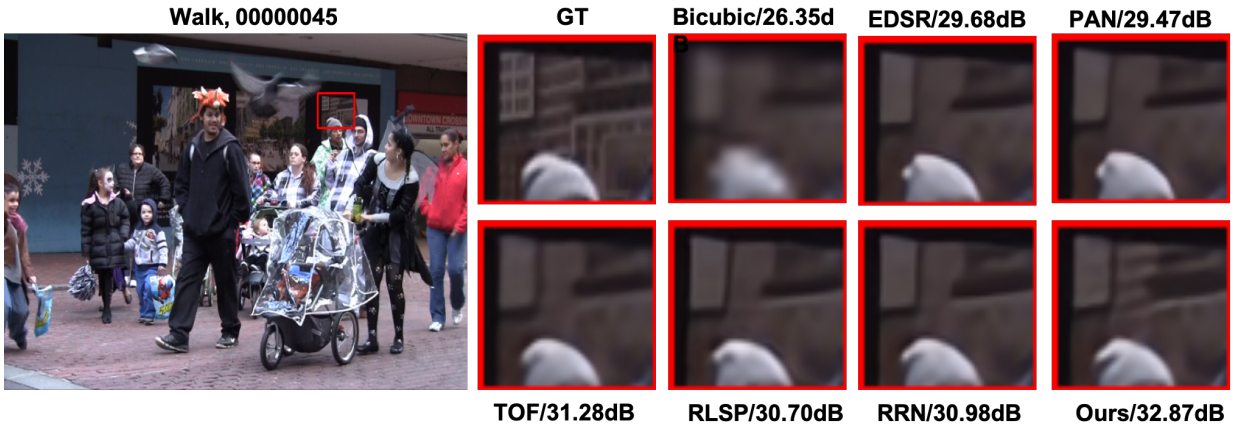


Fig. 13. The illustrated image "Walk 00000045" is selected from the Vid4 dataset. The region marked by the red box is generated by different VSR methods for visual comparison.

REFERENCES

[1] X. Nan, X. Guo, Y. Lu, Y. He, L. Guan, S. Li, and B. Guo, "Delay–rate–distortion optimization for cloud gaming with hybrid streaming," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 27, no. 12, pp. 2687–2701, 2016.

[2] C. P. Lau, A. Alabbasi, and B. Shihada, "An efficient live tv scheduling system for 4g lte broadcast," *IEEE Systems Journal*, vol. 11, no. 4, pp. 2737–2748, 2016.

[3] Z. Luo, Z. Wang, M. Hu, Y. Zhou, and D. Wu, "Livesr: Enabling universal hd live video streaming with crowdsourced online learning," *IEEE Transactions on Multimedia*, 2022.

[4] "Video enhancement with task-oriented flow," *International Journal of Computer Vision*, vol. 127, pp. 1106–1125, 2019.

[5] D. Fuoli, S. Gu, and R. Timofte, "Efficient video super-resolution through recurrent latent space propagation," in *Proceedings of the IEEE International Conference on Computer Vision Workshop (CVPRW)*, 2019.

[6] X. Wang, K. C. Chan, K. Yu, C. Dong, and C. Change Loy, "Edvr: Video restoration with enhanced deformable convolutional networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2019.

[7] M. Haris, G. Shakhnarovich, and N. Ukita, "Recurrent back-projection network for video super-resolution," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[8] J. Caballero, C. Ledig, A. Aitken, A. Acosta, J. Totz, Z. Wang, and W. Shi, "Real-time video super-resolution with spatio-temporal networks and motion compensation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[9] W. Li, X. Tao, T. Guo, L. Qi, J. Lu, and J. Jia, "Mucan: Multi-correspondence aggregation network for video super-resolution," in *European Conference on Computer Vision (ECCV)*, 2020.

[10] P. Yi, Z. Wang, K. Jiang, J. Jiang, and J. Ma, "Progressive fusion video super-resolution network via exploiting non-local spatio-temporal correlations," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2019.

[11] X. Ding, X. Zhang, J. Han, and G. Ding, "Diverse branch block: Building a convolution as an inception-like unit," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.

[12] X. Ding, X. Zhang, N. Ma, J. Han, G. Ding, and J. Sun, "Repvgg: Making vgg-style convnets great again," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.

[13] B. Lim, S. Son, H. Kim, S. Nah, and K. Mu Lee, "Enhanced deep residual networks for single image super-resolution," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2017.

[14] Y. Zhang, K. Li, K. Li, L. Wang, B. Zhong, and Y. Fu, "Image super-resolution using very deep residual channel attention networks," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 286–301.

[15] J. Kim, J. Kwon Lee, and K. Mu Lee, "Deeply-recursive convolutional network for image super-resolution," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 1637–1645.

[16] Y. Tai, J. Yang, and X. Liu, "Image super-resolution via deep recursive residual network," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 3147–3155.

[17] Z. Li, J. Yang, Z. Liu, X. Yang, G. Jeon, and W. Wu, "Feedback network for image super-resolution," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 3867–3876.

[18] Q. Li, Z. Li, L. Lu, G. Jeon, K. Liu, and X. Yang, "Gated multiple feedback network for image super-resolution," in *Proceedings of British Machine Vision Conference (BMVC)*, 2019.

[19] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "Mobilenets: Efficient convolutional neural networks for mobile vision applications," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[20] N. Ahn, B. Kang, and K.-A. Sohn, "Fast, accurate, and lightweight super-resolution with cascading residual network," in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 252–268.

[21] Z. Hui, X. Wang, and X. Gao, "Fast and accurate single image super-resolution via information distillation network," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018, pp. 723–731.

[22] Z. Hui, X. Gao, Y. Yang, and X. Wang, "Lightweight image super-resolution with information multi-distillation network," in *Proceedings of the 27th ACM International Conference on Multimedia*, 2019, pp. 2024–2032.

[23] H. Zhao, X. Kong, J. He, Y. Qiao, and C. Dong, "Efficient image super-resolution using pixel attention," in *Proceedings of European Conference on Computer Vision (ECCV)*, 2020.

[24] J. Xiao, Q. Ye, R. Zhao, K.-M. Lam, and K. Wan, "Self-feature learning: An efficient deep lightweight network for image super-resolution," in *Proceedings of the 29th ACM International Conference on Multimedia*, 2021, pp. 4408–4416.

[25] X. Chu, B. Zhang, and R. Xu, "Multi-objective reinforced evolution in mobile neural architecture search," in *Proceedings of European Conference on Computer Vision*. Springer, 2020, pp. 99–113.

[26] D. Song, C. Xu, X. Jia, Y. Chen, C. Xu, and Y. Wang, "Efficient residual dense block search for image super-resolution," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 07, 2020, pp. 12 007–12 014.

[27] X. Zhang, H. Zeng, and L. Zhang, "Edge-oriented convolution block for real-time super resolution on mobile devices," in *Proceedings of the 29th ACM International Conference on Multimedia*, 2021.

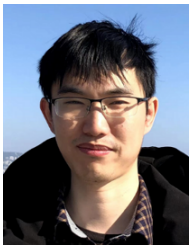
[28] M. Lu, T. Chen, Z. Dai, D. Wang, D. Ding, and Z. Ma, "Decoder-side cross resolution synthesis for video compression enhancement," *IEEE Transactions on Multimedia*, 2022.

[29] H. Lin, X. He, L. Qing, Q. Teng, and S. Yang, "Improved low-bitrate hevc video coding using deep learning based super-resolution

and adaptive block patching," *IEEE Transactions on Multimedia*, vol. 21, no. 12, pp. 3010–3023, 2019.
[30] H. Sun, Z. Cheng, M. Takeuchi, and J. Katto, "Enhanced intra prediction for video coding by using multiple neural networks," *IEEE Transactions on Multimedia*, vol. 22, no. 11, pp. 2764–2779, 2020.
[31] Y. Chen, P. Zhao, M. Qi, Y. Zhao, W. Jia, and R. Wang, "Audio matters in video super-resolution by implicit semantic guidance," *IEEE Transactions on Multimedia*, 2022.
[32] W. Shi, J. Caballero, F. Huszár, J. Totz, A. P. Aitken, R. Bishop, D. Rueckert, and Z. 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 (CVPR)*, 2016, pp. 1874–1883.
[33] Y. Jo, S. W. Oh, J. Kang, and S. J. Kim, "Deep video super-resolution network using dynamic upsampling filters without explicit motion compensation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018, pp. 3224–3232.
[34] J. Dai, H. Qi, Y. Xiong, Y. Li, G. Zhang, H. Hu, and Y. Wei, "Deformable convolutional networks," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2017, pp. 764–773.
[35] J. Lin, Y. Huang, and L. Wang, "Fdan: Flow-guided deformable alignment network for video super-resolution," *arXiv preprint arXiv:2105.05640*, 2021.
[36] T. Isobe, X. Jia, S. Gu, S. Li, S. Wang, and Q. Tian, "Video super-resolution with recurrent structure-detail network," in *Proceedings of European Conference on Computer Vision (ECCV)*. Springer, 2020, pp. 645–660.
[37] K. C. Chan, S. Zhou, X. Xu, and C. C. Loy, "Basicvsr++: Improving video super-resolution with enhanced propagation and alignment," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 5972–5981.
[38] H. Wang, W. Yang, Q. Liao, and J. Zhou, "Bi-rstu: Bidirectional recurrent upsampling network for space-time video super-resolution," *IEEE Transactions on Multimedia*, 2022.
[39] Y. Huang, W. Wang, and L. Wang, "Video super-resolution via bidirectional recurrent convolutional networks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 40, no. 4, pp. 1015–1028, 2017.
[40] K. C. Chan, X. Wang, K. Yu, C. Dong, and C. C. Loy, "Basicvsr: The search for essential components in video super-resolution and beyond," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.
[41] W. Xiong, W. Luo, L. Ma, W. Liu, and J. Luo, "Learning to generate time-lapse videos using multi-stage dynamic generative adversarial networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.
[42] J. Li, X. Wu, Z. Niu, and W. Zuo, "Unidirectional video denoising by mimicking backward recurrent modules with look-ahead forward ones," in *European Conference on Computer Vision*. Springer, 2022, pp. 592–609.
[43] Q. Gao, Y. Zhao, G. Li, and T. Tong, "Image super-resolution using knowledge distillation," in *Proceedings of Asian Conference on Computer Vision (ACCV)*. Springer, 2018, pp. 527–541.
[44] Z. He, T. Dai, J. Lu, Y. Jiang, and S.-T. Xia, "Fakd: Feature-affinity based knowledge distillation for efficient image super-resolution," in *2020 IEEE International Conference on Image Processing (ICIP)*. IEEE, 2020, pp. 518–522.
[45] W. Lee, J. Lee, D. Kim, and B. Ham, "Learning with privileged information for efficient image super-resolution," in *Proceedings of European Conference on Computer Vision (ECCV)*. Springer, 2020, pp. 465–482.
[46] Z. Xiao, X. Fu, J. Huang, Z. Cheng, and Z. Xiong, "Space-time distillation for video super-resolution," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021, pp. 2113–2122.
[47] A. Ranjan and M. J. Black, "Optical flow estimation using a spatial pyramid network," in *Proceedings of the IEEE Conference on Computer Vision and Pattern recognition (CVPR)*, 2017, pp. 4161–4170.
[48] C. Villani, *Optimal transport:old and new*. Springer, 2009, vol. 338.
[49] L. Xie, X. Wang, C. Dong, Z. Qi, and Y. Shan, "Finding discriminative filters for specific degradations in blind super-resolution," *Advances in Neural Information Processing Systems*, vol. 34, pp. 51–61, 2021.
[50] L. Mi, W. Zhang, X. Gu, and Y. Wang, "Variational wasserstein clustering," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018.
[51] S. Son, S. Lee, S. Nah, R. Timofte, and K. M. Lee, "Ntire 2021 challenge on video super-resolution," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, June 2021, pp. 166–181.
[52] C. Liu and D. Sun, "On bayesian adaptive video super resolution," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 36, no. 2, pp. 346–360, 2013.
[53] A. Ignatov, A. Romero, H. Kim, and R. Timofte, "Real-time video super-resolution on smartphones with deep learning, mobile ai 2021 challenge: Report," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.
[54] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proceedings of International Conference on Learning Representation (ICLR)*, 2014.
[55] T. Isobe, F. Zhu, X. Jia, and S. Wang, "Revisiting temporal modeling for video super-resolution," in *Proceedings of the British Machine Vision Conference (BMVC)*, 2020.
[56] A. Mittal, A. K. Moorthy, and A. C. Bovik, "Blind/referenceless image spatial quality evaluator," in *2011 conference record of the forty fifth asilomar conference on signals, systems and computers (ASILOMAR)*. IEEE, 2011, pp. 723–727.
[57] A. Mittal, R. Soundararajan, and A. C. Bovik, "Making a "completely blind" image quality analyzer," *IEEE Signal processing letters*, vol. 20, no. 3, pp. 209–212, 2012.
[58] J. Cai, H. Zeng, H. Yong, Z. Cao, and L. Zhang, "Toward real-world single image super-resolution: A new benchmark and a new model," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 3086–3095.
[59] P. Wei, Z. Xie, H. Lu, Z. Zhan, Q. Ye, W. Zuo, and L. Lin, "Component divide-and-conquer for real-world image super-resolution," in *European Conference on Computer Vision*. Springer, 2020, pp. 101–117.
[60] S.-Y. Lo, H.-M. Hang, S.-W. Chan, and J.-J. Lin, "Efficient dense modules of asymmetric convolution for real-time semantic segmentation," in *Proceedings of the ACM Multimedia Asia*, 2019.

D Biography Section

Jun Xiao received his B.Sc. degree in telecommunication engineering from Guangdong University of Technology, Guangdong, China, in 2016, the M.Sc. degree in electronic and information engineering and the Ph.D. degree from the Department of Electronic and Information Engineering, the Hong Kong Polytechnic University, Hong Kong, in 2018 and 2022, respectively. His research interests include image/video restoration, image/video quality enhancement, and probabilistic machine learning.



Xinyang Jiang received B.E. from Zhejiang University in 2012 and Ph.D. from Zhejiang University in 2017. He is now a researcher from Microsoft Research Asia. Before joining MSRA, he was a researcher from Tencent Youtu Lab. His main research field is computer vision, including person Re-identification, vector graphics recognition and video enhancement and recognition.

Ningxing Zheng received a B.S. degree from the Hua Zhong University of Science and Technology, in 2017, and an M.S. degree from Shanghai Jiao Tong University in 2020, advised by prof. Minyi Guo and Quan Chen. He is currently a research software development engineer II at Microsoft Research Shanghai. His research interests include AI systems, Cloud computing, and Model Compression.

Huan Yang is a senior researcher at Microsoft Research Asia, Multi-Modal Computing Group. He received his B.S. and Ph.D. degrees in computer science in 2014 and 2019 respectively from Shanghai Jiao Tong University, China. His research interest includes image/video restoration, image/video enhancement and GAN/diffusion-based content creation.

Yifan Yang received his M.S. degree from Peking University in 2021 and a B.S. degree from Tongji University. Now he is a Research SDE at Microsoft Research Asia, where his research interests focus on visual self-supervised learning, vision language pre-training, and video enhancement.

Yuqing Yang received his B.S. and Ph.D degrees in Microelectronics from Fudan University in 2006 and 2011, respectively. He is now working for Microsoft Research Asia and mainly focusing on efficient computing systems for emerging scenarios, such as deep learning and video streaming. His recent includes execution optimization for dynamic and sparse neural networks, efficient computing for cloud and edge, neural enhanced video streaming, etc.

Dongsheng Li received B.E. from University of Science and Technology of China in 2007 and Ph.D. from Fudan University in 2012. He is now a principal research manager with Microsoft Research Asia (MSRA) since February 2020. Before joining MSRA, he was a research staff member with IBM Research – China since April 2015. He is also an adjunct professor with School of Computer Science, Fudan University, Shanghai, China. His research interests include recommender systems and machine learning applications. His work on cognitive recommendation engine won the 2018 IBM Corporate Award.

Kin-Man Lam received his Associateship in Electronic Engineering with distinction from The Hong Kong Polytechnic University (formerly called Hong Kong Polytechnic) in 1986, his M.Sc. degree in communication engineering from the Department of Electrical Engineering, Imperial College, U.K., in 1987, and his Ph.D. degree from the Department of Electrical Engineering, University of Sydney, Australia, in 1996. From 1990 to 1993, he was a lecturer at the Department of Electronic Engineering of The Hong Kong Polytechnic University. He joined the Department of Electronic and Information Engineering, The Hong Kong Polytechnic University again as an Assistant Professor in October 1996. He became an Associate Professor in 1999, and has been a Professor since 2010. Currently, he is also an Associate Dean of the Faculty of Engineering. He was actively involved in professional activities. He has been a member of the organizing committee or program committee of many international conferences. Prof. Lam was the Chairman of the IEEE Hong Kong Chapter of Signal Processing between 2006 and 2008, and was the Director-Student Services and the Director-Membership Services of the IEEE SPS between 2012 and 2014, and between 2015 and 2017, respectively. He was also the VP-Member Relations and Development and VP-Publications of the Asia-Pacific Signal and Information Processing Association (APSIPA) between 2014 and 2017, and between 2017 and 2021, respectively. He was an Associate Editor of IEEE Trans. on Image Processing between 2009 and 2014, and Digital Signal Processing between 2014 and 2018. He was also an Editor of HKIE Transactions between 2013 and 2018, and an Area Editor of the IEEE Signal Processing Magazine between 2015 and 2017. Currently, he is the IEEE SPS VP-Membership and the Member-at-Large of APSIPA. Prof. Lam also serves as a Senior Editorial Board member of APSIPA Trans. on Signal and Information Processing and an Associate editor of EURASIP International Journal on Image and Video Processing. His current research interests include image and video processing, computer vision, and human face analysis and recognition.