Revisiting Temporal Alignment for Video Restoration

Kun Zhou1,3*  Wenbo Li2*  Liying Lu2  Xiaoguang Han1  Jiangbo Lu3†

1The Chinese University of Hong Kong (Shenzhen), 2The Chinese University of Hong Kong 3Smartmore Corporation

kunzhou@link.cuhk.edu.cn, {wenboli,lylu}@cse.cuhk.edu.hk
hanxiaoguang@cuhk.edu.cn, jiangbo.lu@gmail.com

Abstract

Long-range temporal alignment is critical yet challenging for video restoration tasks. Recently, some works attempt to divide the long-range alignment into several sub-alignments and handle them progressively. Although this operation is helpful in modeling distant correspondences, error accumulation is inevitable due to the propagation mechanism. In this work, we present a novel, generic iterative alignment module which employs a gradual refinement scheme for sub-alignments, yielding more accurate motion compensation. To further enhance the alignment accuracy and temporal consistency, we develop a non-parametric re-weighting method, where the importance of each neighboring frame is adaptively evaluated in a spatial-wise way for aggregation. By virtue of the proposed strategies, our model achieves state-of-the-art performance on multiple benchmarks across a range of video restoration tasks including video super-resolution, denoising and deblurring. Our project is available in https://github.com/redrock303/Revisiting-Temporal-Alignment-for-Video-Restoration.git.

1. Introduction

Frame alignment plays an essential role in aggregating temporal information in video restoration tasks, e.g., video super-resolution (Video SR), video deblurring, and video denoising. In recent years, great attempts have been made to study this problem. Especially, the deep learning-based methods are successful in building temporal correspondences and achieve promising results.

The existing alignment methods can be roughly categorized into two classes: (i) independent alignment that conducts frame-to-frame alignments totally independently (see Fig. 2(a)) and (ii) progressive alignment that performs temporally consecutive alignments sequentially in a recursive manner (see Fig. 2(b)). Those independent alignment approaches typically focus on designing effective feature descriptors and motion estimation modules to improve the performance. For example, EDVR [28] develops pyramid, cascading and deformable convolutions (PCD) for more accurate alignment. Whereas, without exploiting the correlations between multiple alignments, this strategy is still facing challenges to estimate the long-range motion fields. The second line typically adopts a recurrent framework for gradual alignment. Taking BasicVSR [3] for example, the authors propose an optical-flow-based recurrent architecture for video super-resolution. They predict the bidirectional optical flow between two neighboring frames and then conduct a bidirectional propagation, where the temporal information is aggregated by warping image features produced by previous steps. This kind of methods is mainly proposed to model long-range dependencies since it only needs to handle relatively small motion between neighboring frames in one step. However, such chain-rule-based propagation has no chance to correct the misalignment caused by previ-
As illustrated in Fig. 2(c), we observe that different long-range alignments \(A_i\) actually share some sub-alignments \(a_i\), e.g., \(a_1\) is shared among \(A_1\), \(A_2\) and \(A_3\), so as \(a_2\) in \(A_2\) and \(A_3\). How can we utilize this property to improve the accuracy of the shared sub-alignments? In this work, we propose an iterative alignment module (IAM) built upon the progressive alignment strategy to gradually refine the shared sub-alignments. For a specific shared sub-alignment (e.g., \(a_2\) in \(A_2\) and \(A_3\)), the previously estimated result \((a_2\) in \(A_3\)) is used as a prior in the current iteration \((a_2\) in \(A_3\)). Our IAM has two merits over the progressive alignment scheme. First, the progressive alignment only conducts a single prediction for each sub-alignment so that misalignment can not be corrected. In contrast, our IAM refines each sub-alignment iteratively, yielding more accurate alignment. Second, the progressive alignment performs multi-frame aggregation based on a chain-like propagation so that misalignment will be propagated to the end. In our IAM, each neighboring frame is aligned through individual propagation, making it more reliable. Furthermore, to reduce the computational complexity, we elaborate a simple yet efficient alignment unit for temporal sub-alignments. From Fig. 1, it is observed that our alignment algorithm yields high inference efficiency and superior performance compared with state-of-the-art video SR methods. Particularly, our IAM can be easily plugged into existing deep models. For example, by replacing the original independent alignment module of IconVSR [3] with our “IAM” (denoted as “IconVSR+IAM” in Fig. 1), the PSNR is boosted from 37.47dB to 37.56dB on Vimeo-90K-T [33], while reducing the number of parameters from 8.7M to 7.8M.

Besides, the aggregation of multiple aligned frames remains an essential step, for the purpose of preserving details while eliminating alignment errors. Modern restoration systems either employ a sequential of convolutions to directly fuse the aligned features [3, 27] or adopt spatial-temporal adaptive aggregation strategies [9, 13, 18, 28, 31]. However, all these methods solely rely on the learned parameters, raising the risk of overfitting on a specific domain. In this work, we strive a non-parametric re-weighting module, where two strategies are designed to explicitly evaluate the spatially-adaptive importance of different frames. First, we explore the accuracy of alignments. Patches in the aligned frames are compared with the counterparts in the reference frame, and those of high similarity are assigned with larger weights during fusion. Second, to evaluate the consistency of alignments, we compute the pixel-wise L2 distances of the aligned frames with their average. Pixels with smaller distances are considered to be more consistent with other frames and hence assigned with larger weights. The proposed re-weighting module is parameterless and hence can be plugged into other models.

The main contributions are summarized as:

- We rethink issues of the progressive alignment and accordingly propose an iterative alignment scheme, yielding more accurate estimation, especially over long-range correspondences.
- We propose a non-parametric re-weighting module that simultaneously evaluates the alignment accuracy and temporal consistency.
- The quantitative and qualitative results justify the state-of-the-art performance of our method across several video restoration tasks.

2. Related Work

Temporal Alignment. Many video restoration approaches [3, 21, 30, 33] perform independent temporal alignment between neighboring frames with the central frame. Various strategies have been proposed to improve the performance. For example, to fill the domain gap between opt-
Feature Extraction
Sec. 3.2

Alignment
Sec. 3.3

Reconstruction

Video SR

Feature Fusion.

The majority of video restoration methods fuse the aligned frames for temporal information aggregation by feature concatenation followed by a convolution [17, 27, 33]. For example, FastDVD [25] divides consecutive frames into different groups and designs a two-stage convolutional neural network for multi-frame fusion. In addition, more effective aggregation strategies have been proposed by applying spatial or temporal attention-based mechanism [9, 13, 18, 31]. Isobe et al. [9] design a frame-rate-aware group attention, which can handle various levels of motions. In [30], a motion robustness analysis is adopted to fuse temporal information, where different confidence scores are assigned to the local neighbors of each pixel for merging. Inspired by this work, we design an adaptive re-weighting module for information aggregation, considering both the accuracy and consistency of the alignment.

3. Methodology

3.1. Overview

Fig. 3 shows the proposed framework. Our goal is to reconstruct a high-quality image $I_{k}$ from $2N + 1$ consecutive low-quality images $\{I_{N}, \cdots, I_{0}, \cdots, I_{N}\}$. In the feature extraction module, the input frames are first downsampled with strided convolutions for video deblurring/denoising, while being processed under the same resolution for video SR. Then we utilize the proposed IAM to align input frames referring to the central frame. For simplicity, we only consider the one-side alignment in the following as the other side is processed symmetrically. Afterwards, an adaptive re-weighting module is designed to fuse the aligned features. Finally, the $I_{k}$ is obtained by adding the predicted residue to the original (for video deblurring/denoising) or upsamples (for video SR) input image.

3.2. Feature Extraction

As illustrated in Fig. 4, we conduct feature extraction to transform a RGB frame $I_{k}$ to high-dimensional feature maps $F_{k}$. We first utilize two convolutions with strides of 2 to downsample the feature resolutions for video deblurring and denoising (highlighted in blue dotted box in Fig. 4) for computational efficiency, while keeping the same resolution for video SR (highlighted in green dotted box in Fig. 4). Then we utilize another two convolutions with stride of 2 to obtain the pyramid representations of the input frames. At
3.3. Temporal Alignment

Temporal alignment aims to align multiple neighboring features \( \{ F_{-N}, \ldots, F_1, F_1, \ldots, F_N \} \) to a reference \( F_0 \). Let \( A_k \) be the \( k \)-th temporal alignment between the neighboring frame \( F_k \) and the reference frame \( F_0 \), then we have

\[
A_k(F_k, F_0) = \hat{F}_k^0, \quad k \in \{-N, \ldots, -1, 1, \ldots, N\},
\]

where \( \hat{F}_k^0 \) is the aligned result.

3.3.1 Progressive Alignment

In order to facilitate the long-range alignment, some recent methods [4] adopt a progressive alignment strategy. For the alignment \( A_k \), they divide it into sequential sub-alignments \( \{ a_k, a_{k-1}, \ldots, a_1 \} \) to gradually align the feature \( F_k \) to the reference frame \( F_0 \). We use \( a_i \) to represent the sub-alignment from \( F_i \) to \( F_{i-1} \):

\[
a_i : F_i \rightarrow F_{i-1}.
\]

As illustrated in Fig. 2(b), all neighboring frames are processed through the chained sub-alignments, indicating that the latter sub-alignments strongly depend on the former predictions. Consequently, the error incurred by an intermediate inaccurate sub-alignment will be propagated and accumulated till the end, leading to inferior performance. To alleviate the issue of error accumulation and boost the restoration quality, we propose an iterative alignment algorithm to focus on improving the accuracy of each sub-alignment \( a_i \).

3.3.2 Iterative Alignment

Unlike the progressive alignment that conducts each sub-alignment only once, our algorithm iteratively refines the sub-alignments based on the previous estimation. As shown in Fig. 2(c), we start from the alignment \( A_1 \), which only contains the sub-alignment \( a_1 : F_1 \rightarrow F_0 \), described as:

\[
A_1 : a_1(F_1, F_0, t = 1) \Rightarrow \hat{F}_1^0, h_1^1,
\]

where \( F_k^{-1} \) refers to the aligned result of sub-alignment \( a_i \) in \( A_k \). And \( h_i^1 \) represents the estimated motion field of the sub-alignment \( a_i \) after being refined \( t \) times.

After that, we consider the next alignment \( A_2 \) by sequentially performing two sub-alignments \( \{ a_2, a_1 \} \):

\[
A_2 : \begin{cases} 
 a_2(F_2, F_1, t = 1) \Rightarrow \hat{F}_2^1, h_1^2, \\
 a_1(\hat{F}_2^1, F_0, h_1^1, t = 2) \Rightarrow \hat{F}_2^0, h_1^2.
\end{cases}
\]

For the sub-alignment \( a_i \) in \( A_2 \), it has already been carried out in \( A_1 \) once. Thus we take the pre-estimated motion field \( h_1^1 \) of \( a_1 \) in \( A_1 \) as the initialization and conduct a refinement, formulated as an iterative optimization.

For the subsequent alignment \( A_3 \), two sub-alignments \( \{ a_2, a_1 \} \) will be refined as:

\[
A_3 : \begin{cases} 
 a_3(F_3, F_2, t = 1) \Rightarrow \hat{F}_3^2, h_2^3, \\
 a_2(\hat{F}_3^2, F_1, h_2^3, t = 2) \Rightarrow \hat{F}_3^1, h_2^3, \\
 a_1(\hat{F}_3^1, F_0, h_2^3, t = 3) \Rightarrow \hat{F}_3^0, h_2^3.
\end{cases}
\]

It can be concluded that, apart from the first sub-alignment \( a_k \) in \( A_k \), all other sub-alignments are optimized at least twice. There are two merits: \( (i) \) The sub-alignments will be more accurate through our iterative refinements. \( (ii) \) The sub-alignments not only rely on the pre-aligned features but also the pre-estimated motion field, making it more reliable.

To verify our claim, we evaluate our algorithm together with recent video SR models [2, 3, 14, 28] on REDS4 [20] and REDS4-Fast. As shown in Fig. 5, our model achieves the best performance over the competing methods. Particularly, our method brings about significant improvement in the context of large motion, demonstrating the effectiveness of our IAM in the long-range alignment.
3.3.3 Sub-Alignment Unit

In Sec. 3.3.2, we describe the iterative alignment algorithm in detail. It is observed that for 2N neighboring frames, our method requires N(N + 1) sub-alignments. In contrast, the independent and progressive alignment schemes only need 2N (sub-)alignments. So it is critical to design a simple sub-alignment unit for computational efficiency. To this end, two improvements have been proposed. (i) While the previous methods [3, 28] typically adopt a pyramid alignment scheme that performs multiple-scale processing in the alignment phase, we adopt an early multi-scale fusion strategy in the feature extraction phase so that our IAM only performs single-scale alignments. (ii) We develop a lightweight sub-alignment unit with much fewer parameters than other methods [2, 28]. Specifically, we use a compact structure of residual blocks to reduce computational overhead (see details in supplementary materials.).

Fig. 6 shows the structure of our sub-alignment unit. Taking the i-th sub-alignment ai of Ak for example, we first utilize two convolutions followed by ReLU activation to estimate the initialized motion field hi from the concatenation of source feature Fk and target feature Fk−1. After that, there are two cases for the prediction hi+1,i of ai:

\[
hi+1,i = \begin{cases} 
  h^c_i, & i = k, \\
  \theta(h^c_i, h^{k-1}_i), & \text{others}.
\end{cases}
\]

If ai is the first sub-alignment of Ak (i = k), then no historical prediction can be reused to refine the ai. As a result, we simply set hi as the estimated motion field of ai. Otherwise, we will take the last estimation hi−1 as the input and utilize a single convolution followed by two residual blocks (dubbed as θ) to refine the prediction. Finally, we adopt a deformable convolution [6] to adaptively sample contents from the source feature Fk to adaptively sample contents from the source feature Fk:

\[
\hat{F}^{i-1}_k = \text{DConv}(\hat{F}^{i}_k, F_{k-1}, h^{k+1-i}_i)
\]

Specially, if ai is the first sub-alignment in Ak (i = k), Eq. 7 can be written as:

\[
\hat{F}_k^{k-1} = \text{DConv}(F_k, F_{k-1}, h^1)
\]

The sub-alignment unit is shared for all sub-alignments, largely reducing the number of learnable parameters.

3.4. Adaptive Re-weighting

Although the temporal alignment module performs motion compensation for neighboring frames, it remains vital to fuse them in an effective way. Recently, convolution-based attention mechanism becomes popular to aggregate multi-frame information [9, 13, 18, 31]. By contrast, we present a non-parametric re-weighting module to explicitly evaluate the spatially-adaptive importance of aligned frames from two perspectives. First, we evaluate the accuracy of aligned frames with respect to the reference frame. Second, we measure the consistency of aligned neighboring frames. Fig. 7 describes the pipeline of our re-weighting module.

**Accuracy-Based Re-weighting.** As shown in Fig. 7(a), we measure the accuracy of aligned frames. For the reference frame F0, the feature vector at position (x, y) is denoted as v0, i.e., F0(x, y) = v0. We find its corresponding 3 × 3 patch centered at the same position in the k-th aligned frame Fk. For each feature vector on this patch, we calculate its cosine similarity (normalized inner product) with respect to v0 as:

\[
S^x,y_k(\Delta x, \Delta y) = \frac{F_0^0(x + \Delta x, y + \Delta y)}{\|F_0^0(x + \Delta x, y + \Delta y)\|_2} \odot v_0 \quad \text{with } \|v_0\|_2 = 1.
\]

where \(S^x,y_k\) is the 3 × 3 similarity map at position (x, y) and \(\odot\) represents the inner product. \((x + \Delta x, y + \Delta y)\) is the coordinate of feature vector where \(\Delta x, \Delta y \in \{-1, 0, 1\}\). Then a Softmax function is applied to \(S^x,y_k\) in the spatial dimension, yielding the pixel-wise weights as:

\[
W^x,y_k = \text{Softmax}(S^x,y_k).
\]

Then \(W^x,y_k\) is used to fuse feature vectors on the 3 × 3 patch and the re-weighted result \(F^0_k(x, y)\) is obtained as:

\[
F^0_k(x, y) = \sum_{\Delta x, \Delta y} W^x,y_k(\Delta x, \Delta y) \odot \hat{F}^0_k(x + \Delta x, y + \Delta y).
\]

where \(\odot\) denotes the Hadamard product.
Table 1. The training and network configurations in different video restoration tasks.

| Task             | Video SR | Video Deblurring | Video Denoising |
|------------------|----------|------------------|-----------------|
| Configuration    | $M(128), B(40)$ | $M(128), B(10)$ | $M(64), B(10)$ |
| GPUs             | 6        | 6                | 2               |
| Patch Reso.      | $448 \times 448$ | $128 \times 128$ | $128 \times 128$ |
| nFrames          | 5(7)     | 5                | 5               |
| Mini-Batch       | 4        | 4                | 16              |

Consistency-Based Re-weighting. We first calculate the average of aligned neighboring frames yielding $\hat{F}_k^0$, as illustrated in Fig. 7(b). For the $k$-th aligned frame $F_k^0$, we evaluate its consistency with other aligned frames as

$$C_k = \exp(\alpha \cdot \|\hat{F}_k^0 - \hat{F}_k^0\|_2^2),$$

(12)

where $\alpha$ is set to $-1$ in our experiments. It is noted that $C_k$ maintains the same shape as $F_k^0$.

Finally, we multiply the accuracy-based re-weighted feature $\hat{F}_k^0$ to the consistency map $C_k$ and obtain the result:

$$\hat{F}_k = \hat{F}_k^0 \odot C_k.$$ 

(13)

The refined aligned feature $\hat{F}_k$ is passed to the reconstruction module for high-quality image regression (see Fig. 3).

4. Experiments

4.1. Implementation and Training Details

Configuration. As shown in Fig. 3, our network consists of four modules: feature extraction, alignment, re-weighting, and reconstruction. The feature extraction module in Sec. 3.2 contains 5 residual blocks for all tasks. Table 1 shows other detailed configurations, where $M$ is the number of feature channels in the network and $B$ is the number of residual blocks in the reconstruction module.

Training. We show the training settings in Table 1. We use 2-6 NVIDIA GeForce RTX 2080 Ti GPUs to train our models for 900K iterations for all three video restoration tasks. We adopt random vertical or horizontal flipping or 90° rotation for data augmentation. The initial learning rate is set to $5 \times 10^{-4}$ and a cosine decay strategy is employed.

4.2. Datasets and Metrics

Video Super-Resolution. REDS [20] and Vimeo-90K [33] are two widely used datasets in Video SR. Vimeo-90K contains 64,612 training and 7,840 testing 7-frame sequences with resolution $448 \times 256$. The testing set is denoted as Vimeo-90K-T. In REDS, there are 266 training and 4 testing video sequences. Each sequence consists of 100 consecutive frames with resolution $1280 \times 720$. Following [28], we denote the testing set as REDS4. Apart from these two testing datasets, we also give the quantitative results on Vid4 [16], which consists of 4 video clips.

Video Deblurring. We utilize the video deblurring dataset [24] (short for VDB) to train and evaluate our models. There are a total of 61 training and 10 testing video pairs. Each pair contains a blurry and sharp videos. The testing subset is marked as VDB-T.

Video Denoising. In this task, we aim to remove Gaussian white noises with known noise levels ($\sigma$). Our model is trained on DAIVS [11], which contains 87 training and 30 testing $540p$ videos. Set8 [25] is also adopted for testing. Following [25], we keep a maximum of 85 frames for all training and testing sequences for a fair comparison.

Metrics. PSNR and SSIM [29] are used to evaluate the performance of our models in the video SR and deblurring tasks. For video denoising, following the existing methods [5, 25], only PSNR is reported.

4.3. Analysis

In this section, we perform a comprehensive analysis of our method. We abbreviate the iterative alignment module as IAM and the adaptive re-weighting as ARW for clarity.

IAM and ARW. To evaluate the performance of the proposed IAM and ARW designs, we perform a quantitative comparison in Table 2. Starting from a baseline without these designs, we incrementally add the iterative alignment module (IAM) and adaptive re-weighting (ARW). As illus-
Table 3. Ablation study on different IAM and ARW settings for video SR. The running time of each model is also reported with an input size of $7 \times 112 \times 64$.

|          | PSNR (dB) | SSIM    | Runtime (ms) |
|----------|-----------|---------|--------------|
| Baseline | 37.36     | 0.9468  | 153          |
| IAM      | R2        | 37.68 (+0.32) | 0.9487  | 166 |
|          | R3        | 37.72 (+0.36) | 0.9490  | 169 |
| ARW      | Acc. Con. | 37.39 (+0.03) | 0.9469  | 154 |
|          | Con.      | 37.43 (+0.07) | 0.9469  | 158 |
| Full     |           | 37.84 (+0.48) | 0.9498  | 170 |

(4.4) Comparison with State-of-The-Art Methods

We compare our method with state-of-the-art approaches quantitatively and qualitatively in the video SR, video deblurring and video denoising tasks.

**Video Super-resolution.** Table 5 and Table 9 exhibit the quantitative results of our method and existing video SR methods [2–4, 7, 10, 14, 28, 33] on Vimeo-90K-T [33], REDS4 [20] and Vid4 [16] datasets. Compared to the representative independent [28] and progressive [3] alignment methods, our method obtains superior Y-channel PSNR per-
Table 5. Vimeo-90K-T [33] results in video SR. The PSNR(dB)/SSIM results are obtained under the ×4 setting. Numbers in red and blue refer to the best and second-best results. ‘†’ means BasicVSR++ uses an additional REDS dataset for pretraining.

Table 6. Set8 [25] and DAVIS [11] results in video denoising. PSNR(dB) results are reported.

Video Denoising. Following previous methods [1, 5, 25], we adopt Set8 [25] and DAVIS [11] as our benchmarks in the video denoising task. The quantitative results are reported in Table 6. Our model achieves the best results under most noise levels. Especially, compared with the second-best approaches, our method largely improves the PSNR by 0.37dB and 0.65dB under the noise level σ = 50 on Set8 and DAVIS, respectively. Fig. 11 presents some qualitative results. It is observed that our method restores richer and clearer textures compared with other approaches.

Video Deblurring. We compare our method with several recent video deblurring approaches [8, 21, 24, 28, 37] on VDB-T [24]. As illustrated in Table 1, two models with different sizes (10 or 40 residual blocks) are developed, denoted as “Ours-M” and “Ours”. From Table 7, compared to the second best ARV [21], we see that our model achieves 0.12dB and 0.013 improvement on PSNR and SSIM, respectively. Some visual examples illustrated in Fig. 12 also demonstrate that our model is able to handle soame challenging cases with complex motion blur.

4.5. Limitation

In this work, the proposed designs are mainly for improving the accuracy of the long-range alignment. There remains plenty of room to optimize the modeling of subtle motion. Besides, further improving the efficiency of the
alignment pipeline is also our future target.

5. Conclusion

In this paper, we propose a simple yet effective iterative alignment algorithm (IAM) and an efficient adaptive reweighting strategy (ARW) to better utilize multi-frame information. The quantitative and qualitative results of three video restoration tasks illustrate the effectiveness of our method. Besides, we show that our method is general to be deployed in existing video processing systems to further improve their performance. We will explore more video-based tasks in the future. The code will be publicly available to promote the development of the community.

References

[1] Pablo Arias and Jean-Michel Morel. Video denoising via empirical bayesian estimation of space-time patches. *Journal of Mathematical Imaging and Vision*, 60(1):70–93, 2018.

[2] Jiezhang Cao, Yawei Li, Kai Zhang, and Luc Van Gool. Video super-resolution transformer. *arXiv*, 2021.

[3] Kelvin CK Chan, Xintao Wang, Ke Yu, Chao Dong, and Chen Change Loy. Basicvrs: The search for essential components in video super-resolution and beyond. *arXiv preprint arXiv:2012.02181*, 2020.

[4] Kelvin CK Chan, Shangchen Zhou, Xiangyu Xu, and Chen Change Loy. Basicvrs++: Improving video super-resolution with enhanced propagation and alignment. *arXiv preprint arXiv:2104.13371*, 2021.

[5] Xinyuan Chen, Li Song, and Xiaokang Yang. Deep rns for video denoising. In *Applications of Digital Image Processing XXXIX*, volume 9971, page 99711T. International Society for Optics and Photonics, 2016.

[6] Jifeng Dai, Haozhi Qi, Yuwen Xiong, Yi Li, Guodong Zhang, Han Hu, and Yichen Wei. Deformable convolutional networks. In *Proceedings of the IEEE international conference on computer vision*, pages 764–773, 2017.

[7] Muhammad Haris, Gregory Shakharovich, and Norimichi Ukita. Recurrent back-projection network for video super-resolution. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3897–3906, 2019.

[8] Tae Hyun Kim, Kyoung Mu Lee, Bernhard Scholkopf, and Michael Hirsch. Online video deblurring via dynamic temporal blending network. In *CVPR*, pages 4038–4047, 2017.

[9] Takashi Isebe, Xu Jia, Shuhang Gu, Songjiang Li, Shengjin Wang, and Qi Tian. Video super-resolution with recurrent structure-detail network. In *European Conference on Computer Vision*, pages 645–660. Springer, 2020.

[10] Younghyun Jo, Seoung Wug Oh, Jaeyeon Kang, and Seon Joo Kim. Deep video super-resolution network using dynamic upsampling filters without explicit motion compensation. In *CVPR*, pages 3224–3232, 2018.

[11] Anna Khoreva, Anna Rohrbach, and Bernt Schiele. Video object segmentation with language referring expressions. In *Asian Conference on Computer Vision*, pages 123–141. Springer, 2018.

[12] Dongxu Li, Chenchen Xu, Kaihao Zhang, Xin Yu, Yiran Zhong, Wenqi Ren, Hanna Suominen, and Hongdong Li. Arvo: Learning all-range volumetric correspondence for video deblurring. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7721–7731, 2021.

[13] Feng Li, Huihui Bai, and Yao Zhao. Learning a deep dual attention network for video super-resolution. *IEEE Transactions on Image Processing*, 29:4474–4488, 2020.

[14] Wenbo Li, Xin Tao, Taian Guo, Lu Qi, Jiangbo Lu, and Jiaya Jia. Mucan: Multi-correspondence aggregation network for video super-resolution. In *ECCV*, pages 335–351. Springer, 2020.

[15] Wenbo Li, Kun Zhou, Lu Qi, Nianjuan Jiang, Jiangbo Lu, and Jiaya Jia. Lapar: Linearly-assembled pixel-adaptive regression network for single image super-resolution and beyond. 33, 2020.

[16] Ce Liu and Deqing Sun. On bayesian adaptive video super-resolution. *IEEE transactions on pattern analysis and machine intelligence*, 36(2):346–360, 2013.

[17] 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.

[18] Zhi-Song Liu, Li-Wen Wang, Chu-Tak Li, Wan-Chi Siu, and Yui-Lam Chan. Image super-resolution via attention based back projection networks. In *2019 IEEE/CVF International Conference on Computer Vision Workshop (ICCVW)*, pages 3517–3525. IEEE, 2019.

[19] Matteo Maggioni, Giacomo Boracchi, Alessandro Foi, and Karen Egiazarian. Video denoising, deblocking, and enhancement through separable 4-d nonlocal spatiotemporal transforms. *IEEE Transactions on Image Processing*, 21(9):3952–3966, 2012.

[20] Seunghun Nah, Seugyong Baik, Seokil Hong, Gyeongsik Moon, Sanghyun Son, Radu Timofte, and Kyoung Mu Lee. Niire 2019 challenge on video deblurring and super-resolution: Dataset and study. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 0–0, 2019.

[21] Jinshen Pan, Haoran Bai, and Jinhui Tang. Cascaded deep video deblurring using temporal sharpness prior. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3043–3051, 2020.

[22] Anurag Ranjan and Michael J Black. Optical flow estimation using a spatial pyramid network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4161–4170, 2017.

[23] Yaniv Romano, John Isidoro, and Peyman Milanfar. Raisr: rapid and accurate image super resolution. *IEEE Transactions on Computational Imaging*, 3(1):110–125, 2016.
Appendix A. Residual Block

As illustrated in Table 8, our residual block is comprised of two convolutions, where the first convolution is followed by ReLU activation. In previous works [2, 3, 28], the channel numbers typically keep identical within the residual block. In contrast, we set the channel dimension of hidden representations ($M_1$) no more than 64 to reduce the parameters of our network and speed up the training and inference phases:

$$M_1 = \max(M//2, 64),$$

Table 8. The structure of our residual block. $M$ refers to the channel number of input.

| Layer | Operation |
|-------|------------|
| Layer 1 | $\text{Conv}(M, M_1, 3, 1) + \text{ReLU}$ |
| Layer 2 | $\text{Conv}(M_1, M, 3, 1) \Rightarrow y$ |
| Output | $x+y$ |

Appendix B. More Results

B.1. Temporal Consistency

We compare the temporal consistency of our method with several state-of-art video SR approaches [3, 10, 14, 28]. The visual results are illustrated in Figure 13. It is observed that other methods fail to restore the consistent textures clearly. While our method empowered with iterative alignment and two adaptively reweighting strategies is able to generate realistic image contents that are closest to the ground-truth.

B.2. Comparison with State-of-the-art

In Table 9 and 10, we give the detailed comparison with several state-of-the-art video SR approaches [2, 2, 3, 28] on REDS4 [20] and Vid4 [16]. The PSNR and SSIM of each video sequence are reported. For most video clips of both validation sets, our model consistently achieves the best performance. Moreover, we provide extensive qualitative comparison on UDM10 [34], Vid4 [16], Vimeo-90K-T [33], and REDS4 [20] for video SR (in Figure 14, 15), VDBT [24] for video deblurring (in Figure 16) and Set8 [25], DAVIS [11] for video denoising (in Figure 17). All the qualitative results demonstrate that our method has the capacity to handle various challenging cases in these three video restoration tasks.
Table 9. Quantitative comparison on REDS4 [20] benchmark under ×4 setting for video super-resolution. Numbers in red and blue refer to the best and second-best results. All the results are evaluated in the RGB channel. '*' indicates the results are from [2].

| Clip Name    | Bicubic | RCAN [35] | TOF [33] | DUF [10] | EDVR [28] | MuCAN [14] | "BasicVSR [3]" | "IconVSR [3]" | VSR-T [2] | Ours |
|--------------|---------|-----------|-----------|-----------|-----------|-------------|----------------|---------------|-----------|------|
| Calendar (Y) | 20.39/0.5720 | 24.04/0.8110 | 24.05/0.8147 | - | - | - | 24.08/0.8125 | 24.65/0.8270 |
| City (Y)     | 25.16/0.6028 | 28.27/0.8313 | 28.00/0.8122 | - | - | - | 27.94/0.8107 | 29.92/0.8428 |
| Foliage (Y)  | 23.47/0.5666 | 26.41/0.7709 | 26.34/0.7635 | - | - | - | 26.53/0.7635 | 26.41/0.7652 |
| Walk (Y)     | 26.10/0.7974 | 30.60/0.9141 | 31.02/0.9152 | - | - | - | 31.10/0.9163 | 31.15/0.9167 |
| Average (Y)  | 23.78/0.6347 | 27.33/0.8318 | 27.35/0.8264 | 27.26/0.8215 | 27.24/0.8251 | 27.39/0.8279 | 27.36/0.8258 | 27.90/0.8380 |
| Average (RGB)| 22.37/0.5978 | 25.79/0.8136 | 25.83/0.8077 | - | - | - | 26.57/0.8235 | 26.57/0.8235 |

Table 10. Quantitative comparison on Vid4 [16] under ×4 setting for video super-resolution. We report the PSNR (dB)/SSIM results on both the RGB and the Y channel. Numbers in red and blue refer to the best and second-best results.

```
Table | Clip Name       | Bicubic | DUF [10] | EDVR [28] | MuCAN [14] | BasicVSR [3] | IconVSR [3] | VSR-T [2] | Ours  |
-------|-----------------|---------|----------|-----------|-------------|--------------|-------------|-----------|-------|
Calendar (Y) | 20.39/0.5720 | 24.04/0.8110 | 24.05/0.8147 | - | - | - | 24.08/0.8125 | 24.65/0.8270 |
City (Y) | 25.16/0.6028 | 28.27/0.8313 | 28.00/0.8122 | - | - | - | 27.94/0.8107 | 29.92/0.8428 |
Foliage (Y) | 23.47/0.5666 | 26.41/0.7709 | 26.34/0.7635 | - | - | - | 26.53/0.7635 | 26.41/0.7652 |
Walk (Y) | 26.10/0.7974 | 30.60/0.9141 | 31.02/0.9152 | - | - | - | 31.10/0.9163 | 31.15/0.9167 |
Average (Y) | 23.78/0.6347 | 27.33/0.8318 | 27.35/0.8264 | 27.26/0.8215 | 27.24/0.8251 | 27.39/0.8279 | 27.36/0.8258 | 27.90/0.8380 |
Average (RGB) | 22.37/0.5978 | 25.79/0.8136 | 25.83/0.8077 | - | - | - | 26.57/0.8235 | 26.57/0.8235 |
```

**Figure 13.** Visualization of temporal consistency on Vid4 [16].

**B.3. Video Results**

We also provide three videos for visual inspection.

**“city.mp4”**. This video illustrates the visual comparison between bicubic and our method on a Vid4 clip for video super-resolution. It can be observed that our method retrieves much clear image details (e.g., the finer structure of buildings).

**“IMG0030.mp4”**. This video demonstrates the visual results of our method on a testing sequence of VDB-T [24] for the video deblurring task. The blurry input and the generated frames are shown in it.

**“motorbike.mp4”**. This video shows the restoration results on a sequence of Set8 [25] for video denoising.
Figure 14. Qualitative comparison on UDM10 [34] and Vid4 [16] for video SR.
| Bicubic Patch | DUF | EDVR | MuCAN | Ours | GT |
|---------------|-----|------|-------|------|----|
| ![Image](image1.png) | ![Image](image2.png) | ![Image](image3.png) | ![Image](image4.png) | ![Image](image5.png) | ![Image](image6.png) |
| ![Image](image7.png) | ![Image](image8.png) | ![Image](image9.png) | ![Image](image10.png) | ![Image](image11.png) | ![Image](image12.png) |
| ![Image](image13.png) | ![Image](image14.png) | ![Image](image15.png) | ![Image](image16.png) | ![Image](image17.png) | ![Image](image18.png) |
| ![Image](image19.png) | ![Image](image20.png) | ![Image](image21.png) | ![Image](image22.png) | ![Image](image23.png) | ![Image](image24.png) |

Figure 15. Qualitative comparison on Vimeo-90K-T [33] and REDS4 [20] for video SR.
Figure 16. Qualitative comparison on VDB-T [24] for video deblurring.
Figure 17. Qualitative comparison on Set8 [25], DAVIS [11] for video denoising. The values beneath images represent the PSNR (dB).