Attention-based Fusion for Video Super-Resolution

Vassili Chesterkine   Paul Theron
Massachusetts Institute of Technology
{vchester, paulth}@mit.edu

Abstract

Video Super-Resolution (VSR), the process of enhancing low-resolution videos to high-resolution equivalents, primarily capitalizes on neighboring frame alignment to extract additional information. State-of-the-art methods like BasicVSR [1], while powerful, rely on Recurrent Neural Networks (RNNs) and require the input of an entire video to the model, leading to high computational intensity. This study presents a resource-efficient approach that incorporates temporal information hierarchically, designed to yield substantial performance from a reduced number of frames. Our methodology partitions the input sequence into distinct frame rate groups, each offering complementary information to recover missing details in the reference frame. Utilizing BasicVSR as our core feature extractor and alignment method, we integrate these groups through an inter group attention-based module. The proposed method not only demonstrates competitive performance against state-of-the-art models but also showcases potential for further refinement.
Code and experiments are available here: `https://github.com/paultheron-X/6.8300-Computer-Vision`

1. Introduction & Related Work

Video super-resolution (VSR) is the task of constructing high-resolution (HR) videos from low-resolution (LR) videos. With the increasing demand for high-quality video content, video super-resolution has become a very active topic in recent years [14]. Applications can be found in diverse fields, from medical imagery to entertainment content, to surveillance systems. VSR is somewhat more challenging than classical image super-resolution as it entails identifying and combining complementary information from different images. In fact, information from neighbouring frames can be aggregated to obtain a more accurate estimate of the super-resolved frame.

While traditional approaches used statistical models to estimate motion and consequently align neighbouring frames, the recent years have seen various deep-learning approaches being explored and leading to promising results. Two main types of approaches have emerged. In the first, batches of LR frames are fed into a model, and this sliding-window scheme is done over the entire video [13, 15]. This effectively transforms the problem of VSR into a large number of separate multi-frame SR problems. In the second, Recurrent neural networks (RNNs) are used to leverage the temporal information of the full video [1, 7]. Most recent state-of-the-art models fall into the latter category.

Most VSR methods can be decomposed into the following steps: *motion estimation* using optical flow, *alignment* of the neighbouring frames, followed by *aggregation*, and finally *upsampling* where the HR frames are reconstructed. State-of-the-art models such as BasicVSR [1] or BasicVSR++ [2] have successfully been fine-tuning a pre-trained version of SpyNet [9] to perform optical flow estimation. The latter is complicated due to large motions or invalidity of the brightness constancy assumption. Initial estimation inaccuracies can jeopardize the entire reconstruction performance, leading to noticeable artifacts in the resulting images [13]. Alignment is also critical since the Convolutional Neural Networks used to process images (CNN) can only effectively utilize information that is distributed locally in the inputs. In fact, significant progress in VSR such as EDVR [15], TDAN [13] or BasicVSR [1], have often originated from innovative alignment methods.

However, such models rely heavily on common assumptions such as brightness constancy, and their performance drops when videos contain occlusion, or large and complex motion. An interesting approach tackle this limitation is proposed in [6], where subsets of frames defined by different frame rates are used to extract complementary information from the input video. Additionally, under limited computing power, sliding-window approaches using pretrained models are appropriate, as they alleviate the computational intensity of the task. In this work, we first evaluate the performance of using more modern optical flow estimators. Then, we propose an original method based on the optical flow and alignment modules from BasicVSR, and temporal groups of frames to mitigate challenges of current models.

2. Methodology

2.1. Model

In our proposed model, we leverage information extracted from different temporal sub-groups of frames. Based on the sliding-window framework, we define 3 groups identified by a specific frame rate, each of which provides complementary information on the initial video. For instance, given a rolling window size of 3, and the neighbouring low-resolution frames $\{I_0, \ldots, I_N\}$ around I_i, we define $\mathcal{G}_1, \mathcal{G}_2, \mathcal{G}_3$ groups of frame rate $\Delta = 1, 2, 3$ as:

$$\begin{aligned} \mathcal{G}_1 &= \{I_{i-1},\ I_i,\ I_{i+1},\} \\ \mathcal{G}_2 &= \{I_{i-2},\ I_i,\ I_{i+2}\} \\ \mathcal{G}_3 &= \{I_{i-3},\ I_i,\ I_{i+3}\} \end{aligned}$$

2.1.1 Intra-group flow estimation and compensation

We utilize BasicVSR's bi-directional propagation and feature alignment modules as a feature extractor for our model. Specifically, optical flow is estimated between frames of a group, and the resulting features are independently propagated forward and backward along the sequence of frame to perform alignment of the frames based on the extracted features. Each frame rate group is fed as a batch into this module. Doing so, we obtain 64 feature maps for each group, which are then used for aggregation.

In this module, we used pretrained BasicVSR and SpyNet modules, leveraging their learned representations and thereby significantly reducing the number of trainable parameters.

2.1.2 Attention inter-group fusion

In order to merge the obtained features for each frame rate group, we introduce a temporal attention module, based on previous work [6]. Since each group contains different information, attention is helpful as it enables the model to focus on different features over time and consequently better integrate them. For instance, features from groups of fast frame rates ($\Delta = 1$) may contain information on fine details since all neighbouring frames are very similar. Conversely, slower frame rate groups ($\Delta = 5$) may contain alternative information such as larger motion or details absent in nearby frames because of occlusions. In that regard, the attention mechanism enables better aggregation of complementary information from rate groups.

Single-head Attention. For each frame rate group, the output of BasicVSR is fed into a 3×3 convolutional layer, which produces a new set of feature maps F_n, where information from different initial feature maps is combined. These F_n are concatenated and a softmax function is applied along this new dimension to create an attention mask for each group and channel, as formulated below:

$$M_n(x,y)_j = \frac{e^{F_n(x,y)_j}}{\sum_{i=1}^{3} e^{F_i(x,y)_j}} \tag{1}$$

with n, j referring to the frame rate group and channel respectively. The overall mechanism is described in Fig. 2.

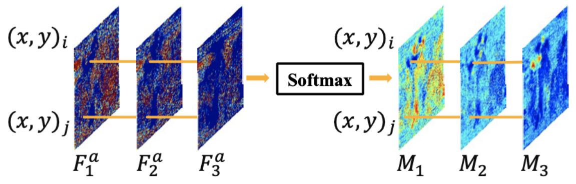

Figure 2. Computation of Group Attention Maps (from [6])

Let us then denote $\tilde{F}_n = M_n \odot F_n$ the attention-weighted feature map, obtained as the element-wise product of the feature map F_n and the attention map M_n. This effectively highlights important information from the BasicVSR feature maps, independently in each group.

The inter-group fusion module's aim is to compile information from all temporal groups and generate a high-resolution residual map. To optimally leverage attention-weighted features across temporal groups, we first concatenate these features along the temporal axis and then input them into a 3D dense block, inspired by [16]. Subsequently, a 2D dense block is utilized for additional fusion, as depicted in Fig. 3. The 3D unit shares the same structure as the 2D unit. A final convolution layer is used at the end of the 2D dense block to decrease the number of channels.

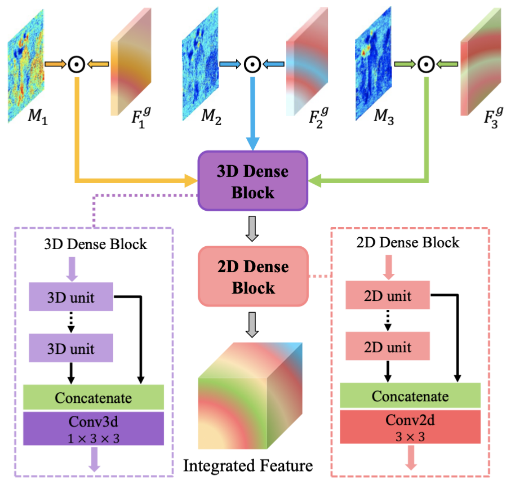

Figure 3. Inter-group fusion module structure (figure from [6])

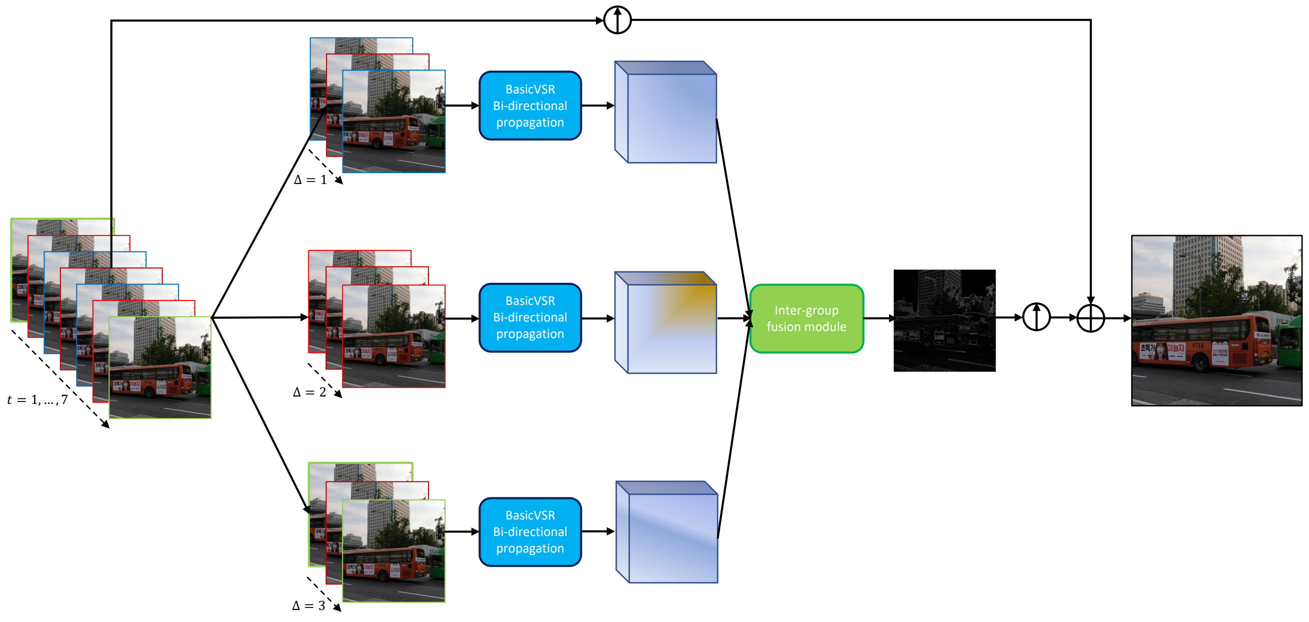
BasicVSR Bi-directional propagation
BasicVSR Bi-directional propagation
BasicVSR Bi-directional propagation
Inter-group fusion module
$t = 1, \ldots, 7$
$\Delta = 1$
$\Delta = 2$
$\Delta = 3$

Figure 1. Architecture of the proposed model

Multi-head Attention. The framework described above utilizes a single head attention, as only one attention map is computed. Recent studies in Transformers and NLP suggest that attention is more powerful when multi-headed.

We consequently implemented a multi-head inter-group module as an improvement to the single-head module. Using similar notations, we first use a 2D convolutional layer to project F_n onto F_n^h, where h indicates a particular attention head. This is illustrated in Fig. 4. By design choice, these specific projection layers are shared across all frame rate groups. For example, the projection layer associated to head h is common for all 3 groups. The main justification for this choice was the significant reduction of the number of trainable weights, making training less computationally-intensive. Additionally, this mechanically ensures that the corresponding projected space is coherent across all groups, and that subsequently computing attention on them is valid. Then, F_n^h is treated as in the single-head module, to compute an head-relative attention map, M_n^h, and a head-relative attention-weighted feature map, $\tilde{F}_n^h$. Those weighted feature maps are then concatenated along the channel dimension. Then, an additional 3D dense block is used, integrating information from all attention heads and compressing it into a dimension suitable for the 3D/2D dense blocks described above. Finally, we apply the usual inter-group fusion module, described in Fig. 3.

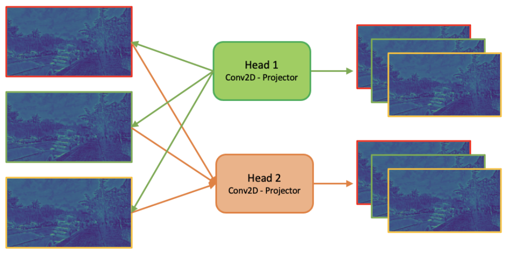

Figure 4. Multi-head projection - on the example of a 2 head scenario, the feature map from each group is projected onto a subspace per attention head, given a shared 2D conv among the different group.

2.2. Upsampling

The final module of our model uses the architecture of BasicVSR's upsampling module, which is composed of 2D convolutional layers – effectively upsampling the input by trading the number of channels for height and width – and PixelShuffle layers – a tensor operation that has been proven to be effective in super-resolution [10]. Because we apply it on top of the output of our custom inter-group fusion module, we simply use the pretrained weights for these layers as a warm start. This upsampling module is used to generate a high-resolution residual map.

Finally, the estimated high-resolution frame is obtained by adding the residual map output from the upsampling module and a bicubic upsampling of the input reference LR frame.

3. Experiments

We use the code and pre-trained weights for BasicVSR, as well as for SpyNet, provided by [1], as the baseline for our experiments. For the trainable parameters, we use the Adam optimizer with $\beta_1 = 0.9$ and $\beta_2 = 0.99$, as well as Cosine Annealing to decay the learning rate from 10^{-4} to 10^{-7}. The training loss is the Charbonnier loss [3]. We were able to access 2 virtual machines[1] (VM), equipped with an NVIDIA GeForce RTX 3090 GPU, as the size of the dataset (50Go+) and lengthy image loading from Google Drive rendered Google Colab sub-optimal.

3.1. Metrics

The quality of the reconstructed frames is evaluated by calculating the peak signal-to-noise ratio (PSNR) and Structural Similarity Index Measure (SSIM). The former metric measures the difference of pixels between the reconstructed image and the ground truth image. PSNR is defined as:

$$\text{PSNR} = 10 \cdot \log_{10}\left(\frac{\text{M}_I^2}{\frac{1}{N}\sum_{i=0}^{N-1}[I_i - \hat{I}_i]^2}\right) \qquad (2)$$

where M_I is the maximal pixel intensity – usually 255 – and the denominator is the mean squared error, with N the number of pixels in the image and $I, \hat{I}$ the ground truth and estimated pixel intensities respectively. The latter metric evaluates the similarity between 2 images based on luminance, contrast and structure. Note that while both metrics are good proxies to identify good reconstructions, they do not necessarily align to human perception. Given a sliding window of frames, we only evaluate these metrics on the middle frame.

3.2. Dataset

We consider the REDS dataset [8], widely-used in VSR tasks. This dataset contains 240 training clips, 30 validations clips and 30 testing clips. Each clip contains 100 consecutive frames of resolution 1280x720, and LR frames are obtained by applying bicubic downsampling with a factor of 4. Following previous studies [15], we conduct validation on REDSval4[2] and testing on REDS4[3], while the remaining training and validation clips are used for training.

[1]Provided by Ecole Polytechnique

[2]Clips 000, 001, 006 and 017 of REDS validation set

[3]Clips 000, 011, 015 and 020 of REDS training set

3.3. Optical flow benchmark

One of our initial experiments was a study of the optical flow estimation module. To better understand BasicVSR and its outputs, we conducted a small ablation study on optical flow extraction in the model, to understand its impact. BasicVSR and BasicVSR++ use an optical flow estimation computed by SpyNet [9], which is a relatively old and poorly performing model in today's standards. A natural question that emerges is whether VSR performance would be improved with a more recent optical flow model, whether this model needs to be fine-tuned with the general model or it can be used as a frozen expert.

Our experiments suggested that the difference in performance between a pipeline using a finetuned optical flow estimator and a pretrained one is relatively small. Further, we replaced SpyNet with more advanced pre-trained optical flow module, such as RAFT [12], PWC-Net [11] or FlowNet [5]. For conciseness, the full results are not included here, but for a 25 frames time window, results (in PSNR) range from 30.18 for RAFT to 31.11 for SpyNet. This could be because modern methods perform particularly well on high resolution frames, whereas the frames used here have a rather low resolution of 360x640. Alternatively, training these flow modules would have been costly and might have only brought a marginal performance gain in the VSR task. This experiment oriented us towards another approach, tackling the aggregation of features after extraction and alignment.

3.4. Model performance results

3.4.1 Quantitative Results

Below is a comparison of VSR performance on the REDS4 test set, between our proposed model and other models.

Method	Modality	Params (M)	PSNR
Bicubic	–	–	26.14
BasicVSR	5 frames	6.3	30.56
	7 frames	6.3	30.65
	11 frames	6.3	30.92
IconVSR	5 frames	8.7	30.81
Our Method	Single Head	6.8	30.74
	Multi Head	7.7	30.67
	Frame rates (1,3,5)	6.6	30.65

Table 1. Quantitative comparison of different approaches PSNR, under different experiment settings. Results are calculated on the RGB channel, and on REDS4 test set.

We re-implemented BasicVSR and IconVSR to ensure fair comparison, and display the obtained results in Table

1. It is important to note that these methods use an RNN-based architecture and therefore feed entire videos as input to the residual network. As the initial motivation for our proposed method was to maintain a reasonable usage of GPU memory, we instead chose to compute the reconstructions of HR frames using only sequences of 5, 7 or 11 input frames. Feeding such sequences to the models, the reconstruction performance is severely degraded in comparison to the results presented in the original study [1]. In a setting with frame rates of 1, 2 and 3, our model takes as input 3 sets of 5 frames. The union of groups 1 and 2 amounts to 7 unique frames, and the union of all groups to 11 unique frames, hence the 5, 7 and 11 frames settings in the table.

All of the results above can be compared to a baseline of bicubic upsampling, with no trainable parameter. We see that our method achieves better performance than BasicVSR in the 5 and 7 frames framework[4]. Interestingly, our single-headed approach is still better than the multi-headed one. This is most likely due to the lack of proper training. In fact, the single-head module has been trained on approximately 100 epochs, while the multi-head has only been trained on 40. Given the difference in the number of parameters, we believe that this reasonably explains the difference in performance, and that with equal training, the multi-head approach would outperform the single-head approach. While we initially defined temporal groups with frame rates of 1, 2 and 3, we also experimented with different frame rates, aiming to extract higher-level information from the videos. Again, we were not able to train this variation on an appropriate number of epochs, although the obtained results are promising.

3.4.2 Qualitative Results

Analysis of the multi-head attention heads yielded interesting observations, displayed in Fig. 5. These images were generated by the multi-head attention module from the same feature maps. In the first row, we see the projections that each attention head has made. It is interesting to notice how the first head (left) focuses on finer details, such as the pavement and the edges of objects, while the second blurs out the latter, and instead seems to produce a depth map of the picture. Examples of reconstructions can be found in Appendix A.

3.5. Discussion

While our results do not reach the level of BasicVSR and subsequent models, we should highlight that all of our experiments were run on 2 separate VMs, equipped with a single GPU, for up to 100 epochs. In comparison, the authors of BasicVSR used 8 parallelized V100 GPUs over more than 5,000 epochs. Due to limited time and resources, we were not able to fully investigate the architecture design and hyperparameters of our model. This would include but not be limited to more rigorous finetuning of the batch size, the number of gradient accumulation steps, the learning rate and weight decay, addition of convolution or batch normalization layers.

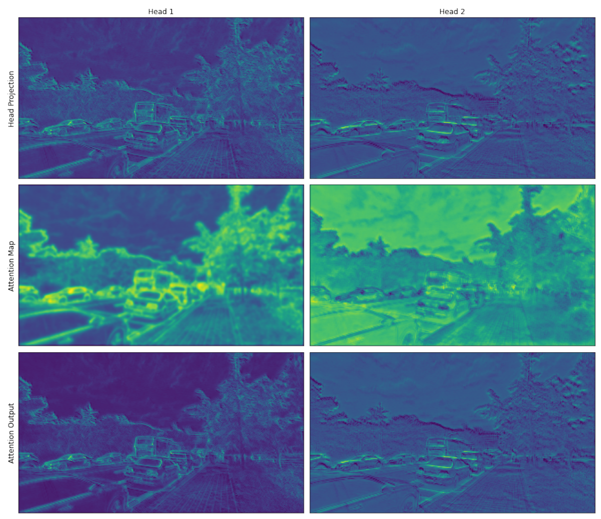

Figure 5. Projection, and attention maps of two different heads in a multi-head setting.

It is also important to note that some of the architecture design choices we made were arbitrary: multi-head attention weights are shared and we use 3D convolution layers to reduce the number of channels instead of a projection matrix as in the original attention method. These choices should be more carefully reviewed to attain optimal performance.

Future work could investigate recent additions such as Deformable Convolutions [4] for better alignment, potentially improving model performance in the downstream task. This approach, used in BasicVSR++ [2], has shown significant performance gains, indicating potential for further improvement.

4. Conclusion

In this study, we developed a novel deep neural network for Video Super Resolution (VSR), proficiently integrating temporal information in a hierarchical manner. The approach reorganizes input sequences into distinct frame rate groups, leveraging BasicVSR for feature extraction and alignment, and an attention-based inter-group module for aggregation. The result is high-quality frame reconstruction with preserved temporal consistency. Demonstrating competitive performance against established models, our technique opens new avenues for further refinement.

[4]For the sake of simplicity, we only use PSNR to compare our results in this report.

Member contributions

Both member contributed equally in the code, design of the model, benchmarking experiments. With regards to the report and slides, individual contributions are as follows:

- Vassili: Slides (content), Report: Introduction, Related Work, Discussion, Results
- Paul: Slides (figures), Report: Figures, Methodology, Results, Conclusion

Both member also participated to reviewing and improving these documents.

References

[1] Kelvin CK Chan, Xintao Wang, Ke Yu, Chao Dong, and Chen Change Loy. Basicvsr: The search for essential components in video super-resolution and beyond. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4947–4956, 2021. 1, 4, 5

[2] Kelvin CK Chan, Shangchen Zhou, Xiangyu Xu, and Chen Change Loy. Basicvsr++: Improving video super-resolution with enhanced propagation and alignment. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 5972–5981, 2022. 1, 5

[3] Pierre Charbonnier, Laure Blanc-Feraud, Gilles Aubert, and Michel Barlaud. Two deterministic half-quadratic regularization algorithms for computed imaging. In *Proceedings of 1st international conference on image processing*, volume 2, pages 168–172. IEEE, 1994. 4

[4] 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. 5

[5] Alexey Dosovitskiy, Philipp Fischer, Eddy Ilg, Philip Hausser, Caner Hazirbas, Vladimir Golkov, Patrick Van Der Smagt, Daniel Cremers, and Thomas Brox. Flownet: Learning optical flow with convolutional networks. In *Proceedings of the IEEE international conference on computer vision*, pages 2758–2766, 2015. 4

[6] Takashi Isobe, Songjiang Li, Xu Jia, Shanxin Yuan, Gregory Slabaugh, Chunjing Xu, Ya-Li Li, Shengjin Wang, and Qi Tian. Video super-resolution with temporal group attention. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 8008–8017, 2020. 1, 2

[7] Yanghui Li, Hong Zhu, Lixin He, Dong Wang, Jing Shi, and Jing Wang. Video super-resolution with regional focus for recurrent network. *Applied Sciences*, 13(1):526, 2023. 1

[8] Seungjun Nah, Sungyong Baik, Seokil Hong, Gyeongsik Moon, Sanghyun Son, Radu Timofte, and Kyoung Mu Lee. Ntire 2019 challenge on video deblurring and super-resolution: Dataset and study. In *CVPR Workshops*, June 2019. 4

[9] 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. 1, 4

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

[11] Deqing Sun, Xiaodong Yang, Ming-Yu Liu, and Jan Kautz. Pwc-net: Cnns for optical flow using pyramid, warping, and cost volume. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 8934–8943, 2018. 4

[12] Zachary Teed and Jia Deng. Raft: Recurrent all-pairs field transforms for optical flow. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part II 16*, pages 402–419. Springer, 2020. 4

[13] Yapeng Tian, Yulun Zhang, Yun Fu, and Chenliang Xu. Tdan: Temporally-deformable alignment network for video super-resolution. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 3360–3369, 2020. 1

[14] Zhigang Tu, Hongyan Li, Wei Xie, Yuanzhong Liu, Shifu Zhang, Baoxin Li, and Junsong Yuan. Optical flow for video super-resolution: A survey, 2022. 1

[15] Xintao Wang, Kelvin CK Chan, Ke Yu, Chao Dong, and Chen Change Loy. Edvr: Video restoration with enhanced deformable convolutional networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 0–0, 2019. 1, 4

[16] Yulun Zhang, Yapeng Tian, Yu Kong, Bineng Zhong, and Yun Fu. Residual dense network for image super-resolution. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2472–2481, 2018. 2

A. Reconstructed Images

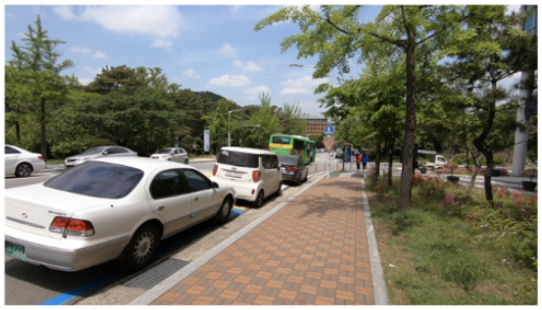

Figure 6. Ground Truth Image

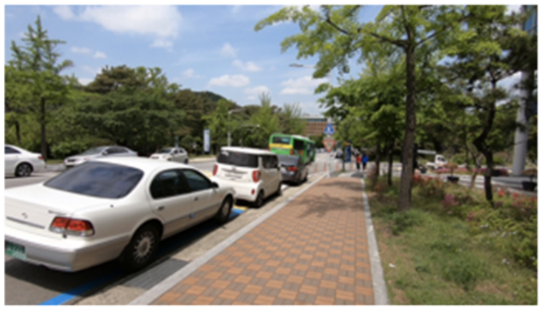

Figure 7. Baseline: LR Upsampled image

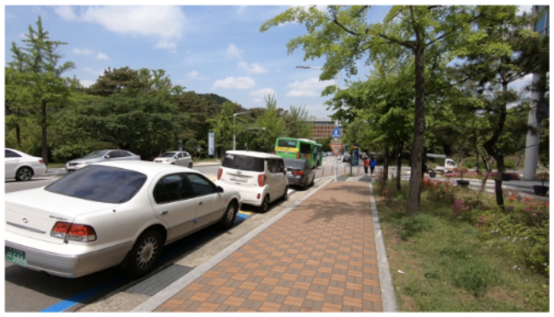

Figure 8. Reconstructed Image