Alignment-aware Patch-level Routing for Dynamic Video Frame Interpolation

Ban Chen[1]
ban.chen@samsung.com
Xin Jin[1]
jin.xin@samsung.com
Longhai Wu[1]
longhai.wu@samsung.com
Jie Chen[1]
ada.chen@samsung.com
Ilhyun Cho[2]
ih429.cho@samsung.com
Cheul-Hee Hahm[2]
chhahm@samsung.com

[1] Samsung Electronics (China) R&D Centre, Nanjing
[2] Samsung Electronics, South Korea

Abstract

Existing dynamic Video Frame Interpolation (VFI) methods either struggle to produce satisfactory results on high-resolution videos due to the lack of explicit image warping, or fail to achieve practical speedup due to the use of inefficient sparse convolution. In this work, we propose a novel dynamic VFI architecture that combines image warping and patch-wise processing: the warping operation can compensate for the large motion in high-resolution videos, while patch-wise dynamic processing can be implemented with efficient regular convolution. Specifically, we develop an Alignment-aware Patch-wise Routing module to select the adaptive sub-networks to synthesize target patches from local patches in warped input frames. Extensive experiments demonstrate that our dynamic VFI method can achieve excellent results on 2K and 4K videos, saving considerable amount of computation and runtime with marginal performance degradation.

1 Introduction

Video frame interpolation (VFI) aims to synthesize intermediate frames between consecutive frames, thereby enhancing the temporal resolution and enriching the visual contents. VFI is widely used in various real-world applications, including video generation [45], video editing [10], video compression [44], *etc*. Despite recent advances in performance, current VFI methods still encounter difficulties in addressing complex and large motion cases (which are common in high-resolution videos), and have limited flexibility and heavy computational burden for practical deployment. In particular, a common limitation is the uniform computation in synthesizing different image regions. It overlooks the varying interpolation

difficulties across different frame regions, leading to redundant computation in areas with minimal motion or clear content.

Recently, a few adaptive approaches are proposed to assign varying amount of computation for different image regions [2, 6]. Choi *et al.* [6] integrate a specialized module to decide patch-level input scale and model depth, dynamically synthesizing each patch with appropriate resolution and sub-network. Despite about 50% reduction in FLOPs, this method does not integrate image warping into the dynamic architecture to compensate for large motion. As a result, the synthesis network in [6] directly aligns the pixels (possibly with large displacement) in original input frames, leading to inferior performance for high-resolution videos. Cheng *et al.* [2] learn pixel-level uncertainty, and employ a spatial pruning method based on sparse convolution [25] to skip redundant computation on pixels with low uncertainty. Although it reduces theoretical FLOPs, the practical speedup on GPUs is constrained, as pixel-wise sparse convolution is unfriendly for parallel processing.

To address aforementioned limitations, we propose a novel dynamic architecture for VFI. It integrates image warping with a dynamic synthesis network, and performs efficient patch-level (rather than pixel-level) dynamic computation. It follows the main steps in forward-warping-based VFI: estimating the bi-directional flow between input frames, forward-warping input frames towards the target frame, and then predicting the target frame from warped frames with a synthesis network. Our key innovation is an Alignment-aware Patch-level Routing (APR) module, which is inserted before the synthesis network, and can select appropriate sub-synthesis-networks for different warped input patches based on the alignment between them. Furthermore, our synthesis network takes warped image patches (rather than unwarped patches [6]) as input, and its lightweight yet highly modularized network design allows dynamic computation and efficient feature fusion. Coupling these designs, our APR-VFI method can reliably handle large motion, and achieve practical speedup for high-resolution videos.

Our contributions can be concluded as follows:

- We propose an Alignment-aware Patch-level Routing (APR) module to adaptively determine the inference path in the synthesis network for each image patch.
- We design a dynamic VFI framework based on APR, with a specially designed synthesis network that takes warped image patches as input.
- In comparison with baseline model without APR, our APR-VFI achieves competitive performance on 2K and 4K Xiph benchmark, saving up to 30% FLOPs and 20% runtime with marginal performance degradation.

2 Related work

Video Frame Interpolation. Most VFI methods can be roughly categorized into kernel-based, flow-based and diffusion-based. Kernel-based methods [3, 4, 9, 23, 33, 34] interpolate intermediate frames by learning adaptive kernels and adopting separable convolutions on input frames without explicit motion estimation. Although these methods are relatively effective, without explicit image warping they can hardly deal with large motions, given the limited receptive field of learned convolutional kernels. Flow-based methods [18, 30, 35] firstly estimate the optical flow between input frames, and then leverage estimated flow to explicitly guide the synthesis of intermediate frame. In particular, Kong *et al.* [21] present an efficient encoder-decoder network for efficient frame interpolation. This network refines

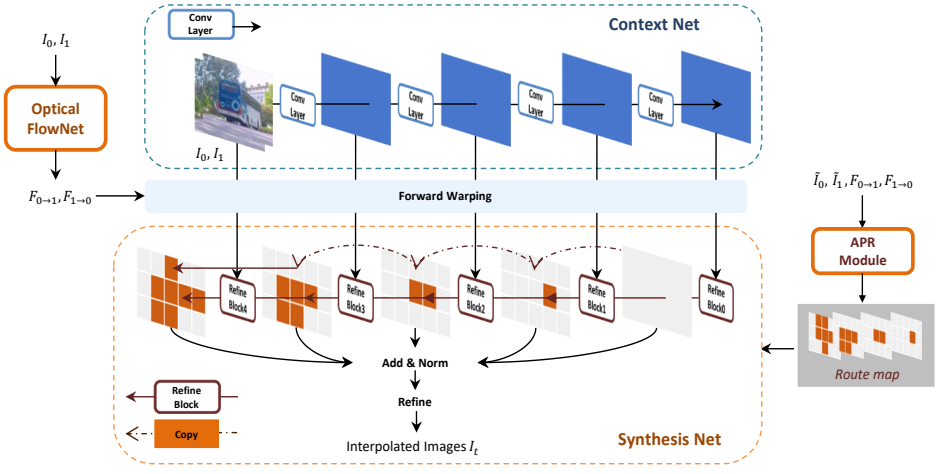


Figure 1: Overview of Alignment-aware Patch-level Routing for Dynamic VFI (APR-VFI) architecture. APR-VFI contains an optical flow network for motion estimation, a context network for context feature extraction, and an APR equipped synthesis network for frame prediction. APR predicts a route map and feeds it into synthesis network as a guidance. $\tilde{I}_0$ and $\tilde{I}_1$ indicate warped image 0 and 1. $F_{0\to1}$ and $F_{1\to0}$ denote the estimated bi-directional flow between input frames I_0 and I_1

optical flow together with intermediate feature until generating the desired result. Despite improved efficiency, it can hardly generalize well from small motion datasets to extremely large motion datasets.

Recently, diffusion models [13, 39] have exhibited remarkable progress in generating high-fidelity and diverse images, and some works also explored its potentials in VFI field. MVCD [41] integrates neighboring frames as condition in a devised convolutional U-net to implicitly learn spatio-temporal dynamics. In contrast, MADIFF [16] explicitly captures the inter-frame motion hints (by an events simulator [48]) as auxiliary guidance for interpolation. To better adapt LDMs to VFI, LDMVFI [8] proposes a novel VFI-specific auto-encoder integrating enhanced feature interaction during reconstruction progress. Instead of revising architecture, VIDIM [40] cascades two standard diffusion models, where a base model generates low resolution results and a super-resolution model upsamples to large resolution. However, these methods are relative slow since progressive denoising procedure.

Dynamic network. Many vision tasks have employed dynamic architectures to allocate different amount of computation for different inputs. These methods can be divided into two groups: spatial-wise and sample-wise [11]. Spatial-wise dynamic networks perform adaptive operations on different image regions, such as changing model depth with early exit [42], adaptively skipping some middle layers [14, 15, 26] and adjusting the input resolution [6]. In contrast, sample-wise networks formally switch model architecture [22, 47] or change parameters [12, 49] for different samples. In this work, we propose a spatial-wise network that selects adaptive inference path for each image patch to achieve good trade-off between efficiency and accuracy.

3 Method

In this section, we firstly introduce our APR-VFI pipeline (Sec.3.1), then elaborate the key components in APR-VFI (Sec.3.2), and describe our training strategies (Sec.3.3).

3.1 Overall Pipeline

Given two frames I_0 and I_1, VFI aims to synthesize the intermediate frame I_t, where t $\in$ (0,1). As illustrated in Fig.1, we use an optical flow network for motion estimation. Meanwhile, a context network captures multi-level context features of input frames. The original input frames and their context features are forward-warped towards the target frame before feeding to APR-based synthesis network for frame interpolation. Formally, let $F_{0\to1}$ and $F_{1\to0}$ denote the estimated bi-directional flow between input frames I_0 and I_1, and linear scaling can be used to approximate the motion from I_0 and I_1 to target I_t. Let $\overrightarrow{W}$ denote forward-warping (average splatting), then the warped images $\tilde{I}_0$ and $\tilde{I}_1$ can be calculated as Eq.(1). Warping context features follows the same procedure.

$$\tilde{I}_0 = \overrightarrow{W}(t \cdot F_{0\to1}, I_0), \quad \tilde{I}_1 = \overrightarrow{W}((1-t) \cdot F_{1\to0}, I_1) \tag{1}$$

Our synthesis network composed of several successive refine blocks for iterative patch-wise feature refinement (similar in spirit but different in structure with IFRNet [21]). By integrating APR module, each refine block can adaptively process a subset of informative patches, instead of processing all patches (shown in Fig.2). Then, all refined features are fused together, and a convolution layer takes the fused feature to predict a mask M for combining the warped images ($\tilde{I}_0$ and $\tilde{I}_1$), and a residual image $\triangle I_t$ for further refinement. Finally, the final interpolated image is calculated as follow:

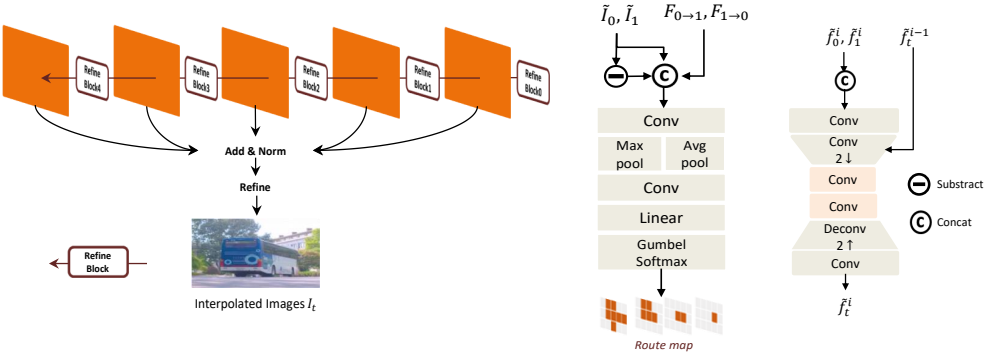
$$I_t = M \times \tilde{I}_0 + (1-M) \times \tilde{I}_1 + \triangle I_t \tag{2}$$

3.2 Model Components

The key model components in our APR-VFI pipeline include an optical flow estimator, a context network and an APR-based dynamic synthesis network. In particular, we follow the motion estimator in [20] to calculate bi-directional motions between input frames.

Context Network. The context network extracts multi-level context features of input frames, using several convolution layers. However, our context network is different with previous pyramidal context network [20]. To accommodate with APR-based synthesis network that operates on fixed resolution, we do not down-sample features, and reduce the number of feature channels to release computation burden.

Alignment-aware Patch-level Routing(APR). Our APR predicts a multi-level route map that determines the synthesis path of each image patch. We assume that image content complexity and temporal derivations play a key role in route planning. Therefore, APR (in Fig. 2) takes concatenated warped image difference, estimated flow and warped images as input, and convolves them to extract spatiotemporal features. To ensure each pixel of route map corresponding to path of one image patch, we apply two pooling methods to simultaneously downscale spatiotemporal features from *(B,C,H,W)* into *(B,C,* n_h, n_w*)*, where n_h, n_w are patch number across height and width. Two pooling methods enhance feature representation. Subsequently, a linear layer predicts the possibility of each patch feeding into each stage of synthesis network. In addition, to make our APR-VFI end-to-end trainable, we follows [7] to use the Straight-Through Gumbel-Softmax trick [17] in training and applying the argmax operation at test time. Our APR only has 5.13k parameters, and introduces little computational cost for whole pipeline.



(a) Static synthesis network (b) APR module (c) Refine Block

Figure 2: Structure of different model components.

Here, we would like to highlight the differences between our APR module and the SD-finder in [7] ("S" indicates scale, and "D" indicates depth). Firstly, our APR module does not predict the scaling factor, and eliminates the need of a super resolution for further refinement. Second, our design of APR aims to predict the route map, mainly based on the alignment between warped patches. This design stems from our intuition that if the warped input patches have been aligned well with each other, the synthesis network should be easily predict the target patch from warped patches.

Synthesis Network. Our synthesis network mainly consists of 5 refine blocks, which progressively refines intermediate feature from coarse-to-fine. In particular, We define the first refine block as static block that is always applied for all patches, as good initial intermediate feature is helpful for maintaining performance under reduced computation.

As shown in Fig. 2, our refine block uses a simple encoder-decoder structure for feature refinement. We compare APR-based dynamic synthesis network (Fig. 1) with its static counterpart (Fig. 2). Static synthesis network infers with all image patches (equivalent to full image), while dynamic synthesis network divides feature into to a grid of $n_h \times n_w$ patches and dynamically selects a subset for refinement. As mentioned before, APR predicts a binary multi-level route map $M \in \mathbb{R}^{n_h \times n_w \times d}$, which indicates the dynamic inference path in synthesis network for all image patches. Here, n_h and n_w are the numbers of patches across height and width, while d means the number of dynamic blocks in synthesis network. For a route vector $v_{p_i}^k \in M$, where i represents the index of patch. $v_{p_i}^k = 1$ means patch p_i will go through the k^{th} refine block, while 0 means p_i will skip this block by directly inheriting the corresponding features from preceding refine block. Let us denote the warped k^{th} stage context features of patch i as $\tilde{f}_{p_{i0}}^k$ and $\tilde{f}_{p_{i1}}^{k-1}$ and its corresponding route vector as $v_{p_i}^k$. The output feature $f_{p_i}^k$ of k^{th} refine block ϕ_k is calculated as:

$$f_{p_i}^k = \begin{cases} \phi_k(\tilde{f}_{p_{i0}}^k, \tilde{f}_{p_{i1}}^k, f_{p_i}^{k-1}) & \text{if } v_{p_i}^k = 1 \\ f_{p_i}^{k-1} & \text{if } v_{p_i}^k = 0 \end{cases} \tag{3}$$

In [34], the authors studied the effect of self-ensembling methods in VFI, and concluded that any form of self-ensembling is superior to a single prediction. Therefore, we propose a simple but effective method to fuse output features from 5 refine blocks for final prediction: adding all refined features together and normalizing the result in channel dimension.

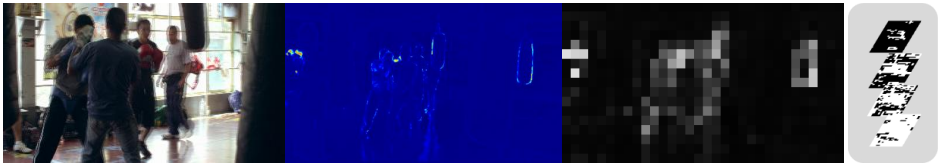


Figure 3: Pipeline of reference route map generation. The luminance of third image reflects the difference magnitude, where bright region means large difference.

3.3 Training

Objective Functions. We use three types of loss: reconstruction loss L_{rec}, census loss [28] L_{cen} and APR loss L_{apr}. Reconstruction loss is Charbonnier distance between our interpolation I_t and ground truth I_t^{GT}. For APR loss, we generate a route label $M^{'} \in \mathbb{R}^{n_h \times n_w \times d}$ as ground truth (described later in 3.3) and compute L_1 distance between $M^{'}$ and predicted route map M. Finally, our loss is weighted sum of reconstruction loss, census loss and APR loss, as in Eq. (4). By default, we set $\lambda_1 = 0.1$ and $\lambda_2 = 0.01$.

$$L = L_{rec}(I_t, I_t^{GT}) + \lambda_1 L_{cen}(I_t, I_t^{GT}) + \lambda_2 \times L_{apr}(M^{'}, M) \tag{4}$$

Curriculum training. End-to-end learning of all components at once leads to unstable training. Therefore, we train APR-VFI with 3 training stages: 1) pretrain the static APR-VFI with whole images; 2) freeze optical flow network and train static APR-VFI with image patches of 64×64; 3) freeze optical flow network and jointly train context network, synthesis net with APR. Freezing optical flow network in step 3 is crucial for model coverage, since significantly increased reconstruction loss in this stage is mainly caused by the introduction of APR but not inaccurate flow. We provide more information of objective functions in supplementary material.

Generation of reference route map $M^{'}$. To train APR module, we adopt a simple heuristic rule to generate reference route map $M^{'}$ as supervision and experiments show that $M^{'}$ helps routing module estimate accurate path. Fig. 3 visualizes the procedure of reference route map generation.

Firstly, we define the patch keeping ratio for d dynamic blocks as $[kr_1, kr_2, ..., kr_d]$, indicating the percentage of patches being processed at a certain stage. The default value is $[0.1, 0.4, 0.6, 0.9]$. Then, we generate reference route map $M^{'} \in \mathbb{R}^{n_h \times n_w \times d}$ as follows:

1. Compute difference between warped images but ignoring difference in hole region (M_{hole}) around warped image edge. The hole region brought by warp operation has significant difference but can be easily reconstructed with few computation.

2. Split difference map from step 1 into $n_h \times n_w$ patches and estimate patch difference in Eq.(5). The kernel size of MaxPool and AvgPool is (16,16) and (4,4).

$$P_{diff} = MaxPool(AvgPool((\tilde{I}_0 - \tilde{I}_1) * M_{hole})) \tag{5}$$

3. Sort the P_{diff} in an ascending order. For the 1^{st} dynamic block, we generate $m_1^{'} \in \mathbb{R}^{n_h \times n_w \times 1}$ by annotating top kr_1 patches as 1 (pass), and remaining patches as 0 (exit).

4. Follow step 3 until the last dynamic block, and concatenate $[m_1^{'}, m_2^{'}, ..., m_d^{'}]$ as $M^{'}$.

	Extra Dataset	Xiph-2k			Xiph-"4k"			Para. (M)
		PSNR/SSIM	GPU Runtime(s)	TFLOPs	PSNR/SSIM	GPU Runtime(s)	TFLOPs	
ToFlow [46]	-	33.93/0.922	-	-	30.74/0.856	-	-	1.4
BMBC [36]	-	32.82/0.928	4.025	5.900	31.19/0.880	4.025	5.900	11.0
ABME [37]	-	36.53/0.944	0.907	3.120	33.73/0.901	0.907	3.120	18.1
SepConv [32]	-	34.77/0.929	0.078	2.078	32.06/0.880	0.078	2.078	19.8
SuperSloMo [19]	-	33.88/0.925	0.087	2.957	31.99/0.880	0.087	2.957	1.15
AdaCoF [24]	-	34.86/0.928	0.069	0.848	31.68/0.870	0.069	0.848	21.8
DAIN [1]	√	35.95/0.940	1.154	13.221	33.49/0.895	1.154	13.221	24.0
FILM [38]	-	36.66/0.951	0.161	4.706	33.78/0.906	0.161	4.706	34.4
EBME-H [20]	-	36.62/0.967	1.086	0.095	33.93/0.945	1.086	0.095	3.9
SoftSplat [31]	√	36.62/0.967	0.223	2.150	33.93/0.946	0.223	2.150	12.2
IFRNet large [21]	-	36.63/0.966	0.081	1.960	33.58/0.944	0.081	1.960	19.7
CAIN [5]	-	35.21/0.937	0.189	3.133	32.56/0.901	0.021	3.133	42.8
CAIN-SD [7]	√	34.68/0.924	0.199	1.600	32.92/0.893	0.203	1.983	-
Ours(w/o APR)	-	36.67/0.967	0.143	1.730	34.02/0.946	0.143	1.730	2.2
Ours(with APR)	-	36.46/0.966	0.116	1.220	33.90/0.944	0.123	1.271	2.2

Table 1: Evaluation results on Vimeo90K and Xiph. Computational complexity is measured in TFLOPs and GPU Time(s), and performance measured in PSNR and SSIM. "-" means the corresponding data is unavailable. Ours(with APR) represent proposed APR-VFI, while Ours(w/o APR) denotes the same model architecture but removing APR.

4 Experiments

Implementation Details

Training Details. We use Vimeo90K [46] to train the interpolation model from scratch. The batch size is 32 and data argumentation such as random flip, rotation, temporal reversal *etc.*, are performed in training process. We choose AdamW [27] as the optimizer with $\beta_1 = 0.9$, $\beta_2 = 0.999$ and weight decay $1e^{-4}$. The iteration numbers at 3 training stages are 800k, 200k and 200k. In first 2000 steps of each stage, we warm up the learning rate to $2e^{-4}$ and then use cosine annealing for remaining steps to reduce learning rate from $2e^{-4}$ to $2e^{-5}$. Our model is trained and tested on a NVIDIA A100.

Evaluation Metrics and Datasets

Standard PSNR and SSIM [43] are adopted for performance evaluation. For computation complexity and inference speed, we calculate FLOPs and GPU runtime on 2048×1080. Our model is evaluated on the following datasets:

Xiph [29]: Following the settings in [7], we use the downsampled version as Xiph-2K and the center-cropped version as Xiph-"4K". Although both their resolution is 2048×1080, Xiph-"4K" keeps the original motion magnitude of 4K (4096×2160) images.

Vimeo90K [46]: It contains frame triplets of 256×448 resolution. This dataset is not our main focus, because its flow magnitude is relatively small. We report corresponding performance in Sec.4

Comparison to the State-of-the-arts

In this section, we compare APR-VFI and its static version (a model shares the same model structure with APR-VFI but removing APR.) with various state-of-the-art VFI methods, and analyze our model performance.

Quantitative Evaluation. Tab. 1 shows comparison results on Xiph. We report two versions of model results; static VFI (w/o APR) infers with complete procedure and adaptive

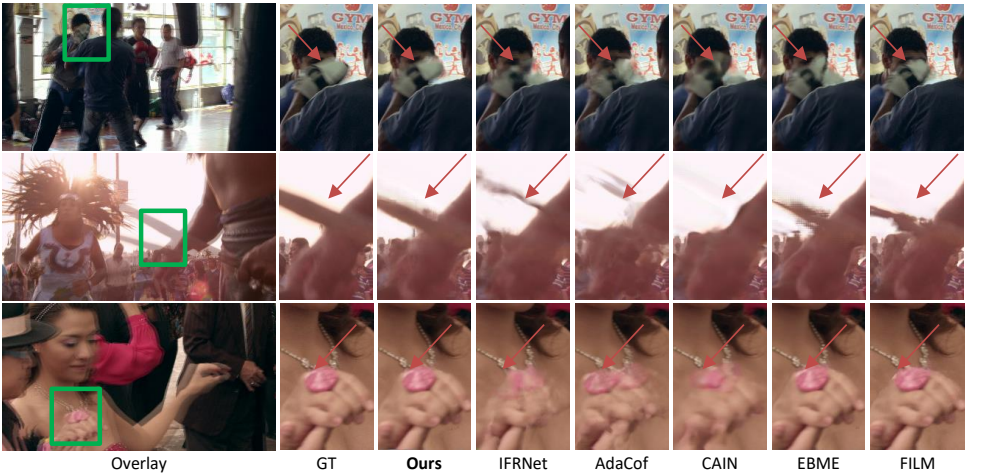


Figure 4: Visual comparison on Xiph-4k.

VFI (with APR) shortens inference path of easy regions to enable better efficiency. The static VFI (w/o APR) achieves the best results in Xiph-2K and Xiph-"4K". It outperforms IFRNet-large by 0.03db in Xiph-2K and 0.44db in Xiph-"4K" with less FLOPs and 8× fewer parameters. In contrast with FILM which is designed for large motion, we perform significant (0.24db) superiority on Xiph-"4K". The recent EBME-H utilizes the explicit forward warping, but still falls behind us on Xiph-"4K". We attribute the reason to the feature fusion in synthesis network which provides complementary features for large motion.

For APR-VFI, we compare it with another adaptive VFI method CAIN-SD. Noticing that CAIN-SD measures runtime on RTX6000, we linearly estimate its runtime on A100 according to capability difference. We surpass CAIN-SD by a large margin and run 1.6× faster with less FLOPs. Although CAIN-SD can reduce more FLOPs while achieving higher performance, it introduces extra training dataset to learn large motion estimation. With help of APR, APR-VFI saves 30% and 26% TFLOPs of its complete procedure (ours w/o APR) on Xiph-2K and Xiph-"4K" with minimal performance degradation.

In addition, our VFI (w/o APR) achieves 36.05 on Vimeo90K and the APR module saves 34% of computational cost with 0.2db degradation. The performance on Vimeo90k is not our focus, but we provides related results in supplementary material.

Qualitative Evaluation. Fig. 4 provides visual comparisons between APR-VFI and other VFI methods on Xiph. It can be seen that our model can interpolate large and complex regions with pleasant results. Our framework shows a superior visual experience within enlarged bounding box, while other models suffer from ghost and blur artifacts when objects move fast.

We visualize predicted inference path from APR in Fig. 5. The regions (white T-shirts, blank sky, empty floor) with minor difference or clear context get fewer synthesis processing while difficult regions (women face, fast moving sticks) tend to pass through more fine refinement. Thanks to accurate inference path, we preserve model capability on complex regions while save redundant computation on easy regions.

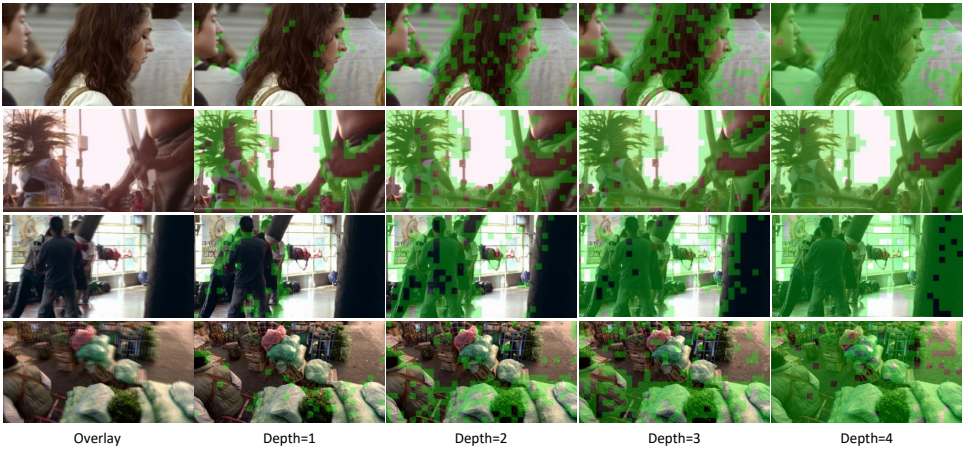


Figure 5: Visualization of route map predicted by APR. The green masks at different depths represent the patches passing through corresponding refine block.

5 Ablation Study

Effects on Patch Size. Patch size is an important parameter for APR-VFI. We first study its effects in static VFI and choose an appropriate setting for APR-VFI. Tab. 2 shows that smaller patch size causes worse performance. The accuracy of model with patch size 8×8 is considerably lower than 128×128, although the FLOPs is same. We analyze that splitting images into smaller patches will lead to loss of global information, and synthesis network may not be able to interpolate good results with limited context. Finally, We choose patch size of 64 instead of 128, because if the patches are too large the number of patches will be reduced, which is not conducive to route module training.

Patch Keeping Ratio. Patch Keeping Ratio is important in balancing efficiency and accuracy by restricting the number of processing patches for each refine block. We compare different settings in Tab. 3. The results shows gradually increasing the patch keeping ratio with model depth is appropriate for our architecture. We analyze that the prediction layer is supervised to fit well with output feature from the last refine block. Keeping more patches at deeper blocks can reduce feature distribution gap caused by changing inference path. We find [0.2, 0.5, 0.7, 1.0] setting introduces more computational cost than [0.1, 0.4, 0.6, 0.9] without obvious improvement on X-"4K". It means computation under [0.1, 0.4, 0.6, 0.9] setting achieves the best trade-off, selecting more patches would reduce efficiency.

Feature Fusion Methods. To vertify the effectiveness of our fusion method (add&norm), we compare it with concat&conv operation (concatenate all features and perform convolution). As shown in Tab. 5, our method achieves more improvement on Xiph-"4k" without extra parameter.

Supervision for APR. As mentioned in Sec.3.3, we generate reference route map to guide APR learning optimal inference path. To prove its effectiveness, we study the effect of different route maps on APR-VFI. We follow the same mechanism but using different types of alignment information when generating route map (step 1). Tab. 4 shows warped image difference achieves the best accuracy, since we warp images and features before feeding them into synthesis network. It provides the most related alignment information for assigning adaptive path. In addition, we analyze that reconstruction loss may misguide model focusing on some high-frequency but well-aligned easy regions, which causes the lowest performance.

We provide more ablation studies on supplementary material.

patch size	X-2K	X-"4K"
8	36.48/0.966	33.89/0.945
16	36.54/0.966	33.92/0.945
32	36.61/0.967	33.90/0.945
64	36.64/0.967	34.02/0.946
128	36.67/0.967	34.05/0.946

Table 2: Ablation on effects of patch size at patch-level inference.

keep ratio	Xiph-2K	Xiph-"4K"	TFLOPs
0.2 0.5 0.7 1.0	36.49/0.965	33.90/0.944	1.341
0.1 0.4 0.6 0.9	36.46/0.965	33.90/0.944	1.271
0.2 0.4 0.6 0.8	36.45/0.965	33.87/0.944	1.296
0.5 0.5 0.5 0.5	36.45/0.965	33.84/0.943	1.389
0.9 0.6 0.4 0.1	36.42/0.965	33.86/0.944	1.504

Table 3: Ablation on different keep ratio. Higher ratio indicates more passing patches at corresponding stage.

supervised information	X-2K	X-"4K"
warped difference	36.46/0.9659	33.90/0.9442
image difference	36.42/0.9654	33.86/0.9435
reconstruction loss	36.42/0.9655	33.84/0.9436

Table 4: Ablation on different supervision for APR training.

fusion method	X-2K	X-"4K"
add&norm	36.66/0.9672	34.02/0.9461
concat&conv	36.68/0.9672	33.97/0.9457
no fusion	36.64/0.9670	33.93/0.9456

Table 5: Ablation on different feature fusion methods.

6 Limitations and Future work

Although our proposed APR enables adaptive video frame interpolation, there are still some limitations worth exploring. First, although APR could relieve computational cost, we still have relative large FLOPs due to limited down-sampling operations. Second, adaptive inference path predicted by APR is restricted by pre-defined reference route map, which impairs model flexibility. In future work, we will rethink synthesis network structure to save computational resource. Meanwhile, we will also investigate effective training strategies without explicit supervision for routing module.

7 Conclusion

In this work, we propose an Alignment-aware Patch-level Routing (APR) module and design a compatible VFI architecture (APR-VFI). Our dynamic VFI adaptively interpolates image patches through different sub-networks, saving redundant computation on easy regions and preserving model capability on challenging regions. Meanwhile, we utilize a simple but effective feature fusion method to enhance feature representation. It shows that the proposed APR-VFI achieves excellent performance on large motion dataset, saving considerable computation with marginal performance degradation.

References

[1] Wenbo Bao, Wei-Sheng Lai, Chao Ma, Xiaoyun Zhang, Zhiyong Gao, and Ming-Hsuan Yang. Depth-aware video frame interpolation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 3703–3712, 2019.

[2] Ri Cheng, Xuhao Jiang, Ruian He, Shili Zhou, Weimin Tan, and Bo Yan. Uncertainty-guided spatial pruning architecture for efficient frame interpolation. In *Proceedings of the 31st ACM International Conference on Multimedia*, MM 23. ACM, October 2023. doi: 10.1145/3581783.3611752. URL http://dx.doi.org/10.1145/3581783.3611752.

[3] Xianhang Cheng and Zhenzhong Chen. Multiple video frame interpolation via enhanced deformable separable convolution. 2020.

[4] Xianhang Cheng and Zhenzhong Chen. Video frame interpolation via deformable separable convolution. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34 (7):10607–10614, 2020.

[5] Myungsub Choi, Heewon Kim, Bohyung Han, Ning Xu, and Kyoung Mu Lee. Channel attention is all you need for video frame interpolation. In *AAAI*, 2020.

[6] Myungsub Choi, Suyoung Lee, Heewon Kim, and Kyoung Mu Lee. Motion-aware dynamic architecture for efficient frame interpolation. 2021.

[7] Myungsub Choi, Suyoung Lee, Heewon Kim, and Kyoung Mu Lee. Motion-aware dynamic architecture for efficient frame interpolation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 13839–13848, 2021.

[8] Duolikun Danier, Fan Zhang, and David Bull. Ldmvfi: Video frame interpolation with latent diffusion models. *arXiv preprint arXiv:2303.09508*, 2023.

[9] Tianyu Ding, Luming Liang, Zhihui Zhu, and Ilya Zharkov. Cdfi: Compression-driven network design for frame interpolation. *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7997–8007, 2021. URL https://api.semanticscholar.org/CorpusID:232290640.

[10] John Flynn, Ivan Neulander, James Philbin, and Noah Snavely. Deepstereo: Learning to predict new views from the world's imagery. In *CVPR*, 2016.

[11] Yizeng Han, Gao Huang, Shiji Song, Le Yang, Honghui Wang, and Yulin Wang. Dynamic neural networks: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(11):74367456, November 2022. ISSN 1939-3539. doi: 10.1109/tpami.2021.3117837. URL http://dx.doi.org/10.1109/TPAMI.2021.3117837.

[12] Adam W. Harley, Konstantinos G. Derpanis, and Iasonas Kokkinos. Segmentation-aware convolutional networks using local attention masks. In *2017 IEEE International Conference on Computer Vision (ICCV)*. IEEE, October 2017. doi: 10.1109/iccv.2017.539. URL http://dx.doi.org/10.1109/ICCV.2017.539.

[13] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *arXiv preprint arxiv:2006.11239*, 2020.

[14] Xiaotao Hu, Jun Xu, Shuhang Gu, Ming-Ming Cheng, and Li Liu. Restore globally, refine locally: A mask-guided scheme to accelerate super-resolution networks. In *Computer Vision ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 2327, 2022, Proceedings, Part XIX*, page 7491, Berlin, Heidelberg, 2022. Springer-Verlag. ISBN 978-3-031-19799-4. doi: 10.1007/978-3-031-19800-7_5. URL https://doi.org/10.1007/978-3-031-19800-7_5.

[15] Xiaotao Hu, Zhewei Huang, Ailin Huang, Jun Xu, and Shuchang Zhou. A dynamic multi-scale voxel flow network for video prediction. In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, June 2023.

doi: 10.1109/cvpr52729.2023.00593. URL http://dx.doi.org/10.1109/CVPR52729.2023.00593.

[16] Zhilin Huang, Yijie Yu, Ling Yang, C. Qin, Bing Zheng, Xiawu Zheng, Zikun Zhou, Yaowei Wang, and Wenming Yang. Motion-aware latent diffusion models for video frame interpolation, 2024.

[17] Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with gumbel-softmax, 2016.

[18] Huaizu Jiang, Deqing Sun, Varan Jampani, Ming-Hsuan Yang, Erik Learned-Miller, and Jan Kautz. Super slomo: High quality estimation of multiple intermediate frames for video interpolation. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*. IEEE, June 2018. doi: 10.1109/cvpr.2018.00938. URL http://dx.doi.org/10.1109/CVPR.2018.00938.

[19] Huaizu Jiang, Deqing Sun, Varun Jampani, Ming-Hsuan Yang, Erik Learned-Miller, and Jan Kautz. Super slomo: High quality estimation of multiple intermediate frames for video interpolation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 9000–9008, 2018.

[20] Xin Jin, Longhai Wu, Guotao Shen, Youxin Chen, Jie Chen, Jayoon Koo, and Cheulhee Hahm. Enhanced bi-directional motion estimation for video frame interpolation. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, 2023.

[21] Lingtong Kong, Boyuan Jiang, Donghao Luo, Wenqing Chu, Xiaoming Huang, Ying Tai, Chengjie Wang, and Jie Yang. Ifrnet: Intermediate feature refine network for efficient frame interpolation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[22] Xiangtao Kong, Hengyuan Zhao, Yu Qiao, and Chao Dong. Classsr: A general framework to accelerate super-resolution networks by data characteristic. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, June 2021. doi: 10.1109/cvpr46437.2021.01184. URL http://dx.doi.org/10.1109/CVPR46437.2021.01184.

[23] Hyeongmin Lee, Taeoh Kim, Tae young Chung, Daehyun Pak, Yuseok Ban, and Sangyoun Lee. Adacof: Adaptive collaboration of flows for video frame interpolation, 2019.

[24] Hyeongmin Lee, Taeoh Kim, Tae-young Chung, Daehyun Pak, Yuseok Ban, and Sangyoun Lee. Adacof: Adaptive collaboration of flows for video frame interpolation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 5316–5325, 2020.

[25] Baoyuan Liu, Min Wang, Hassan Foroosh, Marshall Tappen, and Marianna Penksy. Sparse convolutional neural networks. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 806–814, 2015. doi: 10.1109/CVPR.2015.7298681.

[26] Ming Liu, Zhilu Zhang, Liya Hou, Wangmeng Zuo, and Lei Zhang. *Deep Adaptive Inference Networks for Single Image Super-Resolution*, page 131148. Springer International Publishing, 2020. ISBN 9783030668235. doi: 10.1007/978-3-030-66823-5_8. URL http://dx.doi.org/10.1007/978-3-030-66823-5_8.

[27] Ilya Loshchilov and Frank Hutter. Fixing weight decay regularization in adam. 2018.

[28] Simone Meyer, Abdelaziz Djelouah, Brian McWilliams, Alexander Sorkine-Hornung, Markus Gross, and Christopher Schroers. Phasenet for video frame interpolation. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*. IEEE, June 2018. doi: 10.1109/cvpr.2018.00059. URL http://dx.doi.org/10.1109/CVPR.2018.00059.

[29] Christopher Montgomery. Xiph.org video test media (derfs collection), 1994. URL https://media.xiph.org/video/derf/.

[30] Simon Niklaus and Feng Liu. Softmax splatting for video frame interpolation. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, June 2020. doi: 10.1109/cvpr42600.2020.00548. URL http://dx.doi.org/10.1109/CVPR42600.2020.00548.

[31] Simon Niklaus and Feng Liu. Softmax splatting for video frame interpolation. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2020.

[32] Simon Niklaus, Long Mai, and Feng Liu. Video frame interpolation via adaptive convolution. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2017.

[33] Simon Niklaus, Long Mai, and Feng Liu. Video frame interpolation via adaptive convolution, 2017.

[34] Simon Niklaus, Long Mai, and Oliver Wang. Revisiting adaptive convolutions for video frame interpolation, 2020.

[35] Junheum Park, Keunsoo Ko, Chul Lee, and Chang-Su Kim. *BMBC: Bilateral Motion Estimation with Bilateral Cost Volume for Video Interpolation*, page 109125. Springer International Publishing, 2020. ISBN 9783030585686. doi: 10.1007/978-3-030-58568-6_7. URL http://dx.doi.org/10.1007/978-3-030-58568-6_7.

[36] Junheum Park, Keunsoo Ko, Chul Lee, and Chang-Su Kim. Bmbc: Bilateral motion estimation with bilateral cost volume for video interpolation. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XIV 16*, pages 109–125. Springer, 2020.

[37] Junheum Park, Chul Lee, and Chang-Su Kim. Asymmetric bilateral motion estimation for video frame interpolation. In *International Conference on Computer Vision*, 2021.

[38] Fitsum Reda, Janne Kontkanen, Eric Tabellion, Deqing Sun, Caroline Pantofaru, and Brian Curless. Film: Frame interpolation for large motion. In *European Conference on Computer Vision*, pages 250–266. Springer, 2022.

[39] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models, 2021.

[40] Eric Tabellion Siddhant Jain, Daniel Watson and Janne Kontkanen Aleksander Hoy nski, Ben Poole. Video interpolation with diffusion models. *arXiv preprint arXiv:2404.01203*, 2024.

[41] Vikram Voleti, Alexia Jolicoeur-Martineau, and Christopher Pal. Mcvd: Masked conditional video diffusion for prediction, generation, and interpolation. In *(NeurIPS) Advances in Neural Information Processing Systems*, 2022. URL https://arxiv.org/abs/2205.09853.

[42] Shizun Wang, Ming Lu, Kaixin Chen, Xiaoqi Li, Jiaming Liu, and Yandong Guo. Adaptive patch exiting for scalable single image super-resolution. *ArXiv*, abs/2203.11589, 2022. URL https://api.semanticscholar.org/CorpusID:247596606.

[43] Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*, 13(4):600–612, 2004.

[44] Chao-Yuan Wu, Nayan Singhal, and Philipp Krähenbühl. *Video Compression Through Image Interpolation*, page 425440. Springer International Publishing, 2018. ISBN 9783030012373. doi: 10.1007/978-3-030-01237-3_26. URL http://dx.doi.org/10.1007/978-3-030-01237-3_26.

[45] Xiaoyu Xiang, Yapeng Tian, Yulun Zhang, Yun Fu, Jan P. Allebach, and Chenliang Xu. Zooming slow-mo: Fast and accurate one-stage space-time video super-resolution. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3370–3379, June 2020.

[46] Tianfan Xue, Baian Chen, Jiajun Wu, Donglai Wei, and William T Freeman. Video enhancement with task-oriented flow. *International Journal of Computer Vision*, 127: 1106–1125, 2019.

[47] Jiahui Yu and Thomas Huang. Universally slimmable networks and improved training techniques. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*. IEEE, October 2019. doi: 10.1109/iccv.2019.00189. URL http://dx.doi.org/10.1109/ICCV.2019.00189.

[48] Alex Zihao Zhu, Ziyun Wang, Kaung Khant, and Kostas Daniilidis. Eventgan: Leveraging large scale image datasets for event cameras. *arXiv preprint arXiv:1912.01584*, 2019.

[49] Xizhou Zhu, Han Hu, Stephen Lin, and Jifeng Dai. Deformable convnets v2: More deformable, better results. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, June 2019. doi: 10.1109/cvpr.2019.00953. URL http://dx.doi.org/10.1109/CVPR.2019.00953.