OPTICAL FLOW LESS VIDEO FRAME INTERPOLATION

Shreshth Saini[*]
saini.2@utexas.edu

The University of Texas at Austin
Electrical and Computer Engineering

Krishna Srikar Durbha[*]
krishna.durbha@utexas.edu

The University of Texas at Austin
Electrical and Computer Engineering

***Index Terms*—** Video Interpolation, Video Processing, Computer Vision, Deep Learning

ABSTRACT

Current approaches for video frame interpolation use optical flow features to predict the motions in interpolated frames accurately. Predicting optical flow has additional computational overhead and often becomes the bottleneck for the overall performance of the algorithm. To address this challenge, we propose the use of transformers to learn the long-range interactions with mutual self-attention between frames as a surrogate for motion estimation. Furthermore, instead of feeding in only two frames as input to the model, we empirically show that using more than three frames could better assist the transformers in motion estimation. Since transformers have high computation costs due to self-attention, we resort to the use of novel local self-attention extended to spatiotemporal dimensions. We train our model from scratch and show comparable results despite being trained on a fractional dataset as compared to other state-of-the-art methods. The code, models, and slide deck will be released separately.

1. INTRODUCTION

Video frame interpolation(VFI) aims to temporally upsample an input video by synthesizing new frames between existing ones. It is a fundamental problem in computer vision that involves the understanding of motions, structures, and natural image distributions, which facilitates numerous downstream applications. VFI can be used in novel view synthesis, video compression, video restoration, and slow-motion generation applications. The referenced research can roughly be categorized into two groups, motion-free and motion-based, depending on whether or not cues like optical flow are incorporated. Motion-free models typically rely on kernel prediction or spatiotemporal decoding, which is effective but limited to interpolating frames at fixed time steps and their runtime increases linearly in the number of desired output frames. On the other end of the spectrum, motion-based approaches establish dense correspondences between frames and apply the warp to render the intermediate pixels. Kernel-based methods estimate a kernel for each pixel in the frame indicating the weights for its neighboring pixels to synthesize the new pixel. They generate the synthesized frame in a local convolutional manner. Phase-based methods generate intermediate frames by per-pixel phase modification. Currently, most methods are flow-based, and typically, their networks contain two modules: an optical flow module and a frame-synthesized module. The optical flow is the process of finding pixel-wise motions between sequences of frames. Motion vectors are attached to each pixel to indicate where the pixel is moving to. By warping the input frame based on the estimated optical flow, the frame synthesized module synthesizes the intermediate frames. VFI works target temporal super-resolution on the premise that the frame rate of the input sequence is often already sufficiently high. Researchers have experimentally proven that state-of-the-art(SOTA) methods suffer in overall performance while dealing with low frame rates ($<$ 30fps). Capturing long-range dependencies is of central importance in video interpolation. Transformers, which are initially designed for natural language processing (NLP) to efficiently model long-range dependencies between input and output, naturally overcome the above drawbacks of CNN-based algorithms, and are in particular suitable for the task of video interpolation [1]. Motivated by the success of NLP, several methods recently adapted Transformers in computer vision and demonstrated promising results on various tasks, such as image classification, semantic segmentation, object detection, and 3D reconstruction. Nevertheless, how to effectively apply Transformers to video interpolation that involves an extra-temporal dimension remains an open yet challenging problem. To evaluate the quality of video frame interpolation results, most interpolation methods rely on traditional metrics, such as Peak Signal-to-Noise Ratio (PSNR) and Structural Similarity Index Measure (SSIM) [2]. Some methods also explore Interpolation Error (IE), Normalized Interpolation Error, and Learned Perceptual Image Patch Similarity (LPIPS) [3] to measure their results.

The images or videos of the natural world are not random. They generally obey the natural scene statistics [4], which is generally used in image and video quality assessment algo-

[*]These authors contributed equally to this work

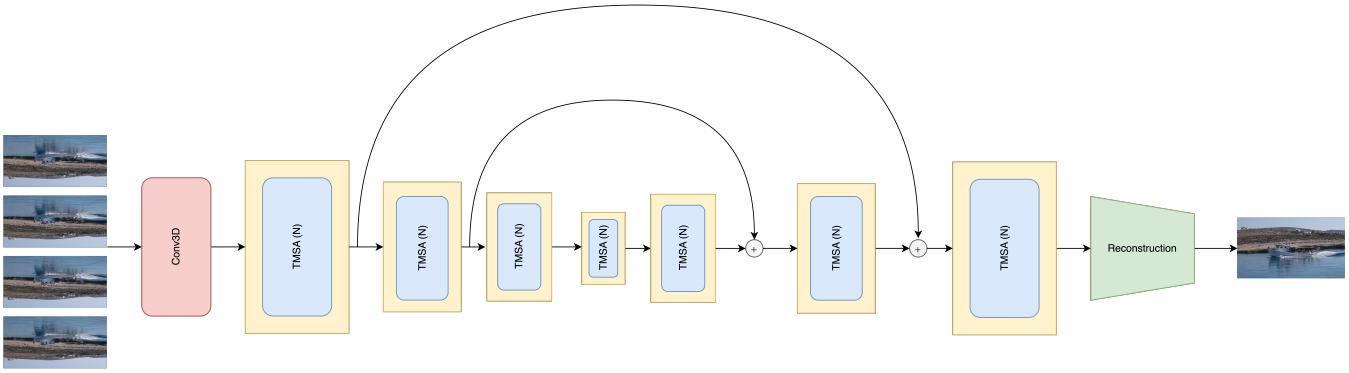

Fig. 1: Block diagram of the proposed architecture of our model.

rithms [5], [6], [7] etc. In video quality assessment, in its early stages, attempts were made to understand the motion vectors and to see if they have any pattern or follow any underlying statistics. Till now, no such pattern or statistics have been identified.

In our work, we try to leverage one idea from physics that objects do follow, which is 'Inertia.' Newton's first law, or the law of inertia, states that an object at rest remains at rest, and an object in motion remains in motion at a constant speed and in a straight line unless acted on by an unbalanced force. As we are dealing with natural videos, we can reasonably assume that the objects of a video do follow the laws of inertia.

From the task at hand, i.e., video interpolation, we have some constraints to get efficient frame predictions. The conditions are as follows:

- The frame rate of the video is high enough so that we can capture subtle changes in movements.
- No frames of video are skipped during the prediction of the video frame.

Considering a set of frames from a video with a frame-rate high enough, a fraction of a second of the video will inadvertently obey Newton's laws. Also, we need a video in which frames are not skipped, as we want to capture the motion details.

In the following sections, we explain some of the prominent related work in the video interpolation field; then, we move on to our proposed model, training methodology, and results.

2. RELATED-WORK

Existing approaches can be classified into three broad categories. Phase-based, Kernel-based, and Motion-based ones.

Phase-based methods represent motion in the phase shift of individual pixels. They interpolate phase and amplitude across the levels of a multi-scale pyramid through optimization [8] or neural networks [9]. These methods are constrained in their solution with their limitation to limited range motion. Kernel-based methods jointly perform motion estimation and motion compensation in a single step. Niklaus et al. [10] estimate a spatially-adaptive convolution kernel for each pixel using a convolutional network. The intermediate frame is then generated by convolving the input frames with the predicted kernels. Newer methods propose the use of separable convolutions [10] to reduce network parameters, adopting deformable convolution followed by estimation of both kernel weights and offset vectors [11], integrating an optical flow and interpolation kernels together [12] to improve the performance.

Often all these methods produce blurry outputs, especially in the case of high-motion frames, mainly because learning-based methods tend to generate pixels directly. Besides, to handle large motion, the estimated kernels are designed to be large, leading to a large number of parameters to learn. For motion-based methods, optical flow is estimated to warp the input frames. Liu et al. [13] introduce a deep network that produces 3D optical flow vectors across space and time and warps input frames by trilinear sampling. More methods use contextual information to interpolate high-quality results [14], forward warping input frames using softmax splatting [15]. A consistent pattern with CNN is that it struggles in modeling long-term dependencies, thus limiting large motion handling.

Transformer: Transformer was first proposed by Vaswani et al. [16] for machine translation. It consists of stacked self-attention layers for modeling dense relations among input tokens and has shown great flexibility. Since then, it has been adopted in computer vision applications vastly. Carion et al. [17] propose an end-to-end detection Transformer (DETR) for direct set prediction. Dosovitskiy et al. [18] propose ViT, which is a pure Transformer for image classification and achieves decent results. Liu et al. [19] present a general-

purpose backbone called Swin Transformer, which achieves linear computational complexity by computing self-attention within non-overlapping windows. A shifted window scheme is also proposed for cross-window connection. Not much work is present that exploits the transformers for video frame interpolation, except more recently, zhihao et al. [20] propose a transformer-based approach with local attention in the spatiotemporal domain.

3. METHOD:

Consider a video $V \in \mathbb{R}^{T\times 3\times H\times W}$ be a sequence of input frames that are input to the model. Video Restoration Transformer (VRT) [21] has achieved well on tasks like video super-resolution, deblurring, denoising, and video interpolation. VRT proposes to use mutual attention for joint feature alignment and fusion. It adaptively utilizes features from supporting frames and fuses them into the reference frame, which can be regarded as implicit motion estimation and feature warping VRT has several temporal mutual self-attention (TMSA) modules followed by a parallel warping module. In TMSA, mutual attention is focused on mutual alignment between neighboring two-frame clips, while self-attention is used for feature extraction. At the end of each scale, parallel warping is used to fuse neighboring frame information into the current frame.

We use a modified version of VRT without any parallel wrapping and optical flow. By this approach, we demonstrate the essence of temporal mutual self-attention and self-attention in frame interpolation without the necessity of computation of optical flow, which is one of the most time-consuming processes. The following explains the key components of our model.

3.1. Overall Framework

At the beginning, we are going to use a 3D convolution to extract low-level features of the video, followed by multiple TMSA and Self-Attention layers. At the end, we use a 2D convolution layer to predict the frame. We use U-Net type architecture to create a balance between the spatial resolution of features and no.of feature maps.

3.2. Temporal Mutual Self Attention

Given a reference frame $X_R \in \mathbb{R}^{N\times C}$ and supporting frame $X_S \in \mathbb{R}^{N\times C}$ where N is the number of feature elements and C is the channel number, we compute the query Q_R, key K_S and value V_S from linear projections of X_R and X_S.

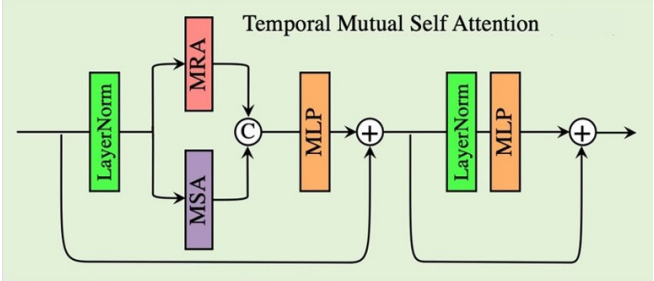

Fig. 2: Temporal Mutual Self Attention (TMSA) layer.

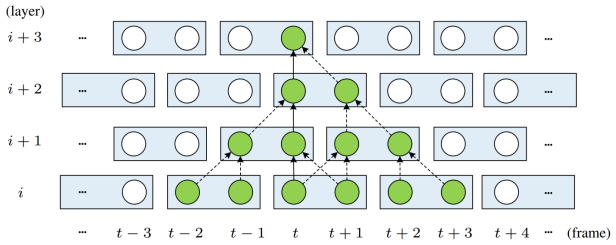

Fig. 3: The above figure shows a stack of temporal mutual self attention (TMSA) layers. The sequence is partitioned into 2-frame clips at each layer and shifted for every other layer to enable cross-clip interactions. Dashed lines represent information fusion among different frames.

$$Q_R = X_R P_Q \tag{1}$$
$$K_S = X_S P_K \tag{2}$$
$$V_S = X_S P_V \tag{3}$$

where $P_Q, P_K, P_V in \mathbf{R}^{C\times D}$ are projection matrices, and D is no.of projected features. We calculate the mutual attention as follows:

$$A = \text{Softmax}(\frac{Q_R K_S^T}{\sqrt{D}}) \tag{4}$$
$$\text{MA}(Q_R, K_S, V_S) = \text{Softmax}(\frac{Q_R K_S^T}{\sqrt{D}})V_S \tag{5}$$
$$Y_{i,:}^R = \sum_{j=1}^{N} A_{i,j} V_{j,:}^S \tag{6}$$

$Y_{i,:}^R$ refers to the new feature of the i-th element in the reference frame. This equals to image warping given an optical flow vector. When $A_{i,j} \rightarrow 1$ does not hold, Eqn.5 can be regarded as a "soft" version of image warping. In practice, the reference frame and supporting frame can be exchanged, allowing mutual alignment between two frames. Besides, similar to multi-head self-attention, we can also perform the attention for h times and concatenate the results as multi-head

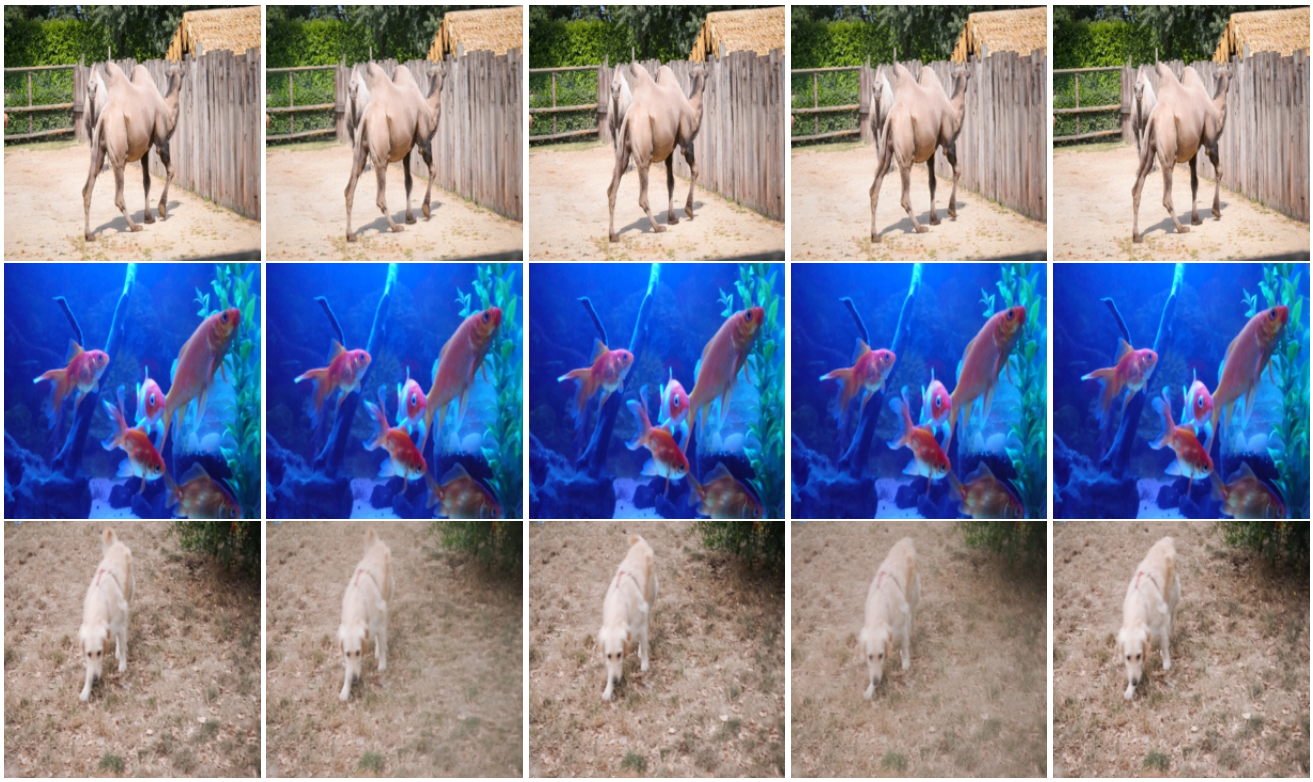

Fig. 4: The first column of images are last frame of video V_1, the second column images are predictions of model for V_1 as input, the third column of images are our targets, the fourth column are predictions of model for V_{-2} as input and last column images are the first frame of video V_2.

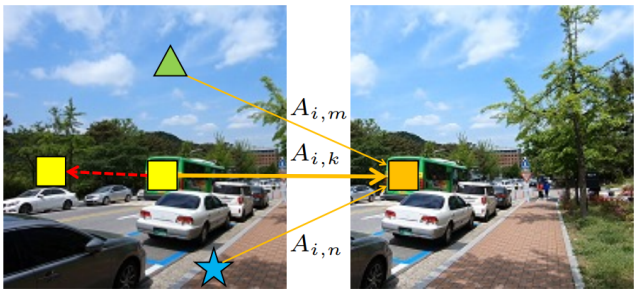

Fig. 5: The orange square (the i-th element of the reference frame) query elements in the supporting frame and use their weighted features as a new representation for the orange square. The weights are shown around solid arrows (we only show three examples for clarity). When $A_{i,k} \to 1$ and the rest $A_{i,j} \to 0 (j \neq k)$ the mutual attention equals to warping the yellow square to the position of the orange square (illustrated as a dashed arrow).

mutual attention (MMA). First, mutual attention can adaptively preserve information from the supporting frame than image warping, which only focuses on the target pixel. It also avoids black hole artifacts when there is no matched positions.

To extract and preserve features from the current frame, we use mutual attention together with self-attention. Let $X \in \mathbb{R}^{2\times N\times C}$ represent two frames, which can be split into $X_1 \in \mathbb{R}^{1\times N\times C}$ and $X_2 \in \mathbb{R}^{1\times N\times C}$.

$$X_1, X_2 = \text{Split}_0(\text{LN}(X)) \tag{7}$$
$$Y_1, Y_2 = \text{MMA}(X_1, X_2), \text{MMA}(X_2, X_1) \tag{8}$$
$$Y_3 = \text{MSA}([Y_1, Y_2]) \tag{9}$$
$$X = \text{MLP}(\text{Concat}_2(\text{Concat}_0(Y_1, Y_2), Y_3)) + X \tag{10}$$
$$X = \text{MLP}(\text{LN}(X)) + X \tag{11}$$

where the subscripts of Split and Concat refer to the specified dimensions. However, due to the design of mutual attention can only deal with two frames at a time.

One naive way to extend the above equations for 'T' frames is to deal with frame-to-frame pairs exhaustively, resulting in the computational complexity of $O(T^2)$. TMSA first partitions the video sequence into non-overlapping 2-frame clips and then applies the above equations to them in parallel. As shown in Fig. 3, it shifts the sequence temporally by 1 frame for every other layer to enable cross-clip con-

nections, reducing the computational complexity to $\mathcal{O}(T)$. The temporal receptive field size is increased when multiple TMSA modules are stacked together. Specifically, at layer $i(i \geq 2)$, one frame can utilize information from up to $2(i-1)$ frames.

4. TRAINING PROCEDURE

Consider a video $V \in \mathbb{R}^{T\times 3\times H\times W}$ be a sequence of input frames. Our objective is to predict the next frame $F \in \mathbb{R}^{1\times 3\times H\times W}$ which follows the law of inertia just by looking the past frame but not the future frames. The generally approach is to design a model that predicts the next F when video V is given as input. But we want the predicted frame such that it is actually correlates to the next set of frames. So, instead of using the general approach mentioned above, we decided to leverage the next set of frames or the next few seconds of the video that are generally available. The following is our training procedure:

We split the video V into two parts, $V_1 \in \mathbb{R}^{T_1\times 3\times H\times W}$ and $V_2 \in \mathbb{R}^{T_2\times 3\times H\times W}$ and we set our objective to predict the frame that lies between V_1 and V_2 i.e if video V has frames of time $\{t_a, t_{a+1}, ..., t_i, t_{i+1}, ...t_{T+a-1}\}$, video V_1 has frames of time $\{t_a, t_{a+1}, ..., t_i\}$ and video V_1 has frames of time $\{t_{i+1}, t_{i+2}, ..., t_{T+a-1}\}$. Next, we reverse the frames of the video V_2 and let's call it V_{-2}. We can understand that video V_{-2} is like to watch the video from back to start. So, the predicted next frames by the model when V_1 and V_{-2} are passed as input F_1 and F_2 respectively should be close to each other and the true target frame.

Let $f(.)$ be the model, then the predicted frame are:

$$F_1 = f(V_1) \tag{12}$$
$$F_2 = f(V_{-2}) \tag{13}$$

If F be the target predicted frame, then the training objective:

$$\min_{F_1,F_2} c_1 l(F_1, F_2) + c_2 l(F, F_1) + c_3 l(F, F_2) + c_4 \|F_1 - F_2\|^2 + c_5 \|F - F_1\|^2 + c_6 \|F - F_2\|^2 \tag{14}$$

where $l(.)$ is the perceptual loss function [22] and $\|.\|$ is l2 loss function. By providing only previous-frames as input, we are making sure the model's prediction of next-frame to obey causality and by modifying the loss function to utilize future frame information we are making sure the predicted frame is not only close to target-frame but also continuity with the future frames.

5. EXPERIMENTS AND RESULTS

Dataset: We have considered videos from DAVIS dataset [23] for training our model. We have set the parameters

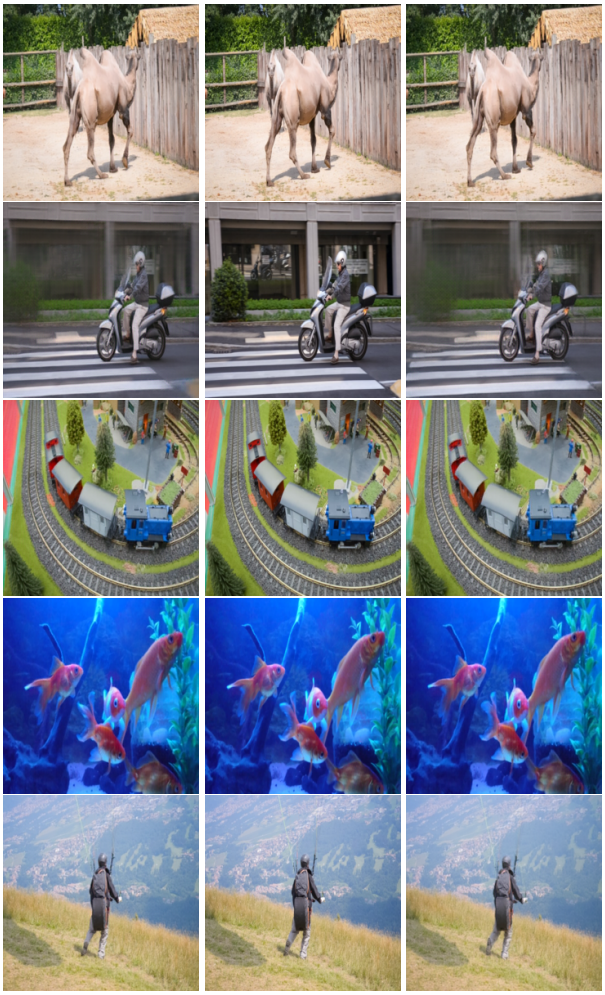

Fig. 6: The left column of images are predictions of the model for V_1 as input, the right column of images are predictions of the model for V_{-2} as input and the middle row are targets to the model during training-stage.

T_1 and T_2 to be 4. We divided video is divided into non-overlappping sets of 9 frames where 8 are used for video splits V_1 and V_2 and the middle frame is used as a target. We have decided to consider non-overlapping frames to make sure our model doesn't overfit on the content. During training we set the spatial-resolution of our video to be 240×240.

Training: Our model has 19 million parameters and our dataset has 644 total samples. We split the dataset into two parts with a validation split ratio of 0.2. We trained our model with a learning-rate 0.0005, with AdamW optimizer with a weight-decay of 0.05, momentum 0.9 for 300 epochs. We used 9 Nvidia A-100 GPUs with batch-size of 1 on each GPU. We used DDP training strategy and it took around 5 hours for training. PSNR and SSIM have been used as evaluation metrics, and we achieved a validation PSNR of 22.14 and SSIM of 0.864.

Table 1: Quantitative Comparison on DAVIS dataset.

Method	#Parameters (M)	Davis	
		PSNR (↑)	SSIM (↑)
SuperSloMo[24]	39.6	25.65	0.857
FLAVR[25]	42.4	27.44	0.874
QVI[26]	29.2	27.17	0.874
VFIT[20]	29.0	**28.09**	**0.888**
Ours	**19**	22.14	0.864

Fig.6 and fig.4 show qualitative results and confirm that our model is learning to predict the next subsequent frame, and on close observation, we can see that the predictions of our model are close to the target, but still, there exist very subtle differences and yet they are continuous with the next set of frames. Table 1 shows the quantitative results and comparison with other SOTA methods, we have the lowest parameters with comparable SSIM.

6. LIMITATIONS AND FUTURE WORK:

While the current state-of-the-art methods use more than 70,000 raw videos, we restricted ourselves to only 300 raw videos due to the scope of the project and time constraints. Despite being trained on the limited dataset from scratch, our model performed reasonably well in generating temporally consistent frames. We believe that simply increasing the training samples could greatly benefit the overall performance and generalizability. Furthermore, we could not capture diversity in contents and scenes with our small dataset. This results in failure cases such as poor performance on low-resolution input; a high-motion and high-camera motion scenes input produce a lot of judders and flickering in the predicted frames. Again, a more diverse and larger training dataset can mitigate such limitations. We also observe empirically that giving more than three frames as input does not always help the model to estimate the motion, especially in high-camera motion scenes.

In order to further improvise on our method and to surpass current state-of-the-art methods, we propose the following:

- Design a novel architecture module to efficiently capture optical flow features to reduce the reliability on multiple input frames, it will also reduce the memory overhead.
- Diversifying the training to data includes larger motion within the scene and camera motion scenes.
- Training on a larger corpus of data to improve overall performance.

7. REFERENCES

[1] Liying Lu, Ruizheng Wu, Huaijia Lin, Jiangbo Lu, and Jiaya Jia, "Video frame interpolation with transformer," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2022, New Orleans, LA, USA, June 18-24, 2022*. 2022, pp. 3522–3532, IEEE.

[2] Zhou Wang, Alan C. Bovik, Hamid R. Sheikh, and Eero P. Simoncelli, "Image quality assessment: from error visibility to structural similarity," *IEEE Trans. Image Process.*, vol. 13, no. 4, pp. 600–612, 2004.

[3] Richard Zhang, Phillip Isola, Alexei A. Efros, Eli Shechtman, and Oliver Wang, "The unreasonable effectiveness of deep features as a perceptual metric," in *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*. 2018, pp. 586–595, Computer Vision Foundation / IEEE Computer Society.

[4] Daniel L. Ruderman, "The statistics of natural images," *Network: Computation In Neural Systems*, vol. 5, pp. 517–548, 1994.

[5] Anish Mittal, Anush Krishna Moorthy, and Alan Conrad Bovik, "No-reference image quality assessment in the spatial domain," *IEEE Transactions on Image Processing*, vol. 21, no. 12, pp. 4695–4708, 2012.

[6] Michele A. Saad, Alan C. Bovik, and Christophe Charrier, "Blind prediction of natural video quality," *IEEE Transactions on Image Processing*, vol. 23, no. 3, pp. 1352–1365, 2014.

[7] Joshua Peter Ebenezer, Zaixi Shang, Yongjun Wu, Hai Wei, Sriram Sethuraman, and Alan C. Bovik, "Chipqa: No-reference video quality prediction via space-time chips," *IEEE Transactions on Image Processing*, vol. 30, pp. 8059–8074, 2021.

[8] Simone Schaub-Meyer, Oliver Wang, Henning Zimmer, Max Grosse, and Alexander Sorkine-Hornung, "Phase-based frame interpolation for video," *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1410–1418, 2015.

[9] Simone Schaub-Meyer, Abdelaziz Djelouah, Brian McWilliams, Alexander Sorkine-Hornung, Markus H. Gross, and Christopher Schroers, "Phasenet for video frame interpolation," *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 498–507, 2018.

[10] Simon Niklaus, Long Mai, and Feng Liu, "Video frame interpolation via adaptive convolution," *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2270–2279, 2017.

[11] Zhihao Shi, Xiaohong Liu, Kangdi Shi, Linhui Dai, and Jun Chen, "Video interpolation via generalized deformable convolution," *ArXiv*, vol. abs/2008.10680, 2020.

[12] Wenbo Bao, Wei-Sheng Lai, Xiaoyun Zhang, Zhiyong Gao, and Ming-Hsuan Yang, "Memc-net: Motion estimation and motion compensation driven neural network for video interpolation and enhancement," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 43, pp. 933–948, 2018.

[13] Ziwei Liu, Raymond A. Yeh, Xiaoou Tang, Yiming Liu, and Aseem Agarwala, "Video frame synthesis using deep voxel flow," *2017 IEEE International Conference on Computer Vision (ICCV)*, pp. 4473–4481, 2017.

[14] Simon Niklaus and Feng Liu, "Context-aware synthesis for video frame interpolation," *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 1701–1710, 2018.

[15] Simon Niklaus and Feng Liu, "Softmax splatting for video frame interpolation," *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5436–5445, 2020.

[16] Ashish Vaswani, Noam M. Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin, "Attention is all you need," *ArXiv*, vol. abs/1706.03762, 2017.

[17] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko, "End-to-end object detection with transformers," in *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part I 16*. Springer, 2020, pp. 213–229.

[18] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al., "An image is worth 16x16 words: Transformers for image recognition at scale," *arXiv preprint arXiv:2010.11929*, 2020.

[19] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo, "Swin transformer: Hierarchical vision transformer using shifted windows," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2021, pp. 10012–10022.

[20] Zhihao Shi, Xiangyu Xu, Xiaohong Liu, Jun Chen, and Ming-Hsuan Yang, "Video frame interpolation transformer," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 17482–17491.

[21] Jingyun Liang, Jiezhang Cao, Yuchen Fan, Kai Zhang, Rakesh Ranjan, Yawei Li, Radu Timofte, and Luc Van Gool, "VRT: A video restoration transformer," *CoRR*, vol. abs/2201.12288, 2022.

[22] Justin Johnson, Alexandre Alahi, and Li Fei-Fei, "Perceptual losses for real-time style transfer and super-resolution," in *Computer Vision - ECCV 2016 - 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part II*, Bastian Leibe, Jiri Matas, Nicu Sebe, and Max Welling, Eds. 2016, vol. 9906 of *Lecture Notes in Computer Science*, pp. 694–711, Springer.

[23] Jordi Pont-Tuset, Federico Perazzi, Sergi Caelles, Pablo Arbeláez, Alexander Sorkine-Hornung, and Luc Van Gool, "The 2017 davis challenge on video object segmentation," *ArXiv*, vol. abs/1704.00675, 2017.

[24] 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*, 2018, pp. 9000–9008.

[25] Tarun Kalluri, Deepak Pathak, Manmohan Chandraker, and Du Tran, "Flavr: Flow-agnostic video representations for fast frame interpolation," *2023 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pp. 2070–2081, 2020.

[26] Xiangyu Xu, Liu Siyao, Wenxiu Sun, Qian Yin, and Ming-Hsuan Yang, "Quadratic video interpolation," in *Neural Information Processing Systems*, 2019.