Decoding Motion-blur using Variational Autoencoder

Sushlok Ajay Shah
Visvesvaraya National Institute of Technology, Nagpur

Sushant Jha
Visvesvaraya National Institute of Technology, Nagpur

Saugata Sinha
Visvesvaraya National Institute of Technology, Nagpur

***Abstract*—Applications of camera-based systems are increasing day by day due to the development in both hardware as well as the algorithmic side. One of the major limiting factors to the extension of these systems in some fields with high-speed applications like automation, self-driving automobiles, etc., is the exposure time of the camera to capture images, which results in motion blur. Although this limitation can be tackled by utilizing suitable hardware to improve the camera's frame rate, it will increase the development costs. This paper introduces a novel learning-based solution for the combined task of video deblurring and intermediate frame generation from a single motion blur image and its corresponding first sharp image. This designed method can be used for low-frame-rate to high-frame-rate video conversion without any change in the existing hardware. The proposed network accurately predicts the structure from a clear image as well as the motion of objects in the blur frame and uses this information to construct the frames in the precise sequence. The Experimental results show that our algorithm performs substantially well on the GoPro dataset for blur decoding and frame interpolation task.**

***Index Terms*—Video Deblurring and intermediate frame generation, Blur decoding,**

I. Introduction

Due to their reasonable cost and wide-ranging applications, cameras have become very popular means to capture information in the form of images and videos. In many applications, such as remote-controlled and self-driving automobiles, video acquisition with a high frame rate is of utmost importance. Any algorithm that can increase the video's frame rate and improve the quality of each frame can boost the applications in the related fields at the same time also keeping it cost-effective.

Any video is captured in frames, where each frame is just an image; the terms frame and image are used interchangeably in this report. The time for which the light is allowed to fall on the camera sensor for each frame is termed exposure time; a camera with a lower frame rate has a higher exposure time. If any object in the frame moves during this time, it gets blurred in the output frame, and the type of movement can be derived from the characteristics of the motion-induced blur (motion blur). Since videos are meant to capture moving objects, there is always some level of blur. The blur in the video affects the performance of any software which uses this video to make any decisions. The study done by [20], [21] shows the adverse effect of both focus and motion blur on the performance of object detection and segmentation algorithms. The quality of videos can be improved by increasing the frame rate of the output video. Also, a high-fps video will have less blur in every frame than a low-fps video for the same scene. One solution to this problem would be to use a camera with a higher frame rate, but that can be expensive; hence, we look at a more economical approach by using a learning-based algorithm that will take in the raw video and return an improved version of the same video to any software working on top of it.

Some earlier works have generated multiple frames using a single frame from a video; previous works such as [1]–[3], [6], [7] tried to generate a sequence of frames from a single blur frame; hence they do not apply to low-fps to high-fps video translation, as the outputs in these methods did not have any limit on the generation time. These methods were majorly used for future frame prediction rather than enhancement of the entire video, which is the topic of interest for our work. [18] used predefined optical flow, whereas others derived optical flow from looking at previous blur or clear frames. Both methods perform well; hence we can assume that extracting optical flow explicitly can lead to better results. Another improvement to generation came with vision-transformers, which could deal with images. These models retained information while memorizing long-term sequences producing extraordinary results for a long sequence of frame generation.

Other related works, including [8]–[13], may provide significant insight into how we can generate clear output on a per-frame basis. Video deblurring here refers to generating a clear frame corresponding to every blur frame in the video. The input of such models is generally a set of consecutive frames of three or more in number, as in [8] to extract temporal information and use it to enhance the main frame or use the accumulated flow from previous frames as in [9]. Optical flow can also be explicitly extracted from the input frames. The transformation is then based on these extracted optical flows as done by [13]. These methods use structural information from consecutive frames to enhance the quality of the middle frame. [12], a recent work, claims the best Peak-Signal-to-Noise-Ratio (PSNR) among all the video deblurring methods. This method uses a three-stage feature extraction, temporal fusion, and reconstruction process to estimate the precise structure of the objects in the frame under reconstruction. [12] also shows the efficiency of transformers over other methods in video deblurring.

Our work aims to use a motion-blurred frame from any video and create multiple frames with reduced blur for every single frame at the input, thus resulting in a higher fps output. Upon close analysis of the work done in [1]–[19], one can conclude that there are three necessary building blocks of any model used for deblurring and controlled frame generation: motion/flow encoder, positional encoder, and transformation block. The transformation block transforms the input frame to an output frame given some time/position instant, and the flow encoding describes the objects' movement in the frame. Some early works [1], [2] are based on an encoder-decoder architecture with normal distribution as the latent space, and the output frames are sampled from this distribution. [1] approximates this distribution as a standard normal distribution, whereas [2] approximates it with the distribution of the previous frame; this change results in better diversity of images in [2] compared to [1]. Other methods, such as [6], were designed for operation on a single input frame as they do not limit the number of output frames generated, and the outputs may surpass the motion in the next blur frame. They also need help with directional ambiguity since the decoding is based on a single frame, and there is no motion guidance from the past or future frames. [19] solves these problems using all three necessary components of flow, position, and transformation block. However, it uses the previous, current, and next frames for resolving directional ambiguity, which can be done using the previous and current frames alone. There is also some scope for improvement regarding the amount of blur the algorithm can handle while simultaneously maintaining the quality of the generated frames.

So, our contributions can be summarized as follow:

- We propose a novel time-sensitive architecture to decode motion blur into Spatio-temporal consistent sequences of images.
- We presented an efficient and scalable learning-based model for video interpolation from motion blur images which can be used to boost the frame rate of the vision systems.
- We quantitatively and qualitatively evaluate our model against the Go-pro dataset.

II. Methodology

An image $y \in \mathbb{R}^{M \times N}$ captured for an exposure time τ can be written as

$$y = g\left(\frac{1}{\tau}\int_0^{\tau} \tilde{x}(t)dt\right) = g\left(\frac{1}{T}\sum_{i=0}^{T-1} x[i]\right) \tag{1}$$

where g is the camera response function, which relates the irradiance of the image plane with the measured image intensity. $x(t)$ is the instant image (irradiance) at time t. We can discretize the time axis into T segments and define a sequence of frames $x[i]$, with $i = 1, ..., T$. So, if the object or camera moves quickly within the exposure time, then it results into motion blur in the generated image. As we can see from eq (1), this motion blur can be assumed as a discrete sum of several images within exposure time T. As mentioned earlier, we want to decode this blur image into this sequence of images/frames $x[i]$, with $i = 1, ..., T$. For Spatio-temporal consistency and controlled generation, any networks need the following three necessary blocks:

- Flow encoder encodes the flow of every object in the frame using the direction and magnitude of blur for every pixel.
- Positional encoding provides the model with a sense of time so that the generated frames in the temporal domain are located at the time instants encoded in the positional encoding.
- Generator block takes in the flow encoding and uses it to transform the encoding of the previous clear frame. It also uses positional encoding to decide how far the next frame should be from the previous frame in the temporal domain.

On the basis of these blocks, we describe our model architecture in the following section:

A. Proposed architecture:

The generic blur decoding process involves a two-step process which includes deblurring the first frame from the sequence that formed the blur and using this sharp frame for the reconstruction of the remaining frames based on the motion guidance provided by the blurred image. Our work is on the second step of this process with the prior assumption that we have got a sharp image from some source like a deblurring network or some high fps camera. So, to summarize, we aim to reconstruct a sequence of sharp images $X_{1:t} \in \mathbb{R}^{H \times W}$ given the first sharp image of that sequence $X_0 \in \mathbb{R}^{H \times W}$ and blur image $Blur_t \in \mathbb{R}^{H \times W}$.

The overview of our proposed architecture of the blur decoder model is shown in Fig: 1. The network consists of a convolutional encoder-decoder architecture similar to a Variational Autoencoder (VAE) with skip connections, a latent modeling Long Short-Term Memory (LSTM), a Motion/flow encoder, and a positional encoder providing the timestamp information corresponding to an image we want to generate. A detailed description of individual blocks and the training procedure is presented in the subsequent subsections.

*1) **Feature Encoder and Decoder**:* This block is similar to most autoencoder architectures, where the encoder takes an image as input and represents it as a compact latent feature representation $F_t \in \mathbb{R}^{D_f \times H' \times W'}$. A pyramidal feature extraction network is used as an encoder, and features at different scales or levels are stored. The decoder utilizes them to reconstruct the required sharp image. The encoder provides the spatial information about the current sharp image to the latent modeling network, which transforms it into a new latent corresponding to the required sharp image.

*2) **Positional Encoder**:* To make the network aware of the temporal location of the image to be generated, we use positional encoding, which encodes the timestamp of

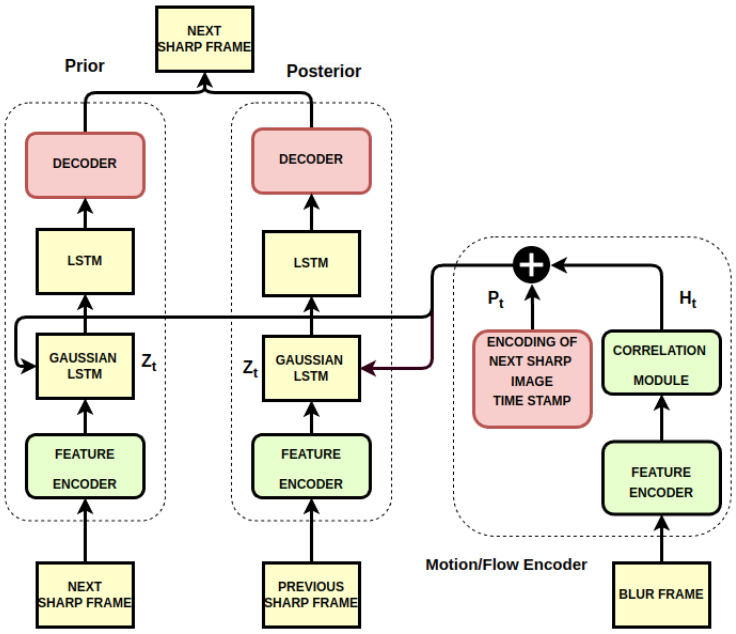

Fig. 1. Proposed model for Blur decoding

generation, the temporal distance of the last frame from the current frame, and current time embedding. The current time embedding $\tau \in \mathbb{R}^{N\times 1}$ is used to represent current time information where the n values of the embedding vector are sampled from n sinusoidal functions, each with exponentially different frequencies.

*3) **Motion or Flow Encoder**:* As discussed earlier, the feature encoder provides spatial information about the scene. However, to reconstruct the following frames, we also need to provide information about the temporal motion of features over time. This is achieved by using a flow encoder.

Inspired by the performance of the current learning-based optical flow methods and their ability to encode temporal variation of feature patterns, [18] tried to use the optical flow maps generated from the blurred image as a motion guide to generate images. [17] tried to linearly sample optical flow maps generated by blur images to generate a sharp image.

However, there are better ways to approximate the projective transformation of spatial information and to encode complete information about the motion over time rather than just end-to-end optical flow maps. The optical flow map encodes the end-to-end motion of a feature starting from frame X_0 to end frame X_T or vice versa. So the intermediate non-linear motion is lost with such encoding. Also, the optical flow maps estimated from blur images are very noisy. So, instead of encoding the complete motion as an optical flow map, we first estimated features using the same encoder and used correlation volume computed from the encoded features to encode the movement that occurred over time.

This correlation volume $H_t \in \mathbb{R}^{D_h\times H'\times W'}$ provides the complete motion history of the feature, which is further used by the latent modeling network to generate latent embedding corresponding to the required frame.

Finally, the positional encoding P_τ, flow encoding H_t, and feature encoding F_t are used as inputs by the latent modeling network for predicting the latent corresponding to the frame to be generated.

*4) **Latent Modeling Network**:* This network is the central part of the complete network; it is responsible for generating a temporally and spatially consistent frame $x_{t+\tau}$ from the input image x_t, the motion history H_t, and corresponding positional encoding P_τ corresponding to the time instant $\tau \in [0..1]$ specified by the user.

Standard Variational Autoencoder (VAE) assumes a prior distribution $p(z)$, and an encoder function is introduced, which tries to model $p(z \mid x)$ to approximate the intractable input prior distribution $p(x)$. Then VAE enforces a distribution constraint over the latent space and trains the network to maximize the reconstruction likelihood $p(x \mid z)$. So, the encoder aims to map the input distribution to the latent distribution, and the decoder samples a latent sample from this predefined distribution to reconstruct input data. The complete network is trained by maximizing the variational lower bound specified by the following inequality:

$$\log p_\theta(\mathbf{x}) = \log \int_{\mathbf{z}} p_\theta(\mathbf{x} \mid \mathbf{z}) p(\mathbf{z}) d\mathbf{z}$$
$$\geq \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})} \log p_\theta(\mathbf{x} \mid \mathbf{z}) - D_{\mathrm{KL}}\left(q_\phi(\mathbf{z} \mid \mathbf{x}) \| p(\mathbf{z})\right).$$

where ϕ and θ are the learnable network parameter of the encoder and decoder respectively.

Here we use a modified version variational lower bound to generate a sharp image $x_{t+\tau}$ given feature encoding of sharp image F_t, flow encoding H_t, and positional encoding P_τ. This modified latent processing is modeled using a latent modeling network consisting of two LSTMs:

- Gaussian LSTM takes in feature encoding, positional encoding, and flow encoding and generates the standard normal distributed latent encoding Z_t.
- Decoding LSTM takes Z_t as input to generate encoding of the frame we want to generate, i.e. h_{t+1}.

The model is optimized by maximizing the modified variational lower bound for the above constraints:

$$\mathcal{L}_{\theta,\phi,\psi}\left(x_{1:T}\right) = \sum_{\tau=1}^{T}\left[\mathbb{E}_{q_\phi(z_{1:\tau}|x_{1:\tau})}\log p_\theta\left(x_\tau \mid F(x_1), z_{1:\tau}\right)\right.$$
$$\left. -D_{\mathrm{KL}}\left(q_\phi\left(z_{1:\tau} \mid x_{1:\tau}\right)||p_\psi\left(z_\tau \mid F_t(x_1), H_t, P_\tau\right)\right)\right].$$

B. Dataset

In our work, we use the GoPro dataset [22], which consists of images taken from videos of different scenes at 240 fps. Each scene is different in terms of illuminance and objects recorded. The diversity in the scenes allows us to test our model for both cases where the moving objects are small and large.

The dataset for blur frames is obtained using sequences from the same dataset. A blur can be represented as the integration of all the light falling on the camera sensor during the exposure time. However, blur can also be approximated to a discrete summation of a sequence of frames, given that the exposure time for each frame is relatively large compared to interframe spacing. Considering the high frame rate of 240fps of the GoPro dataset, it is fair to assume that eq (1) for continuous and discrete time cases is very close when the number of frames is large.

C. Training and Results

The GoPro dataset was used to generate training and testing samples for our model; as mentioned earlier, due to the high fps of 240fps, we can develop a variable-length realistic blur by combining multiple frames from the given dataset. We used a similar strategy for generating blur frame and used the corresponding clear frames for training the generation sequence. For good motion blur, we experimented by combining 16-20 frames to generate a corresponding motion-blurred image. To make the network aware of the temporal encoding or the positional encoding, we generated sequences of variable length ranging from 4 frames to 16 frames.

All the functional blocks mentioned in the proposed architecture section are trained by training two separate networks parallelly. The two networks are posterior and prior labeled in fig 1. The prior is trained to encode the ground truth image $x_{t+\tau}$ which we want to generate, flow encoding Ht, and its current encoding P_τ into a latent representation $z_{t+\tau}$. Posterior is the network that takes F_t feature encoding of the current image x_t, flow encoding H_t, and positional encoding with required generation time p_τ and try to mimic the prior latent distribution. In other words, it tries to generate the same or similar latent to that of prior, which corresponds to image $x_{t+\tau}$

The complete model is trained using the two loss function given as follows:

- **Posterior loss:**

$$\mathcal{L}^{\text{posterior}}_{\theta,\phi,\psi}\left(x_{1:T} \mid \mathbf{c}\right) =$$
$$\alpha_{gen}.Generation\ loss + \beta.alignment\ loss$$
$$+ \alpha_{\text{last}}.Last\ frame\ generation\ loss$$

- **Prior loss:**

$$\mathcal{L}^{\text{posterior}}_{\theta,\phi,\psi}\left(x_{1:T} \mid \mathbf{c}\right) =$$
$$\alpha_{gen}.Generation\ loss + \beta_{var}.KLdivergence$$

where, $\mathbf{c} = [F_t, H_t, P_\tau]$ denotes the input information provided by the feature encoder, flow encoder, and positional encoder.

$$Generation\ loss$$
$$= -\sum_{\tau=1}^{T}\left[\mathbb{E}_{q_\phi(z_{1:\tau}|x_{1:\tau})}\log p_\theta\left(x_t \mid x_{1:\tau-1}, z_{1:\tau}, \mathbf{c}\right)\right.$$
$$= \sum_{\tau=1}^{T} MSELoss(x_t^{gen}, x_t) + \mid 1 - SSIM(x_t^{gen}, x_t) \mid$$

Generation loss controls the reconstruction quality of the output frame. Here we are using Mean Squared Error (MSE) loss and Structural Similarity Index (SSIM) loss to check the reconstruction quality.

$$Alignment\ loss =$$
$$\sum_{\tau=1}^{T} D_{\mathrm{KL}}\left(q_\phi\left(z_{1:\tau} \mid x_{1:t}, \mathbf{c}\right)\|p_\psi\left(z_t \mid x_{1:t-1}, \mathbf{c}\right)\right)$$

Alignment loss is used to check the prior and posterior latent distribution alignment. This is checked by measuring the KL divergence of the two latent distributions. KLdivergence loss is the same as that of alignment loss, except we align the generated latent distribution of prior with the standard normal distribution. And finally,

$$Last\ frame\ generation\ loss$$
$$= \mathbb{E}_{p_\psi(z_T|x_{1:t-1},\mathbf{c})}\log p_\theta\left(x_T \mid x_{1:T-1}, z_{1:T}, \mathbf{c}\right)$$
$$= MSELoss(x_T^{gen}, x_T)$$

based on the training procedure described, we present the following quantitative and qualitative results.

III. Discussion

From Table 1, it is evident that our model is able to achieve results that are at par with the current work in this field. We were able to successfully use LSTMs for storing data related to the flow in the latent space for multiple frame generation. We obtained very good PSNR and SSIM values for the frames at earlier stages of the sequence while these values started to degrade significantly as we moved towards the end of the

sequences	metric	timestamp							avg metric
		0	1	2	3	4	5	6	
GOPR0372_07_00	MSE	0.0189711	0.03663569	0.05593198	0.07643994	0.09870185	0.12320533	0.15056915	0.080065
	PSNR	34.97657487	29.5842955	25.8592133	23.0506326	20.7432968	18.7383673	16.9244924	24.26812
	SSIM	0.74325951	0.6144255	0.524549	0.4635137	0.421044	0.3902972	0.3672129	0.50347
GOPR0384_11_00	MSE	0.01305308	0.01996679	0.02957881	0.04108709	0.054021	0.06810068	0.08309436	0.04412
	PSNR	37.69067334	34.01546509	30.6166635	27.76929963	25.39301983	23.37834853	21.64593637	28.6442
	SSIM	0.73049236	0.63762481	0.55650443	0.48792115	0.43076788	0.38290215	0.34245003	0.50980
GOPR0384_11_01	MSE	0.00991178	0.01805535	0.02868421	0.04079182	0.0540075	0.06809441	0.08289588	0.04320
	PSNR	40.34775611	35.5413735	31.6090365	28.5005239	25.9853338	23.8952757	22.1173886	29.7138
	SSIM	0.76365979	0.65900395	0.56927809	0.49785476	0.44055837	0.39367026	0.35482577	0.52555
GOPR0881_11_00	MSE	0.00869714	0.0138232	0.0201131	0.026728	0.0333521	0.0398472	0.0462179	0.026968
	PSNR	41.30489099	37.45182627	34.26276965	31.79074555	29.84221361	28.26887051	26.95477824	32.8394
	SSIM	0.83069768	0.75362406	0.68183924	0.62455409	0.58074199	0.54717308	0.52072373	0.648479

TABLE I

EVALUTION RESULT OF OUR PROPOSED MODEL ON SEQUENCES FROM GOPRO DATASET FOR VIDEO GENERATION FROM BLUR

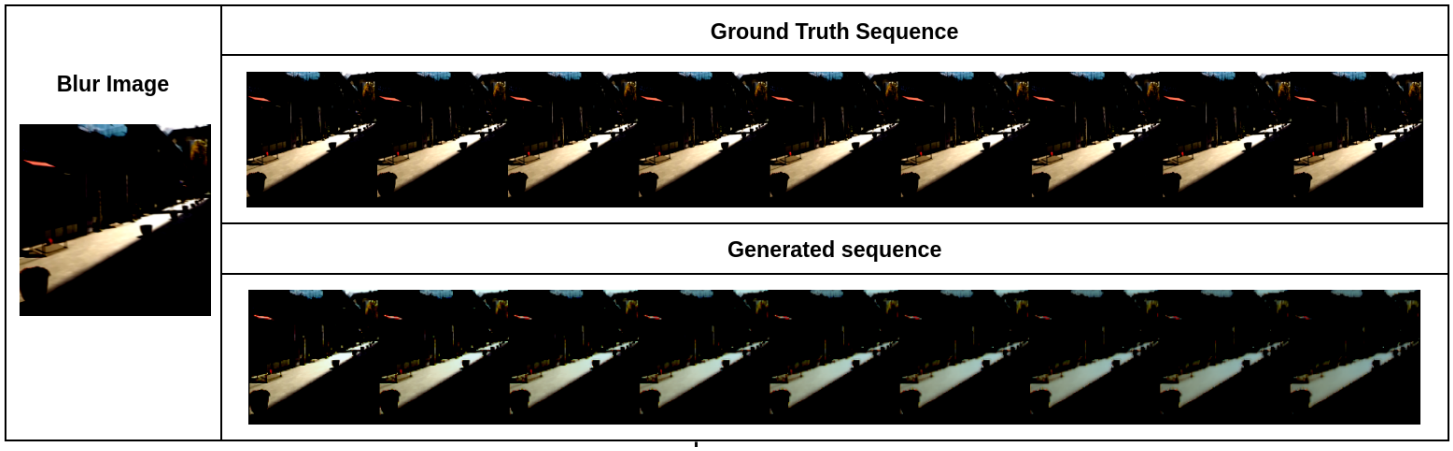

Fig. 2. Qualitative result of our proposed blur decoding model GoPro dataset

sequence. This is especially prominent when the length of the generated sequence is larger than 5-6 frames.

We used a variational autoencoder for video generation by using a time-varying latent space. Two encoders were involved in the process, one to encode the blur frame and the other to encode the previous sharp frame. The training procedure was complicated since we tried to train two networks, the prior and the posterior, to mimic each other. Since this training involved multiple loss functions, appropriate weight to each of them was given to balance between diversity and quality of the output frames.

We observed that the quality of reconstructed frames from the variational autoencoder could have been better if a bigger latent space would have been used in our architecture. To improve the results, we added skip connections between the encoder and decoder at different scales. Further improvements can be made by increasing the dimension of latent space.

In future, we aim to compare our architecture with other works in the same field. Keeping in mind the importance of image quality, we will use the diffusion model for decoding to get better-quality frames.

IV. CONCLUSION

We designed a model for both video deblurring and intermediate frame generation tasks. Our architecture is derived from the variational autoencoder and LSTM and can produce outputs at par with the other works in the same field. The model can be used for low-frame-rate to high-frame-rate video conversion on any video captured from a camera. Further improvement can be made to the model for better generation quality and for dealing with more blur in the input.

REFERENCES

[1] Lee, A. X., Zhang, R., Ebert, F., Abbeel, P., Finn, C., & Levine, S. (2018). "Stochastic adversarial video prediction." arXiv preprint arXiv:1804.01523.

[2] Denton, E., & Fergus, R. (2018, July). "Stochastic video generation with a learned prior." In International conference on machine learning (pp. 1174-1183). PMLR.

[3] Zhao, L., Peng, X., Tian, Y., Kapadia, M., & Metaxas, D. (2018). "Learning to forecast and refine residual motion for image-to-video generation." In Proceedings of the European conference on computer vision (ECCV) (pp. 387-403).

[4] Jin, M., Meishvili, G., & Favaro, P. (2018). "Learning to extract a video sequence from a single motion-blurred image." In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (pp. 6334-6342).

[5] Wang, T. H., Cheng, Y. C., Lin, C. H., Chen, H. T., & Sun, M. (2019). "Point-to-point video generation." In Proceedings of the IEEE/CVF International Conference on Computer Vision (pp. 10491-10500).

[6] Purohit, K., Shah, A., & Rajagopalan, A. N. (2019). "Bringing alive blurred moments." In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 6830-6839).

[7] Argaw, D. M., Kim, J., Rameau, F., Zhang, C., & Kweon, I. S. (2021). "Restoration of Video Frames from a Single Blurred Image with Motion Understanding." In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 701-710).

[8] Zhang, K., Luo, W., Zhong, Y., Ma, L., Liu, W., & Li, H. (2018). "Adversarial spatio-temporal learning for video deblurring." IEEE Transactions on Image Processing, 28(1), 291-301.

[9] Zhou, S., Zhang, J., Pan, J., Xie, H., Zuo, W., & Ren, J. (2019). "Spatio-temporal filter adaptive network for video deblurring." In Proceedings of the IEEE/CVF International Conference on Computer Vision (pp. 2482-2491).

[10] Zhu, C., Dong, H., Pan, J., Liang, B., Huang, Y., Fu, L., & Wang, F. (2022, June). "Deep recurrent neural network with multi-scale bi-directional propagation for video deblurring." In Proceedings of the AAAI conference on artificial intelligence (Vol. 36, No. 3, pp. 3598-3607).

[11] Park, J., Nah, S., & Lee, K. M. (2022). "Pay Attention to Hidden States for Video Deblurring: Ping-Pong Recurrent Neural Networks and Selective Non-Local Attention." arXiv preprint arXiv:2203.16063.

[12] Cao, M., Fan, Y., Zhang, Y., Wang, J., & Yang, Y. (2022). "Vdtr: Video deblurring with transformer." IEEE Transactions on Circuits and Systems for Video Technology.

[13] Zhang, H., Xie, H., & Yao, H. (2022). "Spatio-Temporal Deformable Attention Network for Video Deblurring. In European Conference on Computer Vision (pp. 581-596). Springer, Cham.

[14] Liu, Y. L., Liao, Y. T., Lin, Y. Y., & Chuang, Y. Y. (2019, July). "Deep video frame interpolation using cyclic frame generation." In Proceedings of the AAAI Conference on Artificial Intelligence (Vol. 33, No. 01, pp. 8794-8802).

[15] Gupta, A., Aich, A., & Roy-Chowdhury, A. K. (2020). "ALANET: Adaptive Latent Attention Network forJoint Video Deblurring and Interpolation." arXiv preprint arXiv:2009.01005.

[16] Argaw, D. M., Kim, J., Rameau, F., & Kweon, I. S. (2021, May). "Motion-blurred video interpolation and extrapolation." In Proceedings of the AAAI Conference on Artificial Intelligence (Vol. 35, No. 2, pp. 901-910).

[17] Oh, J., & Kim, M. (2022). "DeMFI: deep joint deblurring and multi-frame interpolation with flow-guided attentive correlation and recursive boosting." In European Conference on Computer Vision (pp. 198-215). Springer, Cham.

[18] Zhong, Z., Sun, X., Wu, Z., Zheng, Y., Lin, S., & Sato, I. (2022). "Animation from blur: Multi-modal blur decomposition with motion guidance." In European Conference on Computer Vision (pp. 599-615). Springer, Cham.

[19] Zhong, Z., Cao, M., Ji, X., Zheng, Y., & Sato, I. (2022). "Blur Interpolation Transformer for Real-World Motion from Blur." arXiv preprint arXiv:2211.11423.

[20] Sayed, M., & Brostow, G. (2021). "Improved handling of motion blur in online object detection." In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 1706-1716).

[21] Vasiljevic, I., Chakrabarti, A., & Shakhnarovich, G. (2016). "Examining the impact of blur on recognition by convolutional networks." arXiv preprint arXiv:1611.05760.

[22] Nah, S., Hyun Kim, T., & Mu Lee, K. (2017). "Deep multi-scale convolutional neural network for dynamic scene deblurring." In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 3883-3891).

[23] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... & Polosukhin, I. (2017). "Attention is all you need." Advances in neural information processing systems, 30.