An Image-to-video Model for Real-Time Video Enhancement

Dongyu She
shedy19@mails.tsinghua.edu.cn
BNRist, Department of CS&T, Tsinghua University
Beijing, China

Kun Xu*
xukun@tsinghua.edu.cn
BNRist, Department of CS&T, Tsinghua University
Beijing, China

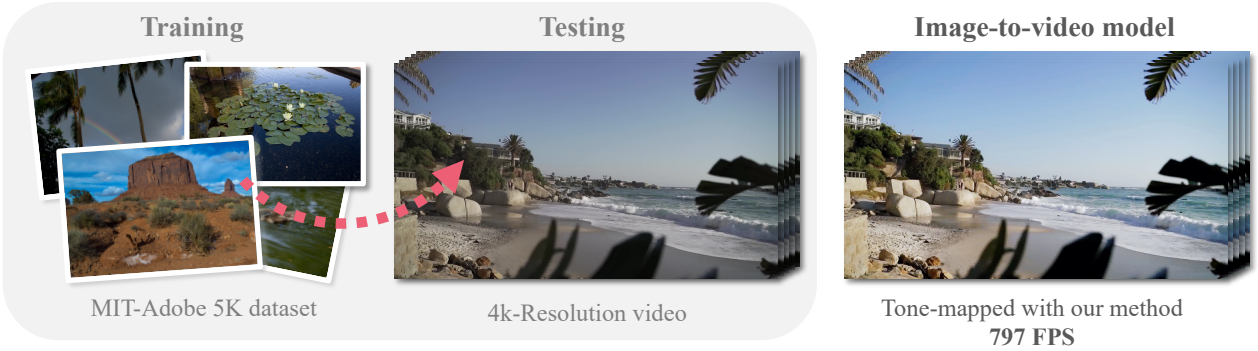

Figure 1: This paper presents an image-to-video model for real-time video enhancement, which is trained using only static images. The learned image-to-video model is able to enhance 4k-resolution videos with inference running in real-time.

ABSTRACT

Recent years have witnessed the increasing popularity of learning-based methods to enhance the color and tone of images. Although these methods achieve satisfying performance on static images, it is non-trivial to extend such image-to-image methods to handle videos. A straight extension would easily lead to computation inefficiency or distracting flickering effects. In this paper, we propose a novel image-to-video model enforcing the temporal stability for real-time video enhancement, which is trained using only static images. Specifically, we first propose a lightweight image enhancer via learnable flexible 2-dimensional lookup tables (F2D LUTs), which can consider scenario information adaptively. To impose temporal constancy, we further propose to infer the motion fields via a virtual camera motion engine, which can be utilized to stabilize the image-to-video model with temporal consistency loss. Experimental results show that our image-to-video model not only achieves the state-of-the-art performance on the image enhancement task, but also performs favorably against baselines on the video enhancement task. Our source code is available at https://github.com/shedy-pub/I2VEnhance.

*Corresponding author.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
MM '22, October 10–14, 2022, Lisboa, Portugal.
© 2022 Association for Computing Machinery.
ACM ISBN 978-1-4503-9203-7/22/10…$15.00
https://doi.org/10.1145/3503161.3548325

CCS CONCEPTS

• **Applied computing** → *Media arts*; • **Computing methodologies** → **Computational photography**.

KEYWORDS

Video enhancement, image enhancement, temporal consistency, image-to-video model

ACM Reference Format:
Dongyu She and Kun Xu. 2022. An Image-to-video Model for Real-Time Video Enhancement. In *Proceedings of the 30th ACM International Conference on Multimedia (MM '22), October 10–14, 2022, Lisboa, Portugal.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3503161.3548325

1 INTRODUCTION

Digital photography has progressed dramatically in recent years due to the ubiquitous presence of cameras on various devices. It is becoming more and more popular to record the beautiful lives via images and videos through online sharing communities, *e.g.* Flickr and Instagram. As captured photographs may still lack quality due to varying factors in various scenes, photo retouching is desirable to improve the visual quality of photographs. Unfortunately, manual retouching requires specialized skills and considerable time, especially for videos, thus is challenging for casual users. The above practical demands have motivated the research of automatic real-time photo enhancement.

With promising development in deep learning, the image enhancement community has observed significant advances. Numerous learning-based methods [7, 10, 16, 23, 34, 48] achieve satisfying performance on static images. However, it is non-trivial to extend such image-to-image methods to handle videos frame by frame due to two main challenges. First, complex network architecture and

high computation overheads prevent them from real-time processing. The image-to-image translation methods [4, 7, 46] generate high-quality enhanced results via encoder-decoder structures. However, such methods consisting of a deep stack of upsampling and conv layers are computationally expensive. The lookup table-based method [40, 48] exhibits significant performance and efficiency while only considering the image retouching task.

Second, videos pose an additional challenge compared to images. Not only do the video frames need to satisfy the intended enhancement, they also need to be temporally consistent. Otherwise, the processed video will exhibit flickering artifacts. Video-to-video translation techniques are trying to address the problem by directly incorporating optical flow-based losses [22], or networks that operate on the time dimension, *e.g.* 3D convolutional layers or recurrent layers. However, collecting the human-retouched dataset for video retouching is too labor-consuming. The blind video consistency models [2, 8, 26] relax the need for task-specific datasets and losses, which, however, include an extra post-processing step.

In this paper, we propose a novel image-to-video model enforcing the temporal stability for real-time video enhancement, which is trained using only static images. Fig. 1 shows the video enhancement results predicted by the proposed model that are trained using only static images. Specifically, we first propose a lightweight image enhancer via learnable flexible 2-dimensional lookup tables (F2D LUTs), which is able to consider scenario information adaptively. To impose temporal constancy, we further propose to infer the motion fields via a virtual camera motion engine, which can be utilized to stabilize the image-to-video model with temporal consistency loss. Thus, the temporal consistency on video task can be imposed by enforcing the transform invariant constraint during training.

Our main contributions are summarized as follows:

- We propose a image-to-video framework for real-time video enhancement. The proposed model achieves nearly 800 FPS on 4k-resolution videos, which is highly efficient for deployment in practical applications.
- We propose a Flexible 2D-LUTs architecture by constructing lookup table under two-dimension setting and introduce a more efficient interpolation function.
- We propose to incorporate motion field prior from still image via a virtual camera motion engine, which is utilized to impose the temporal consistency.
- Our extensive experiments on both image benchmark dataset and the synthetic video dataset validate that our model significantly outperforms state of-the-art image enhancement methods both quantitatively and qualitatively.

2 RELATED WORK

Image Enhancement and Retouching Recently, significant progress has been made by employing neural networks for automatic image enhancement. In general, these learning-based approaches can be divided into image-to-image translation based methods and regression based method.

Image-to-image translation models [7, 17, 18, 20, 24, 33, 47, 52] consider the process of image enhancement as a problem of image-to-image translation. They usually work in an end-to-end fashion, feeding the input image as a whole to specially designed neural networks and obtain the enhanced image as output. While those models could generate satisfactory results, they usually suffer from two limitations. First, large models are usually employed and hence large memory consumption is required; second, they may generate visible artifacts in high frequency and flat regions due to the use of downsampled convolutional layers. Various methods are also proposed to reduce the artifacts.

In regression based methods, enhancement is achieved by applying pixel-wise color transformation to the input image. Different methods employ different models for the predefined color transformation, including color transformation matrices [5, 10, 30], curve-based functions [12, 23, 28, 32, 38], sequential functions [14, 16, 34, 37, 45], and 3D lookup tables (LUTs) [40, 48].

As a typical example for transformation matrix based methods, HDRNet [10] uses 3×4 affine transformation matrices, to map input pixel colors to output colors. The coefficients of color transformation matrices are predicted through low-resolution images and stored in the bilateral grid, which are then applied to the original image guided by a full-resolution single-channel guidance map. However, the capability of transformation matrix in handling sophisticated color transformation is limited. Besides, the employed FCN network is also expensive for high-resolution inputs.

For curve based methods, they mimic color transformation using curves, like those color curve adjustment tools in retouching softwares, *e.g.* Lightroom and Photoshop. Generally, these methods try to regress a limited number of parameters, *e.g.* the representative points of the curve [23, 28, 32, 38], the coefficients of a pre-defined function [12, 16, 34]. However, such functions mostly ignore the relationship between different channels, which is insufficient to approximate complex transformations.

LUTs are widely used in image processing pipelines and commercial softwares for accelerating color adjustment operations such as hue or tonal adjustments. Zeng *et al.* [48] proposed image adaptive 3D-LUTs, which combines the deep learning paradigm with traditional image enhancement paradigm and achives promising results. Specifically, the color transformation is modeled as a weighted interpolation of multiple 3D-LUTs, where the weights are predicted through convolutional networks according to the image content. Moreover, there are some methods extending LUTs to handle spatial aware enhancement [40] and single image super resolution [21]. While 3D-LUTs is an attractive transformation function that can be well integrated with image enhancement tasks, it is not flexible enough since only the interpolation weights are adaptive to the input image while the basis 3D-LUTs are always fixed.

Video Enhancement Video enhancement, especially video retouching, is still an open and challenging problem. While the problem of image enhancement and retouching has been well studied, it is non-trivial to extend image enhancement methods to videos. Directly applying existing image enhancement methods in a frame-by-frame way would simply lead to temporal inconsistency, *i.e.*, flickering artifacts. One way to alleviate the flickering artifacts is to employ blind video temporal consistency techniques [3, 26], which directly operate on the per-frame processed videos as a post-processing step and are blind to the used image enhancement algorithm.

Since videos are 3-dimensional (*i.e.*, while images are 2-dimensional), another solution is to extend 2D networks in image enhancement

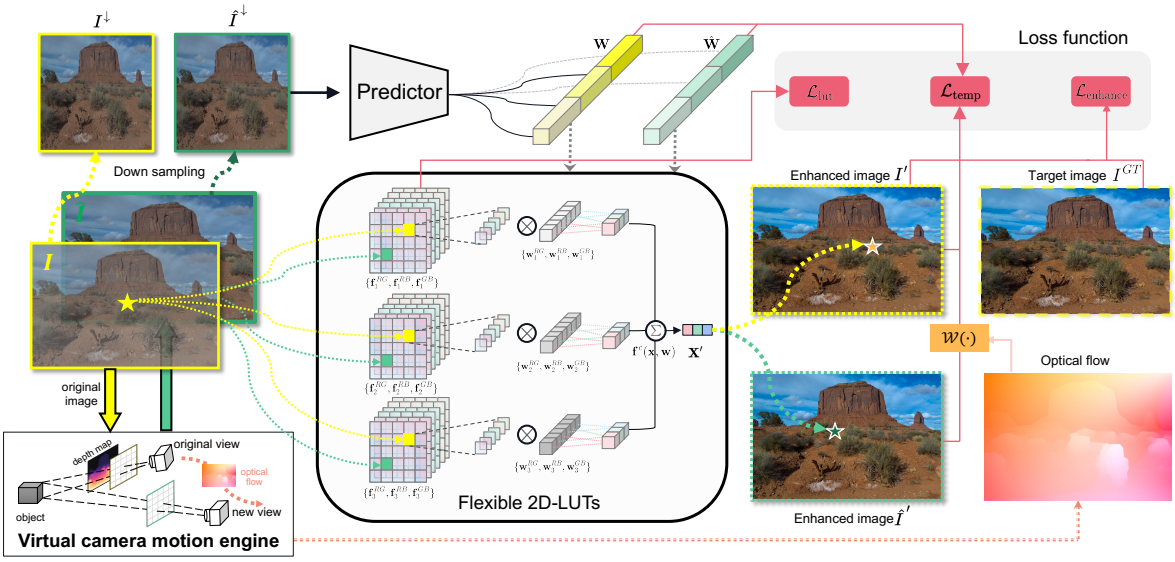

Figure 2: Our proposed framework consists of two steps, *i.e.*, motion field inference, and video enhancement. Given the input image, we first generate a new virtual view from the still image via a virtual camera motion engine, which can be computed offline before training. Then, both the input and generated images are enhanced with the proposed image enhancer built with F2D-LUT. The enhancement results and the obtained optical flow are then used to compute the final loss. During the inference phase, our model directly takes each video frame as input and predicts the corresponding weights for F2D-LUTs.

models to their 3D counterpart [19, 31]. Specifically, Lv *et al.* [31] employ 3D convolution layers to handle image sequences, while Jiang *et al.* [19] propose a modified 3D U-Net for low-light video enhancement. To reduce noises in low light videos, In addition, Wang *et al.* [42] propose a low light video enhancing method with a high sensitivity noise model achieving temporal coherence through a spatial-temporal LSTM module. Chen *et al.* [6] train a siamese network on static raw videos, which enforces temporal consistency between frames directly. Zhang *et al.* [51] propose the RT-VENet for real-time video enhancement. Its color transformation is modeled using curve functions whose parameters are stored in grids. A temporal feature aggregation module is included and a temporal-consistency loss is used. All above images require paired videos for training. However, compared to images, videos, especially paired videos, are rather difficult to collect.

Since images are much more easily available than videos, it would be desirable for video enhancement methods if they can be trained only using images. Eilertsen *et al.* [9] introduce temporally stable CNNs, where temporal-aware regularization losses are additionally included in training. They enable image processing (but only considering the loss function without network structure is not sufficient to model the consistency in the time dimension. Recently, Zhang *et al.* [50] propose the StableLLVE for low light video enhancement with image-based model and alleviate flickering with predicted optical flow. However, the UNet architecture in [50] makes the method impractical when applying to high-resolution videos. In addition, they predict the optical flow by combining an unsupervised motion propagation model [49] with segmentation results, which is suboptimum for the scenes that do not include any foreground. In this paper, we propose a highly efficient framework for real-time enhancement application based on the LUTs method. We also employ a physics-based engine to generate more stable motion fields, described in Sec. 3.2. Readers could refer to the survey [27] for more detailed discussions on image and video enhancement.

3 METHODOLOGY

In this section, we present in detail our proposed image-to-video model for real-time video enhancement. Fig. 2 gives the pipeline of our overall algorithm. The image-to-video enhancement model is based on a LUTs-based image enhancer, which imposes temporal consistency by utilizing the motion field inferred via the virtual camera motion engine.

3.1 Image Enhancer

Before introducing the idea of our flexible 2D lookup tables (F2D-LUTs) , we first briefly review the image adaptive 3D-LUTs (short as A3D-LUTs) [48] for image enhancement.

Adaptive 3D-LUTs Generally, let's consider a global color transformation $\mathbf{f_{ori}} : \mathbf{x} \mapsto \mathbf{x}'$ that maps an input 3D RGB color $\mathbf{x} = (x_r, x_g, x_b)$ to an output RGB color $\mathbf{x}' = (x'_r, x'_g, x'_b)$. For simplicity, all input and output RGB colors are assumed to be inside the RGB unit cube, *i.e.*, $\mathbf{x}, \mathbf{x}' \in [0, 1]^3$.

A 3D-LUT $\mathbf{f}$ uses a 3D uniform grid structure with $(M-1)^3$ grids in the 3D RGB space and stores an output RGB color at each grid vertex. There are M^3 grid vertices in total. A lookup for an input RGB color $\mathbf{x}$ is computed as a multi-linear interpolation of nearby

grid vertices:

$$\mathbf{f}(\mathbf{x}) = \sum_{i=1}^{8} t_i(\mathbf{x})\mathbf{f}_{\mathbf{ori}}(\mathbf{x}_i), \tag{1}$$

where $\mathbf{x}_i$ $(1 \le i \le 8)$ denote the eight vertices of the grid which $\mathbf{x}$ resides in, and $t_i(\mathbf{x})$ $(1 \le i \le 8)$ are trilinear interpolation weights which sum up to one: $\sum_{i=1}^{8} t_i(\mathbf{x}) = 1$. The color mapping defined by a 3D-LUT $\mathbf{f}$ could be viewed as a piecewise linear approximation of the original color mapping $\mathbf{f}_{\mathbf{ori}}$.

In order to make the enhancement process adaptive to image contents, they use multiple 3D-LUTs and combine them in a linear way:

$$\mathbf{f}^c(\mathbf{x}) = \sum_{j=1}^{K} w_j \mathbf{f}_j(\mathbf{x}), \tag{2}$$

where $\mathbf{f}_j(\cdot)$ is the j-th 3D-LUT and w_j denotes its weight $(1 \le j \le K)$. the weights w_j $(1 \le j \le K)$ are image adaptive and are inferred from a CNN weight predictor by analyzing the low resolution input image. K is the number of 3D-LUTs which is usually set between 3 to 5 [29, 48].

Flexible 2D-LUTs While 3D-LUTs are effective in approximating sophisticated color transformations, it requires a relatively large number of parameters. For example, for a typical hyperparameter $M = 33$, one 3D-LUT requires about 108K $(3M^3)$ parameters. On the other hand, while the curved-based methods, which behave like 1D-LUTs, require fewer parameters, they usually have less capability in representing complex color transformations. Our observation is that: 2D-LUT, between 3D and 1D, would strike a good trade-off between the capability of representation and model size. For this purpose, we propose flexible 2D-LUTs (short as F2D-LUTs), more flexible enhancing operators to tackle dramatic variations among different scenes.

A single 2D-LUT is defined in a similar way as a 3D-LUT in Eqn. (1) except that: 1) 2D-LUTs use 2D grids instead 3D grids; 2) when performing a lookup, we use bicubic weights instead of bilinear weights for interpolation as we find bicubic interpolation can further improve the smoothness of enhanced results.

In order to capture the 3D color transformations in RGB space, we define our F2D-LUT as a group of three 2D-LUTs where the operated two color channels vary in a rotating order. Specifically, a RG-channel (*i.e.*, red and green channels) 2D-LUT defines the following color mapping $\mathbf{f}^{RG} : \mathbf{x} \mapsto \mathbf{x}'$:

$$(x'_r, x'_g, 0) = \mathbf{x}' = \mathbf{f}^{RG}(\mathbf{x}) = \mathbf{f}^{RG}(x_r, x_g). \tag{3}$$

Note that only R and G channels (x_r and x_g) are taken as input and B channel (x_b) is not considered. We also only output colors in R and G channels, while the output value in the B channel is simply set as zero. The other two 2D-LUTs, including the GB-channel 2D-LUT (*i.e.*, $\mathbf{f}^{GB}$) and the BR-channel 2D-LUT (*i.e.*, $\mathbf{f}^{BR}$), are defined similarly.

A full 3D color transformation could be obtained by linearly combining multiple F2D-LUTs:

$$\mathbf{f}^c(\mathbf{x}, \mathbf{w}) = \sum_{j=1}^{K} \mathbf{w}_j^{RG} \circ \mathbf{f}_j^{RG}(\mathbf{x}) + \mathbf{w}_j^{GB} \circ \mathbf{f}_j^{GB}(\mathbf{x}) + \mathbf{w}_j^{BR} \circ \mathbf{f}_j^{BR}(\mathbf{x}), \tag{4}$$

where K is the number of F2D-LUTs, $\mathbf{f}_j^{RG}$, $\mathbf{f}_j^{GB}$ and $\mathbf{f}_j^{BR}$ denote the RG-channel, GB-channel, and BR channel 2D-LUTs in the j−th F2D-LUT, respectively, and $\mathbf{w} = \{\mathbf{w}_j^{RG}, \mathbf{w}_j^{GB}, \mathbf{w}_j^{RG} | 1 \le j \le K\}$ is the corresponding learnable weight vector. Note that the operator $\circ$ denotes an element-wise multiplication, for example, for weight $\mathbf{w}^{RG} = (w_r, w_b, 0)$ and color $\mathbf{f}^{RG}(\mathbf{x}) = (x'_r, x'_g, 0)$, their element-wise multiplication results in color $(w_r x'_r, w_b x'_g, 0)$.

The model size of our F2D-LUT is much smaller than that of 3D-LUT. In practice, we also set $M = 33$ (*i.e.*, the number of elements along one color channel), and a F2D-LUT contains only 6K $(6M^2)$ parameters, which is only about 6% of a 3D-LUT's parameters $(3M^3)$.

Self-adaptive weight predictor Following [48], we employ a simple weight predictor $\mathbf{F}(\cdot)$ with the same CNN backbone to support image-adaptive F2D-LUTs. As shown in Fig. 2, given the input image I, we predict the weights for F2D-LUTs, denoted as:

$$\mathbf{w} = \mathbf{F}(I^{\downarrow}; \theta), \tag{5}$$

where $I^{\downarrow}$ is the down-sampled input image, and θ denotes the parameters of the CNN-based weight predictor. During inference phase, after the weights are predicted using Eqn. (5), the final enhanced image I' can be obtained by performing lookups and interpolations as in Eqn. (4) in a pixelwise manner, written as:

$$I' = \mathbf{f}^c(I, \mathbf{w}). \tag{6}$$

3.2 Image-to-Video Enhancement Model

In this subsection, we opt to impart our image enhancer described in Sec. 3.1 with temporal stability for the video enhancement task. The key idea is to mimic the motion between adjacent frames and implicitly model the temporal consistency in an end-to-end trainable network. As shown in Fig. 2, given a still image, we first predict a new view of the input mimicking the adjacent frame. Then, the corresponding optical flow can be computed, which can represent both global and local motions for mimicking the motions of dynamic scenes. Finally, with the help of the inferred motion field, our model can be stabilized by optimizing the proposed temporal consistency constraint. Below, we explain in detail.

Motion Field Inference We first infer the motion field from the still images via a virtual camera motion engine module inspired by [1]. Given the input image I, we first estimate the depth map $\mathcal{D}$ via an off-the-shelf network. The estimated $\mathcal{D}$ is then employed to project pixels in I to 3D space according to inverse intrinsics matrix $\mathcal{M}^{-1}$. Assuming that I is framed by the camera at 3D location c_0, an arbitrary virtual motion is applied on the camera moving it to a new position c_1. Specifically, we generate a rotation R_1 and a translation t_1 by sampling a random triplet of Euler angles and a random 3D vector, respectively. The transformation matrix is then defined as $T_{0\to1} = (R_1 \mid t_1)$. For each pixel p in I, the coordinates $\hat{p}$ of its corresponding pixel in $\hat{I}$ acquired from new viewpoint c_1 can be obtained by:

$$\hat{p} \sim \mathcal{M} T_{0\to1} \mathcal{D}(p) \mathcal{M}^{-1} p. \tag{7}$$

The optical flow can then be distilled using Depthstillation [1]. Fig. 3 shows the generated images and obtained optical flows. As can be seen, such physical-based engine is able to generate meaningful

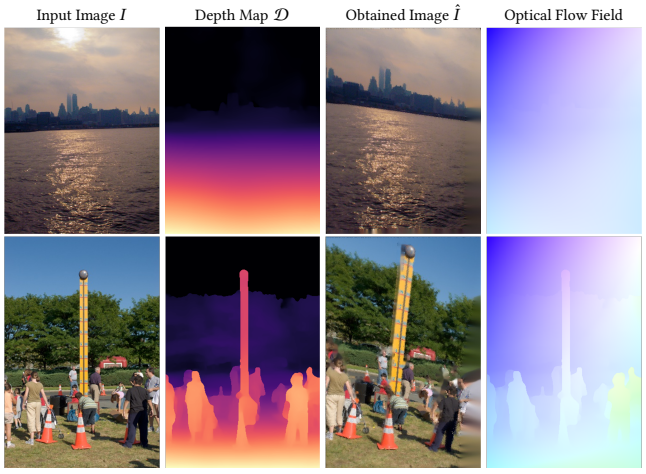

Figure 3: The example results of virtual camera motion engine. From left to right: a) Input, b) Estimated depth map, c) The generated image under a new virtual view, and d) Optical flow field consequence of virtual camera motion.

results for both cases with and without obvious foreground objects.

Imposing temporal consistency Motivated by that temporally stable model should be transform-invariant [9], during training, we enforce the model to get outputs of transformed inputs with the same transformations as if the operations are applied to results directly. Consequently, we generate a motion with the estimated optical flow to mimic actual video sequences and train our image-based model in a Siamese way. As shown in Fig. 2, the input image I is first feed into the weight predictor of our image enhancer to obtain image adaptive weights $\mathbf{w}$ in Eqn. (5). The image under new view $\hat{I}$ as well as the optical flow can be obtained with the virtual camera motion engine descibed in Sec. 3.2. We then feed $\hat{I}$ again to the weight predictor to obtain the weights $\hat{\mathbf{w}}$ for the new view image. Finally, we enforce temporal consistency between the predicted weights $\mathbf{w}$ and $\hat{\mathbf{w}}$, and between the enhanced image and the warped result of the enhanced image, which we will explain later in the loss function subsection.

Note that the above process of estimating optical flows and imposing temporal consistency is only involved during the training stage. During the inference stage, we simply feed video frames to the image enhancer in a frame-by-frame fashion, and the temporal consistency is automatically preserved.

3.3 Loss Function

Our loss function for the video enhancement task is defined as the weighted sum of an enhancement loss, an LUT loss, and a temporal consistency loss:

$$\mathcal{L} = \mathcal{L}_{enhance} + \mathcal{L}_{lut} + \lambda_t \mathcal{L}_{temp}, \tag{8}$$

where the enhancement loss $\mathcal{L}_{enhance}$ ensures content consistency of enhanced images, the LUT loss $\mathcal{L}_{lut}$ is included to enforce the smoothness of LUTs, and the temporal consistency loss $\mathcal{L}_{temp}$ enforces the temporal stability. λ_t is set according to the ablation study.

Specifically, given a pair of input image I and the ground truth enhanced image I^{GT}, the enhancement loss $\mathcal{L}_{enhance}$ is defined as:

$$\mathcal{L}_{enhance} = \left\| \mathbf{f}^c(I, \mathbf{w}) - I^{GT} \right\|^2, \tag{9}$$

where $\mathbf{f}_c$ denotes the image enhancer in Eqn. (6) and $\mathbf{w}_t$ is the predicted weight vector in Eqn. (5).

In order to stably transform the input RGB values into the desired color space without generating much artifacts, following the adaptive 3D-LUT work [48], we also employ a smooth regularization term $\mathcal{R}_s$ and a monotonicity regularization term $\mathcal{R}_m$ on our F2D-LUTs as follows:

$$\mathcal{L}_{lut} = \lambda_s \mathcal{R}_s + \lambda_m \mathcal{R}_m. \tag{10}$$

The smooth regularization consists of a 2D total variation regularization term and a ℓ_2-norm regularization on the predicted weights:

$$\begin{aligned} \mathcal{R}_s = \sum_{\mathbf{f} \in \{\mathbf{f}^{RG}, \mathbf{f}^{GB}, \mathbf{f}^{RB}\}} \sum_{i,j} \Big(& \left\| \mathbf{f}_{(i+1,j)} - \mathbf{f}_{(i,j)} \right\|^2 \\ & + \left\| \mathbf{f}_{(i,j+1)} - \mathbf{f}_{(i,j)} \right\|^2 + \|\mathbf{w}\|^2 \Big). \end{aligned} \tag{11}$$

The monotonicity regularization on 2D-LUT learning is defined as follows:

$$\begin{aligned} \mathcal{R}_m = \sum_{\mathbf{f} \in \{\mathbf{f}^{RG}, \mathbf{f}^{GB}, \mathbf{f}^{RB}\}} \sum_{i,j} \Big(& \varphi\left(\mathbf{f}_{(i,j)} - \mathbf{f}_{(i+1,j)} \right) \\ & + \varphi\left(\mathbf{f}_{(i,j)} - \mathbf{f}_{(i,j+1)} \right) \Big), \end{aligned} \tag{12}$$

where $\varphi(x)$ denotes the ReLU function, *i.e.*, $\varphi(x) = max(0, x)$.

The temporal consistency loss is defined to enforce both consistency on enhanced results and on weight parameters between warped and unwarped counterparts:

$$\mathcal{L}_{temp} = \beta_1 \left\| \mathcal{W}\left(\mathbf{f}^c(I, \mathbf{w}) \right) - \mathbf{f}^c(\hat{I}, \hat{\mathbf{w}}) \right\|^2 + \beta_2 \left\| \mathbf{w} - \hat{\mathbf{w}} \right\|^2, \tag{13}$$

where $\mathcal{W}$ denotes the warping function that warps image with optical flow, β_1 and β_2 are the weights to ensure the similar magnitude of these two loss terms.

4 EXPERIMENT

Implementation Details We apply random cropping, horizontal flipping for data augmentation following [48]. During training, the mini-batch size is set to 1 for image task and 64 for video task. The learning rate is initialized as 10^{-4}, and the model is trained by Adam optimizer [25] with default parameters for 400 epochs. For the loss function, we adopt the default parameter settings as [48], *i.e.*, λ_s is set to $1e-4$, and λ_m is set to 10 in all our experiments. Empirically, we set β_1 and β_2 as 0.1 and 1, respectively. The optimal weight λ_t of our proposed temporal consistency loss is discussed in the ablation study.

We have implemented and tested our method on two deep learning frameworks, *i.e.*, PyTorch [35] and Jittor [15]. The inference performance of both frameworks is given in Tab. 1. Generally, Jittor's implementation is faster than that of PyTorch. All experiments are performed on a PC with an NVIDIA RTX 3090 Ti GPU.

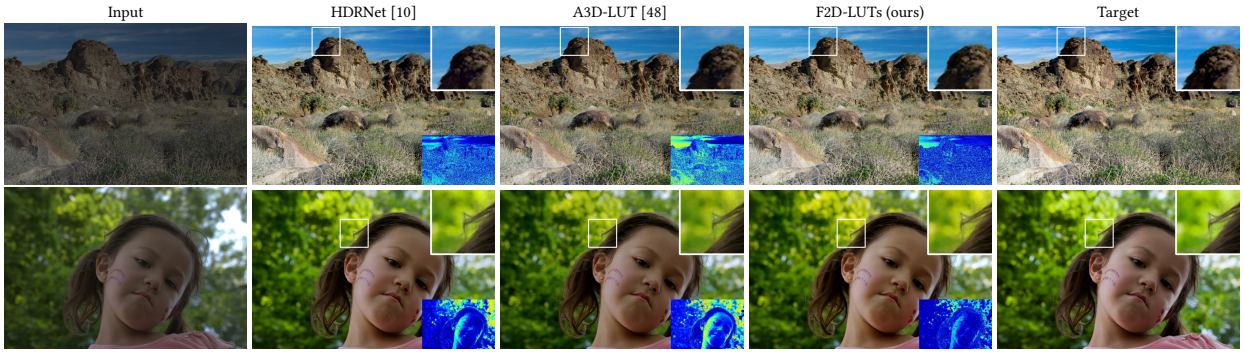

Figure 4: Image retouching comparisons. Shown are the test images from FiveK, results from the HDRNet [10], results from the A3D-LUT [48], our results, and the ground truth. The corresponding error maps are also given. *(Best viewed with zoom)*

Table 1: Inference speed comparison between PyTorch and Jittor. The boldface denotes the faster framework.

Platform	Time per image (ms/im)	Iterations per second (it/s)
PyTorch [35]	1.2435	6.4092
Jittor [15]	**1.1302**	**6.7801**

We have trained and evaluated our method on two datasets: MIT-Adobe 5K [4] (FiveK) and a synthetic video dataset based on Arrow of Time (AOT) dataset [36].

Image Dataset The MIT-Adobe 5K dataset contains 5,000 raw images and the paired retouched results, each of which is retouched by five human experts. We use the standard training and testing split and use images retouched by expert C as the ground truth following the common practice [10, 39, 48]. Following [48], we conduct experiments on two resolutions, *i.e.*, 480p and the original image resolution.

Synthetic Video Dataset We also construct a synthetic video dataset to evaluate our method for video enhancement task. Following [51], we use the original AOT dataset [36] as ground truth and generate different degrees of under-exposed and over-exposed video clips as low-quality inputs (see details in Appendix B). The synthetic dataset includes 180 video clips collected from YouTube by retrieving a diverse set of keywords. Consequently, we use about 31K paired frames from 160 sequences for training and the remaining 20 sequences for testing. Note that our image-to-video model is trained by taking a single image as input at each iteration rather than using sequential inputs. During the inference phase, our model enhances the test video clips frame by frame.

Metrics We evaluate their performances with two common metrics, *i.e.*, Peak Signal-to-Noise Ratio (PSNR) and Structural Similarity (SSIM) [43] [1]. In addition, following [50], we also validate temporal stability of models with AB(Var) [31], Mean Absolute Brightness Difference (MABD) [19]. The lower values in the three metrics stand for better temporal stability.

[1]We compute SSIM using the implementation in [48].

Table 2: Quantitative comparison of image retouching results on the FiveK dataset with different resolutions. Here, "N.A." means that the result is not available. Note that some results are replicated from [48]. Running speed (in FPS) is measured on 4K-resolution images using a single GTX 3090 Ti.

Method	FiveK-480p		FiveK-original		Speed
	PSNR↑	SSIM↑	PSNR↑	SSIM↑	
UPE [39]	21.88	0.853	21.65	0.859	6
Dis-Rec [34]	21.98	0.856	21.81	0.862	0.01
DPE [7]	23.75	0.908	N.A.	N.A.	N.A.
HDRNet [10]	24.32	0.912	24.03	0.919	25
A3D-LUT [48]	25.21	0.922	25.10	0.930	**821**
F2D-LUTs	**25.34**	**0.929**	**25.19**	**0.935**	797

4.1 Evaluation and Results

Image Enhancement We first independently evaluate the effectiveness of our image enhancer (Sec. 3.1) on image enhancement tasks. We have compared with several representative baselines, *i.e.*, UPE [39], Dis-Rec [34], HDRNet [10], DPE [7], and A3D-LUT [48]. All methods are trained on the FiveK dataset with both resolutions. Tab. 2 reports the PSNR and SSIM scores, where the best performance for each evaluation metric is highlighted in bold black. Compared with the reinforcement-learning-based method [34] and the transformation function based methods [10, 39], the LUTs-based methods achieve better performance with much fewer parameters. Our proposed image enhancer is superior to the compared methods in both 480p and full-resolution settings attributed to the 2D structure and the interpolation function. We also report the running speed on 4K-resolution images and model parameters. Fig. 4 shows the qualitative comparisons with corresponding error maps of different image models. Our observation is that the HDRNet tends to enhance images excessively with a broader range of RGB space, while our method is more consistent with the tonal style of targets compared with A3D-LUT.

Video Enhancement We compare our overall video enhancement method with state-of-the-art methods, including image-based enhancement methods, video-based enhancement methods as well as

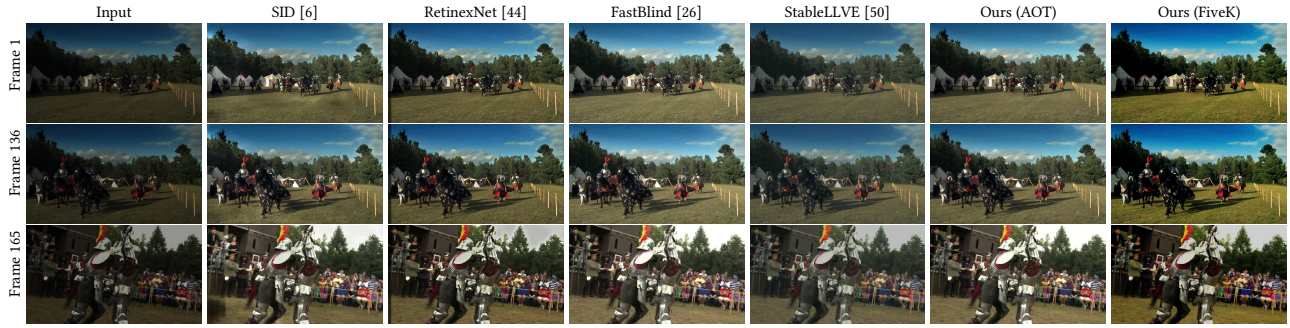

Figure 5: Qualitative comparison on video enhancement task. Shown are the input test video from the AOT dataset, and results from four baselines including SID [6], RetinexNet [44], FastBlind [26], StableLLVE [50]. We also show the video enhancement results using our model trained on the AOT and FiveK datasets in last two column. Please zoom in for a better view.

Table 3: Quantitative comparison on the synthetic video dataset. Our model as well as image-based methods are trained with image data, while ⋆ denotes methods using sequential video data for training.

Methods	PSNR ↑	SSIM ↑	MSE ↓	AB(Var)↓	MADB↓
LIME [13]	18.28	0.8381	1.31	0.295	49.960
MELLEN [31]	18.50	0.7713	1.21	0.552	19.669
RetinexNet [44]	21.53	0.9182	0.65	0.077	**8.403**
A3D-LUT [48]	27.95	0.9335	0.32	0.255	17.880
SID [6]	24.19	0.9345	0.41	0.065	12.664
SMOID⋆ [19]	25.21	0.9332	0.39	0.054	11.664
vid2vid⋆ [41]	23.93	0.9259	0.55	0.196	12.586
Fast blind⋆ [26]	26.73	0.9326	0.35	0.053	11.984
StableLLVE [50]	27.55	0.9334	0.37	0.051	11.034
A3D-LUT+$\mathcal{L}_{temp}$	28.30	0.9346	0.29	0.106	12.052
Ours	**29.08**	**0.9409**	**0.22**	**0.051**	10.958

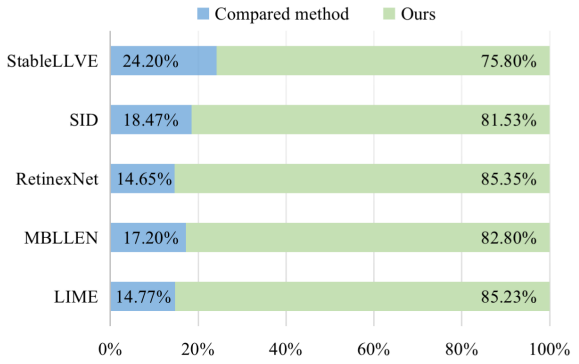

Figure 6: Results of user study. Our method is compared with five image-based methods for blind A/B tests. In each comparison, we show the preference percentage of the compared method and ours, and require the participants to select the result that are more stable and visually pleasant.

methods utilizing self-consistency. The image-based methods include a traditional method LIME [13] as well as four deep-learning based methods, *i.e.*, MBLLEN [31], RetinexNet [44], SID [6] and A3D-LUT [48]. They are used to enhance videos by processing video frames one by one. The two competing video-based methods include vid2vid [41] and SMOID [19]. In addition, we also compare two methods utilizing self-consistency, *i.e.*, Fast blind [26] and StableLLVE [50]. All methods are trained using the synthetic video dataset (from AOT) for a fair comparison.

Quantitative results are provided in Tab. 3. The image-to-image methods do not perform well especially on the AB(Var) metric since they fail to preserve temporal consistency, *e.g.* suffering from flickering. Compared with image-based methods, video-based methods generally achieve better PSNR and SSIM and are more stable according to AB(Var) and MADB. The post-processing method Fast blind, based on the enhancement results of HDRNet, achieves better performance on both quality and stability than the above methods. The image-to-video model, StableLLVE, can also generate stable results, however, the visual performance is limited by the prediction model which visual quality is unsatisfactory due to lacking color diversity. Our proposed model is superior to the compared methods in terms of visual quality and stability. We additionally measure the performance of A3D-LUTs on the video task by replacing our F2D-LUTs with the originally A3D-LUTs while keeping other components unchanged, *i.e.*, A3D-LUT+$\mathcal{L}_{temp}$. The results show that our proposed F2D-LUTs are more effective, *i.e.*, having a higher dimension of self-adaptive weights w but with much smaller model sizes, which are more suitable for video tasks.

Fig. 5 further shows qualitative comparisons. It can be found that the image enhancement methods SID and RetinexNet suffer from artifacts, while the generated results of StableLLVE are unsatisfactory due to incorrect temperature. For the video temporal consistency works, the method of Fast blind takes the original and per-frame processed videos as inputs to produce a temporally consistent video. In the last column of Fig. 5, we also show results generated by our method trained with the FiveK dataset. Those results tend to have more vivid colors due to the color diversity of the FiveK dataset.

User Study We have conducted a user study to evaluate the subjective visual quality of our method on the video retouching task. We compare with five image-based methods including LIME [13], MELLEN [31], RetinexNet [44], SID [6] and StableLLVE [50]. We

Table 4: Ablation results on the image enhancer model. We show the PSNR results of using number (M) of A3D-LUT [48] and the proposed F2D-LUTs with two interpolation functions.

LUTs Num.	1	2	3	4	5
A3D-LUT	23.15	24.86	25.21	25.26	25.29
F2D (cubic)	24.84	25.17	25.24	25.25	25.26
F2D (bicubic)	25.01	25.25	25.34	25.36	25.40

Table 5: Ablation studies on the weights of temporal consistency loss on the synthetic video dataset. Note that the model with λ_t=0 equals our proposed F2D-LUTs.

Setting	PSNR ↑	SSIM ↑	MSE ↓	AB(Var)↓	MADB↓
λ_t=0	28.22	0.9339	0.30	0.254	17.860
λ_t=0.01	28.40	0.9342	0.28	0.127	14.384
λ_t=0.1	28.83	0.9386	0.24	0.092	11.982
λ_t=0.5	**29.08**	**0.9409**	**0.22**	0.051	10.958
λ_t=1.0	29.02	0.9398	**0.22**	**0.048**	**10.324**

randomly select seven testing videos from the AOT dataset for the blind A/B test, and invite 29 participants for the study. Only two enhanced videos are shown at a time, and the participants are asked to choose a better one considering the tones and stability. Fig. 6 summaries the feedback results, suggesting that our retouched videos are visually more appreciated than those of competing approaches.

4.2 Ablation Study

We conducted several ablation studies to investigate the effectiveness of the proposed image enhancer as well as the temporal consistency loss in our model. For image enhancer, we perform ablation studies on the 480p FiveK dataset, while the temporal consistency loss is discussed on the video dataset.

Image Enhancer We evaluate the performance of our proposed F2D-LUTs with different settings on image enhancement, including the number K of F2D-LUTs and the choice of the interpolation function. Tab. 4 reports the performance of A3D-LUTs and our F2D-LUTs with different numbers of LUTs from $K = 1$ to $K = 5$. The results of $K = 1$ verify the effectiveness of our proposed F2D-LUT compared with a single 3D-LUT. When increasing the number of LUTs from 1 to 3, both F2D-LUTs and A3D-LUTs are boosted significantly. In addition, further increasing the number of LUTs leads to minor improvement for A3D-LUT, while our proposed F2D-LUTs with bicubic interpolation function show the increased capacity. For a fair comparison with A3D-LUTs, we still set K to 3 in other experiments. Fig. 7 visualizes three F2D-LUTs with $K = 1$ learned with both smooth and monotonicity regularization, with only monotonicity regularization and with only smooth regularization. As seen, the using either monotonicity regularization or smooth regularization only results in irregular mapping results, while jointly optimization leads to smoother and better visual results.

Temporal Consistency Loss We also conduct the ablation study on the temporal consistency loss verifying its effect on the visual quality and temporal stability. Tab. 5 shows the performance of our image-to-video model with different weight λ_t. As reported, with the increase of λ_t, our model is gradually temporally stable and achieves better visual quality. Compared with a pure image enhancer (*i.e.*, λ_t=0), our model with λ_t=0.5 decreases the MADB metric by about 7.5, which is fixed in our remaining experiments.

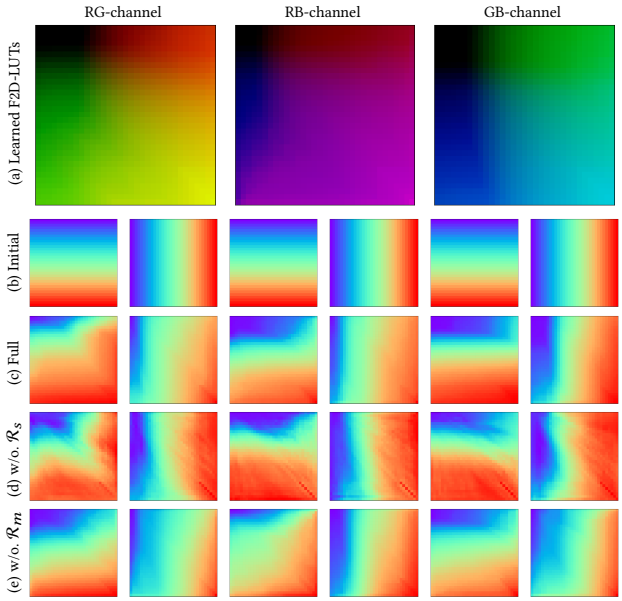

Figure 7: Visualization of the F2D-LUT ($K = 1$) learned with different weights for $\mathcal{L}_{lut}$ on FiveK. (a) Visualization of 2D-LUTs $\{\mathbf{f}_1^{RG}, \mathbf{f}_1^{RB}, \mathbf{f}_1^{GB}\}$ in the RGB format by adding an additional channel filled with zeros. (b) Visualization of initialized identical 2D-LUTs in one-channel format. Visualization of learned 2D-LUTs optimized (c) with both smooth and monotonicity regularization, (d) with only monotonicity regularization, (e) with only smooth regularization.

5 CONCLUSION

In this paper, we propose a novel model for real-time video enhancement with LUTs-based image enhancer, which is trained on static images to address the image-to-video task. Our proposed F2D-LUTs exhibit capacity of representation and efficiency surpassing the image-to-image methods on the MIT-Adobe-5K dataset. In addition, by utilizing motion field inferred from the still images, our proposed model is further imparted with temporal consistency. Extensive experiments on the synthetic video dataset demonstrate the superiority of our image-to-video model against SOTA methods on both performance and efficiency.

ACKNOWLEDGMENTS

This work was supported by the National Natural Science Foundation of China (No. 61932003).

REFERENCES

[1] Filippo Aleotti, Matteo Poggi, and Stefano Mattoccia. 2021. Learning optical flow from still images. In *IEEE Conf. Comput. Vis. Pattern Recog.* 15201–15211.

[2] Nicolas Bonneel, James Tompkin, Kalyan Sunkavalli, Deqing Sun, Sylvain Paris, and Hanspeter Pfister. 2015. Blind video temporal consistency. *ACM Trans. Graphic* 34, 6 (2015), 196:1–196:9.

[3] Nicolas Bonneel, James Tompkin, Kalyan Sunkavalli, Deqing Sun, Sylvain Paris, and Hanspeter Pfister. 2015. Blind video temporal consistency. *ACM Trans. Graphic* 34, 6 (2015), 1–9.

[4] Vladimir Bychkovsky, Sylvain Paris, Eric Chan, and Frédo Durand. 2011. Learning photographic global tonal adjustment with a database of input/output image pairs. In *IEEE Conf. Comput. Vis. Pattern Recog.* 97–104.

[5] Yoav Chai, Raja Giryes, and Lior Wolf. 2020. Supervised and Unsupervised Learning of Parameterized Color Enhancement. 981–989.

[6] Chen Chen, Qifeng Chen, Minh N Do, and Vladlen Koltun. 2019. Seeing motion in the dark. In *Int. Conf. Comput. Vis.* 3185–3194.

[7] Yu-Sheng Chen, Yu-Ching Wang, Man-Hsin Kao, and Yung-Yu Chuang. 2018. Deep photo enhancer: Unpaired learning for image enhancement from photographs with gans. In *IEEE Conf. Comput. Vis. Pattern Recog.* 6306–6314.

[8] Xuan Dong, Boyan Bonev, Yu Zhu, and Alan L. Yuille. 2015. Region-based temporally consistent video post-processing. In *IEEE Conf. Comput. Vis. Pattern Recog.* 714–722.

[9] Gabriel Eilertsen, Rafal K Mantiuk, and Jonas Unger. 2019. Single-frame regularization for temporally stable cnns. In *IEEE Conf. Comput. Vis. Pattern Recog.* 11176–11185.

[10] Michaël Gharbi, Jiawen Chen, Jonathan T Barron, Samuel W Hasinoff, and Frédo Durand. 2017. Deep bilateral learning for real-time image enhancement. *ACM Trans. Graphic* 36, 4 (2017), 1–12.

[11] Michael D Grossberg and Shree K Nayar. 2003. What is the space of camera response functions?. In *2003 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2003. Proceedings.*, Vol. 2. IEEE, II–602.

[12] Chunle Guo, Chongyi Li, Jichang Guo, Chen Change Loy, Junhui Hou, Sam Kwong, and Runmin Cong. 2020. Zero-reference deep curve estimation for low-light image enhancement. In *IEEE Conf. Comput. Vis. Pattern Recog.* 1780–1789.

[13] Xiaojie Guo, Yu Li, and Haibin Ling. 2016. LIME: Low-light image enhancement via illumination map estimation. *IEEE Trans. Image Process.* 26, 2 (2016), 982–993.

[14] Jingwen He, Yihao Liu, Yu Qiao, and Chao Dong. 2020. Conditional Sequential Modulation for Efficient Global Image Retouching. In *Eur. Conf. Comput. Vis.* 679–695.

[15] Shi-Min Hu, Dun Liang, Guo-Ye Yang, Guo-Wei Yang, and Wen-Yang Zhou. 2020. Jittor: a novel deep learning framework with meta-operators and unified graph execution. *Information Sciences* 63, 222103 (2020), 1–21.

[16] Yuanming Hu, Hao He, Chenxi Xu, Baoyuan Wang, and Stephen Lin. 2018. Exposure: A white-box photo post-processing framework. *ACM Trans. Graphic* 37, 2 (2018), 1–17.

[17] Andrey Ignatov, Nikolay Kobyshev, Radu Timofte, Kenneth Vanhoey, and Luc Van Gool. 2017. DSLR-quality photos on mobile devices with deep convolutional networks. In *Int. Conf. Comput. Vis.* 3277–3285.

[18] Andrey Ignatov, Nikolay Kobyshev, Radu Timofte, Kenneth Vanhoey, and Luc Van Gool. 2018. WESPE: weakly supervised photo enhancer for digital cameras. In *IEEE Conf. Comput. Vis. Pattern Recog. Worksh.* 691–700.

[19] Haiyang Jiang and Yinqiang Zheng. 2019. Learning to see moving objects in the dark. In *Int. Conf. Comput. Vis.* 7324–7333.

[20] Yifan Jiang, Xinyu Gong, Ding Liu, Yu Cheng, Chen Fang, Xiaohui Shen, Jianchao Yang, Pan Zhou, and Zhangyang Wang. 2021. EnlightenGAN: Deep light enhancement without paired supervision. *IEEE Trans. Image Process.* 30 (2021), 2340–2349.

[21] Younghyun Jo and Seon Joo Kim. 2021. Practical Single-Image Super-Resolution Using Look-Up Table. In *IEEE Conf. Comput. Vis. Pattern Recog.* 691–700.

[22] Dahun Kim, Sanghyun Woo, Joon-Young Lee, and In So Kweon. 2019. Deep Video Inpainting. In *IEEE Conf. Comput. Vis. Pattern Recog.* 5792–5801.

[23] Han-Ul Kim, Young Jun Koh, and Chang-Su Kim. 2020. Global and local enhancement networks for paired and unpaired image enhancement. In *Eur. Conf. Comput. Vis.* 339–354.

[24] Han-Ul Kim, Young Jun Koh, and Chang-Su Kim. 2020. PieNet: Personalized Image Enhancement Network. In *Eur. Conf. Comput. Vis.* 374–390.

[25] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[26] Wei-Sheng Lai, Jia-Bin Huang, Oliver Wang, Eli Shechtman, Ersin Yumer, and Ming-Hsuan Yang. 2018. Learning blind video temporal consistency. In *Eur. Conf. Comput. Vis.* 170–185.

[27] Chongyi Li, Chunle Guo, Linghao Han, Jun Jiang, Ming-Ming Cheng, Jinwei Gu, and Chen Change Loy. 2021. Low-Light Image and Video Enhancement Using Deep Learning: A Survey. *IEEE Trans. Pattern Anal. Mach. Intell.* (2021).

[28] https://cg.cs.tsinghua.edu.cn/latex/project/6238b8aae098d5006f28f5adChongyi Li, Chunle Guo, Qiming Ai, Shangchen Zhou, and Chen Change Loy. 2020. Flexible piecewise curves estimation for photo enhancement. *arXiv preprint arXiv:2010.13412* (2020).

[29] Jie Liang, Hui Zeng, Miaomiao Cui, Xuansong Xie, and Lei Zhang. 2021. PPR10K: A Large-Scale Portrait Photo Retouching Dataset With Human-Region Mask and Group-Level Consistency. In *IEEE Conf. Comput. Vis. Pattern Recog.* 653–661.

[30] Enyu Liu, Songnan Li, and Shan Liu. 2020. Color enhancement using global parameters and local features learning. In *Asia Conf. Comput. Vis.*

[31] Feifan Lv, Feng Lu, Jianhua Wu, and Chongsoon Lim. 2018. MBLLEN: Low-light image/video enhancement using CNNs.. In *Brit. Mach. Vis. Conf.*, Vol. 220. 4.

[32] Sean Moran, Steven McDonagh, and Gregory Slabaugh. 2021. Curl: Neural curve layers for global image enhancement. In *Int. Conf. Pattern Recog.* 9796–9803.

[33] Zhangkai Ni, Wenhan Yang, Shiqi Wang, Lin Ma, and Sam Kwong. 2020. Towards unsupervised deep image enhancement with generative adversarial network. *IEEE Trans. Image Process.* 29 (2020), 9140–9151.

[34] Jongchan Park, Joon-Young Lee, Donggeun Yoo, and In So Kweon. 2018. Distort-and-recover: Color enhancement using deep reinforcement learning. In *IEEE Conf. Comput. Vis. Pattern Recog.* 5928–5936.

[35] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. 2019. Pytorch: An imperative style, high-performance deep learning library. In *Adv. Neural Inform. Process. Syst.*, Vol. 32.

[36] Lyndsey C Pickup, Zheng Pan, Donglai Wei, YiChang Shih, Changshui Zhang, Andrew Zisserman, Bernhard Scholkopf, and William T Freeman. 2014. Seeing the arrow of time. In *IEEE Conf. Comput. Vis. Pattern Recog.* 2035–2042.

[37] Jing Shi, Ning Xu, Yihang Xu, Trung Bui, Franck Dernoncourt, and Chenliang Xu. 2021. Learning by Planning: Language-Guided Global Image Editing. In *IEEE Conf. Comput. Vis. Pattern Recog.* 13590–13599.

[38] Yuda Song, Hui Qian, and Xin Du. 2021. StarEnhancer: Learning Real-Time and Style-Aware Image Enhancement. In *Int. Conf. Comput. Vis.* 4126–4135.

[39] Ruixing Wang, Qing Zhang, Chi-Wing Fu, Xiaoyong Shen, Wei-Shi Zheng, and Jiaya Jia. 2019. Underexposed photo enhancement using deep illumination estimation. In *IEEE Conf. Comput. Vis. Pattern Recog.* 6849–6857.

[40] Tao Wang, Yong Li, Jingyang Peng, Yipeng Ma, Xian Wang, Fenglong Song, and Youliang Yan. 2021. Real-time Image Enhancer via Learnable Spatial-aware 3D Lookup Tables. In *Int. Conf. Comput. Vis.* 2471–2480.

[41] Ting-Chun Wang, Ming-Yu Liu, Jun-Yan Zhu, Guilin Liu, Andrew Tao, Jan Kautz, and Bryan Catanzaro. 2018. Video-to-Video Synthesis. *Adv. Neural Inform. Process. Syst.* 31 (2018).

[42] Wei Wang, Xin Chen, Cheng Yang, Xiang Li, Xuemei Hu, and Tao Yue. 2019. Enhancing low light videos by exploring high sensitivity camera noise. In *Int. Conf. Comput. Vis.* 4111–4119.

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

[44] Chen Wei, Wenjing Wang, Wenhan Yang, and Jiaying Liu. 2018. Deep Retinex Decomposition for Low-Light Enhancement. In *Brit. Mach. Vis. Conf.*

[45] Jianzhou Yan, Stephen Lin, Sing Bing Kang, and Xiaoou Tang. 2014. A learning-to-rank approach for image color enhancement. In *IEEE Conf. Comput. Vis. Pattern Recog.* 2987–2994.

[46] Zhicheng Yan, Hao Zhang, Baoyuan Wang, Sylvain Paris, and Yizhou Yu. 2016. Automatic Photo Adjustment Using Deep Neural Networks. *ACM Trans. Graphic* 35, 2 (2016), 11:1–11:15.

[47] Syed Waqas Zamir, Aditya Arora, Salman Khan, Munawar Hayat, Fahad Shahbaz Khan, Ming-Hsuan Yang, and Ling Shao. 2020. Learning Enriched Features for Real Image Restoration and Enhancement. In *Eur. Conf. Comput. Vis.*

[48] Hui Zeng, Jianrui Cai, Lida Li, Zisheng Cao, and Lei Zhang. 2020. Learning image-adaptive 3d lookup tables for high performance photo enhancement in real-time. *IEEE Trans. Pattern Anal. Mach. Intell.* (2020).

[49] Xiaohang Zhan, Xingang Pan, Ziwei Liu, Dahua Lin, and Chen Change Loy. 2019. Self-supervised learning via conditional motion propagation. In *IEEE Conf. Comput. Vis. Pattern Recog.* 1881–1889.

[50] Fan Zhang, Yu Li, Shaodi You, and Ying Fu. 2021. Learning temporal consistency for low light video enhancement from single images. In *IEEE Conf. Comput. Vis. Pattern Recog.* 4967–4976.

[51] Mohan Zhang, Qiqi Gao, Jinglu Wang, Henrik Turbell, David Zhao, Jinhui Yu, and Yan Lu. 2020. RT-VENet: A Convolutional Network for Real-time Video Enhancement. In *ACM Int. Conf. Multimedia.* 4088–4097.

[52] Lin Zhao, Shao-Ping Lu, Tao Chen, Zhenglu Yang, and Ariel Shamir. 2021. Deep Symmetric Network for Underexposed Image Enhancement With Recurrent Attentional Learning. In *Int. Conf. Comput. Vis.* 12075–12084.

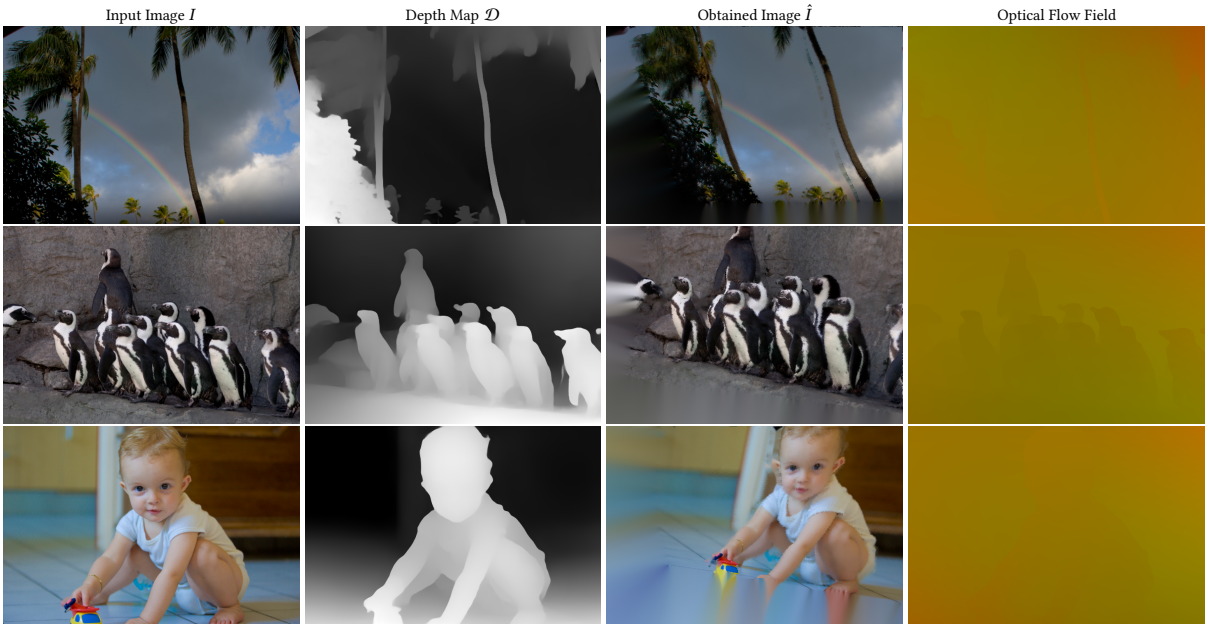

Figure 8: The example results of virtual camera motion engine. From left to right: a) Input, b) Estimated depth map, c) The generated image under a new virtual view, and d) Optical flow field consequence of virtual camera motion.

A MOTION FIELD INFERENCE

We generate non-existed optical flow from the still images to represent the motion field of video sequences via a virtual camera motion engine [1]. We use the default setting in the official implementation without the segmentation tool and randomly sample the transformation matrix for the datasets. Fig. 8 gives the visual results of the generated image and obtained optical flow on the FiveK dataset. As can be seen, such a physics-based engine can obtain reasonable new images for mimicking the adjacent frame, benefit from which our image-to-video model can model the temporal consistency directly.

B VIDEO DATASET SYNTHESIS

Following [51], we synthesize the under-exposed and over-exposed samples by adjusting the exposure ratios of irradiances in the raw image. As the raw image is not always available, the pseudo raw image is first transformed with the reverse operation of the camera response function (CRF), *i.e.*, inverse camera response function (ICRF). The CRF f relates the actual measured intensity value I_B at a photosensitive element to the image irradiance I_E by $I_B = f(I_E)$. We use the sigmoid function as $f(\cdot)$, thus the ICRF $g := f^{-1}$ can be formulated as:

$$I_E = g(I_B) = \sqrt[n]{\frac{\sigma I_B}{1+\sigma-I_B}}, \tag{14}$$

where $n = 9$ and $\sigma = 0.6$ according to the mean of the collected camera curves [11]. Then the exposure of the input images can be adjusted by multiplying the image irradiance by scaling factor s.

We denote the exposure scaling factor as s_o for the overexposure image and determine it by the clipped fraction of the image information v_o after adjusting the exposure. We use the percentage histogram function $H(\cdot)$ to measure the image information, thus, the exposure scaling s_o for overexposure augmentation is determined by:

$$s_o = \frac{1}{i_t}, \quad s.t. \sum_{i=0}^{i_t} H(i) = 1 - v_o, \tag{15}$$

Similarly, we suppose s_u is the exposure scaling factor for the underexposure image, and v_u is the clipped fraction of the image information after exposure adjustment. Thus, the exposure scaling factor underexposure augmentation is determined by:

$$s_u = i_t, \quad s.t. \sum_{i=0}^{i_t} H(i) = v_u. \tag{16}$$

According to [51], we set $v_o \in [0.1, 0.2]$ for the overexposure adjustment, and $v_u \in [0.4, 0.6]$ for the underexposure adjustment. After sampling the exposure scaling factor, the augmented images I'_B can be obtained by performing exposure adjustment and mapping to the intensity value via CRF, *i.e.*, $I'_B = f(s \cdot I_E)$. In our video dataset, 85% videos are randomly sampled as under-exposed data, and the remaining 15% videos are over-exposed data. In each video, frames in each video clip (5 frames) employ the same exposure scaling factor s, while the following clip uses the exposure scaling factor sampled from $[s-\delta, s+\delta]$. We set δ as 0.02 for synthesizing videos shot in the environment with normal illumination.