Blurry Video Compression
A Trade-off between Visual Enhancement and Data Compression

Dawit Mureja Argaw
KAIST

Junsik Kim
Harvard University

In So Kweon
KAIST

Abstract

Existing video compression (VC) methods primarily aim to reduce the spatial and temporal redundancies between consecutive frames in a video while preserving its quality. In this regard, previous works have achieved remarkable results on videos acquired under specific settings such as instant (known) exposure time and shutter speed which often result in sharp videos. However, when these methods are evaluated on videos captured under different temporal priors, which lead to degradations like motion blur and low frame rate, they fail to maintain the quality of the contents. In this work, we tackle the VC problem in a general scenario where a given video can be blurry due to predefined camera settings or dynamics in the scene. By exploiting the natural trade-off between visual enhancement and data compression, we formulate VC as a min-max optimization problem and propose an effective framework and training strategy to tackle the problem. Extensive experimental results on several benchmark datasets confirm the effectiveness of our method compared to several state-of-the-art VC approaches.

1. Introduction

Video compression (VC) methods primarily aim to jointly compress the *motion* estimated between consecutive frames and the *residual* computed between the reconstructed frames and their original counterpart. In this regard, existing VC approaches [2, 11, 12, 17, 21, 22, 24–27, 40, 46, 49, 50] have achieved remarkable results on videos acquired under specific settings such as instant (known) exposure time and shutter speed that often result in distinctively sharp videos with sufficiently high frame rate. However, when these methods are evaluated on videos captured under different temporal priors, such as slow shutter speed, long exposure time, and fast-moving objects, which lead to degradations like motion blur and low frame rate, they perform very poorly and fail to preserve the input video quality. This is mainly because motion and residual information cannot be precisely computed and compressed between frames with degraded contents and relatively large temporal distances, and hence existing approaches cannot effectively generalize to complicated real-world scenarios. Furthermore, this limitation cannot be trivially solved by retraining previous works with blurry footage owing to the nature of their problem formulation.

With the recent progress in deep network-based motion deblurring [10, 31, 38, 41, 44], a straightforward approach to tackling the task at hand would be to cascade off-the-shelf deblurring and video compression models. However, using cascade models results in sub-optimal performance. First, the pixel errors introduced in the deblurring stage would propagate to the compression stage, thus degrading the overall performance. Second, due to the arbitrariness of the artifacts introduced in the deblurring stage, since each frame is processed independently, temporal smoothness can not be ensured. As a result, the decoded video will suffer from flickering artifacts. The alternative strategy of reverse cascading (compression followed by deblurring) also suffers from the same limitations.

An attempt to address these limitations by deploying a naïve end-to-end optimization scheme on cascade models, unexpectedly, results in an even worse performance. This is mainly due to the inherent trade-off between visual enhancement and data compression when performing the joint task. Intuitively speaking, a blurry video can be regarded as the pseudo-compressed version of its sharp equivalent. On the other hand, a blind attempt to enhance the visual quality of the given blurry video before compression would in turn have a decompressing effect, *i.e.* it increases the amount of bits required to encode the given data.

We have observed that an end-to-end training of cascade models with the standard rate-distortion optimization [25, 49, 50] fails to exploit this trade-off and makes the cascade models converge to a *bad local minima* [14, 20, 37], where the deblurring network collapses to an identity function in deblurring + compression (`D` + `C`) cascade models, whereas the compression network incurs a heavy encoding cost in compression + deblurring (`C` + `D`) cascade models.

This paper tackles the video compression problem in a situation, where a video may contain blurry regions due to a

predefined camera setting or dynamics in a scene. Inspired by the aforementioned trade-off, we formulate the task at hand as a min-max optimization problem, *i.e.* we adaptively maximize the visual quality of a given video while simultaneously minimizing the number of bits needed to encode it. To this end, we present a single, end-to-end trainable network with interrelated visual enhancement and compression modules. To mitigate erroneous motion estimation due to blur corrupted pixels or large displacements, we propose a context-aware flow refinement mechanism. Finally, we devise an effective training strategy to tailor each component in our network toward optimal compression performance. We show the versatility of our approach by demonstrating that *one* well-trained model can be used for 1) generating sharp, yet compressed videos from raw blurry videos and 2) compressing videos while preserving the input prior, *i.e.* either sharp or blur prior.

Contributions. To summarize, our contributions are twofold: (**1**) To the best of our knowledge, this is the first work that studies neural compression for blurry videos. Besides proper problem formulation, our work introduces an effective framework to tackle the problem. (**2**) We comprehensively analyze our work and competing baselines in several settings and confirm the merit of the proposed approach.

2. Related works

Early VC algorithms such as H.264 [45] and H.265 [39] achieve highly efficient compression performance, however, they are based on manually designed modules. In the following years, several works [7, 18, 19, 23, 47] have attempted to substitute some of the components in traditional codecs with DNN-based methods, yet none of these works are end-to-end trainable. Meanwhile, Wu *et al.* [46] formulated the VC problem as an interpolation between compressed images and trained a deep network in an end-to-end manner. However, their model performance is below the widely used video coding standards. Lu *et al.* [25] mimicked the pipeline in conventional methods and proposed the first deep VC framework that has achieved state-of-the-art performance. Most follow-up works focused on improving compression performance using: multiple reference frames [21, 34, 49, 50], hierarchical compression [50], recurrent models [49], adaptive blurring [2], feature-level compression [8, 9, 12], contextual video coding [17], transformer-based architecture [27] and GAN losses to add realism to decoded videos [26, 51]. However, most previous works are prone to motion artifacts, such as blur and low frame rate, and fail to preserve image quality when performing compression under these settings. Our work tackles this problem via joint formulation of visual enhancement and data compression.

3. Methodology

Problem Formulation. Given a sequence of blurred inputs $\{B_t, B_{t+1}, \ldots, B_{t+n}\}$, we aim to output a visually sharp yet compressed video $\{x_t, x_{t+1}, \ldots, x_{t+n}\}$. Let $\mathcal{Q}$ represent a metric to measure the visual quality of frames and $\mathcal{S}$ represent the number of bits required to encode a given data. We formulate the joint blur reduction and video compression task as a min-max problem by optimizing a single model $\mathcal{F}$ with parameters Θ in an end-to-end manner with a compression loss $\mathcal{L}$ as follows,

$$\begin{aligned} \min_{\mathbb{U}\subseteq\Theta} \max_{\mathbb{V}\subseteq\Theta} \quad & \sum \mathcal{L}\Big(\mathcal{F}_\Theta\left(\{B_t\}_{t=1}^{n} | x_{t-1}\right) \Big| \{X_t\}_{t=1}^{n}\Big) \\ \text{subject to} \quad & \mathcal{Q}\big(\{x_t\}_{t=1}^{n}\big) \gg \mathcal{Q}\big(\{B_t\}_{t=1}^{n}\big) \\ & \mathcal{S}\big(\{x_t\}_{t=1}^{n}\big) \ll \mathcal{S}\big(\{B_t\}_{t=1}^{n}\big) \end{aligned} \tag{1}$$

where x_{t-1} denotes a keyframe encoded using an image compression method similar to 'I-frame' in traditional codecs [29, 32, 42], $\{X_t, X_{t+1}, \ldots, X_{t+n}\}$ represents the ground truth (GT) sequence, $\mathbb{V}$ denotes a subset of model parameters that contribute towards maximizing the quality of the input blurry frames with respect to their sharp counterparts and $\mathbb{U}$ represents a subset of model parameters associated with quantizing motion and residual information to minimize temporal redundancy between frames.

Since the visual enhancement process increases the Bpp of a video, optimizing $\mathbb{V}$ has a decompressing effect that contradicts the compression objective $\mathcal{L}$. Therefore, joint training of *blur reduction* and *video compression* becomes a min-max problem. As mentioned previously, naïve joint training does not work since the two tasks counter each other. In this work, we propose a framework that optimizes these two contradicting objectives. The overview of our proposed framework is depicted in Fig. 1.

3.1. Visual Enhancement

To reduce the blur of the given input B_t, we design a visual enhancement network (`VENet`) which has two interrelated components. The first part estimates an additive vector representation b_t, which we refer to as *blur residual*, to offset the blur from B_t. If the given input is a sharp frame, then b_t is approximately a zero vector of the same size as the given input. The second part of the `VENet`, on the other hand, compresses the blur residual information. The compressed blur residual $\hat{b}_t$ will then be added to B_t to output a visually enhanced frame $\hat{x}_t$ that will be used in the next steps (see Fig. 1a). The main motivation behind such a design choice comes in twofold. First, unlike blind deblurring approaches [10, 31, 38, 41, 44] that directly output a deblurred frame, our `VENet` enables us to simultaneously optimize the quality and size of the enhanced frame. Second, as our network is trained in an end-to-end manner, the

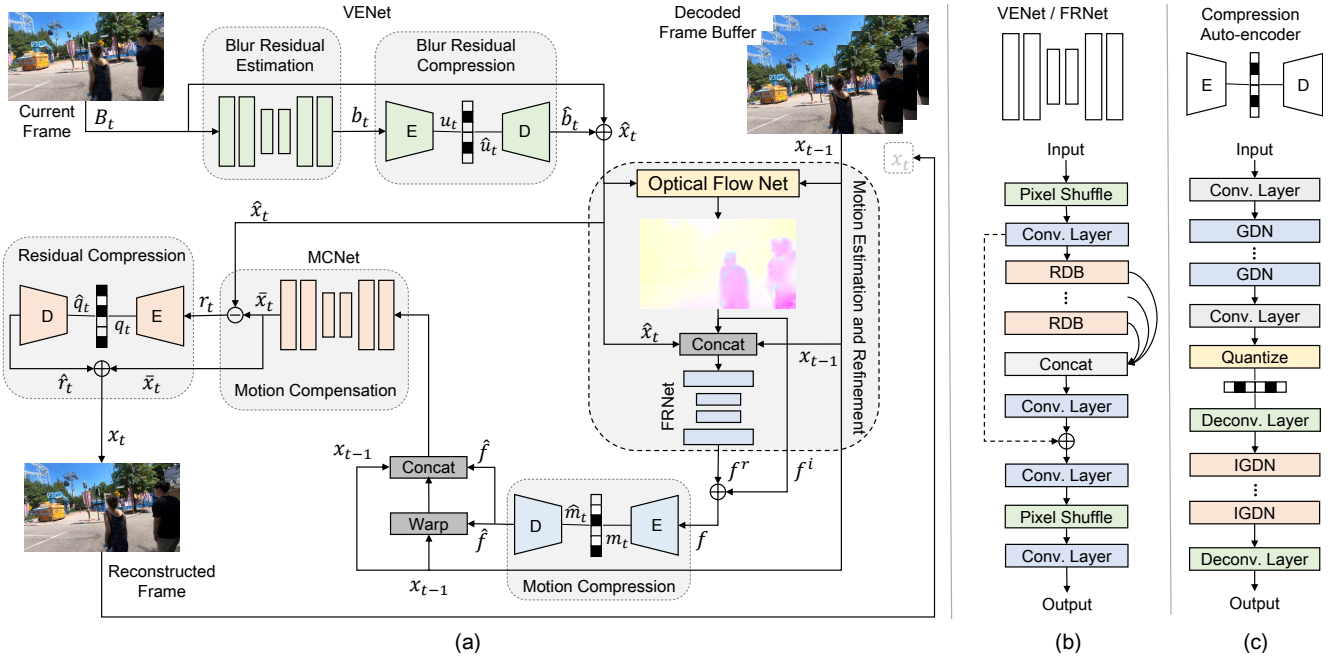


Figure 1. (a) Overview of our proposed framework. (b) The backbone structure for blur and flow residual estimation networks. (c) Network architecture for the compression autoencoder.

VENet learns to maximize the quality of the input frame for optimal video compression.

To estimate the blur residual b_t conditioned on the input B_t, we used a variation of residual dense network [53] as a backbone network. As shown in Fig. 1b, the backbone network consists of a pixel shuffle layer, convolutional layers, residual dense blocks (RDB) [53], and a sub-pixel convolution (pixel reshuffle) layer [36]. In order to compress the estimated blur residual information b_t, we adopted an auto-encoder style network [3,25,50]. As depicted in Fig. 1c, the blur residual is fed into a series of convolution (deconvolution) and nonlinear transform layers. Given a blur residual b_t of size $M \times N \times 3$, the encoder generates a blur residual representation u_t of size $M/16 \times N/16 \times 128$. Then u_t is quantized to $\hat{u}_t$. We used the *factorized* entropy model [4] for quantization. The decoder inputs the quantized representation and reconstructs the blur residual information $\hat{b}_t$. The enhanced frame $\hat{x}_t$ is then obtained by adding $\hat{b}_t$ to the blurred input B_t, *i.e.* $\hat{x}_t = B_t + \hat{b}_t$.

3.2. Motion Estimation and Compression

To reduce temporal redundancy in the given video, we first estimate the motion between the current enhanced frame $\hat{x}_t$ and the previous reconstructed frame x_{t-1}. We employ a pre-trained optical flow network [33] to predict an initial flow $f^i_{t\to t-1}$. Most existing video compression works fine-tune the pre-trained flow network using a warp loss (Eq. (2)) for better motion estimation. However, we argue that such a modification has two important limitations. First, it assumes that $\hat{x}_t$ and x_{t-1} are absolute references, and thus does not attend to potential motion artifacts in the inputs. This in turn leads to erroneous results (see Sec. 5). Second, as the receptive field of the flow network is fixed, it is challenging to adaptively handle large motions. To address these limitations, we propose a flow refinement network (FRNet) and an attention-based loss function for effective motion estimation and compression.

As shown in Fig. 1a, FRNet inputs the initial flow $f^i_{t\to t-1}$, $\hat{x}_t$ and x_{t-1} and outputs a residual flow $f^r_{t\to t-1}$ which will be added to the initial flow to generate a refined flow (Eq. (3), Eq. (4)). We used a residual dense architecture (see Fig. 1b) with three RDBs to generate $f^r_{t\to t-1}$.

The refined flow information $f_{t\to t-1}$ is then encoded, quantized, and reconstructed as $\hat{f}_{t\to t-1}$ using a flow auto-encoder network (see Fig. 1a).

$$\mathcal{L}_{\text{warp}} = \sum \left\| \hat{x}_t - \mathcal{W}_b(x_{t-1}, \hat{f}^i_{t\to t-1}) \right\|_2 \tag{2}$$

$$f^r_{t\to t-1} = \text{FRNet}(\hat{x}_t \,||\, x_{t-1} \,||\, f^i_{t\to t-1}) \tag{3}$$

$$f_{t\to t-1} = f^i_{t\to t-1} + f^r_{t\to t-1} \tag{4}$$

where $\mathcal{W}_b$ denotes a back-warping layer, $\hat{f}^i_{t\to t-1}$ denotes the reconstructed initial flow and $||$ stands for channel-wise concatenation.

To tailor the flow refinement process for the task at hand, we designed a context-aware training loss that enforces FRNet to attend to the visually enhanced regions. Specifically, we generate an attention map by scoring the different regions of $\hat{x}_t$ according to their degree of enhancement (with respect to B_t) so that the motion refinement stage

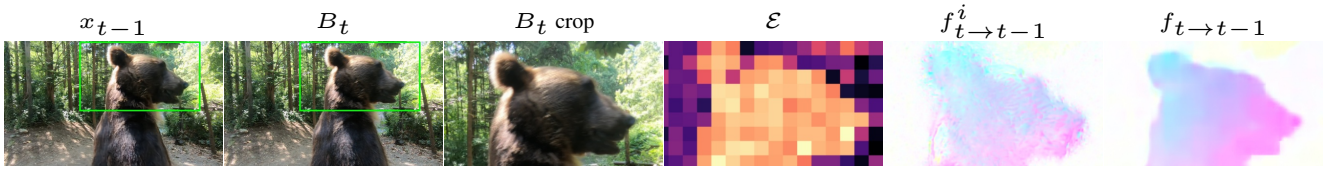


Figure 2. Visualization of motion refinement. $\mathcal{E}$, $f^i_{t\to t-1}$ and $f_{t\to t-1}$ denote the attention map, initial flow, and refined flow, respectively.

knows which regions to particularly focus on (see Fig. 2). To achieve this, we first compute the error map $\mathcal{E}$ which is defined as the mean squared error between $\hat{x}_t$ and the corresponding GT frame X_t, *i.e.* $\|\hat{x}_t - X_t\|_2$. $\mathcal{E}$ is a 2D tensor of size $M \times N$, where values are averaged across channel. To avoid a noisy map, we further applied an average pooling layer of kernel size k and stride k and assign each pixel in the error map with the corresponding average value of its neighborhood, *i.e.* we segmented $\mathcal{E}$ into $\frac{M \cdot N}{k^2}$ regions of size $k \times k$. Then we rank and label each region with integer values $\{v, \ldots, v\frac{M \cdot N}{k^2}\}$, where v is a constant to ensure increased variance in the error map distribution.

$$\mathcal{E} = \texttt{rank}_{[v, v\frac{M \cdot N}{k^2}]}\Big[\texttt{AvgPool}_{(k,k)}\big(\|\hat{x}_t - X_t\|_2\big)\Big] \quad (5)$$

The higher values in $\mathcal{E}$ represent segments that still have motion artifacts while the lower values represent regions that are enhanced (or that were sharp initially). By using $\mathcal{E}$ as an attention weight, we propose a new loss, which we refer to as *context-aware loss* ($\mathcal{L}_{\texttt{CaL}}$), for informed motion refinement and compression as shown in Eq. (7). $\mathcal{E}$ is min-max scaled between 0 and 1 (Eq. (6)) in order not to disrupt the natural trade-off between distortion and bit-rate during training (refer to Sec. 3.4).

$$w = \frac{\mathcal{E} - \min(\mathcal{E})}{\max(\mathcal{E}) - \min(\mathcal{E})} \quad (6)$$

$$\mathcal{L}_{\texttt{CaL}} = \sum \Big\| w \cdot \big[\hat{x}_t - \mathcal{W}_b(x_{t-1}, \hat{f}_{t\to t-1})\big] \Big\|_2 \quad (7)$$

3.3. Residual Compression

Given the refined flow $\hat{f}_{t\to t-1}$, we back-warp the reference frame x_{t-1} to reconstruct the current frame. To compensate for the artifacts in the warped frame, we further process it using a motion compensation network (`MCNet`). As depicted in Fig. 1a, `MCNet` inputs the warped frame, the reference frame x_{t-1} and the motion vector $\hat{f}_{t\to t-1}$ and outputs a motion compensated frame $\overline{x}_t$ which is expected to be as close as the enhanced frame $\hat{x}_t$ (Eq. (8)). We used a residual-UNet architecture like the one used in [25, 50] for `MCNet`.

$$\overline{x}_t = \texttt{MCNet}\big(\mathcal{W}_b(x_{t-1}, \hat{f}_{t\to t-1}) \,||\, x_{t-1} \,||\, \hat{f}_{t\to t-1}\big) \quad (8)$$

Finally, the residual between the enhanced raw frame $\hat{x}_t$ and the motion compensated frame $\overline{x}_t$, *i.e.* $r_t = \hat{x}_t - \overline{x}_t$, is compressed using a residual encoder-decoder network. Like the blur residual and motion compression, the residual information r_t is first encoded to a latent representation q_t, then quantized to $\hat{q}_t$ and finally decoded to $\hat{r}_t$ (see Fig. 1a). The reconstructed residual information $\hat{r}_t$ is added to the motion compensated frame $\overline{x}_t$ to obtain the compressed frame x_t, *i.e.* $x_t = \overline{x}_t + \hat{r}_t$.

3.4. Training Strategy

The goal of our video compression framework is to minimize the number of bits used for encoding a given video frame B_t, while simultaneously enhancing its quality with respect to its sharp counterpart X_t and reducing the distortion between the enhanced frame $\hat{x}_t$ and the reconstructed frame x_t. Therefore, we formulate the optimization problem as follows,

$$\mathcal{L} = \lambda_e E + \lambda_d D + R \quad (9)$$

where λ_e and λ_d are hyperparameters to control the three-way trade-off between the enhancement E, distortion D and bit-rate R. For the visual enhancement part, we jointly optimize the number of encoding bits for the quantized blur residual $\hat{u}_t$ and the $\ell 1$ photometric loss between the enhanced frame $\hat{x}_t$ and the corresponding GT frame X_t as shown in Eq. (10). Note that we also included the $\ell 1$ photometric loss between $B_t + b_t$ and X_t so that the blur residual b_t auto-encoder will not collapse into a *bad local minima* where $\hat{b}_t = 0$, *i.e.* $B_t = \hat{x}_t$.

$$\mathcal{L}_{\texttt{VENet}} = \lambda_e \Big[\sum \big\|X_t - (B_t + b_t)\big\|_1 + \big\|X_t - \hat{x}_t\big\|_1\Big] + R(\hat{u}_t) \quad (10)$$

where $R(\cdot)$ denotes the number of bits used for encoding the representations. We used the density model of [3] to estimate R. λ_e is defined as $\lambda_e = s\lambda_d$, where s is a step decay parameter to maintain the trade-off between the visual enhancement and compression as the training progresses.

Following previous works [21, 25, 49, 50], we used a progressive scheme to train the different components in the compression part. First, the motion estimation and compression step in Sec. 3.2 is trained by optimizing the proposed context-aware loss $\mathcal{L}_{\texttt{CaL}}$ and the bit-rate for encoding the quantized motion vector $\hat{m}_t$ (Eq. (11)). The optical flow network [33] used to obtain the initial flow $f^i_{t\to t-1}$ is initialized with pre-trained weights and remains unchanged.

Then, the motion compensation network in Sec. 3.3 is added into the training using the loss function defined in Eq. (12).

$$\mathcal{L}_{\texttt{M}} = \lambda_d \mathcal{L}_{\texttt{CaL}} + R(\hat{m}_t) \tag{11}$$

$$\mathcal{L}_{\texttt{MCNet}} = \lambda_d \sum \left\| \overline{x}_t - \hat{x}_t \right\|_2 + R(\hat{m}_t) \tag{12}$$

Finally, the distortion between the reconstructed frame x_t and the enhanced frame $\hat{x}_t$ is optimized using the loss function formulated in Eq. (13), where $D(x, y)$ is defined as a distortion metrics. We used the mean square error (MSE), *i.e.* $D(x, y) = \texttt{MSE}(x, y)$ when optimizing for PSNR and $D(x, y) = 1 - \texttt{MS-SSIM}(x, y)$ when optimizing for MS-SSIM. This is in accordance with the experiment protocol in previous works [25, 49, 50]. The total training loss for end-to-end optimization of the whole network is summarized in Eq. (14).

$$\mathcal{L}_{\texttt{D}} = \lambda_d D(x_t, \hat{x}_t) + R(\hat{m}_t) + R(\hat{q}_t) \tag{13}$$

$$\mathcal{L}_{\texttt{total}}(i) = \begin{cases} \mathcal{L}_{\texttt{VENet}} & i \leq a \\ \mathcal{L}_{\texttt{VENet}} + \mathcal{L}_{\texttt{M}} & a < i \leq b \\ \mathcal{L}_{\texttt{VENet}} + \mathcal{L}_{\texttt{MCNet}} & b < i \leq c \\ \mathcal{L}_{\texttt{VENet}} + \mathcal{L}_{\texttt{D}} & c < i \leq \max_{\texttt{iter}} \end{cases} \tag{14}$$

where a, b, c are different iteration steps and $\max_{\texttt{iter}}$ is the maximum iteration during training.

4. Experiments

4.1. Experimental Setting

Datasets. Existing works use Vimeo-90k dataset [48], which contains $89,800$ clips each with 7 sharp frames, for a model training. However, as this dataset cannot be applied to train a model in a blurry scenario, we follow the common practice in computer vision research and generate a blur dataset by synthesizing low-frame-rate videos from a sharp high-frame-rate sequence [35, 54]. To simulate a frame B_t taken by a low-frame-rate camera, we average n consecutive frames taken by a 240 fps camera.

$$B_t = \frac{1}{n} \sum_{j=\kappa t - \frac{n}{2}}^{j=\kappa t + \frac{n}{2}} I_j$$

where I_j is the j-th high-frame-rate latent image and the parameter κ determines the frame rate of the acquired frames. As n is related to the degree of blur [6], we experiment with different values of $n \in \{5, 7, 9\}$. To take the shutter closing time into account during video acquisition [54], we discard m number of consecutive frames in between before synthesizing the next frame B_{t+1}, where $m \in \{5, 3, 1\}$. We set $\kappa = m + n = 10$ to downsample the 240 fps to 24 fps, which is a common fps setting for commodity cameras. In this manner, we create blurry videos captured under different exposure settings.

We apply this scheme to Adobe240 [38], GOPRO [31] and REDS [30] datasets which provide high-frame-rate videos with a resolution of 1280×720. Most of the videos in these datasets, however, have less than a thousand frames which makes it challenging to synthesize enough and diverse training set. Hence, instead of training separately on each dataset, we used a total of 325 videos (100 from Adobe240, 25 from GOPRO, 200 from REDS) and built a training dataset that approximately has $32,500$ clips each with 7 frames. The remaining non-overlapping videos in each dataset (30 from Adobe240, 8 from GOPRO, and 70 from REDS) are used for testing. We also make use of other benchmark datasets such as UVG [28] and MCL-JCV [43] for generalization experiments.

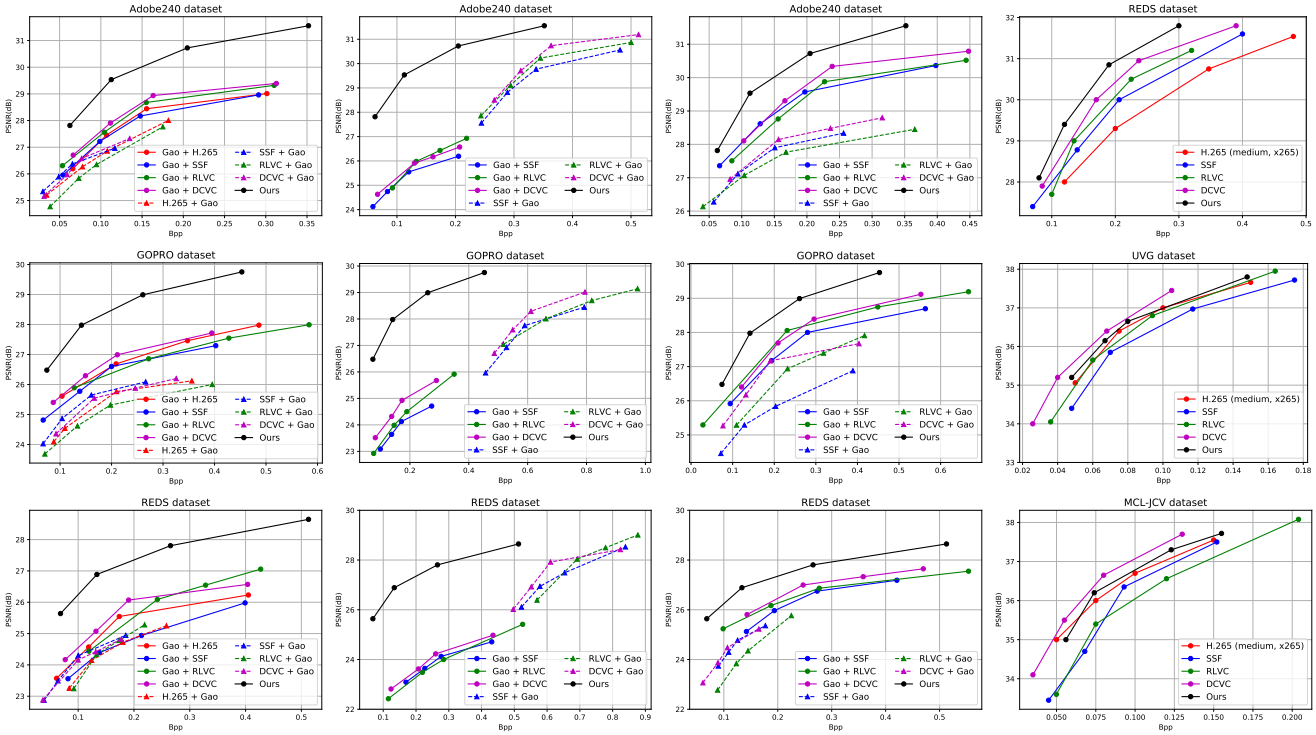
Implementation Details. Following the previous works [25, 49, 50], we train eight models by setting $\lambda_d = 256, 512, 1024, 2048$ when optimizing for PSNR and $\lambda_d = 8, 16, 32, 64$ when optimizing for MS-SSIM. For compressing the reference 'I-frame', we use BPG [5] and Lee *et al.* [16] in our PSNR and MS-SSIM models, respectively. For each model, an Adam optimizer [15] with an initial learning rate $1e-4$ and momentum 0.9 is used. We set the different iteration steps a, b, c and $\max_{\texttt{iter}}$ to $1e+5$, $2e+5$, $3e+5$ and $1e+6$, respectively. The learning rate is decayed by a factor of 10 at $5e+5$ and $8e+5$ iterations. The step decay parameter s for λ_e is initialized with 1 and reduced by 0.25 every $2e+5$ iterations. We use a mini-batch size of 4 by randomly cropping images of size 256×256 during training. Our network is implemented using Tensorflow [1].

Evaluation Metrics. We measure the distortion between the reconstructed frame x_t and the GT frame X_t on PSNR and MS-SSIM metrics with respect to the number of bits for encoding the representations m_t and q_t. Bits per pixel (Bpp) is used to represent the required bits for each pixel in the current frame.

4.2. Experimental Results

We compare our work with traditional video codecs, *i.e.* H.265 [39], and the latest learned methods for which open-source implementations are available, *i.e.* SSF [2], RLVC [50] and DCVC [17]. However, naive VC models are prone to motion artifacts and fail to preserve image quality. Hence, to compare our model with competitive baselines, we implement a cascade approach by using the state-of-the-art deblurring networks, *i.e.* Gao [10]. We create two types of cascade models, *i.e.* deblurring + compression (D + C) and compression + deblurring (C + D).

For each cascade model, we establish three types of baselines: (i) ***Off-the-shelf***, where pretrained models from the



(a) **Off-the-shelf** (b) **Joint training** (c) **Intermediate supervision** (d) **Preserving the input prior**

Figure 3. (**a**) - (**c**) Rate-distortion performance comparison between our method and (**a**) ***off-the-shelf*** cascade models. (**b**) cascade models optimized with the ***joint training*** scheme ($\mathcal{L}_{\texttt{Joint}}$). (**c**) cascade models optimized with ***intermediate supervision*** scheme ($\mathcal{L}_{\texttt{Int}}$). (**d**) Rate-distortion performance comparison on preserving the blur (*top*) and sharp (*middle* and *bottom*) priors.

respective tasks are successively used for the joint task. (ii) ***Joint training***, where cascade models are jointly trained from scratch using the standard rate-distortion loss [49], *i.e.* $\mathcal{L}_{\texttt{Joint}} = \lambda D_{\texttt{Joint}} + R$, where $D_{\texttt{Joint}}$ denotes the losses computed with respect to the GT frame X_t at different stages of training. For D + C models, $D_{\texttt{Joint}}$ is computed with respect to the output of the compression network (without explicit supervision to the deblurring network) and for C + D models, $D_{\texttt{Joint}}$ is computed with respect to the output of the deblurring network. R represents the rate optimization for motion and residual representations. (iii) ***Intermediate supervision***, where we use additional intermediate supervision between the cascaded components. For D + C models, an $\ell 1$ photometric loss is used to train the deblurring network, and the compression network is simultaneously optimized with respect to the deblurred output. The total training loss is defined in Eq. (15).

$$\mathcal{L}^{\texttt{D+C}}_{\texttt{Int}} = \lambda_d \left\| X_t - \hat{x}_t \right\|_1 + \lambda_c D_{\texttt{Int}} + R \qquad (15)$$

where $D_{\texttt{Int}}$ denotes the distortion losses computed with respect to the deblurred output $\hat{x}_t$. λ_d and λ_c represent hyper-parameters for the deblurring and compression networks, respectively. Note that the loss in Eq. (15) is analogous to the proposed training strategy in Sec. 3.4. For C + D models, the equivalent training scheme would be to first optimize the compression network with the blurry input and enhance the decoded output using the deblurring network as shown in Eq. (16).

$$\mathcal{L}^{\texttt{C+D}}_{\texttt{Int}} = \lambda_c D_{\texttt{Int}} + \lambda_d \left\| X_t - x_t \right\|_1 + R \qquad (16)$$

where $D_{\texttt{Int}}$ represents the distortion losses computed with respect to the blurry input B_t. We use the official code of each compression and deblurring model to implement the cascade baselines. For fair evaluation, each cascade model is trained and evaluated using the same settings and datasets as our method. We implement H.265 [39] in a *medium* setting using $\times 265$ encoder.

Bit-rate and Distortion. Fig. 3a depicts the rate-distortion curves of our method and *off-the-shelf* cascade models on Adobe240 [38], GOPRO [31] and REDS [30] test sets. As can be inferred from the figure, our approach performs favorably against all possible baselines on PSNR metric. For instance, given a blurred input and a budget of 0.3 Bpp, our approach can reconstruct sharp frames at a quality of 29.52 dB on average, while the second best performing method (Gao [10] + DCVC [17]) reconstructs frames at a quality of 27.68 dB. The main reason behind such a performance gap is due to error propagation in *off-the-shelf* cascade models, where blur/compression artifacts propagate from the first stage to the second stage, degrading

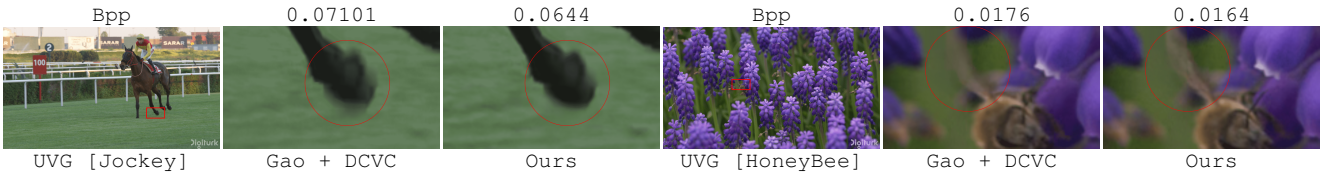


Figure 4. Qualitative analysis on real-world videos. Note that PSNR is not possible to compute as sharp GT is not available.

the overall performance. Moreover, as each network is optimized for the individual task, naïvely cascading them for the joint task often gives sub-optimal results as observed in other works [35, 54]. It can also be inferred from Fig. 3a that D + C models perform significantly better than their C + D counterparts. This is mainly because deblurring networks [10] perform poorly in the presence of both blur and compression artifacts.

One way to address the limitations of *off-the-shelf* cascade models would be to end-to-end optimize trainable cascade models. In Fig. 3b, we study the *joint training* scheme, where cascade models are trained as one network with the standard rate-distortion loss ($\mathcal{L}_{\texttt{Joint}}$). As can be seen from Fig. 3b, the *joint training* scheme results in performance worse than *off-the-shelf* models. This is mainly because of the inherent trade-off between the two tasks as discussed in Sec. 1. Our experimental analysis reveals that, for D + C models, the deblurring network basically collapses to an identity function. This is intuitive because it is easier to compress a blurry input compared to its sharper equivalent since the latter one will have more information to encode. As a consequence, the compression network alone has to reconstruct the input video while reducing its temporal redundancies but deblurring its contents at the same time, which are incompatible. Similarly, we observed that C + D models incur a heavy encoding cost as shown in Fig. 3b. This is because the compression network diverges to preserving as much information as possible for the deblurring network to enhance. These results show that the trade-off balance always tips to the second task when cascade models are trained in the *joint training* scheme, resulting in a subpar overall performance on the joint task.

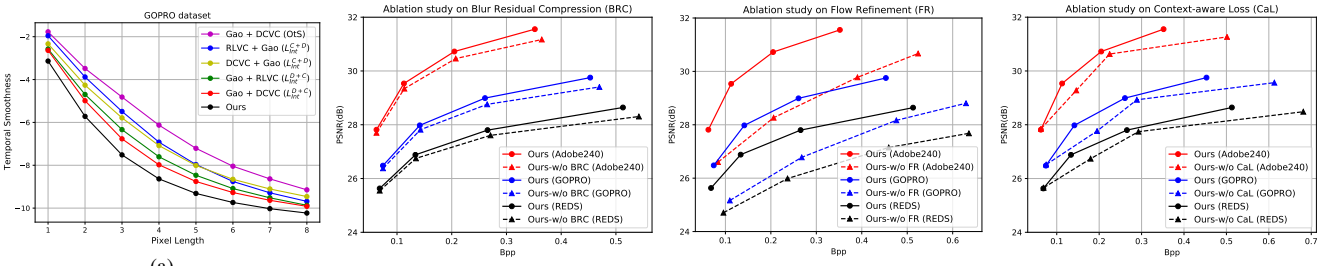
To create a stronger baseline, we optimize the cascade models using a training scheme analogous to our proposed training strategy, *i.e.* *intermediate supervision* in Eq. (15) and Eq. (16), where we introduce additional intermediate supervision between the cascaded components. As can be seen from Fig. 3c, the proposed training strategy significantly improves the performance of cascade models. It can also be inferred from Fig. 3c that D + C methods generally outperform their corresponding C + D counterparts despite the fact that C + D take much fewer bits given that the compression network encodes the blurry input in this setting. This is mainly because the deblurring network fails to blindly enhance the compressed output due to the high ill-posedness of the task. Enhancing the quality of compressed videos is a very challenging problem even for sharp videos [52], let alone for *blurred* and *compressed* inputs.

In comparison to the strongest D + C models, our network still gives a notably better result. To further analyze the results, we compute the quality gain between the highest and smallest distortions for the curves in Fig. 3c over the corresponding increase in bit-rate cost. In this metric, our approach achieves 10.80 dB/Bpp on average whereas Gao [10] + DCVC [17] could only achieve 7.86 dB/Bpp. The performance gain of our method compared to cascade D + C models trained under the same setting can be attributed to the motion refinement step in Sec. 3.2 which is crucial for optimal compression performance in a blurry scenario as opposed to fine-tuning a pretrained flow network (see Sec. 5). In Fig. 4, we show qualitative results on blurry scenes from real-world videos. As shown in the figure, our model gives a visually better output at a lower encoding cost compared to the strongest baseline.

4.3. Experimental Analyses

Preserving the Input Prior. Although motion blur is mostly considered an unwanted artifact, it can be sometimes useful to add realism to a scene. Taking that into consideration, we experiment with the idea of having the option to preserve the blur when performing compression. We achieve this during *inference* time by simply using input B_t (instead of the enhanced frame $\hat{x}_t$) to compute the frame residual r_t in Sec. 3.3, *i.e.* $r_t = B_t - \overline{x}_t$. The remaining steps follow accordingly.This mechanism is an inverse process of the visual enhancement in Sec. 3.1 where r_t is equivalent to the blur residual b_t. As previous works [2, 17, 49] typically output a blurry video given a blurry input, we use them as a baseline to evaluate our work on preserving a prior. As can be inferred from Fig. 3d (*top*), our approach (with the simple fix) performs better than state-of-the-art VC approaches in maintaining the input blur when compressing a blurry video. This result further highlights the flexibility of our proposed framework to be easily adapted to diverse conditions without the need for re-training.

The notion of preserving a prior also applies when the input video is distinctively sharp with predominantly static contents. To compare our approach and previous works in this scenario, we use the UVG [28] and MCL-JCV [43] datasets. As can be seen in Fig. 3d (*middle* and *bottom*), our model trained in a blurry setup gives a very competitive performance compared to the state-of-the-art models



(a) (b) (c) (d)

Figure 5. (**a**) Temporal smoothness comparison, (**b**) - (**d**) Ablation studies on blur residual compression, flow refinement and $\mathcal{L}_{\texttt{CaL}}$

[25, 49, 50] optimized on a sharp dataset.

Complexity Analysis. In Table 1, we analyze the runtime and parameter size of our model and the competing baselines. As shown in the table, our approach takes 0.67 seconds to reconstruct an input frame of size 1280 × 720px while the best-performing cascade models, *i.e.* Gao [10] + DCVC [17] and Gao [10] + RLVC [50], take 1.97 and 1.98 seconds, respectively.

Temporal Smoothness. To evaluate the smoothness of videos decoded by our method and competing baselines, we adopt a flow-based smoothness metric [13, 35]. We first compute the second-order differential flow between the decoded video and its GT counterpart using 3 consecutive frames at a time, *i.e.* $df = (f_{x_2 \to x_1} - f_{x_1 \to x_0}) - (f_{X_2 \to X_1} - f_{X_1 \to X_0})$, where x_0, x_1, x_2, and X_0, X_1, X_2 are frame sequences in the decoded and GT videos, respectively. The temporal smoothness metric $\mathcal{T}(l)$ is then defined as function of the pixel error length l,

$$\mathcal{T}(l) = \log \sum_{v \in df} \mathbf{1}_{[l, l+1)}(\|v\|_2) - \log |df| \qquad (17)$$

where v denotes a vector of matrix df, $|\cdot|$ represents the size of a matrix and the indicator function $\mathbf{1}_S(x)$ equals to 1 if x belongs to set S. The lower value of $\mathcal{T}(l)$ indicates better temporal stability. As can be seen from Fig. 5(a), *off-the-shelf* cascade models perform worse on the temporal smoothness metric as they are prone to error propagation. Compared to the strongest baseline (Gao [10] + DCVC [17] trained with $\mathcal{L}_{\texttt{Int}}^{\texttt{D+C}}$), our approach decodes a temporally smoother video.

5. Ablation Study

In this section, we perform extensive experiments to show the effectiveness of our designed modules. First, we study the importance of blur residual compression by training a network without the b_t auto-encoder, where the enhanced frame $\hat{x}_t$ is obtained directly by adding b_t to the input B_t, *i.e.* $\hat{x}_t = B_t + b_t$. It can be inferred from Fig. 5(b) that blur residual compression consistently leads to better performance. As mentioned previously, compressing the blur residual information plays a significant role in maintaining the balance between visual enhancement and compression for optimal overall performance.

Table 1. Evaluation on time complexity and parameter size

Method	Gao [10] + H.265 [39]	Gao [10] + SFF [2]	Gao [10] + RLVC [50]	Gao [10] + DCVC [17]	Ours
Parameters ($\times 10^6$)	**3.87**	38.08	20.56	11.81	12.74
Runtime (s)	1.52	2.35	1.98	1.97	**0.67**

We analyze the importance of the motion refinement step by training our network without `FRNet`, *i.e.* we fine-tuned the optical flow network using a warp loss. As can be seen from Fig. 5(c), directly fine-tuning the optical flow network in a blurry scenario results in significantly worse performance compared to a network trained with flow refinement. This is mainly because of erroneous initial flow estimation since the pixels in the input frames are corrupted by blur and a blind warp loss in Eq. (2) can not address this limitation (refer to Sec. 3.2). To demonstrate the benefit of the proposed context-aware loss ($\mathcal{L}_{\texttt{CaL}}$), we train the `FRNet` using the warp loss ($\mathcal{L}_{\texttt{warp}}$ in Eq. (2) using $f_{t \to t-1}$) and compare the results with a network trained using $\mathcal{L}_{\texttt{CaL}}$. As shown in Fig. 5(d) and Fig. 2, attending to the visual enhancement using $\mathcal{E}$ leads to better motion estimation and notably superior compression performance.

6. Conclusion

In this work, we tackle the video compression problem in a general situation, where unwanted blurs may be present in videos. We design various cascade models as baselines and analyze their limitations. To overcome these limitations, we propose a novel framework that can be efficiently optimized in an end-to-end manner. We have demonstrated the effectiveness and flexibility of our approach through extensive analyses on different datasets. However, there remain a few limitations. In extreme cases, where videos are severely blurred or temporally undersampled, we experimentally observed that our enhancement module `VENet` fails, therefore compromising the overall compression performance.

Acknowledgment This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT, No. RS-2023-00212845).

References

[1] Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. Tensorflow: A system for large-scale machine learning. In *12th {USENIX} symposium on operating systems design and implementation ({OSDI} 16)*, pages 265–283, 2016. 5

[2] Eirikur Agustsson, David Minnen, Nick Johnston, Johannes Balle, Sung Jin Hwang, and George Toderici. Scale-space flow for end-to-end optimized video compression. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020. 1, 2, 5, 7, 8

[3] Johannes Ballé, Valero Laparra, and Eero P Simoncelli. End-to-end optimized image compression. *arXiv preprint arXiv:1611.01704*, 2016. 3, 4

[4] Johannes Ballé, David Minnen, Saurabh Singh, Sung Jin Hwang, and Nick Johnston. Variational image compression with a scale hyperprior. *arXiv preprint arXiv:1802.01436*, 2018. 3

[5] Fabrice Bellard. Bpg image format. *URL https://bellard.org/bpg*, 1(3), 2015. 5

[6] Tim Brooks and Jonathan T Barron. Learning to synthesize motion blur. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6840–6848, 2019. 5

[7] Yuanying Dai, Dong Liu, and Feng Wu. A convolutional neural network approach for post-processing in hevc intra coding. In *International Conference on Multimedia Modeling*, pages 28–39. Springer, 2017. 2

[8] Abdelaziz Djelouah, Joaquim Campos, Simone Schaub-Meyer, and Christopher Schroers. Neural inter-frame compression for video coding. In *ICCV*, 2019. 2

[9] Runsen Feng, Yaojun Wu, Zongyu Guo, Zhizheng Zhang, and Zhibo Chen. Learned video compression with feature-level residuals. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, June 2020. 2

[10] Hongyun Gao, Xin Tao, Xiaoyong Shen, and Jiaya Jia. Dynamic scene deblurring with parameter selective sharing and nested skip connections. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3848–3856, 2019. 1, 2, 5, 6, 7, 8

[11] Zhihao Hu, Zhenghao Chen, Dong Xu, Guo Lu, Wanli Ouyang, and Shuhang Gu. Improving deep video compression by resolution-adaptive flow coding. In *European Conference on Computer Vision*, pages 193–209. Springer, 2020. 1

[12] Zhihao Hu, Guo Lu, and Dong Xu. Fvc: A new framework towards deep video compression in feature space. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1502–1511, June 2021. 1, 2

[13] Meiguang Jin, Zhe Hu, and Paolo Favaro. Learning to extract flawless slow motion from blurry videos. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019. 8

[14] Kenji Kawaguchi and Leslie Kaelbling. Elimination of all bad local minima in deep learning. In *International Conference on Artificial Intelligence and Statistics*, pages 853–863. PMLR, 2020. 1

[15] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 5

[16] Jooyoung Lee, Seunghyun Cho, and Seung-Kwon Beack. Context-adaptive entropy model for end-to-end optimized image compression. *arXiv preprint arXiv:1809.10452*, 2018. 5

[17] Jiahao Li, Bin Li, and Yan Lu. Deep contextual video compression. *Advances in Neural Information Processing Systems*, 34, 2021. 1, 2, 5, 6, 7, 8

[18] Tianyi Li, Mai Xu, Ren Yang, and Xiaoming Tao. A densenet based approach for multi-frame in-loop filter in hevc. In *2019 Data Compression Conference (DCC)*, pages 270–279. IEEE, 2019. 2

[19] Tianyi Li, Mai Xu, Ce Zhu, Ren Yang, Zulin Wang, and Zhenyu Guan. A deep learning approach for multi-frame in-loop filter of hevc. *IEEE Transactions on Image Processing*, 28(11):5663–5678, 2019. 2

[20] Shiyu Liang, Ruoyu Sun, Jason D Lee, and Rayadurgam Srikant. Adding one neuron can eliminate all bad local minima. *Advances in Neural Information Processing Systems*, 31:4350–4360, 2018. 1

[21] Jianping Lin, Dong Liu, Houqiang Li, and Feng Wu. M-lvc: Multiple frames prediction for learned video compression. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020. 1, 2, 4

[22] Jerry Liu, Shenlong Wang, Wei-Chiu Ma, Meet Shah, Rui Hu, Pranaab Dhawan, and Raquel Urtasun. Conditional entropy coding for efficient video compression. In *European Conference on Computer Vision*, pages 453–468. Springer, 2020. 1

[23] Jiaying Liu, Sifeng Xia, Wenhan Yang, Mading Li, and Dong Liu. One-for-all: Grouped variation network-based fractional interpolation in video coding. *IEEE Transactions on Image Processing*, 28(5):2140–2151, 2018. 2

[24] Guo Lu, Chunlei Cai, Xiaoyun Zhang, Li Chen, Wanli Ouyang, Dong Xu, and Zhiyong Gao. Content adaptive and error propagation aware deep video compression. In *European Conference on Computer Vision*, pages 456–472. Springer, 2020. 1

[25] Guo Lu, Wanli Ouyang, Dong Xu, Xiaoyun Zhang, Chunlei Cai, and Zhiyong Gao. Dvc: An end-to-end deep video compression framework. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11006–11015, 2019. 1, 2, 3, 4, 5, 8

[26] Fabian Mentzer, Eirikur Agustsson, Johannes Ballé, David Minnen, Nick Johnston, and George Toderici. Neural video compression using gans for detail synthesis and propagation. In *European Conference on Computer Vision*, pages 562–578. Springer, 2022. 1, 2

[27] Fabian Mentzer, George Toderici, David Minnen, Sung-Jin Hwang, Sergi Caelles, Mario Lucic, and Eirikur Agustsson. Vct: A video compression transformer. *arXiv preprint arXiv:2206.07307*, 2022. 1, 2

[28] Alexandre Mercat, Marko Viitanen, and Jarno Vanne. Uvg dataset: 50/120fps 4k sequences for video codec analysis and development. In *Proceedings of the 11th ACM Multimedia Systems Conference*, pages 297–302, 2020. 5, 7

[29] Debargha Mukherjee, Jim Bankoski, Adrian Grange, Jingning Han, John Koleszar, Paul Wilkins, Yaowu Xu, and Ronald Bultje. The latest open-source video codec vp9-an overview and preliminary results. In *2013 Picture Coding Symposium (PCS)*, pages 390–393. IEEE, 2013. 2

[30] Seungjun Nah, Sungyong Baik, Seokil Hong, Gyeongsik Moon, Sanghyun Son, Radu Timofte, and Kyoung Mu Lee. Ntire 2019 challenge on video deblurring and super-resolution: Dataset and study. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, June 2019. 5, 6

[31] Seungjun Nah, Tae Hyun Kim, and Kyoung Mu Lee. Deep multi-scale convolutional neural network for dynamic scene deblurring. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3883–3891, 2017. 1, 2, 5, 6

[32] Jens-Rainer Ohm, Gary J Sullivan, Heiko Schwarz, Thiow Keng Tan, and Thomas Wiegand. Comparison of the coding efficiency of video coding standards—including high efficiency video coding (hevc). *IEEE Transactions on circuits and systems for video technology*, 22(12):1669–1684, 2012. 2

[33] Anurag Ranjan and Michael J Black. Optical flow estimation using a spatial pyramid network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4161–4170, 2017. 3, 4

[34] Oren Rippel, Sanjay Nair, Carissa Lew, Steve Branson, Alexander G. Anderson, and Lubomir Bourdev. Learned video compression. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2019. 2

[35] Wang Shen, Wenbo Bao, Guangtao Zhai, Li Chen, Xiongkuo Min, and Zhiyong Gao. Blurry video frame interpolation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5114–5123, 2020. 5, 7, 8

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

[37] Jascha Sohl-Dickstein and Kenji Kawaguchi. Eliminating all bad local minima from loss landscapes without even adding an extra unit. *arXiv preprint arXiv:1901.03909*, 2019. 1

[38] Shuochen Su, Mauricio Delbracio, Jue Wang, Guillermo Sapiro, Wolfgang Heidrich, and Oliver Wang. Deep video deblurring for hand-held cameras. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1279–1288, 2017. 1, 2, 5, 6

[39] Gary J Sullivan, Jens-Rainer Ohm, Woo-Jin Han, and Thomas Wiegand. Overview of the high efficiency video coding (hevc) standard. *IEEE Transactions on circuits and systems for video technology*, 22(12):1649–1668, 2012. 2, 5, 6, 8

[40] Wenyu Sun, Chen Tang, Weigui Li, Zhuqing Yuan, Huazhong Yang, and Yongpan Liu. High-quality single-model deep video compression with frame-conv3d and multi-frame differential modulation. In *European Conference on Computer Vision*, pages 239–254. Springer, 2020. 1

[41] Xin Tao, Hongyun Gao, Xiaoyong Shen, Jue Wang, and Jiaya Jia. Scale-recurrent network for deep image deblurring. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 1, 2

[42] Jarno Vanne, Marko Viitanen, Timo D Hamalainen, and Antti Hallapuro. Comparative rate-distortion-complexity analysis of hevc and avc video codecs. *IEEE Transactions on Circuits and Systems for Video Technology*, 22(12):1885–1898, 2012. 2

[43] Haiqiang Wang, Weihao Gan, Sudeng Hu, Joe Yuchieh Lin, Lina Jin, Longguang Song, Ping Wang, Ioannis Katsavounidis, Anne Aaron, and C-C Jay Kuo. Mcl-jcv: a jnd-based h. 264/avc video quality assessment dataset. In *2016 IEEE International Conference on Image Processing (ICIP)*, pages 1509–1513. IEEE, 2016. 5, 7

[44] Xintao Wang, Kelvin CK Chan, Ke Yu, Chao Dong, and Chen Change Loy. Edvr: Video restoration with enhanced deformable convolutional networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 0–0, 2019. 1, 2

[45] Thomas Wiegand, Gary J Sullivan, Gisle Bjontegaard, and Ajay Luthra. Overview of the h. 264/avc video coding standard. *IEEE Transactions on circuits and systems for video technology*, 13(7):560–576, 2003. 2

[46] Chao-Yuan Wu, Nayan Singhal, and Philipp Krahenbuhl. Video compression through image interpolation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 416–431, 2018. 1, 2

[47] Mai Xu, Tianyi Li, Zulin Wang, Xin Deng, Ren Yang, and Zhenyu Guan. Reducing complexity of hevc: A deep learning approach. *IEEE Transactions on Image Processing*, 27(10):5044–5059, 2018. 2

[48] Tianfan Xue, Baian Chen, Jiajun Wu, Donglai Wei, and William T Freeman. Video enhancement with task-oriented flow. *International Journal of Computer Vision*, 127(8):1106–1125, 2019. 5

[49] Ren Yang, Fabian Mentzer, Luc Van Gool, and Radu Timofte. Learning for video compression with hierarchical quality and recurrent enhancement. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6628–6637, 2020. 1, 2, 4, 5, 6, 7, 8

[50] Ren Yang, Fabian Mentzer, Luc Van Gool, and Radu Timofte. Learning for video compression with recurrent auto-encoder and recurrent probability model. *IEEE Journal of Selected Topics in Signal Processing*, 2020. 1, 2, 3, 4, 5, 8

[51] Ren Yang, Luc Van Gool, and Radu Timofte. Perceptual learned video compression with recurrent conditional gan. *arXiv preprint arXiv:2109.03082*, 2021. 2

[52] Ren Yang, Mai Xu, Zulin Wang, and Tianyi Li. Multi-frame quality enhancement for compressed video. In *Proceedings*

of the IEEE Conference on Computer Vision and Pattern Recognition, pages 6664–6673, 2018. 7

[53] Yulun Zhang, Yapeng Tian, Yu Kong, Bineng Zhong, and Yun Fu. Residual dense network for image super-resolution. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2472–2481, 2018. 3

[54] Youjian Zhang, Chaoyue Wang, and Dacheng Tao. Video frame interpolation without temporal priors. *Advances in Neural Information Processing Systems*, 33, 2020. 5, 7