






# Feedback Recurrent Autoencoder for Video Compression

Adam Goliński<sup>3</sup> , Reza Pourreza<sup>1</sup>, Yang Yang<sup>1</sup> , Guillaume Sautière<sup>2</sup> ,  
and Taco S. Cohen<sup>2</sup>

<sup>1</sup> Qualcomm AI Research, Qualcomm Technologies, Inc., San Diego, USA  
{pourreza,yyang}@qti.qualcomm.com

<sup>2</sup> Qualcomm AI Research, Qualcomm Technologies Netherlands B.V.,  
Amsterdam, Netherlands  
{gsautie,tacos}@qti.qualcomm.com

<sup>3</sup> Department of Engineering Science, University of Oxford, Oxford, England  
adamg@robots.ox.ac.uk

**Abstract.** Recent advances in deep generative modeling have enabled efficient modeling of high dimensional data distributions and opened up a new horizon for solving data compression problems. Specifically, autoencoder based learned image or video compression solutions are emerging as strong competitors to traditional approaches. In this work, We propose a new network architecture, based on common and well studied components, for learned video compression operating in low latency mode. Our method yields competitive MS-SSIM/rate performance on the high-resolution UVG dataset, among both learned video compression approaches and classical video compression methods (H.265 and H.264) in the rate range of interest for streaming applications. Additionally, we provide an analysis of existing approaches through the lens of their underlying probabilistic graphical models. Finally, we point out issues with temporal consistency and color shift observed in empirical evaluation, and suggest directions forward to alleviate those.

## 1 Introduction

With over 60% of internet traffic consisting of video [1], lossy video compression is a critically important problem, promising reductions in bandwidth, storage, and generally increasing the scalability of the internet. Although the relation between probabilistic modelling and compression has been known since Shannon, video codecs in use today are only to a very small extent based on learning and

A. Goliński, R. Pourreza and Y. Yang—Equal Contribution.

Work completed during internship at Qualcomm Technologies Netherlands B.V. Qualcomm AI Research is an initiative of Qualcomm Technologies, Inc.

**Electronic supplementary material** The online version of this chapter ([https://doi.org/10.1007/978-3-030-69538-5\\_36](https://doi.org/10.1007/978-3-030-69538-5_36)) contains supplementary material, which is available to authorized users.

are not end-to-end optimized for rate-distortion performance on a large and representative video dataset.

The last few years have seen a surge in interest in novel codec designs based on deep learning [2–9], which are trained end-to-end to optimize rate-distortion performance on a large video dataset, and a large number of network designs with various novel, often domain-specific components have been proposed. In this paper we show that a relatively simple design based on standard, well understood and highly optimized components such as residual blocks [10], convolutional recurrent networks [11], optical flow warping and a PixelCNN [12] prior yields competitive rate-distortion performance among the learned methods.

We focus on the online compression setting, where video frames can be compressed and transmitted as soon as they are recorded (in contrast to approaches which require buffering several frames before encoding), which is necessary for applications such as video conferencing and cloud gaming. Additionally, in both applications, the ability to finetune the neural codec to its specific content holds promise to further significantly reduce the required bandwidth [6].

There are two key components to our approach beyond the residual block based encoder and decoder architecture. Firstly, to exploit long range temporal correlations, we follow the approach proposed in Feedback Recurrent AutoEncoder (FRAE) [13], which was shown to be effective for speech compression, by adding a convolutional GRU module in the decoder and feeding back the recurrent state to the encoder. Secondly, we apply a motion estimation network at the encoder side and enforce optical flow learning by using an explicit loss term during the initial stage of the training, which leads to better optical flow output at the decoder side and consequently much better rate-distortion performance. The proposed network architecture is compared with existing learned approaches through the lens of their underlying probabilistic models in Sect. 3.

We compare our method with the state-of-the-art traditional codecs and learned approaches on the UVG 1080p [14] and HEVC [15] video datasets by plotting rate-distortion curves. We show that our method performs competitively on the MS-SSIM [16] distortion metric in the low to high rate regime for the high-resolution data, and particularly in the 0.09-0.13 bits per-pixel (bpp) region, which is of practical interest for video streaming [17].

To summarize, our main contributions are as follows:

1. We develop a simple feedback recurrent video compression architecture based on widely used building blocks (Sect. 2).
2. We study the differences and connections of existing learned video compression methods by detailing the underlying sequential latent variable models (Sect. 3).
3. Our solution achieves competitive rate-distortion performance when compared with other learned video compression approaches as well as traditional codecs under equivalent settings (Sect. 4).

## 2 Methodology

### 2.1 Problem Setup

Let us denote the image frames of a video as  $\mathbf{x} = \{\mathbf{x}_i\}_{i \in \mathbb{N}}$ . Compression of the video is done by an autoencoder that maps  $\mathbf{x}$ , through an encoder  $f_{\text{enc}}$ , into compact discrete latent codes  $\mathbf{z} = \{\mathbf{z}_i\}_{i \in \mathbb{N}}$ . The codes are then used by a decoder  $f_{\text{dec}}$  to form reconstructions  $\hat{\mathbf{x}} = \{\hat{\mathbf{x}}_i\}_{i \in \mathbb{N}}$ .

We assume the use of an entropy coder together with a probabilistic model on  $\mathbf{z}$ , denoted as  $\mathbb{P}_{\mathbf{z}}(\cdot)$ , to losslessly compress the discrete latents. The ideal codeword length can then be characterized as  $R(\mathbf{z}) = -\log \mathbb{P}_{\mathbf{z}}(\mathbf{z})$  which we refer to as the rate term<sup>1</sup>.

Given a distortion metric  $D : \mathbf{X} \times \mathbf{X} \rightarrow \mathbb{R}$ , the lossy compression problem can be formulated as the optimization of the following Lagrangian functional

$$\min_{f_{\text{enc}}, f_{\text{dec}}, \mathbb{P}_{\mathbf{z}}} \mathcal{L}_{\text{RD}} \triangleq \min_{f_{\text{enc}}, f_{\text{dec}}, \mathbb{P}_{\mathbf{z}}} \sum_{\mathbf{x}} D(\mathbf{x}, \hat{\mathbf{x}}) + \beta R(\mathbf{z}), \quad (1)$$

where  $\beta$  is the Lagrange multiplier that controls the balance of rate and distortion. It is known that this objective function is equivalent to the evidence lower bound in  $\beta$ -VAE [19] when the encoder distribution is deterministic or has a fixed entropy. Hence  $\mathbb{P}_{\mathbf{z}}$  is often called the *prior* distribution. We refer the reader to [6, 8, 20] for more detailed discussion. Throughout this work we use MS-SSIM [16] measured in RGB space as our distortion metric for both training and evaluation.

### 2.2 Overview of the Proposed Method

In our work, we focus on the problem of *online* compression of video using a *causal* autoencoder, i.e. one that outputs codes and a reconstruction for each frame on the fly without the need to access future context. In classic video codec terms, we are interested in the *low delay P (LDP)* setting where only I-frames<sup>2</sup> (Intra-coded; independent of other frames) and P-frames (Predictive inter-coded; using previously reconstructed past but not future frames) are used. We do not make use of B-frames (Bidirectionally interpolated frames).

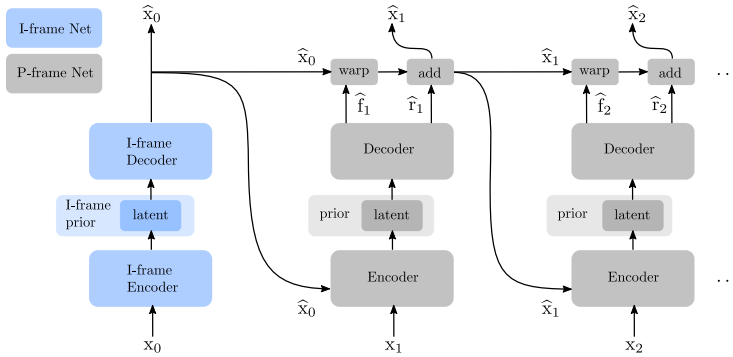
The full video sequence is broken down into *groups of pictures* (GoP)  $\mathbf{x} = \{\mathbf{x}_0, \mathbf{x}_1, \dots, \mathbf{x}_{N-1}\}$  that starts with an I-frame and is followed by  $N-1$  P-frames. We use a separate encoder, decoder and prior for the I-frames and P-frames. The rate term is then decomposed as

$$R(\mathbf{z}) = -\log \mathbb{P}_{\mathbf{z}}^{\text{I}}(\mathbf{z}_0) - \log \mathbb{P}_{\mathbf{z}}^{\text{P}}(\mathbf{z}_1, \dots, \mathbf{z}_{N-1} | \mathbf{z}_0), \quad (2)$$

where a superscript is used to indicate frame type.

<sup>1</sup> For practical entropy coder, there is a constant overhead per block/stream, which is negligible with a large number of bits per stream and thus can be ignored. For example, for adaptive arithmetic coding (AAC), there is up to 2-bit inefficiency [18].

<sup>2</sup> We refer the reader to [21, 22] and Sect. 2 of [4] for a good overview of frame structures in classic codecs.



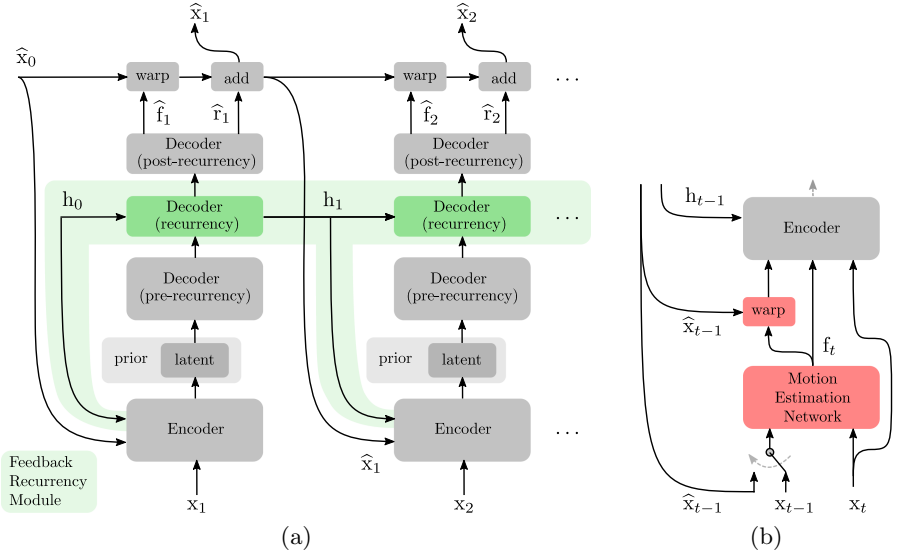
**Fig. 1.** Base architecture of our video compression network. I-frame  $\mathbf{x}_0$  is compressed using a stand-alone image compression autoencoder. The subsequent P-frames are compressed with previous reconstructed frame as input and decoded with optical flow based motion compensation plus residual.

### 2.3 Network Architecture

I-frame compression is equivalent to image compression and there are already many schemes available [20, 23–26]. Here we applied the encoder and decoder design of [25] for I-frame compression. Subsequently, we only focus on the design of the P-frame compression network and discuss how previously transmitted information can be best utilized to reduce the amount of information needed to reconstruct subsequent frames.

#### Baseline Architecture

One straightforward way to utilize history information is for the autoencoder to transmit only a motion vector (e.g., block based or a dense optical field) and a residual while using the previously decoded frame(s) as a reference. At the decoder side, decoded motion vector (in our case optical flow) is used to warp previously decoded frame(s) using bilinear interpolation [27] (referred to later as **warp** function), which is then refined by the decoded residual. This framework serves as the basic building block for many conventional codecs such as AVC and HEVC. One instantiation of such framework built with autoencoders is illustrated in Fig. 1. Here an autoencoder based image compression network, termed I-frame network, is used to compress and reconstruct the I-frame  $\mathbf{x}_0$  independent of any other frames. Subsequent P-frames  $\mathbf{x}_t$  are processed with a separate autoencoder, termed P-frame network, that takes both the current input frame and the previous reconstructed frame as encoder inputs and produces the optical flow tensor  $\hat{\mathbf{f}}_t$  and the residual  $\hat{\mathbf{r}}_t$  as decoder output. The architecture of our P-frame network’s encoder and decoder is the same as the I-frame. Two separate prior models are used for the entropy coding of the discrete latents in I-frame and P-frame networks. The frame is eventually reconstructed as  $\hat{\mathbf{x}}_t \triangleq \text{warp}(\hat{\mathbf{f}}_t, \hat{\mathbf{x}}_{t-1}) + \hat{\mathbf{r}}_t$ .



**Fig. 2.** (a) Video compression network with feedback recurrent module. Detailed network architecture for the encoder and decoder is described in Appendix A. (b) Encoder equipped with MENet, an explicit optical flow estimation module. The switch between  $\hat{x}_{t-1}$  and  $x_{t-1}$  is described at the end of Sect. 2.

For the P-frame network, the latent at each time  $t$  contains information about  $(\hat{\mathbf{f}}_t, \hat{\mathbf{r}}_t)$ . One source of inefficiency, then, is that the current time step’s  $(\hat{\mathbf{f}}_t, \hat{\mathbf{r}}_t)$  may still exhibit temporal correlation with the past values  $(\hat{\mathbf{f}}_{<t}, \hat{\mathbf{r}}_{<t})$ , which is not exploited. This issue remains even if we consider multiple frames as references.

To explicitly equip the network with the capability to utilize the redundancy w.r.t.  $(\hat{\mathbf{f}}_{t-1}, \hat{\mathbf{r}}_{t-1})$ , we would need to expose  $(\hat{\mathbf{f}}_{t-1}, \hat{\mathbf{r}}_{t-1})$  as another input, besides  $x_{t-1}$ , to both the encoder and the decoder for the operation at time step  $t$ . In this case the latents would only need to carry information regarding the incremental difference between the flow field and residual between two consecutive steps, i.e.  $(\hat{\mathbf{f}}_{t-1}, \hat{\mathbf{r}}_{t-1})$  and  $(\hat{\mathbf{f}}_t, \hat{\mathbf{r}}_t)$ , which would lead to a higher degree of compression. We could follow the same principle to utilize even higher order redundancies but it inevitably leads to a more complicated architecture design.

**Feedback Recurrent Module**

As we are trying to utilize higher order redundancy, we need to provide both the encoder and the decoder with a more suitable decoder history context. This observation motivates the use of a *recurrent* neural network that is meant to accumulate and summarize relevant information received previously by the decoder, and a decoder-to-encoder *feedback* connection that makes the recurrent state available at the encoder [13] – see Fig. 2(a). We refer to the added component as the *feedback recurrent module*.

This module can be viewed as a *non-linear predictive coding* scheme, where the decoder predicts the next frame based on the current latent as well as a summary of past latents. Because the recurrent state is available to both encoder and decoder, the encoder is aware of the state of knowledge of the decoder, enabling it to send only complementary information.

The concept of feedback recurrent connection is present in several existing works: In [13], the authors refer to an autoencoder with the feedback recurrent connection as FRAE and demonstrate its effectiveness in speech compression in comparison to different recurrent architectures. In [28, 29], the focus is on progressive coding of images, and the decoder to encoder feedback is adopted for an iterative refinement of the image canvas. In the domain of video compression, [5] proposes the concept of *state-propagation*, where the network maintains a state tensor  $S_t$  that is available at both the encoder and the decoder and updated by summing it with the decoder output in each time step (Fig. 5 in [5]). In our network architecture, we propose the use of generic convolutional recurrent modules such as Conv-GRU [11] for the modeling of history information that is relevant for the prediction of the next frame. In Appendix E, we show the necessity of such feedback connection from an information theoretic perspective.

### Latent Quantization and Prior Model

Given that the encoder needs to output discrete values, the quantization method applied needs to allow gradient based optimization. Two popular approaches are: (1) Define a learnable codebook, and use nearest neighbor quantization in the forward pass and a differentiable softmax in the backward pass [6, 25], or (2) Add uniform noise to the continuous valued encoder output during training and use hard quantization at evaluation [8, 23, 24]. For the second approach, the prior distribution is often characterized by a learnable monotonic CDF function  $f$ , where the probability mass of a single latent value  $z$  is evaluated as  $f(z + 1/2) - f(z - 1/2)$ . In this work we adopt the first approach.

As illustrated in Fig. 2(a), a time-independent prior model is used. In other words, there is no conditioning of the prior model on the latents from previous time steps, and  $\mathbb{P}_{\mathbf{Z}}^P(\mathbf{z}_1, \dots, \mathbf{z}_{N-1} | \mathbf{z}_0)$  in Eq. (2) is factorized as  $\prod_{i=1}^{N-1} \mathbb{P}_{\mathbf{Z}}^P(\mathbf{z}_i)$ . In Sect. 3 we show that such factorization does not limit the capability of our model to capture any empirical data distribution.

A gated PixelCNN [12] is used to model  $\mathbb{P}_{\mathbf{Z}}^P$ , with the detailed structure of the model built based on the description in Fig. 10 of [6]. The prior model for the I-frame network  $\mathbb{P}_{\mathbf{Z}}^I$  is a separate network with the same architecture.

### Explicit Optical Flow Estimation Module

In the architecture shown in Fig. 2(a), the optical flow estimate tensor  $\hat{\mathbf{f}}_t$  is produced explicitly at the end of the decoder. When trained with the loss function  $\mathcal{L}_{RD}$  in Eq. (1), the learning of the optical flow estimation is only incentivized implicitly via how much that mechanism helps with the rate-effective reconstruction of the original frames. In this setting we empirically found the decoder was almost solely relying on the residuals  $\hat{\mathbf{r}}_t$  to form the frame reconstructions.

This observation is consistent with the observations of other works [2, 5]. This problem is usually addressed by using pre-training or more explicit supervision for the optical flow estimation task. We encourage reliance on both optical flow and residuals and facilitate optical flow estimation by (i) equipping the encoder with a U-Net [30] sub-network called *Motion Estimation Network* (MENet), and (ii) introducing additional loss terms to explicitly encourage optical flow estimation.

MENet estimates the optical flow  $\mathbf{f}_t$  between the current input frame  $\mathbf{x}_t$  and the previously reconstructed frame  $\widehat{\mathbf{x}}_{t-1}$ . Without MENet, the encoder is provided with  $\mathbf{x}_t$  and  $\widehat{\mathbf{x}}_{t-1}$ , and is supposed to estimate the optical flow and the residuals and encode them, all in a single network. However, when attached to the encoder, MENet provides the encoder directly with the estimated flow  $\mathbf{f}_t$  and the previous reconstruction warped by the estimated flow  $\text{warp}(\widehat{\mathbf{x}}_{t-1}, \mathbf{f}_t)$ . In this scenario, all the encoder capacity is dedicated to the encoding task. A schematic view of this explicit architecture is shown in Fig. 2(b) and the implementation details are available in Appendix A.

When MENet is integrated with the architecture in Fig. 2(a), optical flow is originally estimated using MENet denoted as  $\mathbf{f}_t$  and later reconstructed in the decoder denoted as  $\widehat{\mathbf{f}}_t$ . In order to alleviate the problem with optical flow learning using rate-distortion loss only, we incentivize the learning of  $\mathbf{f}_t$  and  $\widehat{\mathbf{f}}_t$  via two additional dedicated loss terms,

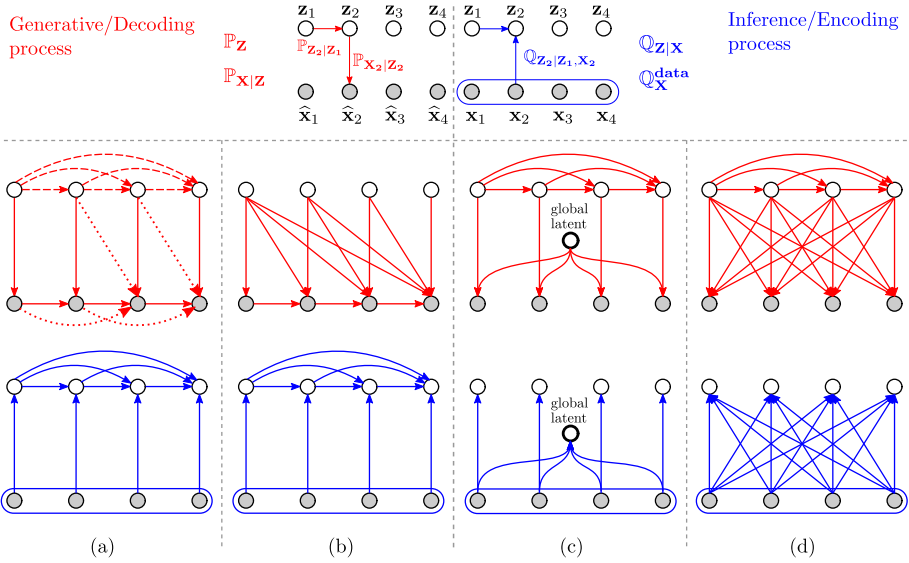
$$\mathcal{L}_{\text{fe}} = D(\text{warp}(\widehat{\mathbf{x}}_{t-1}, \mathbf{f}_t), \mathbf{x}_t), \mathcal{L}_{\text{fd}} = D(\text{warp}(\widehat{\mathbf{x}}_{t-1}, \widehat{\mathbf{f}}_t), \mathbf{x}_t).$$

Hence the total loss we start the training with is  $\mathcal{L} = \mathcal{L}_{\text{RD}} + \mathcal{L}_{\text{fe}} + \mathcal{L}_{\text{fd}}$  where  $\mathcal{L}_{\text{RD}}$  is the loss function as per Eq. (1). We found that it is sufficient if we apply the losses  $\mathcal{L}_{\text{fe}}$  and  $\mathcal{L}_{\text{fd}}$  only at the beginning of training, for the first few thousand iterations, and then we revert to using just  $\mathcal{L} = \mathcal{L}_{\text{RD}}$  and let the estimated flow be adapted to the main task, which has been shown to improve the results across variety of tasks utilizing optical flow estimation [31].

The loss terms  $\mathcal{L}_{\text{fe}}$  and  $\mathcal{L}_{\text{fd}}$  are defined as the distortion between  $\mathbf{x}_t$  and the warped version of  $\widehat{\mathbf{x}}_{t-1}$ . However, early in the training,  $\widehat{\mathbf{x}}_{t-1}$  is inaccurate and as a result, such distortion is not a good choice of a learning signal for the MENet or the autoencoder. To alleviate this problem, early in the training we use  $\mathbf{x}_{t-1}$  instead of  $\widehat{\mathbf{x}}_{t-1}$  in both  $\mathcal{L}_{\text{fe}}$  and  $\mathcal{L}_{\text{fd}}$ . This transition is depicted in Fig. 2(b) with a switch. It is worth to mention that the tensor fed into the encoder is always a warped previous reconstruction  $\text{warp}(\widehat{\mathbf{x}}_{t-1}, \mathbf{f}_t)$ , never a warped previous ground truth frame  $\text{warp}(\mathbf{x}_{t-1}, \mathbf{f}_t)$ .

### 3 Graphical Model Analysis

Recent success of autoencoder based learned image compression [9, 20, 23–26] (see [32] for an overview) has demonstrated neural networks’ capability in modeling spatial correlations from data and the effectiveness of end-to-end joint rate-distortion training. It has motivated the use of autoencoder based solution to



**Fig. 3.** Different combinations of sequential latent variable models and inference models proposed in the literature. (a) with the solid lines only is used in Lu *et al.* [2]. (a) with both solid and dashed lines is used in Liu *et al.* [7]. (a) with the solid and dotted lines is used in M-LVC (presented is a model for a buffer size of 2 past frames) [9]. (b) is used in both Rippel *et al.* [5] and our approach. (c) is used in Han *et al.* [8]. (d) is used in Habibiyan *et al.* [6]. The blue box around nodes  $\mathbf{x}_{1:4}$  means that they all come from a single joint distribution  $\mathbb{Q}_{\mathbf{x}}^{\text{data}}(\mathbf{x}_{1:4})$  and that we make no assumptions on the conditional structure of that distribution. See Appendix F for detailed reasoning about these graphical models. (Color figure online)

further capture the temporal correlation in the application of video compression and there has since been many designs for learned video compression algorithms [2, 5–8, 13]. These designs differ in many detailed aspects: the technique used for latent quantization; the specific instantiation of encoder and decoder architecture; the use of atypical, often domain-specific operations beyond convolution such as Generalized Divisive Normalization (GDN) [23, 33] or Non-Local Attention Module (NLAM) [34]. Here we leave out the comparison of those aspects and focus on one key design element: the graphical model structure of the underlying sequential latent variable model [35].

As we have briefly discussed in Sect. 2.1, the rate-distortion training objective has strong connections to amortized variational inference. In the special case of  $\beta = 1$ , the optimization problem is equivalent to the minimization of  $D_{\text{KL}}(\mathbb{Q}_{\mathbf{x}}^{\text{data}}\mathbb{Q}_{\mathbf{z}|\mathbf{x}}||\mathbb{P}_{\mathbf{z}}\mathbb{P}_{\mathbf{x}|\mathbf{z}})$  [36] where  $\mathbb{P}_{\mathbf{z}}\mathbb{P}_{\mathbf{x}|\mathbf{z}}$  describes a sequential generative process and  $\mathbb{Q}_{\mathbf{x}}^{\text{data}}\mathbb{Q}_{\mathbf{z}|\mathbf{x}}$  can be viewed as a sequential inference process.  $\mathbb{P}_{\mathbf{x}|\mathbf{z}}$  denotes the decoder distribution induced by our distortion metric (see Sect. 4.2 of [6]),  $\mathbb{Q}_{\mathbf{x}}^{\text{data}}$  denotes the empirical distribution of the training data, and  $\mathbb{Q}_{\mathbf{z}|\mathbf{x}}$  denotes the encoder distribution induced on the latents, in our case a Dirac- $\delta$



distribution because of the deterministic mapping from the encoder inputs to the value of the latents.

To favor the minimization of the KL divergence, we want to design  $\mathbb{P}_Z$ ,  $\mathbb{P}_{X|Z}$ ,  $\mathbb{Q}_{Z|X}$  to be flexible enough so that (i) the marginalized data distribution  $\mathbb{P}_X \triangleq \sum_Z \mathbb{P}_Z \mathbb{P}_{X|Z}$  is able to capture complex temporal dependencies in the data, and (ii) the inference process  $\mathbb{Q}_{Z|X}$  encodes all the dependencies embedded in  $\mathbb{P}_Z \mathbb{P}_{X|Z}$  [37]. In Fig. 3 we highlight four combinations of sequential latent variable models and the inference models proposed in literature.

Figure 3(a) without the dashed line describes the scheme proposed in Lu *et al.* [2] (and the low latency mode of HEVC/AVC) where prior model is fully factorized over time and each decoded frame is a function of the previous decoded frame  $\hat{\mathbf{x}}_{t-1}$  and the current latent code  $\mathbf{z}_t$ . There are two major limitations of this approach: firstly, the marginalized sequential distribution  $\mathbb{P}_X$  is confined to follow Markov property  $\mathbb{P}_X(\mathbf{x}_{1:T}) = \mathbb{P}_{X_1}(\mathbf{x}_1) \prod_{t=2}^T \mathbb{P}_{X_t|X_{t-1}}(\mathbf{x}_t|\mathbf{x}_{t-1})$ ; secondly, it assumes that the differences in subsequent frames (e.g., optical flow and residual) are independent across time steps. To overcome these two limitations, Liu *et al.* [7] propose an auto-regressive prior model through the use of ConvLSTM over time, which corresponds to Fig. 3(a) including dashed lines. In this case, the marginal  $\mathbb{P}_X(\mathbf{x}_{1:T})$  is fully flexible in the sense that it does not make any conditional independence assumptions between  $\mathbf{x}_{1:T}$ , see Appendix F for details. With a similar goal in mind, in a work concurrent to ours, Lin *et al.* [9] propose M-LVC which utilizes temporal correlations between consecutive frames' optical flows by explicitly predicting the flow of the currently coded frame using previous flows. In contrast our approach aims to extract the same optical flow (as well as higher order) redundancy implicitly via a learned recurrent state.

Figure 3(b) describes another way to construct a fully flexible marginalized sequential data distribution, by having the decoded frame depend on all previously transmitted latent codes. Both Rippel *et al.* [5] and our approach fall under this category by introducing a recurrent state in the decoder which summarizes information from the previously transmitted latent codes and processed features. Rippel *et al.* [5] use a multi-scale state with a complex state propagation mechanism, whereas we use a convolutional recurrency module (ConvGRU) [11]. In this case,  $\mathbb{P}_X(\mathbf{x}_{1:T})$  is also fully flexible, details in Appendix F.

The latent variable models in Fig. 3(c) and (d) break the causality constraint, i.e. the encoding of a GoP can only start when all of its frames are available. In (c), a global latent is used to capture time-invariant features, which is adopted in Han *et al.* [8]. In (d), the graphical model is fully connected between each frame and latent across different time steps, which described the scheme applied in Habibian *et al.* [6].

One advantage of our approach (as well as Rippel *et al.* [5]), which corresponds to Fig. 3(b), is that the prior model can be fully factorized across time,  $\mathbb{P}_Z(\mathbf{z}_{1:T}) = \prod_{t=1}^T \mathbb{P}_Z(\mathbf{z}_t)$ , without compromising the flexibility of the marginal distribution  $\mathbb{P}_X(\mathbf{x}_{1:T})$ . Factorized prior allows *parallel* entropy decoding of the latent codes across time steps (but potentially still *sequential* across dimensions of  $\mathbf{z}_t$  within each time step). On the inference side, in Appendix E we show that

any connection from  $\mathbf{x}_{<t}$  to  $\mathbf{z}_t$ , which could be modeled by adding a recurrent component on the encoder side, is not necessary.

## 4 Experiments

### 4.1 Training Setup

Here we provide the key information about the training setup, while the comprehensive description of the details of the architecture, training and datasets used (and how they were processed) are in Appendix A. Our training dataset was based on Kinetics400 [38], and we evaluate on UVG 1080p [14] and HEVC Classes BCDE [15]. The distortion metric used was 1–MS-SSIM [16]. The GoP size of 8 was used during training.

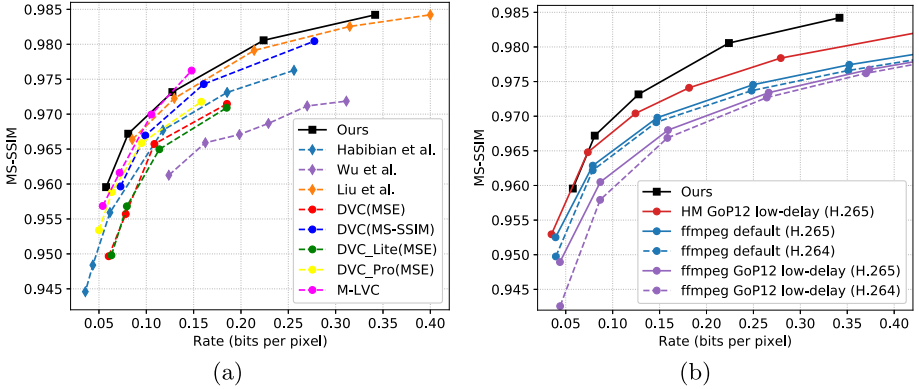
### 4.2 Comparison with Other Methods

**Comparison with Learning-Based Methods.** In Fig. 4(a), we compare the performance of our solution with several learned approaches on UVG 1080p dataset in terms of MS-SSIM versus bitrate where MS-SSIM was first averaged per video and then averaged across different videos. The figures for the HEVC Classes BCDE datasets and comparison using PSNR metric are in Appendix H. When comparing using MS-SSIM metric, on UVG and HEVC-E, our method outperforms all the compared ones in MS-SSIM across the range of bitrates between around 0.05 bpp to around 0.13 bpp. For 1080p resolution this is the range of bitrates of interest for practitioners, e.g., Netflix uses the range of about 0.09–0.13 bpp for their 1080p resolution video streaming [17]. On HEVC-BD, our method is outperformed by M-LVC and DVC, whereas on HEVC-C our method outperforms other methods for bitrate below  $<0.13$  bpp. When comparing using PSNR metric, our method is outperformed by most other methods. This effect is expected since our method was not trained using MSE distortion metric, however we note that when compared on PSNR metric the performance drop of our method is larger than for the DVC(MSSSIM) model.

DVC [2,3], Liu *et al.* [7] and M-LVC [9] are all causal solutions, and they are trained and evaluated with GoP size of 12(UVG)/10(HEVC), 12, 16, respectively. Habibian *et al.* and Wu *et al.* are non-causal solutions and their results are reported based on GoP sizes of 8 and 12, respectively. Our results are evaluated at GoP of 8, same as in training.

We omit comparisons to Han *et al.* [8] because they provide results only for low resolution videos, and to Rippel *et al.* [5] because we lack the licensing rights to use all the Xiph dataset videos [39] they evaluate their method on.

**Comparison with Traditional Video Codecs.** We compared our method with the most popular standard codecs i.e. H.265 [22] and H.264 [40] on UVG 1080p dataset. The results are generated with three different sets of settings (more details are provided in Appendix C):



**Fig. 4.** Results on UVG dataset. (a) Comparison to the state-of-the-art learned methods. The DVC [3] results are presented for both models trained with MSE and MSSSIM distortion metrics. (b) Comparison with classic codecs.

- `ffmpeg` [41] implementation of H.264 and H.265 in low latency mode with GoP size of 12.
- `ffmpeg` [41] implementation of H.264 and H.265 in default mode.
- HM [42] implementation of H.265 in low latency mode with GoP size of 12.

The low latency mode was enforced to H.264 and H.265 to make the problem settings the same as our causal model. GoP size 12 on the other hand, although different from our GoP size 8, was consistent with the settings reported in other papers and provided H.264 and H.265 an advantage as they perform better with larger GoP sizes.

The comparisons are shown in Fig. 4(b) in terms of MS-SSIM versus bitrate where MS-SSIM was calculated in RGB domain. As can be seen from this figure, our model outperforms the HM implementation of H.265 and the `ffmpeg` implementation of H.265 and H.264 in both low latency and default settings, at bitrates above 0.09 bpp which, again, is the bpp range of interest for 1080p resolution videos. We note there are also computational aspects one should consider when comparing to these traditional codecs, see a discussion in Appendix I.

**Qualitative Comparison.** In Fig. 6 we compare the visual quality of our lowest rate model with `ffmpeg` implementation of H.265 at in low latency mode a similar rate. We can see that our result is free from the blocking artifact usually present in H.265 at low bitrate – see around the edge of fingers – and preserves more detailed texture – see structures of hand veins and strips on the coat. In Appendix D, we provide more examples with detailed error and bitrate maps.

As noted in Sect. 2.3, the dedicated optical flow enforcement loss terms were removed at a certain point and the training continued using  $\mathcal{L}_{RD}$  only. As a result, our network learned a form of optical flow that contributed maximally to  $\mathcal{L}_{RD}$  and did not necessarily follow the ground truth optical flow (if such

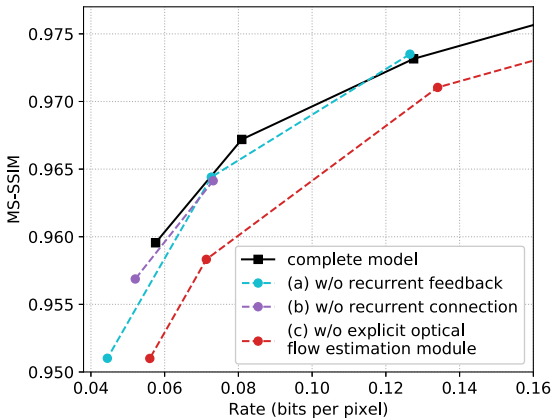
were available). Figure 7 compares an instance of the optical flow learned in our network with the corresponding optical flow generated using a pre-trained FlowNet2 network [43]. The optical flow reconstructed by the decoder  $\hat{\mathbf{f}}$  has larger proportion of values set to zero than the flow estimated by FlowNet2 – this is consistent with the observations by Lu *et al.* [2] in their Fig. 7 where they argue that it allows the flow to be more compressible.

### 4.3 Ablation Study

To understand the effectiveness of different components in our design, in Fig. 5 we compare the performance after removing (a) decoder to encoder recurrent feedback, or (b) the feedback recurrent module in Fig. 2(a), or (c) the explicit optical flow estimation module. We focus the comparison on low rate regime where the difference is more noticeable.

Empirically we find the explicit optical flow estimation component to be quite essential – without it the optical flow output from the decoder has very small magnitude and is barely useful, resulting in consistent loss of at least 0.002 in MS-SSIM score for rate below 0.16 bpp. In comparison, the loss in performance after removing either the decoder to encoder feedback or the recurrent connection all together is minor and only show up at very low rate region below 0.1 bpp.

Similar comparisons have been done in existing literature. Rippel *et al.* [5] report large drop in performance when removing the learned state in their model (Fig. 9 of [5]; about 0.005 drop in MS-SSIM at bpp of 0.05), while Liu *et al.* [7], which utilizes recurrent prior to exploit longer range temporal redundancies, reports relatively smaller degradation after removing their recurrent module (about 0.002 drop in MS-SSIM in Fig. 9 of [7]).



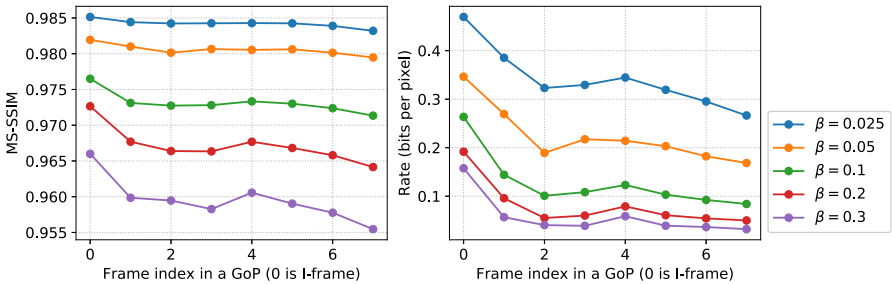
**Fig. 5.** Ablation study on UVG dataset. All models were evaluated at GoP of 8. Model (a) is obtained by removing the decoder to encoder feedback connection in Fig. 2(a). Model (b) removes the the feedback recurrent module. Model (c) removes the explicit optical flow estimation module in Fig. 2(b).



**Fig. 6.** An illustration of qualitative characteristics of our architecture versus H.265 (*ffmpeg*) at a comparable bitrate. (a) and (b) are the original frame and its closeup, (c) and (d) are the closeup on the reconstructed frames from our method and H.265. To make the comparison fair, we used HEVC with fixed GoP setting (*min-keyint=8:scenecut=0*) at a similar rate, so both methods are at equal bpp and show the 4th P-frame of a GoP of 8. Frame 229 of Tango video from Netflix Tango in Netflix El Fuente; see Appendix B for license information.



**Fig. 7.** An example of the estimated optical flow, left to right: two consecutive frames used for estimation, FlowNet2 results, our decoder output  $\hat{\mathbf{f}}$ . Note that the flow produced by our model is decoded from a compressed latent, and is trained to maximize total compression performance rather than warping accuracy. See Appendix B for license information.



**Fig. 8.** Average MS-SSIM and rate as a function of frame-index in a GoP of 8.

We suspect that the lack of gain we observe from the feedback recurrent module is due to both insufficient recurrent module capacity and receptive field, and plan to investigate further as one of our future studies.

#### 4.4 Empirical Observations

**Temporal Consistency.** We found, upon visual inspection, that for our low bitrate models, the decoded video displays a minor yet noticeable flickering artifact occurring at the GoP boundary, which we attribute to two factors.

Firstly, as shown in Fig. 8, there is a gradual degradation of P-frame quality as it moves further away from the I-frame. It is also apparent from Fig. 8 that the rate allocation tends to decline as P-frame network unrolls, which compounds the degradation of P-frame quality. This shows that the end-to-end training of frame-averaged rate-distortion loss does not favor temporal consistency of frame quality, which makes it difficult to apply the model with larger GoP sizes and motivates the use of temporal-consistency related losses [44, 45]. For a figure investigating generalization of the model trained with GoP of 8 to evaluation with a GoP of 10, see Appendix J.

Second part of the problem is due to the nature of the MS-SSIM metric and hence is not reflected in Fig. 8. This aspect is described in the following section.

**Color Shift.** Another empirical observation is that the MS-SSIM metric seems to be partially invariant to uniform color shift in flat regions and due to that the low bitrate models we have trained are prone to exhibit slight color shift, similar to what has been reported in [46]. This color change is sometimes noticeable during I-frame to P-frame transitioning and contributes to the flickering artifact. One item for future study is to find suitable metrics or combination of metrics that captures both texture similarity and color fidelity. Details in Appendix G.

## 5 Conclusion

In this paper, we proposed a new autoencoder based learned lossy video compression solution featuring a feedback recurrent module and an explicit optical flow estimation module. It achieves competitive performance among learned video compression methods and delivers comparable or better rate-distortion results when compared with classic codecs in low latency mode. In future work, we plan to improve its temporal consistency, solve the color fidelity issue, and reduce computational complexity, paving the way to a practical learned video codec.

## References

1. Sandvine: 2019 Global Internet Phenomena Report. <https://www.ncta.com/whats-new/report-where-does-the-majority-of-internet-traffic-come> (2019) Accessed 28 Feb 2020

2. Lu, G., Ouyang, W., Xu, D., Zhang, X., Cai, C., Gao, Z.: DVC: an end-to-end deep video compression framework. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (2019)
3. Lu, G., Zhang, X., Ouyang, W., Chen, L., Gao, Z., Xu, D.: An end-to-end learning framework for video compression. *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2020)
4. Wu, C.Y., Singhal, N., Krähenbühl, P.: Video compression through image interpolation. In: Proceedings of the European Conference on Computer Vision (2018)
5. Rippel, O., Nair, S., Lew, C., Branson, S., Anderson, A.G., Bourdev, L.: Learned video compression. In: Proceedings of the International Conference on Computer Vision (2019)
6. Habibiyan, A., van Rozendaal, T., Tomczak, J.M., Cohen, T.S.: Video compression with rate-distortion autoencoders. In: Proceedings of the International Conference on Computer Vision (2019)
7. Liu, H., Shen, H., Huang, L., Lu, M., Chen, T., Ma, Z.: Learned video compression via joint spatial-temporal correlation exploration. In: Proceedings of the national conference on Artificial Intelligence (2020)
8. Han, J., Lombardo, S., Schroers, C., Mandt, S.: Deep probabilistic video compression. In: Advances in Neural Information Processing Systems (2019)
9. Lin, J., Liu, D., Li, H., Wu, F.: M-lvc: multiple frames prediction for learned video compression. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (2020)
10. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (2016)
11. Shi, X., Chen, Z., Wang, H., Yeung, D., Wong, W., Woo, W.: Convolutional LSTM network: a machine learning approach for precipitation nowcasting. In: Advances in Neural Information Processing Systems (2015)
12. van den Oord, A., Kalchbrenner, N., Espeholt, L., Kavukcuoglu, K., Vinyals, O., Graves, A.: Conditional image generation with PixelCNN decoders. In: Advances in Neural Information Processing Systems (2016)
13. Yang, Y., Sautié, G., Ryu, J.J., Cohen, T.S.: Feedback Recurrent AutoEncoder. In: IEEE International Conference on Acoustics, Speech and Signal Processing (2019)
14. Mercat, A., Viitanen, M., Vanne, J.: Uvg dataset: 50/120fps 4k sequences for video codec analysis and development. In: Proceedings of the 11th ACM Multimedia Systems Conference, MMSys 2020, New York, Association for Computing Machinery, pp. 297–302 (2020)
15. Bossen, F.: Common test conditions and software reference configurations. *JCTVC-F900* (2011)
16. Wang, Z., Bovik, A.C., Sheikh, H.R., Simoncelli, E.P., et al.: Image quality assessment: from error visibility to structural similarity. *IEEE Trans. Image Process.* **13**, 600–612 (2004)
17. Summerson, C.: How much data does Netflix use? <https://www.howtogeek.com/338983/how-much-data-does-netflix-use/> (2018) Accessed 28 Feb 2020
18. Pearlman, W.A., Said, A.: Digital Signal Compression: Principles and Practice. Cambridge University Press, Cambridge (2011)
19. Higgins, I., et al.: beta-VAE: learning basic visual concepts with a constrained variational framework. In: International Conference on Learning Representations (2017)

20. Theis, L., Shi, W., Cunningham, A., Huszár, F.: Lossy image compression with compressive autoencoders. In: International Conference on Learning Representations (2017)
21. Moreira, L.: Digital video introduction. <https://github.com/leandromoreira/digital-video-introduction/blob/master/README.md#frame-types> (2017) Accessed 02 Mar 2020
22. Sullivan, G.J., Ohm, J.R., Han, W.J., Wiegand, T.: Overview of the high efficiency video coding (HEVC) standard. *IEEE Trans. Circuits Syst. Video Technol.* **22**, 1649–1668 (2012)
23. Ballé, J., Laparra, V., Simoncelli, E.P.: End-to-end optimized image compression. In: International Conference on Learning Representations (2017)
24. Ballé, J., Minnen, D., Singh, S., Hwang, S.J., Johnston, N.: Variational image compression with a scale hyperprior. In: International Conference on Learning Representations (2018)
25. Mentzer, F., Agustsson, E., Tschannen, M., Timofte, R., Van Gool, L.: Conditional probability models for deep image compression. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (2018)
26. Rippel, O., Bourdev, L.: Real-time adaptive image compression. In: Proceedings of the International Conference on Machine Learning (2017)
27. Jaderberg, M., Simonyan, K., Zisserman, A., Kavukcuoglu, K.: Spatial Transformer Networks. In: Advances in Neural Information Processing Systems (2015)
28. Gregor, K., Danihelka, I., Graves, A., Rezende, D.J., Wierstra, D.: DRAW: a recurrent neural network for image generation. In: Proceedings of the International Conference on Machine Learning (2015)
29. Gregor, K., Besse, F., Rezende, D.J., Danihelka, I., Wierstra, D.: Towards conceptual compression. In: Advances in Neural Information Processing Systems (2016)
30. Ronneberger, O., Fischer, P., Brox, T.: U-Net: convolutional networks for biomedical image segmentation. In: Medical Image Computing and Computer-Assisted Intervention (2015)
31. Xue, T., Chen, B., Wu, J., Wei, D., Freeman, W.T.: Video enhancement with task-oriented flow. *Int. J. Comput. Vis.* **127**, 1106–1125 (2019)
32. Hu, Y., Yang, W., Ma, Z., Liu, J.: Learning end-to-end lossy image compression: a benchmark. [arXiv:2002.03711](https://arxiv.org/abs/2002.03711) (2020)
33. Ballé, J., Laparra, V., Simoncelli, E.P.: Density modeling of images using a generalized normalization transformation. In: International Conference on Learning Representations (2016)
34. Liu, H., et al.: Non-local Attention Optimized Deep Image Compression. [arXiv:1904.09757](https://arxiv.org/abs/1904.09757) (2019)
35. Koller, D., Friedman, N.: Probabilistic Graphical Models: Principles and Techniques. MIT Press, Cambridge (2009)
36. Alemi, A.A., Poole, B., Fischer, I., Dillon, J.V., Saurous, R.A., Murphy, K.: Fixing a Broken ELBO. In: Proceedings of the International Conference on Machine Learning (2018)
37. Webb, S., et al.: Faithful Inversion of Generative Models for Effective Amortized Inference. In: Advances in Neural Information Processing Systems (2018)
38. Kay, W., et al.: The Kinetics Human Action Video Dataset. [arxiv:1705.06950](https://arxiv.org/abs/1705.06950) (2017)
39. Xiph.org: Xiph.org video test media [derf's collection]. <https://media.xiph.org/video/derf/> (2004) Accessed: 21 Feb 2020



40. Wiegand, T., Sullivan, G.J., Bjontegaard, G., Luthra, A.: Overview of the H.264/AVC video coding standard. *IEEE Trans. Circuits Syst. Video Technol.* **13**, 560–576 (2003)
41. Tomar, S.: Converting video formats with ffmpeg. *Linux J.* **2006**, 10 (2006)
42. HM developers: High Efficiency Video Coding (HEVC). <https://hevc.hhi.fraunhofer.de/> (2012) Accessed 21 Feb 2020
43. Ilg, E., Mayer, N., Saikia, T., Keuper, M., Dosovitskiy, A., Brox, T.: FlowNet 2.0: evolution of optical flow estimation with deep networks. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2017)
44. Gao, C., Gu, D., Zhang, F., Yu, Y.: ReCoNet: real-time coherent video style transfer network. In: *Proceedings of the Asian Conference on Computer Vision* (2018)
45. Lai, W.S., Huang, J.B., Wang, O., Shechtman, E., Yumer, E., Yang, M.H.: Learning blind video temporal consistency. In: *Proceedings of the European Conference on Computer Vision* (2018)
46. Zhao, H., Gallo, O., Frosio, I., Kautz, J.: Loss functions for image restoration with neural networks. In: *IEEE Transactions on Computational Imaging* (2017)