Scale-space flow for end-to-end optimized video compression

Eirikur Agustsson, David Minnen, Nick Johnston, Johannes Ballé, Sung Jin Hwang, George Toderici
Google Research, Perception Team
{eirikur, dminnen, nickj, jballe, sjhwang, gtoderici}@google.com

Abstract

Despite considerable progress on end-to-end optimized deep networks for image compression, video coding remains a challenging task. Recently proposed methods for learned video compression use optical flow and bilinear warping for motion compensation and show competitive rate–distortion performance relative to hand-engineered codecs like H.264 and HEVC. However, these learning-based methods rely on complex architectures and training schemes including the use of pre-trained optical flow networks, sequential training of sub-networks, adaptive rate control, and buffering intermediate reconstructions to disk during training. In this paper, we show that a generalized warping operator that better handles common failure cases, e.g. disocclusions and fast motion, can provide competitive compression results with a greatly simplified model and training procedure. Specifically, we propose scale-space flow, an intuitive generalization of optical flow that adds a scale parameter to allow the network to better model uncertainty. Our experiments show that a low-latency video compression model (no B-frames) using scale-space flow for motion compensation can outperform analogous state-of-the art learned video compression models while being trained using a much simpler procedure and without any pre-trained optical flow networks.

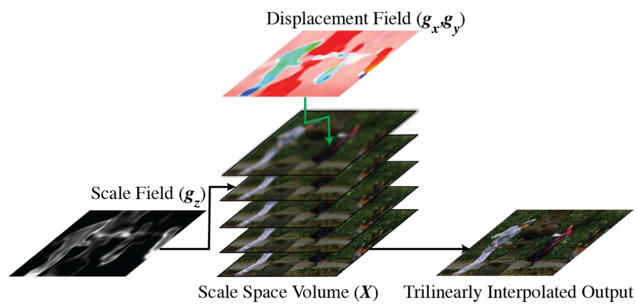

Figure 1. Our proposed scale-space warping module. From the source image $\mathbf{x}$, we construct a fixed-resolution scale-space volume $\mathbf{X}$. In contrast to bilinear warping, where the warped output is sampled directly from the 2-D source image using a 2-channel displacement field $(\mathbf{f}_x, \mathbf{f}_y)$, we trilinearly sample from the 3-D scale-space volume using a 3-channel displacements+scale field $(\mathbf{g}_x, \mathbf{g}_y, \mathbf{g}_z)$. The scale value gives a continuous, differentiable knob that can adaptively blur the source image when warping if the warp is not a good prediction of the target image.

1. Introduction

Recently, there has been significant progress in the area of end-to-end optimized image compression, which went from barely matching JPEG [33] to methods such as [8, 26, 5] that can outperform the best hand-engineered codecs when evaluated in terms of multi-scale structural similarity (MS-SSIM) [36], PSNR, and subjective quality assessments from user studies. While this is very encouraging, over 60% of downstream internet traffic currently consists of streaming video data [1], which means that in order to maximize impact on bandwidth reduction, researchers should focus on video compression.

Since the area of neural video compression is in early stages, it is not yet clear which network architectures are most effective for different application scenarios. We can roughly categorize the existing research methods into the following three categories:

1) *3D autoencoders* are a natural extension of the work done for learned image compression, but [27] demonstrated that representing video using spatiotemporal transformations alone does not lead to better performance compared to standard methods. However, when combined with temporally conditioned entropy models [19], such methods can perform on par with standard methods in terms of MS-SSIM.

2) *Frame interpolation methods* use neural networks to temporally interpolate between frames in a video and then encode the residuals [38, 17]. This approach is commonly used in standard video coding (called "bidirectional prediction" or "B-frame coding") [37], but has the disadvantage that it is generally not suitable for low-latency streaming since such methods need information "from the future" to decode each B-frame. However, in standard codecs, the use of B-frames typically provides the best rate–distortion (RD)

performance when low-latency decoding is not required.

3) *Motion compensation via optical flow* is based on estimating and compressing optical flow which is applied with bilinear warping to a previously decoded frame to obtain a prediction of the frame currently being encoded [24, 30]. The residual error is then separately compressed to reduce total distortion and minimize temporal error accumulation. Recently published methods in this setting achieve compression that outperforms H.264 in terms of PSNR and that outperforms HEVC in terms of MS-SSIM [24, 30]. However, these methods rely on complex architectures and training schemes, such as pre-trained optical flow networks [24], sequential training of sub-networks [30, 24], adaptive rate control during encoding [30] and buffering intermediate reconstructions to disk during training [24].

Our research focuses on the third class of approaches, since it provides a good balance between rate–distortion performance and applicability to low-latency video streaming. However, we argue that using pre-trained optical flow networks [24] and bilinear warping [24, 30] may not be ideal for motion compensation:

1. General flow estimation needs to solve the *aperture problem*, which is not an issue for compression, so the model needlessly solves a harder problem than required. Moreover, optical flow networks aim to minimize motion vector error, while compression seeks to minimize a compromise between bitrate (the entropy of the latent representation of the flow and residual) and distortion (reconstruction error).

2. The need to rely on existing optical flow network architectures thus potentially adds unnecessary constraints and complexity to the design of compression networks.

3. The best optical flow models require a supervised training stage for state-of-the-art performance, which relies on annotated flow data, complicates the training procedure, and limits the domains of applicability.

4. Unlike standard video codecs that use motion compensation vectors, optical flow is dense, meaning that every pixel is warped. Since there is no concept of "not using" a flow prediction, unnecessarily large residuals are expected in the case of disocclusions.

To address these concerns, we propose generalizing optical flow and bilinear warping to *scale-space flow* and *scale-space warping* (see Figure 1), where a *scale field* is added as a third dimension to the typical 2-channel flow field. This per-location scale parameter allows the warping operation to better handle difficult cases and to more gracefully degrade when no flow-based prediction is possible. The scale dimension allows the model to learn to adaptively blur the source content before warping based on how well it predicts the next frame. Intuitively, this should lead to a smaller intermediate residual error and, in turn, to a more compressible residual since the model won't need to spend as many bits to "undo errors" introduced by the warping step.

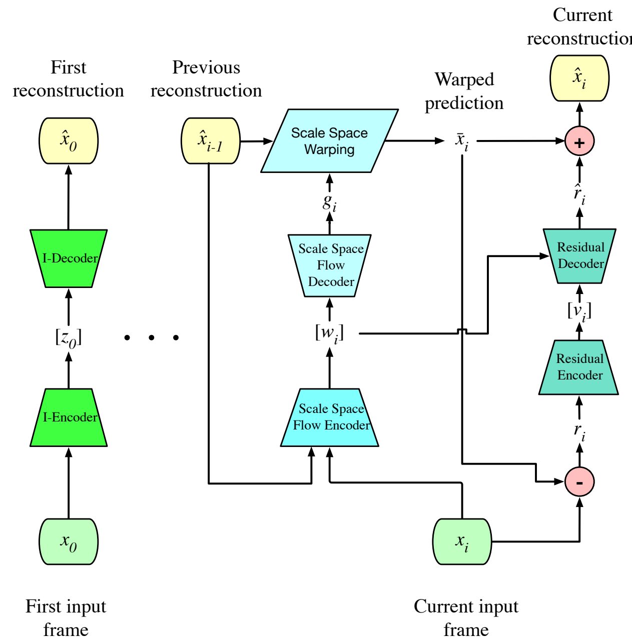

Figure 2. Overview of our end-to-end optimized, low-latency compression system: 1) the scale-space flow is jointly estimated and encoded to a quantized latent, $[w_i]$; 2) the previous reconstruction, $\hat{x}_{i-1}$, is warped using the decoded scale-space flow field, g_i, yielding the prediction, $\bar{x}_i$; 3) the remaining residual, $r_i = x_i - \bar{x}_i$, is encoded to a quantized latent, $[v_i]$, and is decoded to $\hat{r}_i$, which is added to the warped prediction to get the final reconstruction, $\hat{x}_i = \bar{x}_i + \hat{r}_i$. All of the encoder & decoder networks are simple four layer CNNs trained concurrently after random initialization.

Furthermore, we show that a scale-space warping operation integrated into a simple low-latency compression pipeline (depicted in Figure 2) can yield rate–distortion results outperforming recent state-of-the-art learning-based methods. Specifically, for equal PSNR, our method provides an average Bjøntegaard Delta (BD) rate reduction [12] of 13.4% compared to [24] and a savings of 42.9% over [38], while we see a 30.3% savings over [19] for equal MS-SSIM (see Section 5 for a detailed evaluation). Compared to prior approaches for flow-based motion compensation [24, 30], our system is significantly simpler since we do not need to separately estimate flow or use pre-trained networks. We also do not need to use advanced training or encoding strategies such as buffering reconstructions [24] or spatially adaptive rate control [30].

Our ablation studies show that compared to bilinear warping, the proposed scale-space warping significantly improves the rate–distortion performance with gains of more than 1dB at some bitrates (see Section 5 for details).

In summary, our contributions are the following:

1. We propose scale-space flow and warping, an intuitive generalization to flow + bilinear warping that reduces the need for complex residuals in failure cases.

2. Using a simple architecture and training procedure, we are able to train our model end-to-end without utilizing a pre-trained optical flow network.

3. Our experiments show that scale-space flow outperforms recent state-of-the-art models such as [24, 19], while our ablation study shows that the same system trained for flow and bilinear warping performs significantly worse.

2. Related Work

Image Compression Research on learning-based image compression [7, 10, 32, 4, 29, 8, 25, 5] has shown significant progress in terms of rate–distortion performance compared to standard codecs such as JPEG [34], JPEG2000 [21] and BPG [11]. Recent state-of-the-art models [40, 15, 26] use hyperprior-based architectures [8] with improvements including autoregressive context models [26] and multi-rate training [15]. We consider these models to be foundational building blocks for learned compression and use the hyperprior architecture as part of our video compression model.

Standard Video Compression There is a long history of progress for hand-engineered video compression algorithms used to create video format standards. Compression rates have progressively improved, e.g., from H.263 [16], to H.264 [31] and more recently to HEVC [3]. These codecs provide a strong baseline for assessing the quality of learned video compression models, and HEVC in particular remains a strong competitor that often outperforms state-of-the-art learning-based methods.

Learned Video Compression As mentioned above, recent work on learned video compression roughly falls into three categories, of which motion compensation via optical flow is most related to our work. The architecture we adopt can be viewed as a greatly simplified version of the method in [24], which uses a pre-trained flow network [28] combined with a flow compression module. In contrast, we directly learn the motion estimation module from scratch (see *Scale Space Flow Encoder* in Figure 2) which jointly estimates and encodes the motion from the current input frame and the previous reconstruction.

The training process of [24] happens in sequential steps: the I-frame model is trained first and then the P-frame model, which only sees one frame at a time, is optimized. To ensure the P-frame model can handle its own output as input, reconstructions from the P-frame model are buffered to disk during training and fed back to the model. This complicates the training process and means that the P-frame model is trained using "stale" reconstructions from an older version of the model. In contrast, we concurrently train the I-frame and P-frame models from scratch, unrolling the P-frame model over multiple frames during training, which greatly simplifies the training procedure.

Scale-space for flow estimation The use of scale-space techniques has a long history in optical flow estimation, both with classical techniques (e.g. [6, 18, 13]) as well as the use of multi-scale pyramids in deep flow estimation networks [28, 14]. However, these works make use of the scale-space only for flow *estimation*, while the final result is still a standard 2-channel displacement field. In contrast, our estimated 3-channel scale-space flow directly integrates into our proposed scale-space warping operation (see Figure 1) – irrespective of whether a scale-space or multi-scale pyramid is used to estimate it.

Uncertainty estimates for optical flow The scale parameter of our proposed scale-space flow (see Figure 1) can be interpreted as an "uncertainty parameter" in the sense that it is natural to use a high scale value in regions where it is not feasible to obtain a good prediction via warping. While prior work on supervised optical flow studied how to integrate uncertainty into the predictions of flow estimation networks (see [20] for overview), such methods operate in the supervised setting: i.e. they predict the uncertainty in the prediction of *ground truth* flow. In contrast, this work focuses on generalizing the flow + warping operations so that the warped result forms a good prediction irrespective of the relationship between the displacement field and ground truth flow.

3. Method

3.1. Scale-space flow

Our proposed scale-space flow (see Figure 1 for an overview) generalizes flow and bilinear warping to also incorporate Gaussian blurring. Given an image $\mathbf{x}$ with a spatial shape of $H \times W$ and a flow field $\mathbf{f} = (\mathbf{f}_x, \mathbf{f}_y)$, the bilinear warping of $\mathbf{x}$ by $\mathbf{f}$ is denoted as

$$\begin{aligned} &\mathbf{x}' := \text{Bilinear-Warp}(\mathbf{x}, \mathbf{f}) \qquad \text{s.t.} \\ &\mathbf{x}'[x, y] = \mathbf{x}[x + \mathbf{f}_x[x, y], y + \mathbf{f}_y[x, y]] \end{aligned} \tag{1}$$

where $\mathbf{x}[x, y]$ denotes sampling the image $\mathbf{x}$ at (continuous) coordinates (x, y) using bilinear interpolation. We refer to the flow channels $\mathbf{f}_x, \mathbf{f}_y \in \mathbb{R}^{H\times W}$ as the x- and y-displacement fields of the flow $\mathbf{f}$.

For scale-space warping, we construct a fixed-resolution scale-space volume $X = [\mathbf{x}, \mathbf{x}*G(\sigma_0), \mathbf{x}*G(2\sigma_0), \cdots, \mathbf{x}*G(2^{M-1}\sigma_0)]$, where $\mathbf{x} * G(\sigma)$ denotes the convolution of $\mathbf{x}$ with a Gaussian kernel with scale σ. $\mathbf{X}$ represents a stack of progressively blurred versions of x with dimensions $H \times W \times (M+1)$, which we can sample at continuous coordinates (x, y, z) via trilinear interpolation.

We can now define a scale-space flow field as a 3-channel field $\mathbf{g} := (\mathbf{g}_x, \mathbf{g}_y, \mathbf{g}_z)$, and the corresponding scale-space

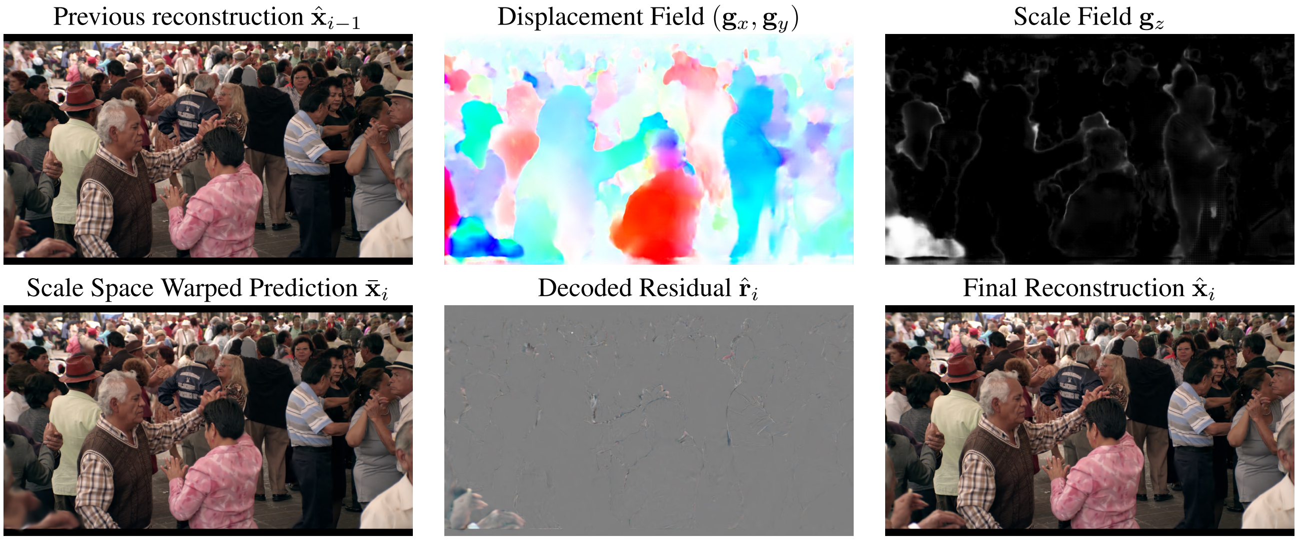

Figure 3. Visualization of the the internals of our model. The network learns to predict spatial flow even for a crowded scene. Note how the scale parameter increases around the boundaries of the people where warping is least likely to provide an accurate reconstruction. Similarly, in the bottom left corner of the image, the motion of the hands is not modeled well by warping so the network predicts a larger scale value that results in a blurrier intermediate reconstruction that ultimately helps minimize the global RD loss.

warp of the image $\mathbf{x}$ as

$$\begin{aligned} &\mathbf{x}' := \text{Scale-Space-Warp}(\mathbf{x}, \mathbf{g}) \qquad \text{s.t.} \\ &\mathbf{x}'[x, y] = \mathbf{X}[x + \mathbf{g}_x[x, y], y + \mathbf{g}_y[x, y], \mathbf{g}_z[x, y]] \end{aligned} \tag{2}$$

We refer to the newly introduced third flow channel $\mathbf{g}_z \in \mathbb{R}_+^{H\times W}$ as the *scale field* of the scale-space flow $\mathbf{g}$.

We note that Scale-Space-Warp is strictly more general than both bilinear warping and Gaussian smoothing. In particular, for $\mathbf{g} = (\mathbf{g}_x, \mathbf{g}_y, \mathbf{g}_z)$:

- When $\mathbf{g}_z = 0$ we obtain bilinear warping as a special case:

$$\begin{aligned} \text{Scale-Space-Warp}(\mathbf{x}, (\mathbf{g}_x, \mathbf{g}_y, \mathbf{0})) = \\ \text{Bilinear-Warp}(\mathbf{x}, (\mathbf{g}_x, \mathbf{g}_y)) \end{aligned} \tag{3}$$

- When $\mathbf{g}_x = \mathbf{g}_y = 0$ and $\mathbf{g}_z = \log_2(\sigma/\sigma_0)$ for $\sigma > \sigma_0$ we recover (approximate) Gaussian blur as a special case:

$$\text{Scale-Space-Warp}(\mathbf{x}, (0, 0, 1+\log_2(\sigma/\sigma_0)) \approx \mathbf{x} * G(\sigma), \tag{4}$$

  where equality holds if $\log_2(\sigma/\sigma_0) \in \{0, \cdot, M-1\}$.

Differentiability Since we use trilinear interpolation (across the 2+1 space + scale dimensions) for the Scale-Space-Warp operation, it is differentiable with respect to all the arguments $(\mathbf{x}, \mathbf{g}_x, \mathbf{g}_y, \mathbf{g}_z)$.

Complexity The additional complexity of Scale-Space-Warp as described above comes from having to construct the volume $\mathbf{X}$ as a stack of progressively blurred versions of the frame and the larger memory associated with storing it, which is linear in the number of scale levels M (we set $M = 5$ in all of our experiments). We chose this representation because it simplifies the implementation of trilinear warping. However, we note that one could technically replace $\mathbf{X}$ with a multi-scale pyramid where the image is decimated at each level, since the signal can be safely decimated after Gaussian filtering [23]. This would reduce the memory cost to a factor of $1 + 1/4 + 1/8 + \cdots = 1.33$ but would complicate the implementation, since it is no longer a matter of interpolating within a single 3-D tensor, but rather within a stack of 2-D tensors.

Reparameterization As mentioned above, the Gaussian kernel size as a function of the volume level is $[0, \sigma_0, 2\sigma_0, ..., 2^{M-1}\sigma_0]$, where the first level corresponds to the original image without filtering. In-between two levels levels $i \le z < i+1$, (with corresponding Gaussian kernel sizes σ_a and σ_b), the interpolated value corresponds to a filter that is a mixture of the two Gaussians. The mixture has an effective kernel size corresponding to the standard deviation, $\sigma = \sqrt{(z-i)\sigma_a^2 + (1-z+i)\sigma_b^2}$, which we use as an approximation of a Gaussian with a size in-between the two.

So given a desired effective kernel size $0 < \sigma < 2^{M-1}\sigma_0$, we can easily solve for the corresponding value of $z = i + (\sigma_b^2 - \sigma^2)/(\sigma_b^2 - \sigma_a^2)$. Thus, for a more natural parameterization, instead of predicting z we directly predict the effective kernel size σ, and then use the corresponding z for trilinear interpolation in the scale-space volume.

Composition While we do not study multi-scale architec-

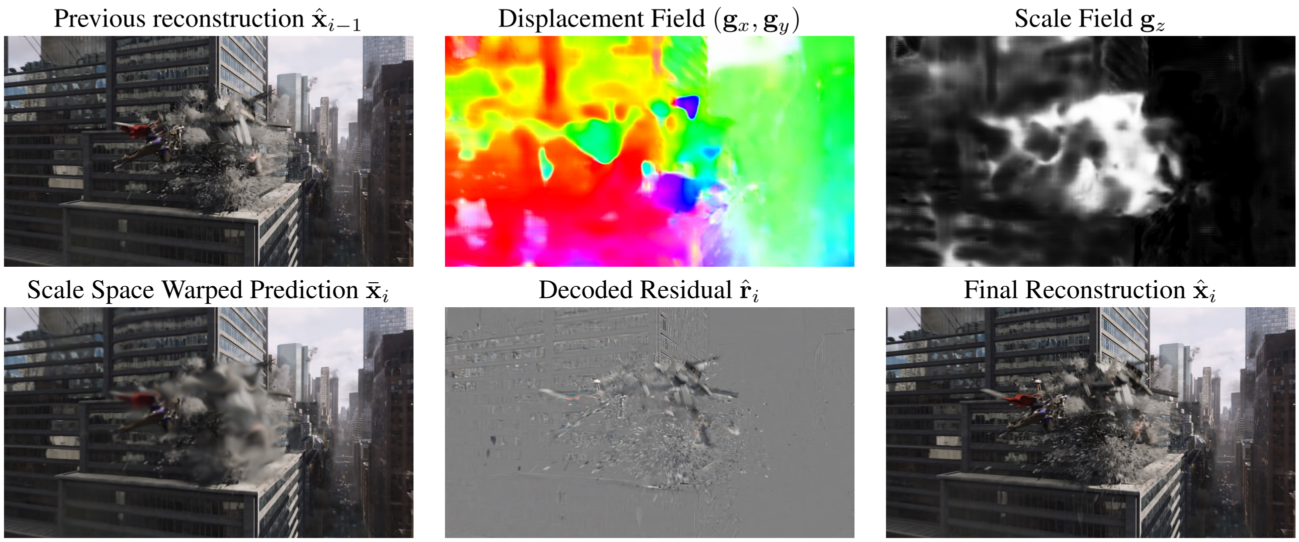

Figure 4. For this pair of frames, the camera motion is relatively well-modeled by the flow predictor, but the explosion in the center can not be modeled accurately by warping. To compensate, the network uses the scale field to blur the explosion and relies on the compressed residual to generate a high-quality reconstruction. Without the scale field, inaccurate warping can lead to a larger residual with a worse rate–distortion trade-off.

tures in this paper, it is common practice to do so for optical flow estimation [28, 14] where the *compositionality* of bilinear warping is exploited: when warping with a (potentially upscaled) field $\mathbf{f}_1$ followed by $\mathbf{f}_2$, one can specify an equivalent field $\mathbf{f}_3$ that achieves the same in a single operation. We note that it would in principle also be possible to integrate scale-space warping into such architectures, since Gaussian filtering has such compositionality [23].

3.2. Compression Model

Our model is targeted for **low-latency scenarios**, which refers to the setting where only previous (decoded) frames are available when encoding (or decoding) a given image. Figure 2 provides an overview of how scale-space warping can be integrated into such a compression architecture.

Given a sequence of frames $\mathbf{x}_0, \cdots, \mathbf{x}_N$ we encode the first (I) frame to a latent $\mathbf{z}_0$ which is quantized to integer values $[\mathbf{z}_0]$, obtaining a reconstruction $\hat{\mathbf{x}}_0$. Now, for a currently given (P-) frame $\mathbf{x}_i$, we use a single network to jointly estimate and encode the quantized scale-space warp latents $[\mathbf{w}_i]$, from which we decode a scale-space flow $\mathbf{g}_i$. We then scale-space warp the previous reconstruction $\hat{\mathbf{x}}_{i-1}$ to obtain an estimate of the current frame $\bar{\mathbf{x}}_i$. Since the estimate $\bar{\mathbf{x}}_i$ will be imperfect, a second branch will encode the residual $\mathbf{r}_i = \mathbf{x}_i - \bar{\mathbf{x}}_i$ to a latent $[\mathbf{v}_i]$ and apply the decoded residual $\hat{\mathbf{r}}_i$ to obtain a final reconstruction $\hat{\mathbf{x}}_i = \bar{\mathbf{x}}_i + \hat{\mathbf{r}}_i$.

For each of the three latent types, $\mathbf{z}_0, \mathbf{v}_i, \mathbf{w}_i$ we use a separate hyperprior [8, 26] to model the corresponding density. To improve computational efficiency, no autoregressive models are used within the hyperprior.

To summarize, we employ the hierarchical autoencoder architecture proposed for image compression [8, 26] for the purposes of I-frame compression, residual compression, and scale-space flow computation. This is different from previous work, where specialized optical flow networks are typically used.

3.3. Quantization and entropy estimation

While we generally adopt the approach of [10] to replace quantization with additive uniform noise to approximate Shannon cross entropy during training with differential cross entropy, we found that for the purpose of propagating "quantized" latents/residuals through further transformations, it was beneficial to use a straight-through estimator (i.e., quantize during training as well as evaluation, and substitute the gradient of the quantizer with the identity function for training). Our approach is thus a combination of the proposals in [10] and [32].

3.4. Loss

We optimize the whole system for the total rate–distortion loss unrolled over N frames [7]:

$$\sum_{i=0}^{N-1} d(\mathbf{x}_i, \hat{\mathbf{x}}_i) + \lambda \left[H(\mathbf{z}_0) + \sum_{i=1}^{N-1} H(\mathbf{v}_i) + H(\mathbf{w}_i) \right], \quad (5)$$

where $H(\cdot)$ denotes the entropy estimate of the respective latent, including the side information extracted by its hyperprior (see [26] for details), and d denotes the distortion metric such as mean squared error (MSE) or multiscale structural similarity (MS-SSIM) [36]. This means that during training, the bitrate allocation for the image latent $\mathbf{z}_0$, the

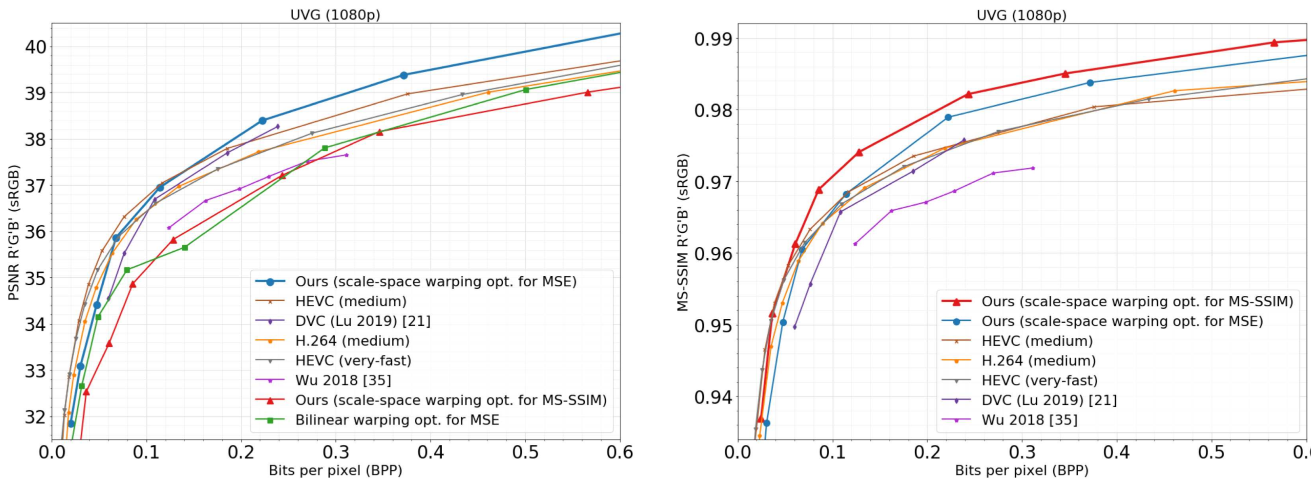

Figure 5. Rate–distortion comparison on the UVG dataset [2] using PSNR (*left*) and MS-SSIM (*right*). Our approach outperforms existing low-latency research models ([24, 38, 19]) at all bit rates for both metrics. Our method outperforms HEVC above 0.15 bpp for PSNR and above 0.05 bpp according to MS-SSIM.

motion compensation latents $\mathbf{w}_i$ and the residual latents $\mathbf{v}_i$ are all automatically determined by the system.

Equation 5 does not contain any loss terms specific to optical flow such as warping losses or total variation regularization. Instead, our networks learn to perform motion compensation with the scale-space flow directly as a byproduct of minimizing the rate–distortion equation.

4. Experimental setup

Architecture Our system uses a simplified version of the architecture from the hyperprior image compression system [26] as a building block, using ReLU activations instead of GDN [9] (see Supplementary for full details). In particular, we used the encoder architecture of [8] for the "I-Encoder", "Scale Space Flow Encoder", and "Residual Encoder', and the corresponding decoder architecture for "I-Decoder", "Scale Space Flow Decoder" and "Residual Decoder" (Figure 2).

Training data The models were trained on video frames extracted from approx. 700,000 high definition (1920×1080) videos with a frame rate of 30Hz (which have been transcoded by YouTube). In an ideal scenario it would be better to use uncompressed video. From each video sequence, we extracted 60 consecutive frames, which were partitioned into temporal chunks of $N = 3$ frames. To reduce pre-existing compression artifacts, since we don't have access to uncompressed videos, the chunks were then downsampled by a randomized factor averaging $\frac{2}{3}$, and randomly cropped to 256×256 or 384×384 pixels (see details below). These video fragments were then randomized, and batches of 8 fragments each were fed to the training algorithm. We note that our method saw at most $1250000 \cdot 8 \cdot 3/30/3600 \approx 278$ hours of video during training ($<$ 1.25M steps, batch size of 8 with 3 frames, 30FPS average across videos), which is an order of magnitude less data than is e.g. available in Vimeo-90K [39].

Colorspace We train and evaluate our models in the sRGB colorspace. This is not ideal, because the native format for most video content is Y'CbCr with 4:2:0 chroma subsampling, and the conversion to and from sRGB is not lossless. However, we adopt sRGB to facilitate comparison with almost all published work in neural video compression [38, 24, 19, 27, 17].

Trained models We optimized our model both for MSE and the MS-SSIM distortion metric, using 9 rate points covering a bitrate range of 0.025 to 0.8 bpp. In particular, we used $\lambda = 0.01 \cdot 2^i$ for MSE and $\lambda = 10 \cdot 2^i$ for MS-SSIM, where $i = -3, \ldots, 5$. We refer to these models as 'Ours (scale-space warping opt. for MSE)' and 'Ours (scale-space warping opt. for MS-SSIM)' respectively.

To measure the benefit of scale-space warping, we optimized 9 models for MSE in an identical fashion, with the only difference being the warp method (i.e. in Figure 2 we change Scale-Space-Warp to Bilinear-Warp and output a 2-channel flow $\mathbf{f}$ instead of the 3 channel scale-space flow $\mathbf{g}$ in the corresponding decoder). We refer to this model as 'Bilinear warping opt. for MSE.'

Training schedule For training we used the Adam[22] optimizer with a base learning rate of 10^{-4}, batch size of 8 and a crop size of 256 pixels. To further reduce training costs, we trained all models for an MSE loss for the first 1,000,000 steps (which could be shared across the MSE and MS-SSIM models), and then further trained the MS-SSIM models for 200,000 steps with the MS-SSIM loss. Finally, for all models we decayed the learning rate to 10^{-5} for 50,000 steps, increasing the crop size to 384×384 pixels[1] at the same

[1]We found that increasing the crop size improved performance (pre-

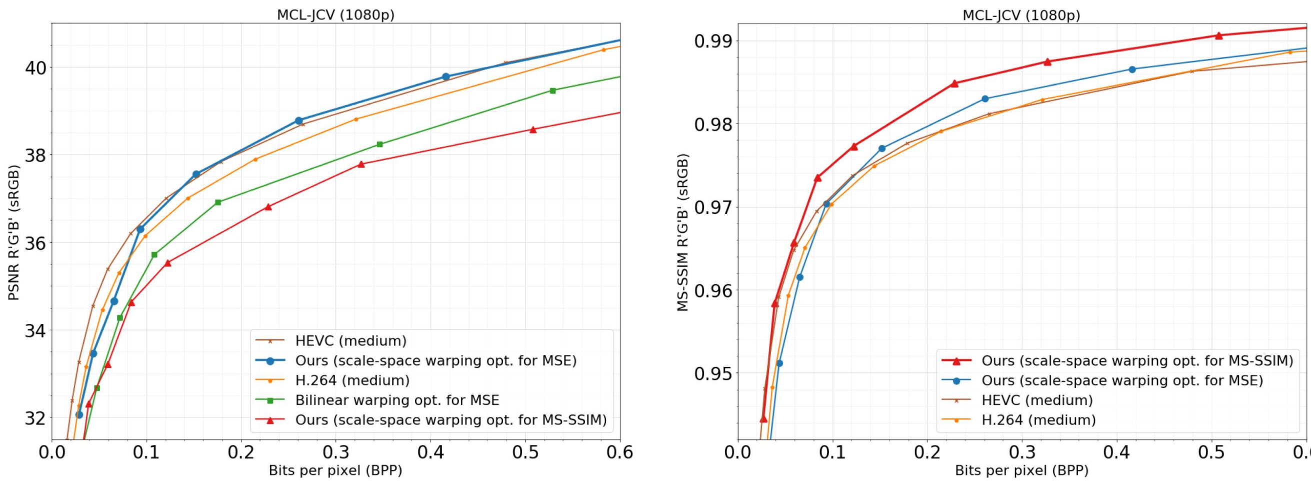

Figure 6. Rate–distortion comparison on the MCL-JCV dataset [35]. Our approach outperforms H.264 above 0.08 bpp on PSNR but has worse rate–distortion performance than HEVC. On MS-SSIM, however, our method outperforms H.264 at all bit rates and exceeds HEVC above 0.08 bpp. We believe the relatively poor PSNR results are due to the presence of several animated videos in the MCL-JCV dataset for which our model was not optimized (see Figure 7 for details).

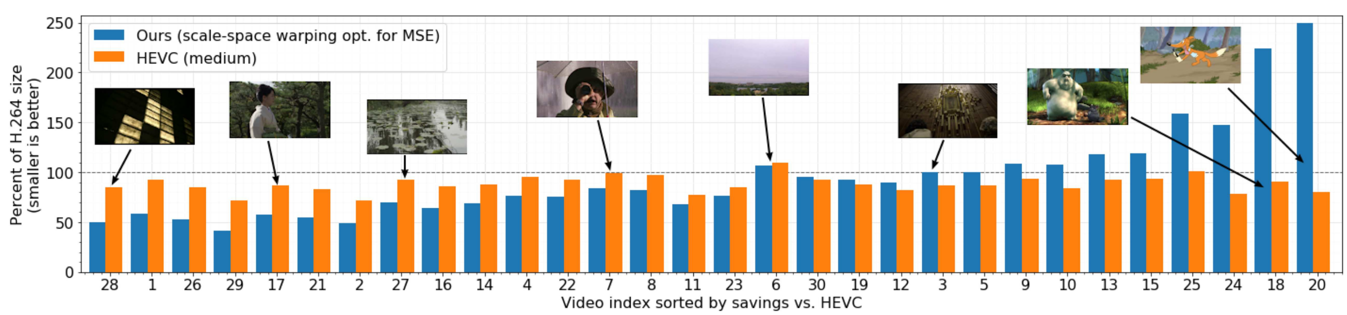

Figure 7. Rate savings for each video in the MCL-JCV dataset [35]. Values represent the file size relative to H.264 as estimated by BD rate for equal PSNR. Our method has smaller or equal file sizes compared to both H.264 and HEVC for most videos (21/30 and 17/30 respectively), but performs much worse on animations (videos 25, 24, 18, and 20), which is not surprising since the training data primarily contains natural videos.

time.

Number of unrolled frames While training for $N = 9$ or $N = 12$ unrolled frames yielded good results for the initial models we explored, the training speed was too slow for practical experimentation with 1,000,000 training steps taking 30 days to train on a NVidia V100 GPU. We found that similar results could be obtained by training with $N = 3$ frames and without passing gradients from the I-frame reconstruction to the P-frame branch (to avoid the I-frame loss dominating the optimization). We trained all the models in this setting, which reduces the training time to approx. 4 days and allows for much faster experimentation.

Standard baselines and compared methods We evaluate the RD performance of our method and compare it with recently published learning-based methods [24, 19, 38] as well as standard codecs (H.264 [31] and HEVC [3]). We evaluate H.264 and HEVC using typical `ffmpeg` settings for low-latency mode, i.e. medium profile with B-frames disabled (see Supplementary for the full command line), and we refer to the results as *H.264 (medium)* and *HEVC (medium)* below. To ensure we perform an apples-to-apples comparison with recent methods, we also evaluate HEVC with the settings used in [24, 19, 38] to verify the baseline matches what is reported in those papers, which we refer to as *HEVC (very-fast)*.

5. Results

Qualitative results In Figures 3 & 4, we visualize the internals of our models for input frames taken from two different evaluation videos. We observe that the model learns to compensate for complex motion in crowded scenes, predicting flow-like displacement fields while purely being trained for the rate–distortion objective in Eq. (5). When the motion is

sumably because of a reduction in border artefacts), at the cost of slowing the training speed – which is why we only did it for the last 50,000 steps.

too complex to be captured by bilinear warping, the model utilizes the scale field to produce a simpler residual.

Quantitative results on the UVG dataset In Figure 5, we show the RD performance on the UVG dataset [2], both in terms of PSNR and MS-SSIM. For PSNR, our MSE-optimized model outperforms the recently introduced DVC method [24], which uses a much more complex architecture with a pre-trained multi-scale optical flow network for motion compensation. Furthermore, we outperform HEVC (`-preset very-fast`) at bitrates above ~0.07 bpp and exceed HEVC (`-preset medium`, the default setting) above ~0.15 bpp. When optimized for MS-SSIM, our model significantly outperforms all of the learning-based methods and H.264 over the entire range of bitrates, and its performance exceeds HEVC above ~0.05 bpp.

Quantitative results on the MCL-JCV dataset In Figure 6, we evaluate our model on the MCL-JCV dataset [35]. In terms of MS-SSIM, our MS-SSIM optimized model outperforms H.264 at all bit rates and exceeds HEVC above ~0.075 bpp. However, when evaluated using PSNR, our MSE-optimized model only outperforms H.264 above ~0.08 bpp, and trails HEVC at bitrates above 0.1bpp.

Per-Video level analysis on MCL-JCV In Figure 7, we compute the Bjøntegaard Delta (BD) rate reduction [12] for equal PSNR relative to H.264 [31] for each video in the MCL-JCV dataset (see the supplemental materials for details on BD rate calculations). We then plot the relative size of each encoded video compared to H.264. For example, a BD rate savings of 15% means that the relative size is 85% (100% – 15%). By construction, the H.264 result is always 100.0%.

We can see that in terms of BD rate, for a majority of videos (21/30) we have smaller or equal file sizes than H.264. In Figure 7, this is shown by blue bars that are below the dotted line representing the H.264 baseline. However, for a fraction of the videos (4/30), the compressed files generated by our method are more than 50% larger than H.264, and two are more than twice as large. The most challenging videos for our method (videos 18, 20, 24, and 25) are all animations, which could be explained by the lack of animated videos in our training dataset as well as the challenge of estimating motion for animations which tend to have relatively little texture. For further details, see the supplemental materials, which include separate RD graphs over just the "natural" videos and just the animated videos in MCL-JCV.

Compared to HEVC (represented by orange bars in Figure 7), our method has significantly smaller file sizes for about half of the videos (17/30). Nonetheless, in terms of PSNR, HEVC outperforms our model on MCL-JCV at low bit rates as shown in Figure 6 (*left*).

Bilinear warp vs Scale-Space warp Comparing our method with the (identically trained) "Bilinear warping" baseline, we find in Figures 5 & 6 that the performance gain of scale-space warping is significant both on the UVG and the MCL-JCV dataset, with a gap of 1dB for bitrates above 0.1bpp.

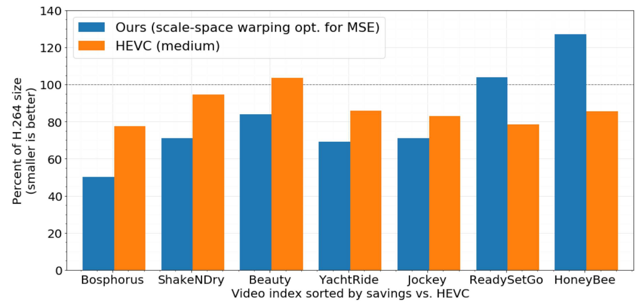

Figure 8. Rate savings for each video in the UVG dataset [2]. Values represent the file size relative to H.264 as estimated by BD rate for equal PSNR, e.g. an average rate savings of 25% yields a value of 75% (100% – 25%).

6. Discussion

In this paper, we proposed scale-space flow and scale-space warping as a generalization of flow and bilinear warping for use in the motion compensation step of learned video compression. Scale-space warping allows our network to better model regions which are poorly predicted with bilinear warping due to issues like disocclusion and fast or erratic motion.

We studied the scale-space warping operation in a simple low-latency motion compensation pipeline, without any pretrained optical flow or complex training or evaluation procedures. Our evaluation shows that it outperforms recent state-of-the-art learning-based methods [24, 19, 38] as well as the standard codecs H.264 and HEVC when evaluated using MS-SSIM.

While the field of learned video compression is still in its infancy, and the research community is still investigating architectures, we believe scale-space warping provides a useful component and a novel and competitive direction for future model explorations. Future directions could include studying more complex architectures (including multi-scale models) and generalizations that use more than one previous frame for warping. Research is also needed to improve generalization to animated videos and to intelligently place I-frames to better handle scene cuts and other abrupt changes.

References

[1] Report: Where Does the Majority of Internet Traffic Come From? https://www.ncta.com/whats-new/report-where-does-the-majority-of-internet-traffic-come. Accessed: 2019-11-12. 1

[2] Ultra video group test sequences. http://ultravideo.cs.tut.fi. 6, 8

[3] ITU-R rec. H.265 & ISO/IEC 23008-2: High efficiency video coding, 2013. 3, 7

[4] Eirikur Agustsson, Fabian Mentzer, Michael Tschannen, Lukas Cavigelli, Radu Timofte, Luca Benini, and Luc Van Gool. Soft-to-Hard Vector Quantization for End-to-End Learning Compressible Representations. In *NIPS*, 2017. 3

[5] Eirikur Agustsson, Michael Tschannen, Fabian Mentzer, Radu Timofte, and Luc Van Gool. Generative adversarial networks for extreme learned image compression. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 221–231, 2019. 1, 3

[6] Luis Alvarez, Joachim Weickert, and Javier Sánchez. Reliable estimation of dense optical flow fields with large displacements. *International Journal of Computer Vision*, 39(1):41–56, 2000. 3

[7] Johannes Ballé, Valero Laparra, and Eero P Simoncelli. End-to-end Optimized Image Compression. *ICLR*, 2016. 3, 5

[8] Johannes Ballé, David Minnen, Saurabh Singh, Sung Jin Hwang, and Nick Johnston. Variational Image Compression with a Scale Hyperprior. *ICLR*, 2018. 1, 3, 5, 6

[9] Johannes Ballé, Valero Laparra, and Eero P. Simoncelli. Density modeling of images using a generalized normalization transformation. *arXiv e-prints*, 2016. presented at the 4th Int. Conf. on Learning Representations. 6

[10] Johannes Ballé, Valero Laparra, and Eero P. Simoncelli. End-to-end optimized image compression. *arXiv e-prints*, 2017. presented at the 5th Int. Conf. on Learning Representations. 3, 5

[11] F. Bellard. BPG image format (http://bellard.org/bpg/). Accessed: 2017-01-30. 3

[12] Gisle Bjøntegaard. Calculation of average PSNR differences between RD-curves. Doc. VCEG-M33, ITU-T SG16/Q6 VCEG, Austin, TX, USA, Apr. 2001. 2, 8

[13] Andrés Bruhn, Joachim Weickert, and Christoph Schnörr. Lucas/kanade meets horn/schunck: Combining local and global optic flow methods. *International journal of computer vision*, 61(3):211–231, 2005. 3

[14] Jia-Ren Chang and Yong-Sheng Chen. Pyramid stereo matching network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5410–5418, 2018. 3, 5

[15] Yoojin Choi, Mostafa El-Khamy, and Jungwon Lee. Variable rate deep image compression with a conditional autoencoder. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3146–3154, 2019. 3

[16] Guy Cote, Berna Erol, Michael Gallant, and Faouzi Kossentini. H. 263+: Video coding at low bit rates. *IEEE Transactions on circuits and systems for video technology*, 8(7):849–866, 1998. 3

[17] Abdelaziz Djelouah, Joaquim Campos, Simone Schaub-Meyer, and Christopher Schroers. Neural inter-frame compression for video coding. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 6421–6429, 2019. 1, 6

[18] Haifeng Gong, Chunhong Pan, Qing Yang, Hanqing Lu, and Songde Ma. Generalized optical flow in the scale space. *Computer Vision and Image Understanding*, 105(1):86–92, 2007. 3

[19] Amirhossein Habibian, Ties van Rozendaal, Jakub M Tomczak, and Taco S Cohen. Video compression with rate-distortion autoencoders. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 7033–7042, 2019. 1, 2, 3, 6, 7, 8

[20] Eddy Ilg, Ozgun Cicek, Silvio Galesso, Aaron Klein, Osama Makansi, Frank Hutter, and Thomas Brox. Uncertainty estimates and multi-hypotheses networks for optical flow. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 652–667, 2018. 3

[21] Information technology–JPEG 2000 image coding system. Standard, International Organization for Standardization, Geneva, CH, Dec. 2000. 3

[22] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 6

[23] Tony Lindeberg. *Scale-space theory in computer vision*, volume 256. Springer Science & Business Media, 2013. 4, 5

[24] 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 Conference on Computer Vision and Pattern Recognition*, pages 11006–11015, 2019. 2, 3, 6, 7, 8

[25] Fabian Mentzer, Eirikur Agustsson, Michael Tschannen, Radu Timofte, and Luc Van Gool. Conditional probability models for deep image compression. In *2018 IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2018. 3

[26] David Minnen, Johannes Ballé, and George D Toderici. Joint Autoregressive and Hierarchical Priors for Learned Image Compression. In *NeurIPS*. 2018. 1, 3, 5, 6

[27] Jorge Pessoa, Helena Aidos, Pedro Tomás, and Mário AT Figueiredo. End-to-end learning of video compression using spatio-temporal autoencoders. 2018. 1, 6

[28] 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, 5

[29] Oren Rippel and Lubomir Bourdev. Real-time adaptive image compression. In *Proc. of Machine Learning Research*, volume 70, pages 2922–2930, 2017. 3

[30] Oren Rippel, Sanjay Nair, Carissa Lew, Steve Branson, Alexander G Anderson, and Lubomir Bourdev. Learned video compression. In *Proc. ICCV*, pages 3454–3463, 2019. 2

[31] Heiko Schwarz, Detlev Marpe, and Thomas Wiegand. Overview of the scalable video coding extension of the h. 264/avc standard. *To appear in IEEE Transactions on Circuits and Systems for Video Technology*, page 1, 2007. 3, 7, 8

[32] Lucas Theis, Wenzhe Shi, Andrew Cunningham, and Ferenc Huszár. Lossy image compression with compressive autoencoders. 2017. presented at the 5th Int. Conf. on Learning Representations. 3, 5

[33] George Toderici, Damien Vincent, Nick Johnston, Sung Jin Hwang, David Minnen, Joel Shor, and Michele Covell. Full resolution image compression with recurrent neural networks. In *2017 IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2017. 1

[34] Gregory K. Wallace. The jpeg still picture compression standard. *Communications of the ACM*, pages 30–44, 1991. 3

[35] 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. 7, 8

[36] Zhou Wang, Eero P Simoncelli, and Alan C Bovik. Multiscale structural similarity for image quality assessment. In *Signals, Systems and Computers, 2004. Conference Record of the Thirty-Seventh Asilomar Conference on*, volume 2, pages 1398–1402. IEEE, 2003. 1, 5

[37] wikipedia. Video compression picture types, 2019g. [Online; accessed 15 November 2019]. 1

[38] 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, 6, 7, 8

[39] Tianfan Xue, Baian Chen, Jiajun Wu, Donglai Wei, and William T Freeman. Video enhancement with task-oriented flow. *International Journal of Computer Vision (IJCV)*, 127(8):1106–1125, 2019. 6

[40] Lei Zhou, Zhenhong Sun, Xiangji Wu, and Junmin Wu. End-to-end optimized image compression with attention mechanism. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 0–0, 2019. 3