DSIC: Deep Stereo Image Compression

Jerry Liu [‡], Shenlong Wang, and Raquel Urtasun
Uber ATG
jerryl@uber.com, slwang@uber.com, urtasun@uber.com

Abstract

In this paper we tackle the problem of stereo image compression, and leverage the fact that the two images have overlapping fields of view to further compress the representations. Our approach leverages state-of-the-art single-image compression autoencoders and enhances the compression with novel parametric skip functions to feed fully differentiable, disparity-warped features at all levels to the encoder/decoder of the second image. Moreover, we model the probabilistic dependence between the image codes using a conditional entropy model. Our experiments show an impressive $30\text{-}50\%$ reduction in the second image bitrate at low bitrates compared to deep single-image compression, and a $10\text{-}20\%$ reduction at higher bitrates.

1. Introduction

Many applications such as autonomous vehicles and 3D movies involve the use of stereo camera pairs. These arrays of cameras oftentimes capture and store massive quantities of data per day, which require good image compression algorithms to ensure an efficient use of space. A naive approach to image compression would be to compress the image streams from each camera separately. However, this ignores the shared information given by the overlapping fields of view between the cameras. Hence, there is a need for compression methods that can efficiently compress a stereo image pair further by utilizing the shared information.

Stereo image compression can be seen as in-between the work of image and video compression. While we get to utilize shared information between two images, we are not able to exploit the spatial-temporal redundancies within a tightly coupled image sequence. There has been an abundance of work on traditional multi-view and stereo compression [12, 14] as well as deep-learning based image and video compression [5, 33, 36, 49]. However, the space of deep multi-view compression is relatively unexplored.

In this work, we present a novel end-to-end deep architecture for stereo image compression. Our contributions revolve around trying to extract as much information out of the first image in order to reduce the bitrate of the second. Towards this goal we leverage state-of-the-art single-image compression autoencoders, and enhance them with novel *parametric skip functions* to feed fully differentiable, disparity-warped features at all levels to the encoder/decoder of the second image. This allow us to store fewer bits for the second image code since multi-level information is being passed from the encoder/decoder of the first image. Moreover, we model the probabilistic dependence between the image codes using a *conditional entropy model*. Since the codes of a stereo pair are highly correlated with each other, a model that can capture this dependence will help reduce the joint entropy, and hence the joint bitrate, of the two latent codes.

We demonstrate a $30\text{-}50\%$ reduction in the second image bitrate at low bitrates compared to deep single-image compression, and a $10\text{-}20\%$ reduction at higher bitrates, when evaluated over an internal self-driving dataset (NorthAmerica), as well as stereo images from Cityscapes. Our experiments additionally demonstrate that we outperform all image codecs and motion-compensation+residual coding baselines on MS-SSIM, a perceptual metric capturing the structural quality of an image.

2. Background and Related Work

We start this section with a brief overview of deep image compression algorithms, including the general formulation and previous works. We then discuss related work on deep video compression, multi-view compression and stereo estimation, respectively.

2.1. A Review on Deep Image Compression

There has been a plethora of work on learned, single-image, lossy image compression [44, 45, 4, 5, 43, 31, 35, 33, 24]. These works generally use nonlinear transforms through convolutional neural network (CNN) layers to encode an image into a latent space, which is then quantized

---
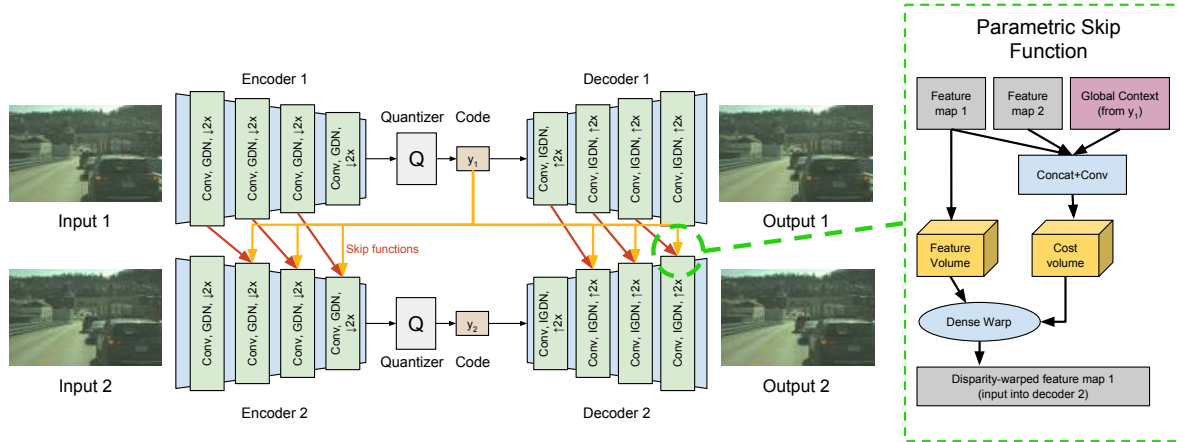[‡] Work done as part of the Uber AI Residency program.

Figure 1: Left: the end-to-end stereo compression architecture; right: the proposed deep parameteric skip function.

into discrete symbols. An entropy coding function using a learned entropy model is then applied to losslessly produce the final bitstream.

More formally, given the input x, deep image compression learns an encoding+quantization function $E(\cdot)$ mapping the input to a discrete-valued vector representation $\hat{y} = E(x)$ as well as a decoder function $D(\cdot)$ that reconstructs the image from the latent code: $\hat{x} = D(\hat{y})$. Both the encoder and decoder are trained to optimize for a balance between minimizing the expected code length of the latent code and maximizing the quality of the reconstruction; this is otherwise known as the rate-distortion tradeoff

$$\ell(x, \hat{x}) + \beta R(\hat{y}) \tag{1}$$

where ℓ is the reconstruction error between the input and the decoded output, typically measured by MSE (mean squared error) or a differentiable image quality metric such as MS-SSIM [46]; R is the cost for encoding the latent representation to a bitstream measured by bitrate. The bitrate is oftentimes approximated in a differentiable manner by measuring the cross-entropy between the latent code distribution and a learned prior:

$$R(\hat{y}) \approx \mathbb{E}_{\hat{y} \sim p_{\hat{y}}}[\log p(\hat{y}; \theta)] \tag{2}$$

Towards these goals, researchers have devised various ways to make the discrete binary encoding operations suitable for end-to-end learning, such as straight-through estimation [44, 43], soft quantization [2, 31] and noise sampling [4, 5]. Moreover, sophisticated prior models have been designed for the quantized representation in order to minimize the cross-entropy with the code distribution. Different approaches to model the prior include autoregressive models [33, 31, 45], hyperprior models [5, 33], and factorized models [43, 4, 5].

2.2. Deep Video Compression

Traditional video compression techniques exploit temporal redundancies by encoding independent frames (I-frames), then using motion compensation / residual coding to encode neighboring frames (P-frames, B-frames) [41, 47]. Recently, several deep-learning based video compression frameworks [49, 36, 15, 25] have been developed. Wu *et al.* [49] employs techniques based upon traditional video compression methods, while Rippel *et al.* [35] uses an end-to-end learning approach and achieves state-of-the-art results compared to traditional video codecs, including HEVC/H.265 and AVC/H.264.

Video compression techniques may not necessarily translate directly to a stereo setting because they typically rely on temporal redundancies between a larger block of images for most bitrate savings. We show in our experiments that indeed motion/residual coding can struggle for stereo.

2.3. Multi-view Compression

There has been much prior work on designing and analyzing multi-view compression techniques, usually in a video compression setting [12, 11, 32, 27, 30, 23]. In this setting, a multi-view video stream is reorganized as a matrix of pictures capturing temporal similarity between successive frames in one view and inter-view similarity between adjacent camera views. Numerous techniques [32, 27] use disparity compensated prediction to code each view given a reference view, similar to motion-compensated prediction in single-view video. The Multi-View Video Coding (MVC) extension developed for H.264/AVC uses this approach [32]. Other techniques involve using dense depth maps to synthesize a more precise view prediction for compression [30, 23].

Stereo specific compression techniques exist, and range from using a Markov random field (MRF) for

disparity prediction to separate transforms for residual images [40, 14, 48, 3, 34, 42]. Also closely related is light field image compression, where learning-based view synthesis techniques are used to take advantage of the vast redundancy between the subaperture images [20, 21].

In contrast, we use an end-to-end deep architecture for stereo image compression. Implicit depth estimation and compression is performed jointly in our model.

2.4. Stereo Matching

Traditional stereo matching methods range from local similarity estimation [8, 37, 16], particle propogation methods such as PatchMatch [7], to variational inference such as conditional random fields [38] and semi-global matching (SGM) [17]. There have been advances in deep learnable stereo matching, utilizing both supervised losses (training against ground truth disparities) [52, 28, 9, 22] as well as unsupervised losses (using a photometric/consistency loss) [53]. Stereo matching can be seen as a specific case of disparity-compensated prediction for the stereo image compression setting. Nonetheless, supervised learning-based stereo matching methods require ground-truth (GT) to train, and acquiring GT for stereo is difficult and expensive.

3. Deep Stereo Image Compression

In this paper we tackle the problem of compressing a pair of stereo images. Intuitively, if the overlapping field of view between the stereo pair is very high, then the bitrate of the combined latent code should be lower than the sum of the bitrates if we compressed the images separately; at the very least it cannot be higher. More formally, let us denote x_1, x_2 as a pair of rectified stereo images and let $H(x_1, x_2)$ be the entropy of the stereo pair. Given the fact that the content of the two images are highly correlated, the mutual information

$$I(x_1, x_2) = H(x_1) + H(x_2) - H(x_1, x_2) \geq 0$$

This observation motivates our general approach: we propose a single compression model that jointly compresses two stereo images. Towards this goal, we focus on extracting as much information as possible from one image in order to reduce the bitrate in the second, such that the total bitrate is lower than the result of independent single-image compression. Our approach is a two-stream deep encoder-decoder network as shown in Fig. 1. Each image in a stereo pair is passed to a separate encoder/quantizer to get a discretized latent code; then, a decoder is utilized to reconstruct the image from the latent code. Compared to previous work, we have two major contributions: 1) we add *multi-level, parametric skip functions* from the first image's feature maps to propagate information to the second image;

2) we utilize a *conditional entropy model* to model the correlation between the latent codes of the two images. Next we will describe each component in details.

3.1. Encoding/Decoding and Quantization

Our encoder, decoder, and quantizer functions borrow their architectures from the single-image compression model in Ballé et al. [5]. As shown in Fig. 1, each encoder is implemented with a series of 4 downsampling convolutions (by 2x) and Generalized Divisive Normalization (GDN) layers [6]. Each decoder is implemented with a series of 4 upsampling deconvolutions (by 2x) and Inverse-GDN layers. Each quantizer Q applies a rounding function to the floating-point output of the encoder $Q(E(x)) = Q(y) = \hat{y}$ to output the discrete code representation.

3.2. Parametric Skip Function

To reduce the joint bitrate across the stereo pair, we design a network module called a *parametric skip function* to propagate information from the first image's encoder/decoder to the second. We conjecture that for a given stereo pair, there exists a correlation between the feature maps of the two images at all levels in the encoder and decoder. Moreover, if we estimate the disparity between each pair of feature maps, we can warp one feature to the other and improve the pixel-level alignment between the two feature maps; this in turn allows us to pass information from one feature map accurately to the corresponding spatial positions of the other.

Specifically, in order to compute the feature map of the second image at level t, each skip function takes its previous layer's feature h_2^{t-1}, the previous layer feature from image one h_1^{t-1} and the first image's code $\hat{y}_1$ as input. Each skip function module consists of four parts. First, a fully convolutional **global context** encoding module $f(\hat{y}_1; w)$ encodes the first image's latent code to a feature descriptor d_1, to capture global context information of the first image, contained in its latent code. The global context feature is shared across all the different levels. Secondly, a **stereo cost volume** module estimates a cost volume c^{t-1} from the input of the first feature map, second feature map and the global context. The cost volume's size is $C \times H^{t-1} \times W^{t-1}$ where C is the maximum disparity and H^{t-1} and W^{t-1} are the height/width of h_1^{t-1}. A softmax layer is applied to ensure the cost is normalized along the disparity dimension per pixel. Each value in the cost volume can be seen as a probability/confidence measure of the correct disparity at that coordinate. We then use this cost volume to **densely warp** the feature from the first image to the second. Particularly, for each pixel i the new feature vector is computed through a weighted sum of feature vectors

across all the disparity values in the disparity range:

$$\boldsymbol{g}_{2,i}^{t-1} = \sum_{d=0}^{C} c_{d,i} \cdot \boldsymbol{h}_{1,(i,d)}^{t-1} \qquad (3)$$

where $c_{d,i}$ represents the cost of disparity d at pixel i. (i,d) represents the pixel index that is d pixels right of pixel i. The volumetric warping gives us a warped feature map $\boldsymbol{g}_2^{t-1}$ which better aligns with the feature map of the second image; this can be also seen as an attention mechanism for each pixel i into the first image's feature map within a disparity range. This design is conceptually similar to previous image synthesis work [13, 51]. Compared to regressing a single disparity map and warping with bilinear sampling [19], our design allows more flexible connections between the target pixel and a range of pixels from the source image. Finally, we use an **aggregation** function to predict the feature map as the final output of our parametric skip function:

$$\boldsymbol{h}_2^t = a(\boldsymbol{g}_2^{t-1}, \boldsymbol{h}_2^{t-1}) \qquad (4)$$

with $\boldsymbol{g}_{2,i}^{t-1}$ the volumetric warped feature from the first iamge and $\boldsymbol{h}_2^{t-1}$ the previous layer's feature from the second image. We refer to supplementary material (Sec. C) for architecture details of context encoding, stereo cost volume and aggregation.

3.3. Conditional Entropy Model

Accurately modeling the entropy of the quantized code is critical in navigating the rate-distortion trade-off, as entropy is a tight lower bound of bitrate [39]. There exists a correlation between the latent codes of the two images, given the highly correlated image content. In order to exploit these relationships, we develop a joint entropy model with neural networks to estimate the joint distribution of the code. In order to maintain differentiability while mimicking the effect of rounding error during quantization, we consider a noisy version of $\boldsymbol{y}$ as input: $\bar{\boldsymbol{y}} = \boldsymbol{y} + \boldsymbol{\epsilon}$, where $\boldsymbol{\epsilon} \sim \mathcal{U}(-0.5, 0.5)$.

Our goal is to model the joint distribution $p(\bar{\boldsymbol{y}}_1, \bar{\boldsymbol{y}}_2; \boldsymbol{\theta})$, which takes into account the dependence of $\bar{\boldsymbol{y}}_2$ on $\bar{\boldsymbol{y}}_1$. Inspired by [5, 33], we additionally want to include *side information as hyperpriors*, $\bar{\boldsymbol{z}}_1, \bar{\boldsymbol{z}}_2$, whose sole purpose is to reduce the entropy of $\bar{\boldsymbol{y}}_1, \bar{\boldsymbol{y}}_2$. Note that $\bar{\boldsymbol{z}}_1, \bar{\boldsymbol{z}}_2$ are derived from $\bar{\boldsymbol{y}}_1, \bar{\boldsymbol{y}}_2$ respectively during encoding and must also be counted in the bitstream. Thus, we factorize the joint probability of $\bar{\boldsymbol{y}}_1, \bar{\boldsymbol{y}}_2, \bar{\boldsymbol{z}}_1, \bar{\boldsymbol{z}}_2$ as follows:

$$p(\bar{\boldsymbol{y}}_1, \bar{\boldsymbol{y}}_2, \bar{\boldsymbol{z}}_1, \bar{\boldsymbol{z}}_2; \boldsymbol{\theta}) = \\ p(\bar{\boldsymbol{y}}_2|\bar{\boldsymbol{y}}_1, \bar{\boldsymbol{z}}_2; \boldsymbol{\theta}_{\bar{\boldsymbol{y}}_2}) p(\bar{\boldsymbol{y}}_1|\bar{\boldsymbol{z}}_1; \boldsymbol{\theta}_{\bar{\boldsymbol{y}}_1}) p(\bar{\boldsymbol{z}}_2; \boldsymbol{\theta}_{\bar{\boldsymbol{z}}_2}) p(\bar{\boldsymbol{z}}_1; \boldsymbol{\theta}_{\bar{\boldsymbol{z}}_1}) \qquad (5)$$

where $p(\bar{\boldsymbol{y}}_1|\bar{\boldsymbol{z}}_1; \boldsymbol{\theta}_{\bar{\boldsymbol{y}}_1})$ denotes the probability of the first image code and $p(\bar{\boldsymbol{y}}_2|\bar{\boldsymbol{y}}_1, \bar{\boldsymbol{z}}_2; \boldsymbol{\theta}_{\bar{\boldsymbol{y}}_2})$ denotes the probability of the second image code, which is conditioned on the
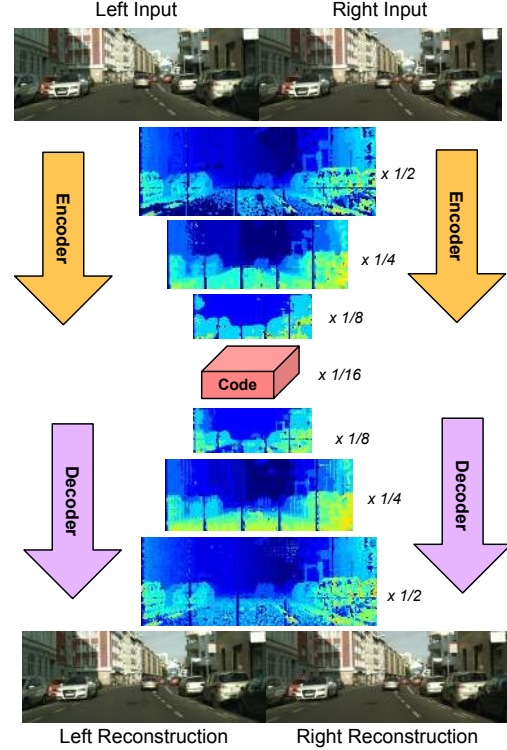

Figure 2: Visualization of the disparity volumes at each resolution level in the encoder/decoder, by taking the mode over the disparity dimension for each feature pixel. Tiling effects are inherently due to unsupervised training on crops.

first image. $\boldsymbol{\theta}_{\bar{\boldsymbol{y}}_2}, \boldsymbol{\theta}_{\bar{\boldsymbol{y}}_1}, \boldsymbol{\theta}_{\bar{\boldsymbol{z}}_2}, \boldsymbol{\theta}_{\bar{\boldsymbol{z}}_1}$ are the full set of parameters for each univariate distribution. All models are factorized into the product of each individual code's distribution under the full independence and conditional independence assumptions:

$$p(\bar{\boldsymbol{z}}_1; \boldsymbol{\theta}_{\bar{\boldsymbol{z}}_1}) = \prod_i p_{1,i}(\bar{z}_{1,i}; \boldsymbol{\theta}_{\bar{\boldsymbol{z}}_1}) \qquad (6)$$

$$p(\bar{\boldsymbol{z}}_2; \boldsymbol{\theta}_{\bar{\boldsymbol{z}}_2}) = \prod_i p_{2,i}(\bar{z}_{2,i}; \boldsymbol{\theta}_{\bar{\boldsymbol{z}}_2}) \qquad (7)$$

$$p(\bar{\boldsymbol{y}}_1|\bar{\boldsymbol{z}}_1; \boldsymbol{\theta}_{\bar{\boldsymbol{y}}_1}) = \prod_i p_{1,i}(\bar{y}_{1,i}|\bar{\boldsymbol{z}}_1; \boldsymbol{\theta}_{\bar{\boldsymbol{y}}_1}) \qquad (8)$$

$$p(\bar{\boldsymbol{y}}_2|\bar{\boldsymbol{y}}_1, \bar{\boldsymbol{z}}_2; \boldsymbol{\theta}_{\bar{\boldsymbol{y}}_2}) = \prod_i p_{2,i}(\bar{y}_{2,i}|\bar{\boldsymbol{y}}_1, \bar{\boldsymbol{z}}_2; \boldsymbol{\theta}_{\bar{\boldsymbol{y}}_2}) \qquad (9)$$

Directly modeling a probability density function (PDF) with a deep parametric function may not be suitable for PDFs with discontinuous shapes, e.g., a uniform distribution between [-0.5, 0.5]. This restricts the power of the entropy model. Following [5], we overcome this issue by modeling probabilities as an area under the cumulative density function (CDF) as opposed to a point on the PDF. We first design our hyperprior models $p_i(\bar{z}_i; \boldsymbol{\theta}_{\bar{\boldsymbol{z}}})$ as follows:

$$p_i(\bar{z}_i; \boldsymbol{\theta}_{\bar{\boldsymbol{z}}}) = (q_i * u)(\bar{z}_i) \qquad (10)$$
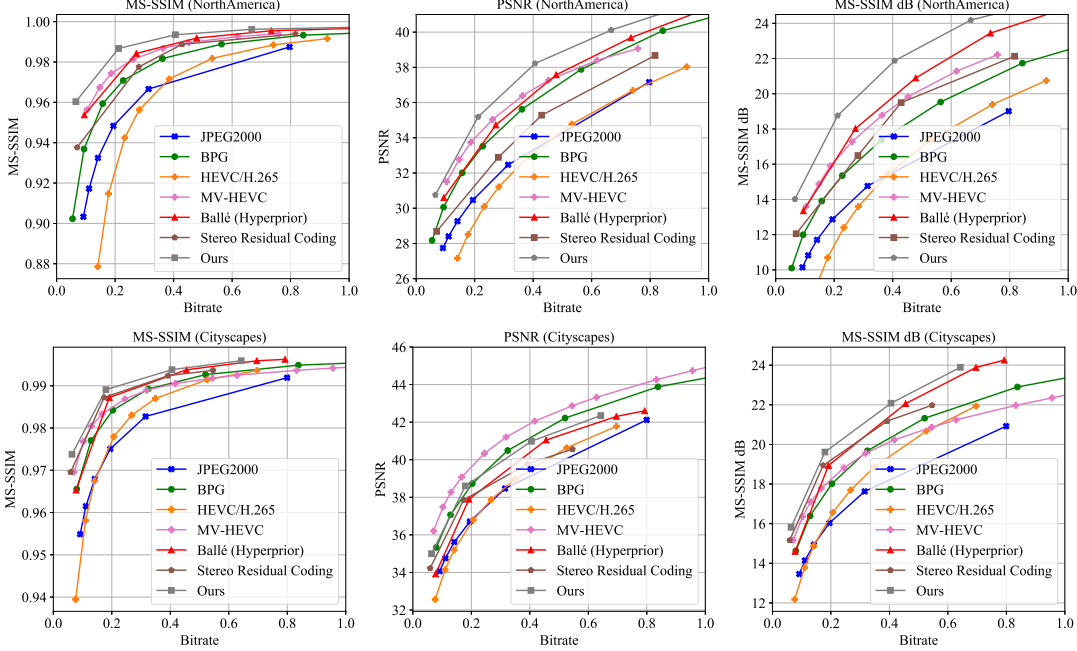
Figure 3: Plot of our stereo model against various compression baselines, for NorthAmerica and Cityscapes.

where $u(\tau) = 1$ if $|\tau| < 0.5$ otherwise $u(\tau) = 0$, and $*$ is the convolution between two functions. Thus we have:

$$
\begin{aligned}
p_i(\bar{z}_i; \boldsymbol{\theta}_{\bar{z}}) &= \int_{-\infty}^{\infty} q_i(\tau; \boldsymbol{\theta}_{\bar{z}}) u(\bar{z}_i - \tau) d\tau \\
&= \int_{\bar{z}_i - 0.5}^{\bar{z}_i + 0.5} q_i(\tau; \boldsymbol{\theta}_{\bar{z}}) d\tau \\
&= c_i(\bar{z}_i + 0.5; \boldsymbol{\theta}_{\bar{z}}) - c_i(\bar{z}_i - 0.5; \boldsymbol{\theta}_{\bar{z}})
\end{aligned} \tag{11}
$$

where $c_i(\bar{z}_i; \boldsymbol{\theta}_{\bar{z}})$ is the cumulative density function (CDF) of some underlying PDF q. This intuitively means that we can define $p_i(\bar{z}_i; \boldsymbol{\theta}_{\bar{z}})$ as an area under the CDF rather than directly as the PDF, and we can use a neural network to directly model $c_i(\bar{z}_i; \boldsymbol{\theta}_{\bar{z}})$. This approach has better capacity to model steep edge PDFs, since even for steep edged PDF, the CDF is still continuous.

We follow a similar approach to model the conditional factorized probabilities for $\bar{y}_1, \bar{y}_2$ - we first highlight the model for $\bar{y}_2$:

$$
p_{2,i}(\bar{y}_{2,i} | \bar{y}_1, \bar{z}_2; \boldsymbol{\theta}_{\bar{y}_2}) = (q_{2,i} * u)(\bar{y}_{2,i}) \tag{12}
$$

Unlike the hyperprior models, we model each individual pixel PDF $q_{2,i}$ as a Gaussian mixture model (GMM):

$$
q_{2,i}(\bar{y}_1, \bar{z}_2) = \sum_k w_{ik} \mathcal{N}(\mu_{ik}, \sigma_{ik}^2)) \tag{13}
$$

where $w_{ik}, \mu_{ik}, \sigma_{ik}^2$ are the distribution parameters depending on $\bar{y}_1, \bar{z}_2$ and $\boldsymbol{\theta}_{\bar{y}_2}$. We also rewrite the convolution as the difference between CDFs as in Eq. (11).

The CDF of a GMM is generally computed numerically in most computational frameworks, while the derivative is analytical. Thus we just need to learn a function that predicts parameters $w_{ik}, \mu_{ik}, \sigma_{ik}^2$ given $\bar{y}_1, \bar{z}_2$ with $\boldsymbol{\theta}_{\bar{y}_2}$ as learnable parameters, instead of modeling the CDF value directly as in the hyperprior entropy model. We found that a mixture model increased performance slightly thanks to its stronger capacity compared to a single Gaussian. Finally, the model for $\bar{y}_1$ follows the same GMM formulation; however given that $\bar{y}_1$ is decoded first, we can only provide $\bar{z}_1$ as input, not $\bar{y}_2$:

$$
p_{1,i}(\bar{y}_{1,i} | \bar{z}_1; \boldsymbol{\theta}_{\bar{y}_1}) = (q_{1,i} * u)(\bar{y}_{1,i}) \tag{14}
$$

Architecture details for our hyper-encoder - deriving $\bar{z}_1, \bar{z}_2$ from $\bar{y}_1, \bar{y}_2$ - as well as for each entropy model can be found in supplementary material (Sec C.3).

3.4. Learning

Our model is trained end-to-end to minimize the following objective function:

$$
\ell + \beta R = \mathbb{E}_{\boldsymbol{x}_1, \boldsymbol{x}_2 \sim p_{\boldsymbol{x}}} [\underbrace{||\boldsymbol{x}_1 - \hat{\boldsymbol{x}}_1||_2^2}_{\text{Distortion (Img. 1)}} + \underbrace{||\boldsymbol{x}_2 - \hat{\boldsymbol{x}}_2||_2^2}_{\text{Distortion (Img. 2)}}] +
$$
$$
\beta \mathbb{E}_{\boldsymbol{x}_1, \boldsymbol{x}_2 \sim p_{\boldsymbol{x}}} [\underbrace{-\log_2 p(\bar{y}_1, \bar{z}_1; \boldsymbol{\theta})}_{\text{Rate (Code 1)}} \underbrace{-\log_2 p(\bar{y}_2, \bar{z}_2 | \bar{y}_1; \boldsymbol{\theta})}_{\text{Rate (Code 2)}}]
$$
$$
\tag{15}
$$

where the first term encodes reconstruction quality of both images and the second term is the bitrate term with the rate predicted by the entropy model. Moreover, we can enforce

a target bitrate for a given model by modifying the rate function to be:

$$R = \max(\mathbb{E}_{\boldsymbol{x}_1, \boldsymbol{x}_2 \sim p_{\boldsymbol{x}}}[-\log_2 p(\bar{\boldsymbol{y}}_1, \bar{\boldsymbol{y}}_2, \bar{\boldsymbol{z}}_1, \bar{\boldsymbol{z}}_2; \boldsymbol{\theta})], H_t) \tag{16}$$

where H_t is our desired target entropy calculated from the target bitrate.

4. Experiments

We present a quantitative and qualitative evaluation of our approach and various baselines on two different datasets. We now provide more details about the datasets and metrics we employ.

4.1. Datasets, Metrics and Baselines

NorthAmerica: We created a dataset captured by driving self driving vehicles in two different cities in North America. This dataset consists of 100k pairs of lossless rectified stereo images. We use 2.5k pairs for validation, 18k for final testing, and the remaining for training. The images in this dataset are 480×300.

Cityscapes: We also train on Cityscapes raw sequences [10], consisting of 89k training pairs and 45k test pairs. For each 2048×1024 image, as a preprocessing step we crop 64 pixels from the top and 128 pixels from the left to remove rectification artifacts. We also crop out the bottom 240 pixels to remove the ego-vehicle in order to focus on the scene imagery.

Metrics: We report results on both peak signal-to-noise ratio (PSNR) and multi-scale structural similarity (MS-SSIM) [46] as a function of bitrate. Both MS-SSIM and PSNR are commonly used perceptual quality metrics, and we measure both to test the robustness of our model. PSNR is defined as $-10 \log_{10}(\text{MSE})$, where MSE is mean-squared error, and better measures the absolute error in a compressed image. On the other hand, MS-SSIM better measures how well overall structural information is preserved. For the MS-SSIM curve, we report both at original scale as well as log-scale, namely "MS-SSIM (dB)" from [5], defined as $-10 \log_{10}(1 - \text{MS-SSIM})$.

Baselines: Our completing algorithms include the single-image hyperprior Ballé model [5] as well as popular image codecs - BPG and JPEG2000. We also try adapting traditional video-compression techniques as additional baselines. Specifically, we run a codec based on the HEVC/H.265 standard [41] over the stereo pair; we also try a Multi-View Coding (MVC) extension of HEVC [1]. Another approach is to try a deep learning method to encode the first image, disparity map and disparity-warped residual image jointly (referred to as "stereo residual coding"): we

compress the first image using the Ballé hyperprior model. Then we use SGM on the stereo pair to generate disparity estimates, and compress them using a second Ballé model. Finally, we compress the disparity-compensated residual image using a third Ballé model.

4.2. Implementation Details

We create multiple stereo compression models, each set to a different desired target bitrate. We set β, the weight on the entropy loss, to a value within 0.5 to 0.001 for the lower to higher bitrate models respectively. For each model at a given target bitrate, we initialize the layers of both encoders and decoders with those from a pre-trained single-image based Ballé model [5] at the corresponding bitrate. This speeds up our training and convergence time significantly. We use a learning rate of $2 \cdot 10^{-4}$ for all models and optimize our parameters with Adam. We train with a total batch size of 4, across 4 GPUs. For NorthAmerica, we train on the full 480×300 image and set C (the maximum disparity) to 32, while for Cityscapes, we train on 384×240 crops and set $C = 64$.

4.3. Experimental Results

Comparison to Baselines: On NorthAmerica, our stereo model consisting of our skip functions and conditional entropy outperforms all of these compression baselines, as shown in Fig. 3. Note that the reported "bitrate" for our stereo models represents the *joint bitrate*, or the average of the bitrate for both images. Assuming our stereo pair is compressed as well under our model as under a single-image setting, this implies our *residual bitrate* savings, the bitrate savings for the second image, to be double our joint bitrate savings. For NorthAmerica, at bitrate 0.0946 for the Ballé model, our model achieves an average 29.0% reduction in the joint bitrate with higher MS-SSIM/PSNR, implying a 58.0% reduction in the residual. At a higher Ballé bitrate 0.734, our model achieves a 17.8% reduction in the residual with higher MS-SSIM/PSNR.

For Cityscapes, our model outperforms all competing baselines in MS-SSIM; it outperforms all baselines except BPG/MV-HEVC in PSNR. At equal PSNR/MS-SSIM, residual savings against Ballé range from 30% in the lower bitrates to 10% at higher bitrates.

Our deep residual coding baseline performs worse than single-image compression at all bitrates for NorthAmerica, and at higher bitrates for Cityscapes. Even though it numerically approaches the performance of our model at lower bitrates, there exist significant boundary/warping artifacts, which we demonstrate in supplementary material. The underperformance of our baseline is consistent with the findings of [49] and [14], the latter of whom noted that residual images exhibit different correlation properties than the full images and may need to be modeled differently.

Input JPEG2000 BPG Ballé Ours

Cityscapes (Cam 1) Bitrate: 0.0648, PSNR: 33.11 Bitrate: 0.0651, PSNR: 34.59 Bitrate: 0.0770, PSNR: 34.62 **Bitrate: 0.0982, PSNR: 36.23**

Cityscapes (Cam 2) Bitrate: 0.0643, PSNR: 32.71 Bitrate: 0.0649, PSNR: 34.38 Bitrate: 0.0792, PSNR: 34.16 **Bitrate: 0.0295, PSNR: 35.05**

NorthAmerica (Cam 1) Bitrate: 0.2825, PSNR: 30.64 Bitrate: 0.278, PSNR: 32.24 Bitrate: 0.319, PSNR: 33.17 **Bitrate: 0.321, PSNR: 33.71**

NorthAmerica (Cam 2) Bitrate: 0.2838, PSNR: 30.43 Bitrate: 0.281, PSNR: 32.14 Bitrate: 0.322, PSNR: 33.00 **Bitrate: 0.200, PSNR: 33.24**
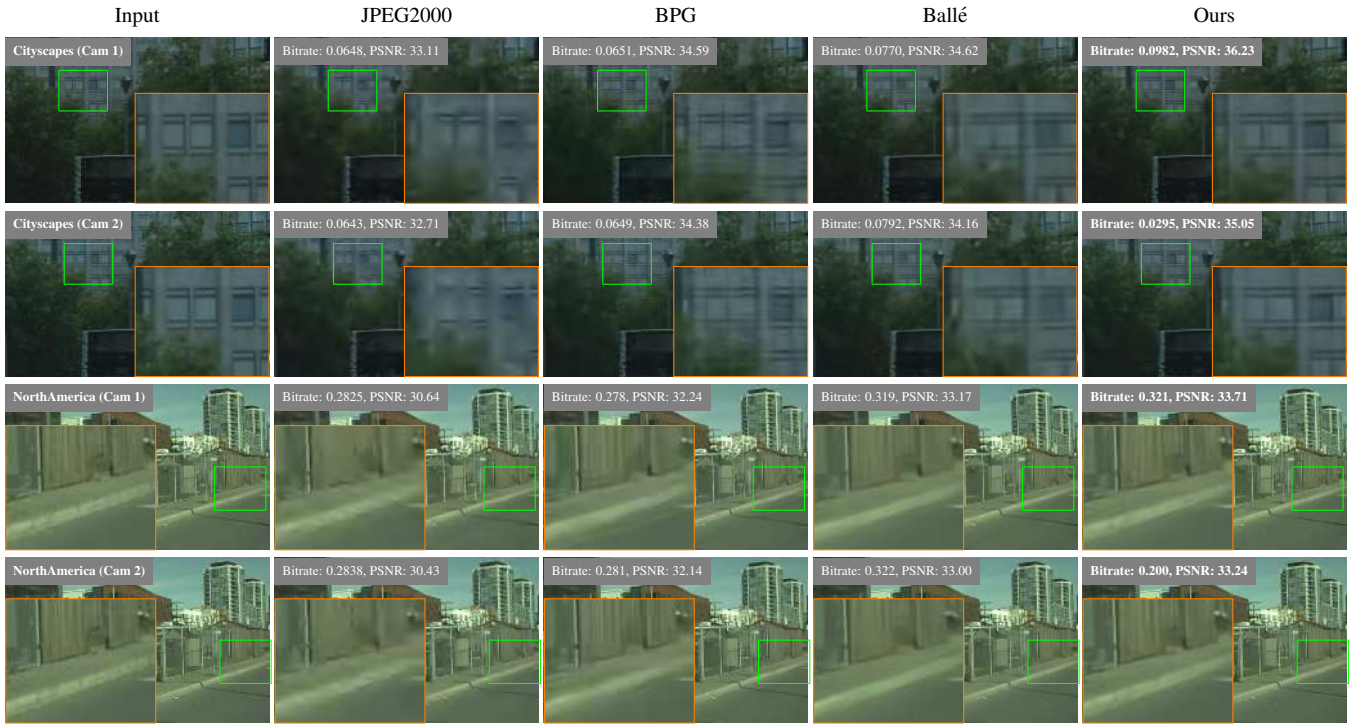
Figure 4: Qualitative results of our method vs. various single-image baselines. Metrics are specified for each entry.
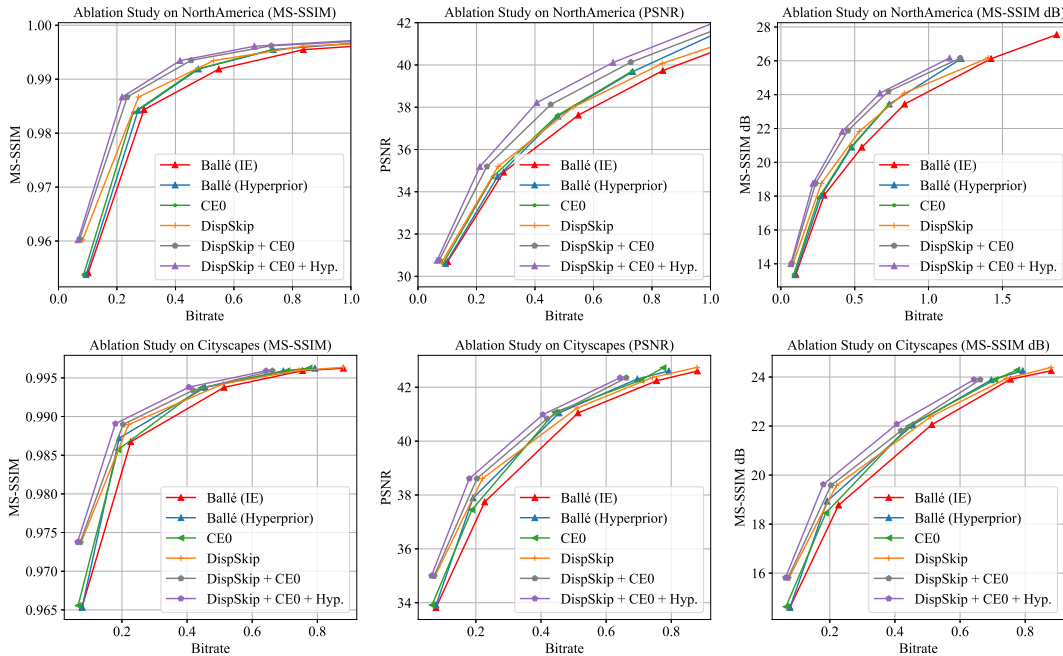
Figure 5: Ablation study. For both datasets, we analyze the independent and combined effects of our skip functions (DispSkip), conditional entropy without hyperprior (CE0), and hyperprior.

Ablation Study: We perform an ablation study in NorthAmerica and Cityscapes in Fig. 12 to isolate the impact of each component in our stereo model: the skip function (DispSkip), conditional entropy (separated from the hyperprior), and hyperprior. We start out with two fully factorized, independent single-image compression models (Ballé [5], denoted as IE), with no skip functions. We then analyze the impact of adding DispSkip on the IE

Resolution	960 x 300	480 x 300	240 x 150
Bitrate (Ours)	**0.361**	**0.406**	**0.437**
MS-SSIM (Ours)	**0.9936**	**0.9935**	**0.9915**
PSNR (Ours)	**40.12**	**38.22**	**35.61**
Bitrate (Ballé)	0.414	0.479	0.518
MS-SSIM (Ballé)	0.9927	0.9919	0.9905
PSNR (Ballé)	39.93	37.57	35.42

Table 1: Analysis of our stereo compression performance on different camera baseline widths/resolutions from NorthAmerica.

model to isolate the effect of DispSkip from our full conditional entropy model. Then, we define a stripped down conditional entropy model (denoted as CE0) that removes all hyperpriors: $\bar{y}_1$ is now modeled by a fully factorized entropy, and $\bar{y}_2$ is modeled by a GMM that only depends on $\bar{y}_1$. We train CE0, both with and without DispSkip, to analyze metrics when no hyperprior side information is used during encoding. Our final stereo model consists of DispSkip, CE0, and the hyperprior, and we showcase that curve as well.

As shown in Fig. 12, DispSkip on its own provides greater gains against the pure IE model at lower bitrates and converges to single-image compression at higher bitrates. In the meantime, CE0 consistently provides performance gains across all bitrates against IE; the improvement is roughly equivalent to that of fitting the Ballé hyperprior entropy model for both image codes. When combined with DispSkip (DispSkip + CE0), this model marginally outperforms the Ballé hyperprior model. Finally, DispSkip + CE0 + hyperprior (forming our full stereo model) provides the greatest metrics gains across all bitrates.

We observe some cannibalization of additional gains when combining DispSkip, CE0, and the hyperprior. The reduction in gains when we combine DispSkip + CE0 makes intuitive sense: our disparity-warped skip connections focus on reusing redundant information and hence reducing the correlation between the image codes, whereas the entropy of our CE0 model is lower when the correlation in both image codes is higher. Moreover, fitting hyperprior side information that can already help reduce the entropy of a single image may somewhat reduce the additional entropy reduction CE0 can provide.

Qualitative Results: A qualitative demonstration of our model on a stereo pair is given in Fig. 4. We show that our approach contains better overall visual quality at a lower bitrate compared to the Ballé model and other codecs. More specifically, our stereo model better preserves overall edges and colors without introducing artifacts. While BPG is competitive with our model on Cityscapes, we observe that BPG magnifies certain high frequency details while distorting lower frequency regions. We leave additional qualitative analysis to supplementary material.

	480 x 300 Res.	1920 x 720 Res.
Ballé	36 GFlops	345 GFlops
Ours	141 GFlops	2700 GFlops

Table 2: Analysis of FLOPs of our approach compared to Ballé.

Effect of Different Baseline Widths To more concretely analyze the impact of the baseline width on our stereo compression model, we recreate copies of NorthAmerica at different resolutions: 960×480, 480×300, and 240×150, with baseline widths of 0.175 m, 0.088 m, 0.044 m respectively. We train with $C = 64$ for the highest resolution and $C = 32$ for the others. As shown in Table 1, we achieve bitrate reductions while increasing perceptual metrics for all resolution levels in NorthAmerica.

Runtime: On a GTX 1080-Ti, our stereo model takes 130ms for a 480x300 NorthAmerica pair, and 2246 ms for a 1920x720 Cityscapes pair. Additionally FLOPS are shown in Tab. 2. Range coding is $\mathcal{O}(N)$ in encoding and $\mathcal{O}(N \log n)$ in decoding, where N is # symbols and n is # unique symbols. Our complexity is dominated by the computation of the cost volumes. We note that we can attempt sparse approximations of the cost volume, or different distributional parametrizations instead of a dense softmax to save compute/memory for future work.

Disparity Volume: To interpret the information learned in our DispSkip connections, Fig. 2 shows a visualization of our disparity volumes in the encoder/decoder at one particular bitrate level (bitrate 0.442), for a Cityscapes stereo pair. These visualizations are generated by taking the mode over the probability vector for each disparity dimension in each volume. The learned disparity maps capture different information at each level, helping to support our justification for predicting a separate disparity volume at each level of the encoder and decoder.

5. Conclusion

We propose a novel deep stereo image compression algorithm, which exploits the content redundancy between the stereo pair to reduce the joint bitrate. Towards this goal, we propose parameteric skip functions and a conditional entropy model to model the dependence between the images. We validate the effectiveness of our method over two large-scale datasets and demostrate that our stereo model reduces the bitrate in the second image by 10-50% from high to low bitrates, compared to a single-image deep compression model. Additionally, we demonstrate that both our skip functions and conditional entropy contribute meaningfully to improving the bitrate and perceptual quality. In the future, we plan to extend our approach to the multi-view image and video compression settings.

References

[1] Mv-hevc implementation. Available at https://github.com/listenlink/3D-HEVC.

[2] Eirikur Agustsson, Fabian Mentzer, Michael Tschannen, Lukas Cavigelli, Radu Timofte, Luca Benini, and Luc V Gool. Soft-to-hard vector quantization for end-to-end learning compressible representations. In *NeurIPS*, 2017.

[3] Haluk Aydinoglu and Monson H. Hayes. Stereo image coding: a projection approach. *IEEE Transactions on Image Processing*, 1998.

[4] Johannes Ballé, Valero Laparra, and Eero P. Simoncelli. End-to-end optimized image compression. In *ICLR*, 2017.

[5] Johannes Ballé, David Minnen, Saurabh Singh, Sung Jin Hwang, and Nick Johnston. Variational image compression with a scale hyperprior. In *ICLR*, 2018.

[6] Johannes Ballé, Valero Laparra, and Eero P. Simoncelli. Density modeling of images using a generalized normalization transformation. In *ArXiv*, 2015.

[7] Connelly Barnes, Eli Shechtman, Adam Finkelstein, and Dan B Goldman. PatchMatch: A randomized correspondence algorithm for structural image editing. In *SIGGRAPH*, 2009.

[8] Stan Birchfield and Carlo Tomasi. Depth discontinuities by pixel-to-pixel stereo. In *ICCV*, 1998.

[9] Jia-Ren Chang and Yong-Sheng Chen. Pyramid stereo matching network. In *CVPR*, 2018.

[10] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *CVPR*, 2016.

[11] Markus Flierl and Bernd Girod. Multiview video compression. *IEEE Signal Processing Magazine*, 2007.

[12] Markus Flierl, Aditya Mavlankar, and Bernd Girod. Motion and disparity compensated coding for multiview video. *IEEE Transactions on Circuits and Systems for Video Technology*, 2007.

[13] John Flynn, Ivan Neulander, James Philbin, and Noah Snavely. Deepstereo: Learning to predict new views from the world's imagery. In *CVPR*, 2016.

[14] Tamas Frajka and Kenneth Zeger. Residual image coding for stereo image compression. In *ICIP*, 2002.

[15] Jun Han, Salvator Lombardo, Christopher Schroers, and Stephan Mandt. Deep probabilistic video compression. In *ArXiv*, 2018.

[16] Marsha Jo Hannah. *Computer Matching of Areas in Stereo Images*. PhD thesis, Stanford, CA, USA, 1974.

[17] Heiko Hirschmuller. Stereo processing by semiglobal matching and mutual information. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2008.

[18] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *ICML*, 2015.

[19] Max Jaderberg, Karen Simonyan, Andrew Zisserman, and Koray Kavukcuoglu. Spatial transformer networks. In *NIPS*, 2015.

[20] Chuanmin Jia, Xinfeng Zhang, Shanshe Wang, Shiqi Wang, Shiliang Pu, and Siwei Ma. Light field image compression using generative adversarial network based view synthesis. *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, 2018.

[21] Xiaoran Jiang, Mikael Le Pendu, and Christine Guillemot. Light field compression using depth image based view synthesis. In *ICMEW*, 2017.

[22] Alex Kendall, Hayk Martirosyan, Saumitro Dasgupta, Peter Henry, Ryan Kennedy, Abraham Bachrach, and Adam Bry. End-to-end learning of geometry and context for deep stereo regression. In *ICCV*, 2017.

[23] Masaki Kitahara, Hideaki Kimata, Shinya Shimizu, Kazuto Kamikura, Yoshiyuki Yashima, Kenji Yamamoto, Tomohiro Yendo, Toshiaki Fujii, and Masayuki Tanimoto. Multi-view video coding using view interpolation and reference picture selection. *ICME*, 2006.

[24] Jooyoung Lee, Seunghyun Cho, and Seung-Kwon Beack. Context-adaptive entropy model for end-to-end optimized image compression. In *ICLR*, 2019.

[25] Haojie Liu, Tong Chen, Ming Lu, Qiu Shen, and Zhan Ma. Neural Video Compression using Spatio-Temporal Priors. In *ArXiv*, 2019.

[26] Jean loup Gailly and Mark Adler. zlib. https://www.zlib.net/, 1995–2017.

[27] Michael E. Lukacs. Predictive coding of multi-viewpoint image sets. In *ICASSP*, 1986.

[28] Wenjie Luo, Alexander G. Schwing, and Raquel Urtasun. Efficient deep learning for stereo matching. *CVPR*, 2016.

[29] Gloria Martín. Range encoding: an algorithm for removing redundancy from a digitised message. In *Video and Data Recording Conference*, 1979.

[30] Emin Martinian, Alexander Behrens, Jun Xin, and Anthony Vetro. View synthesis for multiview video compression. In *Picture Coding Symposium*, 2006.

[31] Fabian Mentzer, Eirikur Agustsson, Michael Tschannen, Radu Timofte, and Luc Van Gool. Conditional probability models for deep image compression. In *CVPR*, 2018.

[32] Philipp Merkle, Karsten Muller, Aljoscha Smolic, and Thomas Wiegand. Efficient compression of multi-view video exploiting inter-view dependencies based on h.264/mpeg4-avc. In *ICME*, 2006.

[33] David Minnen, Johannes Ballé, and George Toderici. Joint autoregressive and hierarchical priors for learned image compression. In *NIPS*, 2018.

[34] Mark S. Moellenhoff and Mark W. Maier. Transform coding of stereo image residuals. *IEEE Transactions on Image Processing*, 1998.

[35] Oren Rippel and Lubomir Bourdev. Real-time adaptive image compression. In *ICML*, 2017.

[36] Oren Rippel, Sanjay Nair, Carissa Lew, Steve Branson, Alexander G. Anderson, and Lubomir Bourdev. Learned video compression. In *ArXiv*, 2018.

[37] Thomas W. Ryan, R.T. Gray, and Bobby R. Hunt. Prediction of correlation errors in stereo pair images. *Optical Engineering*, 1980.

[38] Daniel Scharstein and Chris Pal. Learning conditional random fields for stereo. In *CVPR*, 2007.

[39] Claude E. Shannon. A mathematical theory of communication. *Bell System Technical Journal*, 1948.

[40] Mel Siegel, Priyan Gunatilake, Sriram Sethuraman, and Angel G. Jordan. Compression of stereo image pairs and streams. In *Stereoscopic Displays and Virtual Reality Systems*, 1994.

[41] 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*, 2012.

[42] Katsumi Tahara. Stereoscopic coding system, 1997. US Patent 5,633,682.

[43] Lucas Theis, Wenzhe Shi, Andrew Cunningham, and Ferenc Huszar. Lossy image compression with compressive autoencoders. In *ICLR*, 2017.

[44] George Toderici, Sean M. O'Malley, Sung Jin Hwang, and Damien Vincent. Variable rate image compression with recurrent neural networks. In *ICLR*, 2016.

[45] 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 *CVPR*, 2017.

[46] Zhou Wang, Eero P. Simoncelli, and Alan C. Bovik. Multiscale structural similarity for image quality assessment. In *ACSSC*, 2003.

[47] 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*, 2003.

[48] Antonio Ortega Woontack Woo. Stereo image compression with disparity compensation using the mrf model. 1996.

[49] Chao-Yuan Wu, Nayan Singhal, and Philipp Krahenbuhl. Video compression through image interpolation. In *ECCV*, 2018.

[50] Yuxin Wu and Kaiming He. Group normalization. In *ECCV*, 2018.

[51] Junyuan Xie, Ross Girshick, and Ali Farhadi. Deep3d: Fully automatic 2d-to-3d video conversion with deep convolutional neural networks. In *ECCV*, 2016.

[52] Jure Zbontar and Yann LeCun. Stereo matching by training a convolutional neural network to compare image patches. *JMLR*, 2016.

[53] Yiran Zhong, Yuchao Dai, and Hongdong Li. Self-supervised learning for stereo matching with self-improving ability. In *CVPR*, 2017.