Supplementary Material:
Video Compression through Image Interpolation

Chao-Yuan Wu, Nayan Singhal, and Philipp Krähenbühl

The University of Texas at Austin
{cywu, nayans, philkr}@cs.utexas.edu

1 Model Details

Context feature fusion. In experiments, we found that fusing the U-net features at resolution $W_2 \times H_2$ yields good performance for all models. Fusing features at more resolutions improves the performance slightly, but requires more memory and computation. In this paper, we additionally use $W_4 \times H_4$ features for the encoder of $M_{1,2}$ and $M_{3,3}$, and $W_4 \times H_4$ and $W_8 \times H_8$ features for the decoder of $M_{3,3}$. Models are selected based on the performance on validation set.

Probability estimation in AAC. Adaptive arithmetic coding (AAC) relies on a good probability estimation for each bit, given previously decoded bits, to efficiently encode a binary representation. To estimate the probability, we follow Mentzer et al. and use a 3D-Pixel-CNN model [3, 4]. The model contains 11 layers of masked convolution. Each layer has 128 channels, and is followed by batch normalization [1] and relu. We train the models using Adam [2] with a learning rate of 0.0001 for 30K iterations.

2 Bitrate Optimization

We present detailed results of bitrate optimization. Figure 1a shows the explored performance at the first level of the hierarchy. We pick good combinations from the envelope of the curves, and they proceed to the next level. Figure 1b presents the final combinations. We use these combinations for the experiments in our paper unless otherwise noted.

3 Computational Complexity and Memory Consumption

With 352 $\times$ 288 videos and batch size 1, encoding/decoding takes 20.4/13.0ms per iteration, plus 13.7ms for the context model on a TITAN Xp GPU. This is of the same order as the highly optimized H.264, with takes 4.0/1.4ms for encoding/decoding on a Intel Xeon E5-2630 v3 CPU with FFmpeg implementation. As for memory, our model consumes 506MB at inference time.
4 Full System

Figure 2 illustrates how the full system compresses/decompresses a video step by step. Here we illustrate a 3-level model as an example. (The main model in paper is a 4-level model.)

We first compress the key frames using the I-frame (image compression) model (Figure 2a). Second, we use the decompressed key frames as context and interpolate the frame in next level of hierarchy, i.e. the middle frame (Figure 2b). Finally, we again use the decompressed frames as context, and interpolate the frames in the final level of hierarchy, i.e. the remaining frames (Figure 2c).

References

Fig. 2: Full compression/decompression procedure for a 3-level model. Blue arrows represent motion compensated context features. Gray arrows represent input and output of the network.