A. Appendix

A.1. Commands for standard codecs

The following commands are used to obtain compression results for standard codecs FFmpeg (x264 and x265) and HM. For FFmpeg, we disable B-frames and use default settings otherwise. We use HM-16.25 with default settings using the LowDelay-P config, for more details see https://vcgit.hhi.fraunhofer.de/jvet/HM/-/tags/HM-16.25.

```
# ffmpeg x264
ffmpeg -y -f rawvideo \
  -pix_fmt yuv420p \
  -s:v <width>x<height> \
  -i <input.yuv> \
  -r <framerate> \
  -c:v libx264 \
  -preset <preset> \
  -crf <crf> \
  -x264-params bframes=0 \
  <output>

# ffmpeg x265
ffmpeg -y -f rawvideo \
  -pix_fmt yuv420p \
  -s:v <width>x<height> \
  -i <input.yuv> \
  -r <framerate> \
  -c:v libx265 \
  -preset <preset> \
  -crf <crf> \
  -x265-params bframes=0 \
  <output>

# HM-16.25 LowDelayP
./bin/TAppEncoderStatic -c \
  ./cfg/encoder_lowdelay_P_main.cfg \
  -i <input.yuv> \
  --InputBitDepth=8 \
  -wdt <width> \
  -hgt <height> \
  -fr <framerate> \
  -f <numframes> \
  -q <qp> \
  -o <output>
```

A.2. Source Data

Per-video and per-color channel benchmark results are included in a csv file in the Supplementary Materials, examples of videos decoded with our codec can be viewed at https://www.youtube.com/watch?v=jXH6utaZirU.

A.3. Additional Results

Additional results are shown on the following pages. Tab. 5 lists the hyperparameters used in the various training stages of our model. The full model architecture is detailed in Figure 6. Figure 8 shows the RD performance for the models discussed in the model and quantization ablation in the main text of our paper and Figure 5 details the pipeline for our flow-agnostic model in this ablation. Finally, Figure 7 shows the benchmark of our model and various baselines on the UVG and MCL-JVC datasets.

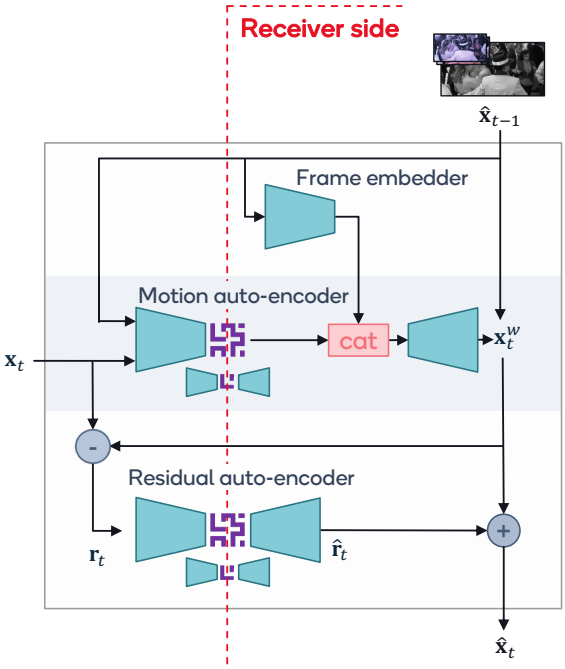

Figure 5. Model architecture of our flow-agnostic model. [3]

[3] Image data from Tango video from Netflix Tango in Netflix El Fuente. Video produced by Netflix, with CC BY-NC-ND 4.0 license: https://media.xiph.org/video/derf/ElFuente/Netflix_Tango_Copyright.txt

		Stage 1 Training	Stage 2 finetuning	Stage 3 PTQ	Stage 4 QAT
Data Size	batchsize	8	16	2	16
	gop	4	7	3	4
	crop size	256x256	256x384	256x256	256x256
Loss Multipliers	β I-frame	β	β	β	β
	β P-frame	2β	2β	-	2β
	P-frame loss modulation	$\tau = 1$ (no modulation)	$\tau = 1.2$	-	$\tau = 1.2$
	predicted flow $\mathbf{f}^P$	$\lambda = 0.1$	$\lambda = 0$	-	$\lambda = 0$
	reconstructed flow $\hat{\mathbf{f}}$	$\lambda = 0.1$	$\lambda = 0$	-	$\lambda = 0$
Optim	lr	1e-04	5e-05	-	5e-07
	lr schedule	-	-	-	cosine decay to 1e-9
Quantization	datatype	float32	float32	int8	int8 (STE)
Training Time	steps	1M	250k	30	100k
	walltime	$\sim$ 4 days	$\sim$ 3 days	$\sim$ 2 minutes	$\sim$ 1 day

Table 5. Different training stages and their corresponding hyperparameters.
We train a model for each value of $\beta \in \{0.0001, 0.0002, 0.0004, 0.0008, 0.0016, 0.0032, 0.0064\}$

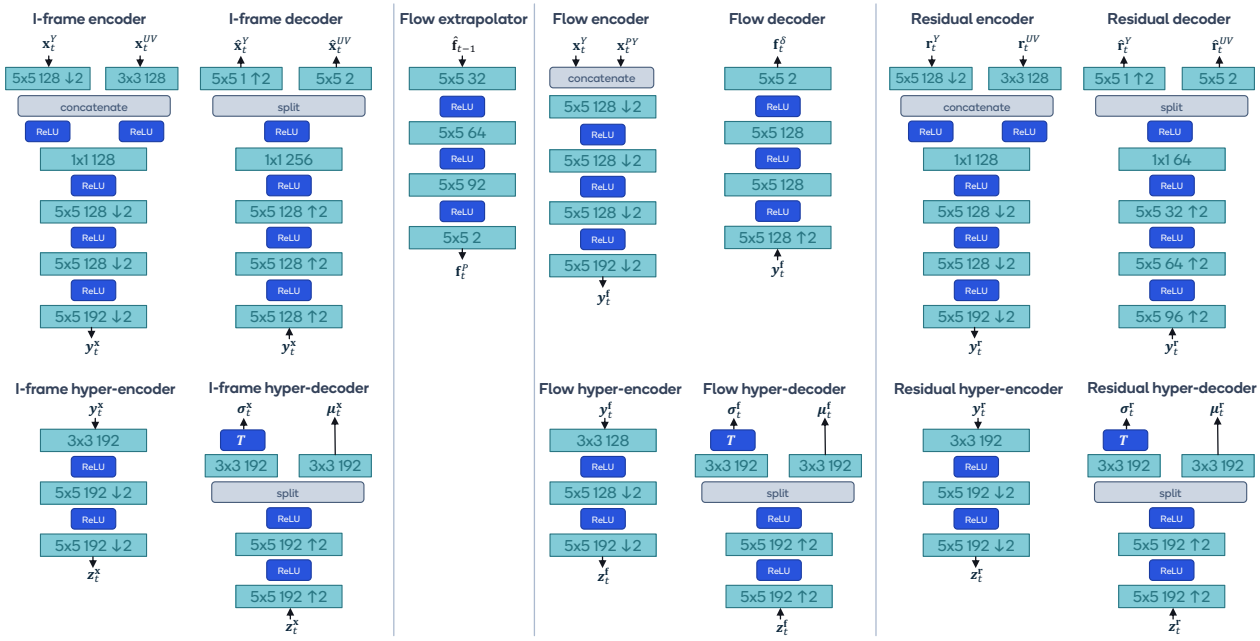

Figure 6. Model architecture for the neural networks inside of our P-frame model. Convolutional layers are displayed as $k \times k$ c where k refers to kernel size and c refers to the number of output channels. Convolutions with stride s are indicated by $\downarrow$ s and transposed convolutions with stride s are shown as $\uparrow$ s.

		Parameters [M, ↓]				kMACs/px [↓]			
		MobileNVC	**MobileCodec**	**SSF**	**SSF-Pred**	**MobileNVC**	**MobileCodec**	**SSF**	**SSF-Pred**
Sender	**I-frame AE**	5.66	6.68	9.47	9.47	116.11	211.60	118.01	118.01
	Motion pred.	0.21	-	-	0.75	1.66	-	-	6.44
	Motion AE	5.59	11.63	9.48	10.10	28.34	175.60	118.70	124.14
	Residual pred.	-	-	-	0.75	-	-	-	6.32
	Residual AE	6.82	6.57	10.09	10.09	36.59	183.60	123.45	123.45
	Pframe total	12.42	18.20	19.57	21.69	64.93	359.20	242.15	260.35
Receiver	**I-frame AE**	2.94	2.94	5.82	5.82	93.39	130.90	94.64	94.64
	Motion pred.	0.21	-	-	0.75	1.66	-	-	6.44
	Motion AE	2.91	5.98	5.83	6.45	9.5	156.20	94.64	100.08
	Residual pred.	-	-	-	0.75	-	-	-	6.32
	Residual AE	3.18	2.75	6.44	6.44	13.36	102.90	100.08	100.08
	Pframe total	6.30	8.65	12.27	14.39	24.52	259.10	194.72	212.92

Table 6. Model complexity per subnetwork for 1080 × 1920 YUV420 input. AE refers to (hypper-prior) autoencoder components and pred. refers to predictor models. Models are: MobileNVC (ours), MobileCodec [21], SSF [1], and SSF-Pred [33].

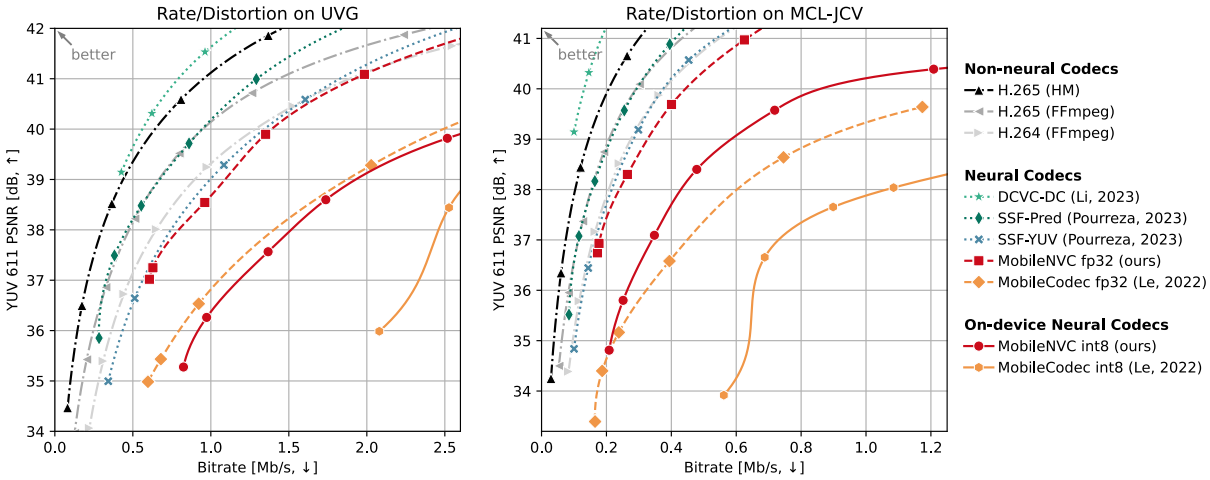

Figure 7. Rate-distortion performance of all models on UVG and MCL-JCV.

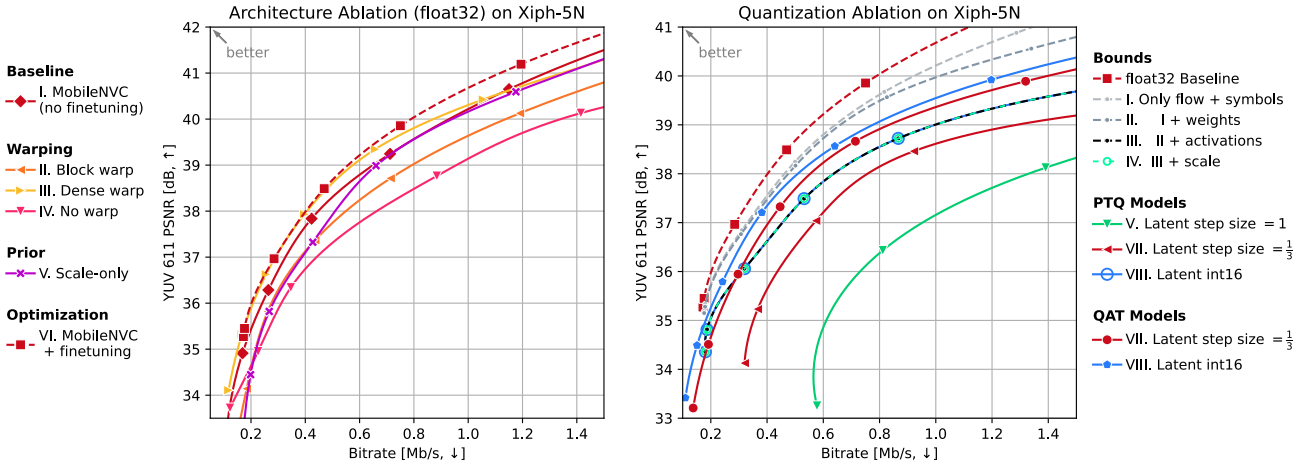

Figure 8. Model ablation (Left) and Quantization Ablation (Right). The models in these plots are described in more detail in Tab. 3 and 4 respectively.

A.4. Warping Samples

The effect of the different warping strategies is shown in Table 7. In this experiment, we compress a single P-frame as usual, but instead of conditioning the model on the previously reconstructed frame $\hat{\mathbf{x}}_{t-1}$ we use the previous groundtruth frame $\mathbf{x}_{t-1}$. Doing so allows us focus on the differences in warping only.

The frame warped with dense warping (III) does not show any clear artifacts. When we use block-warp (II) instead, we see discontinuities where the edges of the object to warp do not align with the blocks (i.e. notice the "gaps" in the yellow mast pole). We see that for the Block-Overlap Warp (I) these artifacts have disappeared. Finally, when we do not use warping but deploy a conditional model (IV) instead, the predicted frame becomes a lot less crisp.

Note that a checkerboard-like pattern can be seen in the flow for our Block-Overlap warping model. This pattern arises as the network learns to exploit the merging of neighboring blocks in blending kernel for a better final result.

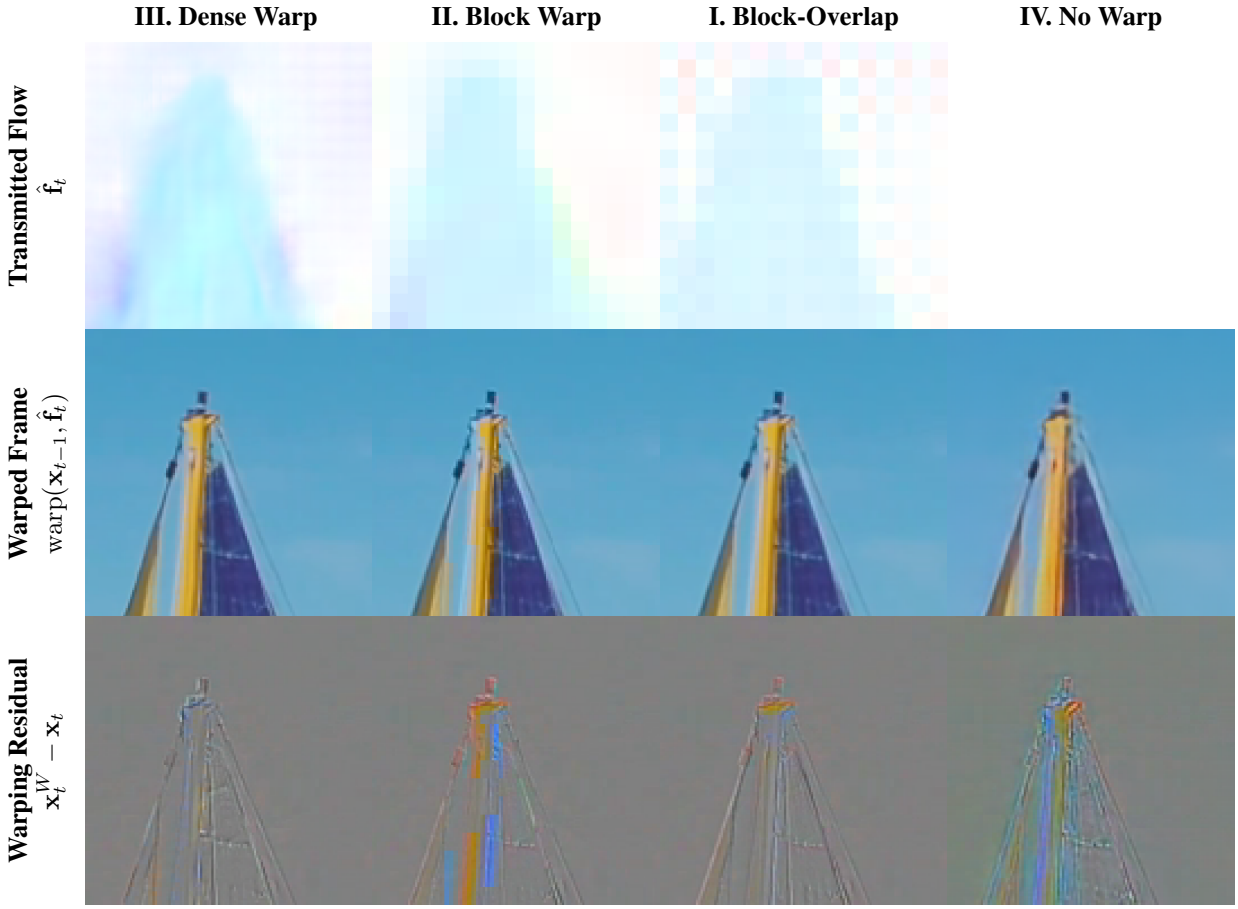

Table 7. Visualization of the output of different warping strategies. Numerals respond to the models in Table 3. Datapoint obtained from https://www.pexels.com/video/sky-blue-boat-sailing-4602958. Crop location: