PhaseNet for Video Frame Interpolation
Supplementary Material

Simone Meyer[1,2] Abdelaziz Djelouah[2] Brian McWilliams[2] Alexander Sorkine-Hornung[2*]
Markus Gross[1,2] Christopher Schroers[2]

[1]Department of Computer Science, ETH Zurich [2]Disney Research

simone.meyer@inf.ethz.ch aziz.djelouah@disneyresearch.com

1. Error Measurements

In Table 1 we report the peak signal-to-noise ration (PSNR) in addition to the SSIM error reported in the main paper. Example input images from the sequences used to compute these error measurements are shown in Figure 1.

2. Details Network Architecture

The architecture of the PhaseNet consists of consecutive PhaseNet blocks. Figure 4 in the paper visualizes the concept of such a block. In Table 2 the specific details for each layer can be found. Each block consists of two convolution layers both followed by batch normalization and leaky ReLU nonlinearity with factor 0.2. The prediction layers pred_i consists of one convolution layer followed by the hyperbolic tangent function. pyr_i summarizes the steerable pyramid decomposition information of the input images at the corresponding level i, i.e. low level residuals for $i = 0$ and phase and amplitude information for $i > 0$. To increase the resolution of the feature maps we use bilinear upscaling noted as *up()* in Table 2. Due to reusing the weights across the color channels and some of the layers, our network has only about 460k trainable parameters in total.

	Method				
	MDP-Flow2	Brox *et al.*	SepConv	Phase	Ours
Barrier	42.33	42.67	40.97	39.93	35.17
Couple	41.02	40.97	41.26	40.26	38.25
Face	40.94	40.89	40.60	40.28	40.31
Hair	37.15	37.69	37.00	36.86	36.34
Handk.	42.32	42.43	38.26	41.74	38.68
Sand	37.43	37.48	36.93	37.17	36.67
Roto	35.39	35.54	35.88	33.73	25.50
Firemen	40.26	40.21	42.54	33.80	34.78
Light	45.82	45.63	45.50	45.63	45.33

Table 1: **Error measurements** of different methods for the different sequences shown in Figure 1 by computing the PSNR (higher is better).

*Alexander Sorkine-Hornung is now at Oculus. He contributed to this work during his time at Disney Research.

3. Details Model Training

Our training dataset consists of about 10k triplets of frames from the DAVIS video dataset [3, 2]. At each iteration we randomly select patches of 256×256 pixels. We perform data augmentation through horizontal and vertical flipping of the patches.

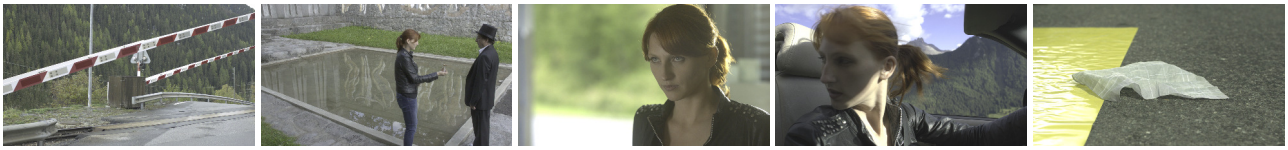
We use Adam optimizer [1] with $\beta_1 = 0.9$, $\beta_2 = 0.999$ and learning rate 0.001. The batch size used is 32 and reduced to 16 and 12, respectively, for the highest two training stages due to memory limitations. We train the lower levels for 12 epochs each and the highest two for 6 epochs due to the reduced batch size to have approximately the same number of iteration steps for each hierarchical level.

References

[1] D. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 1

[2] F. Perazzi, J. Pont-Tuset, B. McWilliams, L. Van Gool, M. Gross, and A. Sorkine-Hornung. A benchmark dataset and evaluation methodology for video object segmentation. In *Computer Vision and Pattern Recognition*, 2016. 1

[3] J. Pont-Tuset, F. Perazzi, S. Caelles, P. Arbeláez, A. Sorkine-Hornung, and L. Van Gool. The 2017 davis challenge on video object segmentation. *arXiv:1704.00675*, 2017. 1



(a) Barrier (b) Couple (c) Face (d) Hair (e) Handkerchief

(f) Sand (g) Firemen (h) Light (i) Roto

Figure 1: Example images from the sequences used for the error measurements.

Name	Input	Kernel	Ch In/Out	Res	Reuse Weights
PhaseNetBlock_0	pyr_0	1×1	$2/64$	8×8	False
pred_0	PhaseNetBlock_0	1×1	$64/1$	8×8	False
PhaseNetBlock_1	up(PhaseNetBlock_0)+up(pred_0)+pyr_1	1×1	$(64+1+16)/64$	12×12	False
pred_1	PhaseNetBlock_1	1×1	$64/8$	12×12	False
PhaseNetBlock_2	up(PhaseNetBlock_1)+up(pred_1)+pyr_2	1×1	$(64+8+16)/64$	16×16	False
pred_2	PhaseNetBlock_2	1×1	$64/8$	16×16	False
PhaseNetBlock_3	up(PhaseNetBlock_2)+up(pred_2)+pyr_3	3×3	$(64+8+16)/64$	22×22	False
pred_3	PhaseNetBlock_3	1×1	$64/8$	22×22	False
PhaseNetBlock_4	up(PhaseNetBlock_3)+up(pred_3)+pyr_4	3×3	$(64+8+16)/64$	32×32	False
pred_4	PhaseNetBlock_4	1×1	$64/8$	32×32	False
PhaseNetBlock_5	up(PhaseNetBlock_4)+up(pred_4)+pyr_5	3×3	$(64+8+16)/64$	46×46	False
pred_5	PhaseNetBlock_5	1×1	$64/8$	46×46	False
PhaseNetBlock_6	up(PhaseNetBlock_5)+up(pred_5)+pyr_6	3×3	$(64+8+16)/64$	64×64	False
pred_6	PhaseNetBlock_6	1×1	$64/8$	64×64	False
PhaseNetBlock_7	up(PhaseNetBlock_6)+up(pred_6)+pyr_7	3×3	$(64+8+16)/64$	90×90	False
pred_7	PhaseNetBlock_7	1×1	$64/8$	90×90	False
PhaseNetBlock_8	up(PhaseNetBlock_7)+up(pred_7)+pyr_8	3×3	$(64+8+16)/64$	128×128	True
pred_8	PhaseNetBlock_8	1×1	$64/8$	128×128	True
PhaseNetBlock_9	up(PhaseNetBlock_8)+up(pred_8)+pyr_9	3×3	$(64+8+16)/64$	182×182	True
pred_9	PhaseNetBlock_9	1×1	$64/8$	182×182	True
PhaseNetBlock_10	up(PhaseNetBlock_9)+up(pred_9)+pyr_10	3×3	$(64+8+16)/64$	256×256	True
pred_10	PhaseNetBlock_10	1×1	$64/8$	256×256	True

Table 2: **Details of the PhaseNet architecture.** The numbers of the channels and resolutions correspond to the case of using one color channel (weights reused for the other two) and a pyramid constructed with $\lambda = \sqrt{2}$ and 4 orientations. The $+$ in the input column corresponds to concatenating the channels. In total the network has about 460k trainable parameters.