DynaVSR: Dynamic Adaptive Blind Video Super-Resolution Supplementary Materials

Suyoung Lee* Myungsub Choi* Kyoung Mu Lee
ASRI, Department of ECE, Seoul National University
{esw0116, cms6539, kyoungmu}@snu.ac.kr

A. Implementation Details

The number of input frames and the loss function in DynaVSR follow the same form as the original VSR method: 5 frames, Charbonnier loss [8] for EDVR, 7 frames with Huber loss for DUF and 7 frames, ℓ_1 loss for TOFlow. Since MFDN is fully convolutional, the pre-trained MFDN network can be used regardless of the number of input frames. We use Adam [7] optimizer for both the inner and outer loop updates, with the corresponding learning rates of $\alpha = \beta = 10^{-5}$. Training requires 30,000 iterations with a mini-batch size of 4, and β is decayed by a factor of 5 at iterations 20,000 and 25,000, while α is kept fixed. We use full images for Vimeo-90K training, but crop the LR patches of size 128×128 for REDS dataset due to GPU memory limits. Starting from a pretrained VSR baseline, the full meta-training of DynaVSR framework takes approximately 15 hours with a single NVIDIA Geforce RTX 2080 Ti GPU, which includes the 6-hour pretraining step for initializing MFDN parameters.

B. Additional Quantitative Results

We provide the SSIM values of Table 3 of the main paper in Table A. Similar to the results for PSNR, EDVR+DynaVSR performs the second best in isotropic Gaussians, and reaches the best in anisotropic and mixed Gaussians. For all tables in this document, **red** denotes the best performance, and **blue** denotes the second best.

Table B, C, D, and E provide the detailed quantitative results for all kernel settings that we used to report the average values. Specifically, Table B and C show the performance for all 9 σ-s for the isotropic Gaussian kernels. Likewise, Table D and E show the performance of all 4 different angles θ for the anisotropic Gaussian kernels.

Table F and G are the results of the 10% of the validation set, which was used to evaluate Correlation-Filter-based models due to its high computational complexity. These data provides a fair comparison between the two models using Correction Filter (CF) algorithm, and the other methods in Table 1 of the main paper. We can observe that the quantitative values are almost same as full evaluation results except for the slight mismatch in the mixed setting, which was originally composed by random sampling of the downscaling kernels that are different for each video sequence.

B.1. Effect of Inner Loop Learning Rate (α)

The inner loop learning rate α is one of the most import hyperparameters that control the stability of training DynaVSR framework. If α is too big, the model parameters can jump to a bad local optimum during the inner loop update, resulting in diminishing performance. On the other hand, if α is too small, the possible performance gain with DynaVSR algorithm may not be fully exploited. We vary α to $10^{-4}, 10^{-5}$, and 10^{-6}, and report the effects in Table H. We found that $\alpha = 10^{-5}$ gives the best overall results and used this value for all experiments.

B.2. Larger Scale Factor ($\times 4$ SR)

Finally, Table I shows the $\times 4$ SR results using EDVR. For larger scaling factor, the size of SLR image will get smaller, and the adaptation using SLR-LR pair becomes more difficult. However, DynaVSR still makes substantial performance improvements in both dataset.

C. Additional Qualitative Results

For extensive visual comparison both for synthetic and real video examples, please refer to our supplementary slides. We also demonstrate the effectiveness of the proposed DynaVSR framework with the attached video demo. Note that, due to the long running time required for KernelGAN [2] and Correction Filter [5] algorithms, making a full video is too inefficient. Therefore, DynaVSR is mainly compared with IKC [3] in our video demo. The video demo can be downloaded by the following URL: https://bit.ly/3pisuGj.

*indicates equal contribution.

Table A: **SSIM results for meta-training with recent VSR models and blind SISR methods**. We evaluate the benefits of DynaVSR algorithm on Vid4 [9] and REDS-val [10] dataset. Red denotes the best performance, and blue denotes the second best.

	Method		Vid4 [9]			REDS-val [10]		
			Iso.	Aniso.	Mixed	Iso.	Aniso.	Mixed
Blind SISR	KG [2] + ZSSR [11]		0.8395	0.8075	0.7495	0.8673	0.8366	0.7721
	CF [5] + DBPN [4]*		0.8675	0.8353	0.7602	0.8741	0.8696	0.8202
	CF [5] + CARN [1]*		0.8826	0.8731	0.8388	0.8855	0.8973	0.8788
	IKC [3]		**0.9125**	0.8308	0.8650	**0.9361**	0.8821	0.9030
Video SR	EDVR [12]	Baseline	0.7846	0.8227	0.8387	0.8470	0.8729	0.8826
		DynaVSR	**0.9031**	**0.8946**	**0.9042**	**0.9286**	**0.9363**	**0.9388**
	DUF [6]	Baseline	0.7815	0.8154	0.8376	0.8426	0.8623	0.8759
		DynaVSR	0.8600	**0.8756**	**0.8812**	0.8935	**0.9023**	**0.9041**
	TOFlow [13]	Baseline	0.7815	0.8132	0.8366	0.8431	0.8648	0.8808
		DynaVSR	0.8509	0.8601	0.8735	0.9031	**0.9087**	**0.9126**

*Since CF takes a long time, we report the performance for random 10% of the validation set.

Table B: **Quantitative results of DynaVSR combined with recent VSR algorithms in isotropic Gaussian kernels on Vid4**. We evaluate the benefits of DynaVSR algorithm on Vid4 [9] dataset.

PSNR

Method	$\sigma = 0.8$	$\sigma = 0.9$	$\sigma = 1.0$	$\sigma = 1.1$	$\sigma = 1.2$	$\sigma = 1.3$	$\sigma = 1.4$	$\sigma = 1.5$	$\sigma = 1.6$	Average
KG + ZSSR	23.70	24.39	25.56	26.21	26.67	27.00	27.03	26.65	26.06	25.92
CF + DBPN	25.62	26.92	27.94	27.81	28.00	27.86	27.65	27.20	26.68	27.30
CF + CARN	27.47	28.09	28.30	28.63	28.52	28.28	27.93	27.37	26.95	27.95
IKC	**30.21**	**30.17**	**30.06**	**29.88**	**29.59**	**29.26**	**28.97**	**28.63**	**28.37**	**29.46**
EDVR Baseline	28.04	27.16	26.38	25.69	25.09	24.57	24.11	23.71	23.35	25.35
EDVR + **DynaVSR**	29.37	29.15	**28.87**	**28.66**	**28.59**	**28.59**	**28.56**	**28.46**	**28.20**	**28.72**
DUF Baseline	27.82	26.99	26.25	25.60	25.02	24.52	24.08	23.68	23.33	25.26
DUF + **DynaVSR**	**29.48**	**29.16**	28.69	28.12	27.50	26.87	26.25	25.66	25.12	27.43
TOF Baseline	27.79	27.01	26.29	25.64	25.06	24.55	24.09	23.70	23.34	25.27
TOF + **DynaVSR**	28.90	28.59	28.20	27.75	27.26	26.73	26.20	25.65	25.12	27.16

SSIM

Method	$\sigma = 0.8$	$\sigma = 0.9$	$\sigma = 1.0$	$\sigma = 1.1$	$\sigma = 1.2$	$\sigma = 1.3$	$\sigma = 1.4$	$\sigma = 1.5$	$\sigma = 1.6$	Average
KG + ZSSR	0.8083	0.8204	0.8454	0.8529	0.8581	0.8576	0.8528	0.8411	0.8189	0.8395
CF + DBPN	0.8195	0.8669	0.8906	0.8901	0.8900	0.8839	0.8724	0.8561	0.8394	0.8675
CF + CARN	0.8932	0.9002	0.9000	0.9005	0.8954	0.8864	0.8750	0.8552	0.8375	0.8826
IKC	**0.9206**	**0.9203**	**0.9196**	**0.9181**	**0.9153**	**0.9116**	**0.9077**	**0.9024**	**0.8965**	**0.9125**
EDVR Baseline	0.8839	0.8607	0.8357	0.8096	0.7834	0.7578	0.7332	0.7096	0.6873	0.7846
EDVR + **DynaVSR**	0.9169	**0.9132**	**0.9089**	**0.9057**	**0.9039**	**0.9023**	**0.8993**	**0.8937**	**0.8838**	**0.9031**
DUF Baseline	0.8772	0.8551	0.8312	0.8062	0.7809	0.7560	0.7317	0.7085	0.6865	0.7815
DUF + **DynaVSR**	**0.9192**	0.9114	0.9004	0.8860	0.8686	0.8485	0.8262	0.8024	0.7776	0.8600
TOF Baseline	0.8757	0.8546	0.8312	0.8065	0.7814	0.7564	0.7321	0.7088	0.6868	0.7815
TOF + **DynaVSR**	0.9053	0.8969	0.8864	0.8736	0.8585	0.8411	0.8213	0.7994	0.7760	0.8509

Table C: **Quantitative results of DynaVSR combined with recent VSR algorithms in isotropic Gaussian kernels on REDS-val**. We evaluate the benefits of DynaVSR algorithm on REDS-val [10] dataset.

PSNR										
Method	$\sigma = 0.8$	$\sigma = 0.9$	$\sigma = 1.0$	$\sigma = 1.1$	$\sigma = 1.2$	$\sigma = 1.3$	$\sigma = 1.4$	$\sigma = 1.5$	$\sigma = 1.6$	Average
KG + ZSSR	25.62	26.81	28.12	29.31	30.29	30.64	30.58	30.30	29.80	29.05
CF + DBPN	28.97	29.74	30.05	31.01	31.18	31.19	30.85	30.48	29.85	30.37
CF + CARN	31.17	31.32	31.53	31.55	31.44	31.00	30.81	30.29	29.72	30.98
IKC	**34.57**	**34.54**	**34.50**	**34.38**	**34.19**	**34.02**	**33.81**	**33.58**	**33.29**	**34.10**
EDVR Baseline	31.83	30.94	30.16	29.49	28.90	28.38	27.93	27.51	27.15	29.14
EDVR + **DynaVSR**	**33.63**	32.32	**32.90**	**32.48**	**32.16**	**31.97**	**31.89**	**31.87**	**31.82**	**32.45**
DUF Baseline	31.44	30.69	30.00	29.37	28.82	28.32	27.88	27.46	27.13	29.01
DUF + **DynaVSR**	33.03	**32.79**	32.40	31.92	31.36	30.77	30.17	29.59	29.04	31.23
TOF Baseline	31.58	30.79	30.06	29.40	28.82	28.31	27.86	27.46	27.10	29.04
TOF + **DynaVSR**	32.61	32.52	32.35	32.09	31.73	31.31	30.82	30.30	29.76	31.50

SSIM										
Method	$\sigma = 0.8$	$\sigma = 0.9$	$\sigma = 1.0$	$\sigma = 1.1$	$\sigma = 1.2$	$\sigma = 1.3$	$\sigma = 1.4$	$\sigma = 1.5$	$\sigma = 1.6$	Average
KG + ZSSR	0.8137	0.8398	0.8623	0.8794	0.8903	0.8922	0.8863	0.8776	0.8643	0.8673
CF + DBPN	0.8535	0.8723	0.8718	0.8960	0.9020	0.8946	0.8787	0.8611	0.8366	0.8741
CF + CARN	0.9172	0.9157	0.9140	0.9090	0.8985	0.8804	0.8678	0.8454	0.8212	0.8855
IKC	**0.9446**	**0.9438**	**0.9429**	**0.9409**	**0.9379**	**0.9350**	**0.9314**	**0.9271**	**0.9217**	**0.9361**
EDVR Baseline	0.9155	0.8989	0.8814	0.8635	0.8458	0.8285	0.8120	0.7962	0.7815	0.8470
EDVR + **DynaVSR**	**0.9446**	**0.9411**	**0.9366**	**0.9319**	**0.9278**	**0.9244**	**0.9213**	**0.9176**	**0.9120**	**0.9286**
DUF Baseline	0.9051	0.8906	0.8750	0.8587	0.8422	0.8259	0.8101	0.7950	0.7806	0.8426
DUF + **DynaVSR**	**0.9327**	0.9274	0.9201	0.9107	0.8992	0.8859	0.8711	0.8553	0.8389	0.8935
TOF Baseline	0.9079	0.8926	0.8763	0.8593	0.8424	0.8258	0.8097	0.7944	0.7799	0.8431
TOF + **DynaVSR**	0.9305	0.8275	0.9231	0.9171	0.9093	0.8995	0.8877	0.8743	0.8593	0.9031

Table D: **Quantitative results of DynaVSR combined with recent VSR algorithms in anisotropic Gaussian kernels on Vid4**. We evaluate the benefits of DynaVSR algorithm on Vid4 [9] dataset.

PSNR					
Method	$\theta = 0°$	$\theta = 45°$	$\theta = 90°$	$\theta = 135°$	Average
KG + ZSSSR	25.09	23.71	25.10	23.55	24.36
CF + DBPN	26.22	25.49	25.70	25.01	25.61
CF + CARN	**27.54**	26.64	26.51	26.57	26.82
IKC	26.10	26.12	26.50	25.97	26.17
EDVR Baseline	25.84	25.87	25.91	25.75	25.84
EDVR + **DynaVSR**	**29.65**	**28.12**	**29.39**	**28.06**	**28.81**
DUF Baseline	25.68	25.71	25.81	25.60	25.70
DUF + **DynaVSR**	27.29	**27.71**	**27.58**	**27.59**	**27.54**
TOF Baseline	25.60	25.71	25.72	25.62	25.66
TOF + **DynaVSR**	27.02	27.10	27.16	26.99	27.07

SSIM					
Method	$\theta = 0°$	$\theta = 45°$	$\theta = 90°$	$\theta = 135°$	Average
KG + ZSSSR	0.8305	0.8257	0.8438	0.8175	0.8075
CF + DBPN	0.8496	0.8310	0.8401	0.8206	0.8353
CF + CARN	**0.8848**	0.8699	0.8662	0.8715	0.8731
IKC	0.8338	0.8266	0.8445	0.8181	0.8308
EDVR Baseline	0.8320	0.8252	0.8142	0.8195	0.8227
EDVR + **DynaVSR**	**0.9179**	**0.8805**	**0.9045**	**0.8756**	**0.8946**
DUF Baseline	0.8225	0.8174	0.8102	0.8117	0.8154
DUF + **DynaVSR**	0.8756	**0.8806**	**0.8699**	**0.8764**	**0.8756**
TOF Baseline	0.8186	0.8158	0.8076	0.8108	0.8132
TOF + **DynaVSR**	0.8640	0.8614	0.8579	0.8569	0.8601

Table E: **Quantitative results of DynaVSR combined with recent VSR algorithms in anisotropic Gaussian kernels on REDS-val**. We evaluate the benefits of DynaVSR algorithm on REDS-val [10] dataset.

PSNR					
Method	$\theta = 0°$	$\theta = 45°$	$\theta = 90°$	$\theta = 135°$	Average
KG + ZSSR	27.65	26.66	28.59	26.28	27.30
CF + DBPN	29.51	28.66	30.16	28.88	29.30
CF + CARN	30.90	29.91	30.96	30.09	30.47
IKC	29.90	30.03	30.30	30.21	30.11
EDVR Baseline	29.33	29.45	30.05	29.82	29.66
EDVR + **DynaVSR**	**33.20**	**33.10**	**33.53**	**33.40**	**33.41**
DUF Baseline	29.06	29.23	29.68	29.56	29.38
DUF + **DynaVSR**	30.84	31.30	31.33	31.69	31.29
TOF Baseline	29.15	29.28	29.76	29.63	29.46
TOF + **DynaVSR**	**31.27**	**31.45**	**31.80**	**31.71**	**31.56**
SSIM					
Method	$\theta = 0°$	$\theta = 45°$	$\theta = 90°$	$\theta = 135°$	Average
KG + ZSSR	0.8465	0.8192	0.8662	0.8143	0.8366
CF + DBPN	0.8635	0.8587	0.8873	0.8689	0.8696
CF + CARN	0.9007	0.8925	0.9003	0.8958	0.8973
IKC	0.8838	0.8801	0.8841	0.8804	0.8821
EDVR Baseline	0.8654	0.8699	0.8797	0.8766	0.8729
EDVR + **DynaVSR**	**0.9356**	**0.9344**	**0.9400**	**0.9351**	**0.9363**
DUF Baseline	0.8549	0.8607	0.8666	0.8669	0.8623
DUF + **DynaVSR**	0.8947	0.9034	0.9026	0.9083	0.9023
TOF Baseline	0.8582	0.8622	0.8702	0.8685	0.8648
TOF + **DynaVSR**	**0.9045**	**0.9070**	**0.9127**	**0.9105**	**0.9087**

Table F: **Quantitative results(PSNR) for meta-training with recent VSR models and blind SISR methods**. We evaluate the benefits of DynaVSR algorithm on Vid4 [9] and REDS-val [10] dataset. Only the 10% of the full data are measured in this table.

	Method		Vid4 [9]			REDS-val [10]		
			Iso.	Aniso.	Mixed	Iso.	Aniso.	Mixed
Blind SISR	KG [2] + ZSSR [11]		25.94	24.38	21.33	28.96	27.43	25.54
	CF [5] + DBPN [4]		27.30	25.61	24.03	30.37	29.30	28.02
	CF [5] + CARN [1]		27.95	26.82	25.62	30.98	30.47	29.70
	IKC [3]		**29.49**	26.17	27.57	**34.10**	30.11	31.42
Video SR	EDVR [12]	Baseline	25.34	25.83	26.31	29.13	29.65	29.58
		DynaVSR	**28.70**	**28.79**	**29.41**	**32.43**	**33.40**	**33.50**
	DUF [6]	Baseline	25.25	25.69	26.53	29.01	29.38	29.71
		DynaVSR	27.42	**27.53**	**27.97**	31.23	31.29	31.11
	TOFlow [13]	Baseline	25.25	25.65	26.66	29.03	29.45	29.97
		DynaVSR	27.15	27.06	27.91	31.47	**31.52**	**31.69**

Table G: **SSIM results for meta-training with recent VSR models and blind SISR methods**. We evaluate the benefits of DynaVSR algorithm on Vid4 [9] and REDS-val [10] dataset. Only the 10% of the full data are measured in this table.

	Method		Vid4 [9]			REDS-val [10]		
			Iso.	Aniso.	Mixed	Iso.	Aniso.	Mixed
Blind SISR	KG [2] + ZSSR [11]		0.8390	0.8093	0.7131	0.8670	0.8394	0.7704
	CF [5] + DBPN [4]		0.8675	0.8353	0.7602	0.8741	0.8696	0.8202
	CF [5] + CARN [1]		0.8826	0.8731	0.8388	0.8855	0.8973	0.8788
	IKC [3]		**0.9125**	0.8301	0.8554	**0.9362**	0.8822	0.9042
Video SR	EDVR [12]	Baseline	0.7845	0.8225	0.8481	0.8468	0.8728	0.8700
		DynaVSR	**0.9026**	**0.8943**	**0.9059**	**0.9285**	**0.9362**	**0.9374**
	DUF [6]	Baseline	0.7815	0.8150	0.8531	0.8425	0.8624	0.8666
		DynaVSR	0.8595	**0.8752**	**0.8906**	0.8936	**0.9026**	**0.8999**
	TOFlow [13]	Baseline	0.7806	0.8124	0.8538	0.8429	0.8645	0.8708
		DynaVSR	0.8503	0.8594	0.8842	0.9028	**0.9082**	**0.9095**

Table H: Effect of inner loop learning rate α for EDVR on REDS-val.

α	Isotropic	Anisotropic	Mixed
10^{-4}	31.47	30.80	30.97
10^{-5}	**32.45**	**33.41**	**33.67**
10^{-6}	30.96	32.94	33.26

Table I: Additional experimental results for $\times 4$ SR in EDVR [12]. The best performance is written in **bold**. Note that DynaVSR shows better performance also in $\times 4$ SR task.

Method	Vid4 [9]			REDS-val [10]		
	Iso.	Aniso.	Mixed	Iso.	Aniso.	Mixed
EDVR baseline	**21.14**	21.53	21.11	24.80	25.22	25.29
EDVR+DynaVSR	**21.14**	**24.74**	**23.84**	**25.81**	**26.19**	**27.11**

References

[1] Namhyuk Ahn, Byungkon Kang, and Kyung-Ah Sohn. Fast, accurate, and lightweight super-resolution with cascading residual network. In *ECCV.*, 2018.

[2] Sefi Bell-Kligler, Assaf Shocher, and Michal Irani. Blind super-resolution kernel estimation using an internal-gan. In *NeurIPS.*, 2019.

[3] Jinjin Gu, Hannan Lu, Wangmeng Zuo, and Chao Dong. Blind super-resolution with iterative kernel correction. In *CVPR.*, 2019.

[4] Muhammad Haris, Gregory Shakhnarovich, and Norimichi Ukita. Deep back-projection networks for super-resolution. In *CVPR.*, 2018.

[5] Shady Abu Hussein, Tom Tirer, and Raja Giryes. Correction filter for single image super-resolution: Robustifying off-the-shelf deep super-resolvers. In *CVPR*, 2020.

[6] Younghyun Jo, Seoung Wug Oh, Jaeyeon Kang, and Seon Joo Kim. Deep video super-resolution network using dynamic upsampling filters without explicit motion compensation. In *CVPR.*, 2018.

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

[8] Wei-Sheng Lai, Jia-Bin Huang, Narendra Ahuja, and Ming-Hsuan Yang. Deep laplacian pyramid networks for fast and accurate super-resolution. In *CVPR.*, 2017.

[9] Ce Liu and Deqing Sun. On bayesian adaptive video super resolution. *TPAMI*, 2013.

[10] Seungjun Nah, Sungyong Baik, Seokil Hong, Gyeongsik Moon, Sanghyun Son, Radu Timofte, and Kyoung Mu Lee. Ntire 2019 challenge on video deblurring and super-resolution: Dataset and study. In *CVPR Workshop.*, 2019.

[11] Assaf Shocher, Nadav Cohen, and Michal Irani. "zero-shot" super-resolution using deep internal learning. In *CVPR.*, 2018.

[12] Xintao Wang, Kelvin CK Chan, Ke Yu, Chao Dong, and Chen Change Loy. Edvr: Video restoration with enhanced deformable convolutional networks. In *CVPR Workshop.*, 2019.

[13] Tianfan Xue, Baian Chen, Jiajun Wu, Donglai Wei, and William T Freeman. Video enhancement with task-oriented flow. *IJCV*, 2019.