Task-Aware Image Restoration using All-In-One Models

2025 Summer Seminar

Sogang University

Vision & Display Systems Lab, Dept. of Electronic Engineering

Presented By

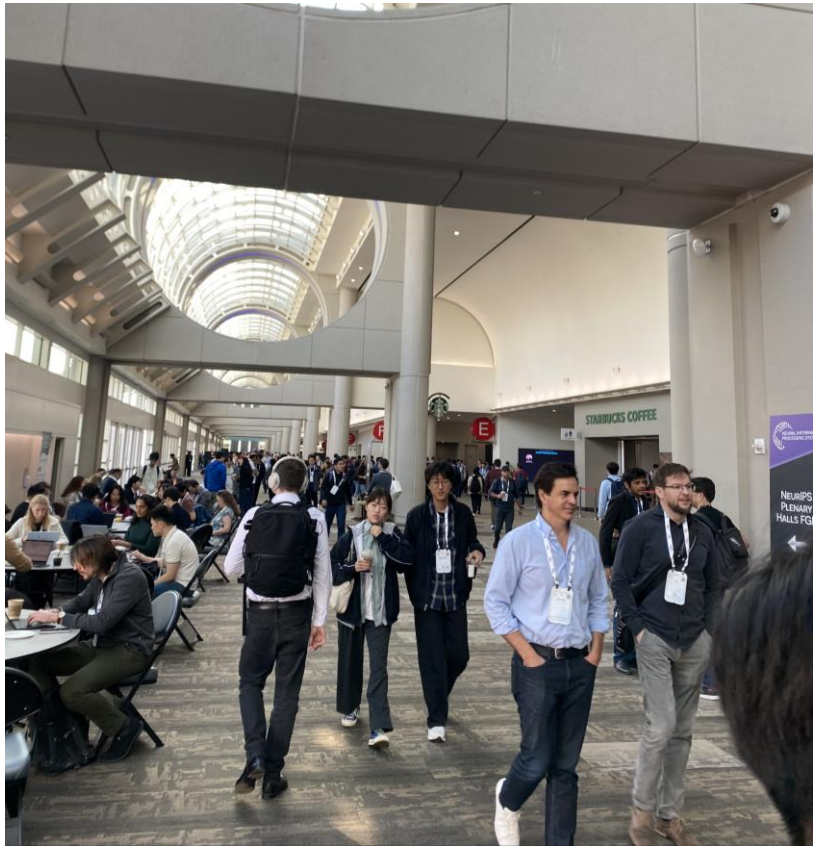
Matti Zinke

Contents

- NeurIPS 학회 경험
- Background
- P2IKT: Prior and Prediction Inverse Kernel Transformer for Single Image Defocus Deblurring [AAAI 2024]
- Revitalizing Convolutional Network for Image Restoration [TPAMI 2025]

NeurIPS 경험

- 2025/12/02-2025/12/07
- San Diego, California

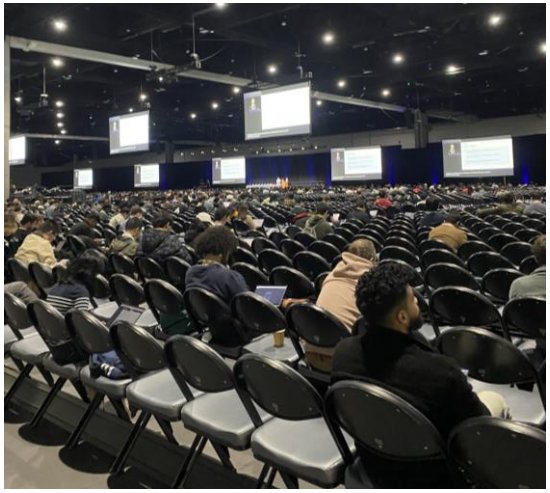


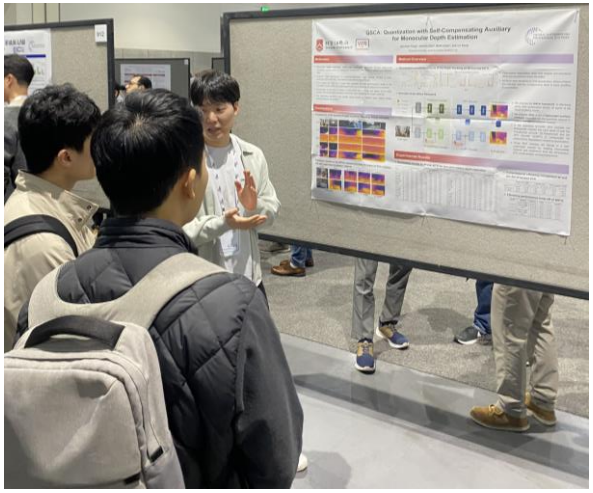
< 학회장 로비 >

< 학회장 앞 >

NeurIPS 경험

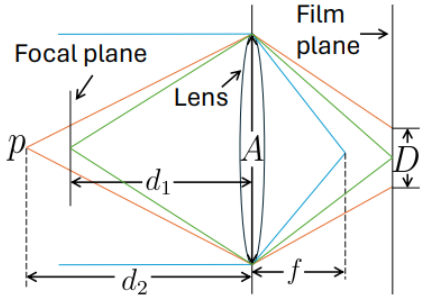
- 2025/12/02-2025/12/07
- San Diego, California
- Over 25.000 submissions and visitors
- 포스터 발표
  - 질문 하러 온 사람 50명 이상
  - 메타, ByteDance, 네이버, 삼성전자, 퀄컴, 노타AI 등





Background

- What is Defocus Deblur?
  - Single image defocus deblurring는 all-in-focus 이미지를 복원하는 것이 목표함
  - Focal plane에 있는 object가 sharp 하며, focal plane에 아닌 것은 blurry 함
    - 이미지 내의 object가 거리에 따라 blur가 달라짐
      - Spatially-varying 특성을 가짐
      - 이미지 전체에 동일한 kernel 적용할 수 없음



  - Defocus deblur 일반적으로 convolution으로 표현됨
    - $I_B = I_S \otimes K + N$
      - Defocus deblurring 보통 식의 역과정을 통해 sharp한 이미지 추정함

- P2IKT: Prior and Prediction Inverse Kernel Transformer for Single Image Defocus Deblurring [AAAI 2024]

Introduction

- Defocus 모델은 보통 irregular이나 generalization 중에 한 가지를 잘 못함
  - Train 이미지에 있는 blur에 대한 mapping을 학습
    - 모르는 blur에 robust 하지 않음
  - Prior kernel은 complexity 줄이기 위해서 쓰게 됨
    - 하지만, irregular blur를 이런 kernel로 근접할 수 없음
- Divide and conquer 방식으로 이 문제를 해결
  - Defocus blur는 보통 Gaussian blur나 비슷한 형태로 근접
    - Inverse Gaussian kernel module로 generalization ability 향상
  - Gaussian 아닌 blur을 제거 위한 inverse kernel prediction module 제안
  - 두 blur kernel 중에 맞는 kernel을 adaptive하게 선택
    - 이를 위해서 scale recurrent transformer 제안

Method

- Blur 모델의 blur을 표현 방식
  - $I_B = k \otimes I_S$
    - I_B는 blurry 이미지, I_S는 sharp 이미지, k는 kernel이며, $\otimes$ 는 convolution
  - Real-world 에서는 blur kernel k는 complex
    - $I_B = \begin{cases} k_{ga} \otimes I_S, & k \text{ approximated as Gaussian} \\ k_{ir} \otimes I_S, & \text{otherwise} \end{cases}$
    - 모든 이미지에 k_{ga}와 k_{ir} 기반인 deconvolution을 적용하며 coefficient map 따라 합침
  - Inverse kernel of k_{ga}
    - k_{ga} 는 linear combination of multi-size Gaussian blur kernels $g(\sigma_j)$
    - $g(\sigma_j)$ 의 inverse kernel은 Wiener deconvolution으로 얻음
    - $I_S = \sum_{j=1}^{J} \alpha_j \odot \left(f^{-1} \left(\frac{\bar{f}(g(\sigma_j))}{f^2(g(\sigma_j)) + n} \right) \otimes I_B \right)$

∴ $g(\sigma)$는 variance σ^2의 Gaussian kernel, α_j는 j-th inverse Gaussian kernel의 coefficient matrix, $\bar{f}(\cdot)$는 $f(\cdot)$의 conjugate, $f(\cdot)$와 $f^{-1}(\cdot)$는 discrete Fourier transform and inverse

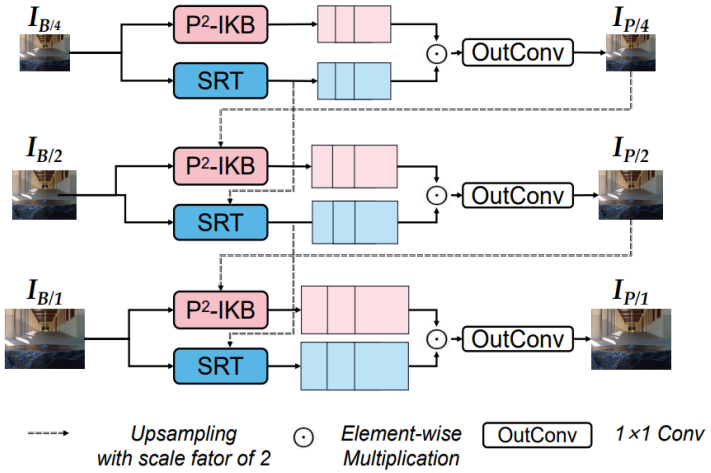
Method

- Blur 모델의 blur을 표현 방식
  - $I_B = k \otimes I_S$
    - I_B는 blurry 이미지, I_S는 sharp 이미지, k는 kernel이며, $\otimes$ 는 convolution
  - Real-world 에서는 blur kernel k는 complex
    - $I_B = \begin{cases} k_{ga} \otimes I_S, & k \text{ approximated as Gaussian} \\ k_{ir} \otimes I_S, & \text{otherwise} \end{cases}$
    - 모든 이미지에 k_{ga}와 k_{ir} 기반인 deconvolution을 적용하며 coefficient map 따라 합침
  - Inverse kernel of k_{ir}
    - Kernel prediction network (KPN) 학습하여 irregular kernel k_{ir} 예측
    - $I_S = f^{-1}(\frac{1}{f(k_{ir})}) \otimes I_B = k_{ir}^{\dagger} \otimes I_B$

      ∴ $k_{ir}^{\dagger}$는 inverse kernel of k_{ir}

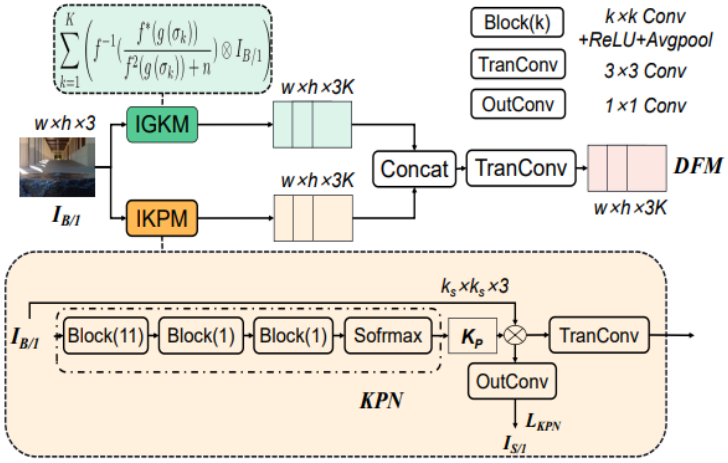
Method

- 모델 구조
  - Coarse-to-fine 모델
  - $I_{B/4}$와 $I_{P/4}$ 는 1/4 사이즈로 줄인 blurry 이미지와 deblurred result을 의미함
  - $I_{B/2}$는 $I_{P/4}$ 를 입력으로 받고 SRT도 이전의 coefficient map을 입력으로 받음
  - Deconvolutional feature map 과 coefficient map는 element-wise product으로 합치며 이미지를 restore함
  - Prior-and-Prediction Inverse Kernel Block (P2IKB)와 Scale Recurrent Transformer (SRT) 제안



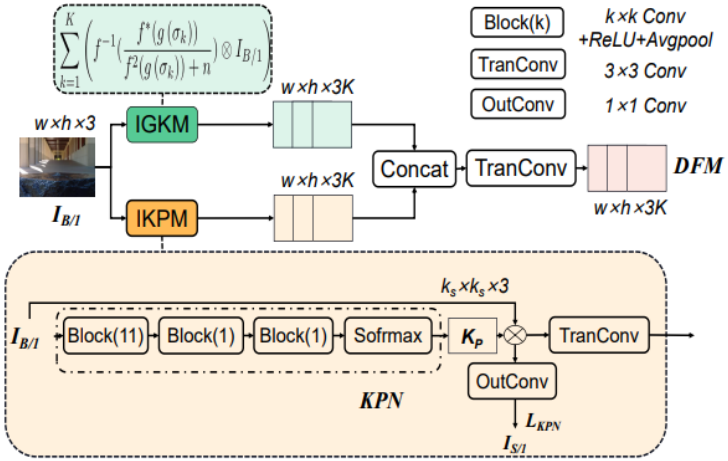
Method

- Prior-and-Prediction Inverse Kernel Block (P2IKB)
  - Defocus blur는 Gaussian blur와 Gaussian blur 아닌 blur로 나눔
    - 각 blur 위해서 모듈을 하나씩 설계
  - 다양한 사이즈인 Gaussian blur을 Inverse Gaussian Kernel Module (IGKM)으로 제거
    - Gaussian kernel은 predefined이며 group convolution으로 이루어짐
  - Irregular blur를 없애기 위해서 Inverse Kernel Prediction Model (IKPM) 설계
    - 이전 kernel prediction networks (KPN)과 비슷하며 irregular kernel을 예측함



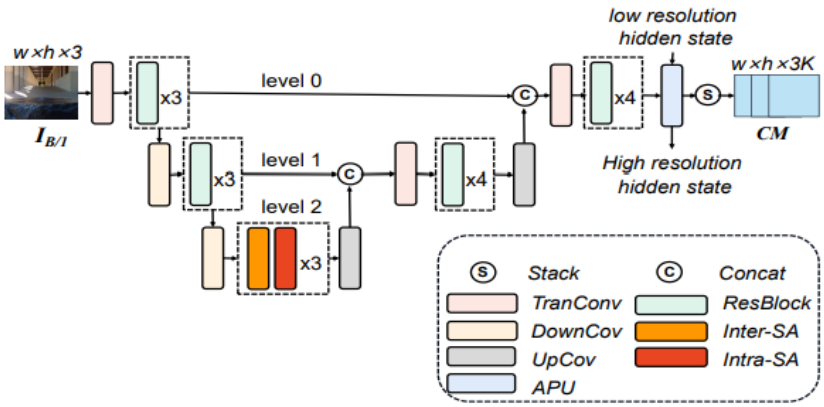
Method

- Prior-and-Prediction Inverse Kernel Block (P2IKB)
  - Inverse kernel prediction module (IKPM)
    - KPN, TransConv, Out-Conv 3부분으로 이루어져 있음
    - KPN은 3x convolution block + softmax을 통해 blurry 이미지 $k_s \times k_s \times 3$ kernel로 mapping
      - k_s는 max kernel size
    - OutConv으로 auxiliary loss L_{KPN}을 설계하며 inverse kernel을 K_p를 정함
      - $L_{KPN} = L_2(I_{S/1}, OutConv(K_p \otimes I_{B/1}))$
    - TransConv으로 IGKM과 IKPM의 channel size를 align을 시킴
    - Concat + TransConv으로 final deconvolutional feature map 계산



Method

- Scale Recurrent Transformer (SRT)
  - Blurry 이미지를 coefficient map으로 변환
    - Coefficient map으로 adaptively IGKM + IKPM의 출력을 deconv feature map으로 변환
  - Encoder-decoder structure, multi-scale feature 얻기 위해서 편리함
  - Coefficient map는 attention map이어서 Stripformer에서 제안한 inter + intra-strip attention 사용
  - Recurrent Unit (APU)는 hidden state를 합치고 coefficient map을 생성하며 다음 stage SRT으로 출력함
  - Restormer와 동일하게 decoder에 basic block을 더 추가함



Method

- Training strategy
  - $L = L_2 + \lambda_1 L_{freq} + \lambda_2 L_{LPIPS} + \lambda_3 L_{KPN}$
    - L_{freq} 는 frequency-domain loss
    - L_{LPIPS}는 Learned Perceptual Image Patch Similarity loss

- Ablation

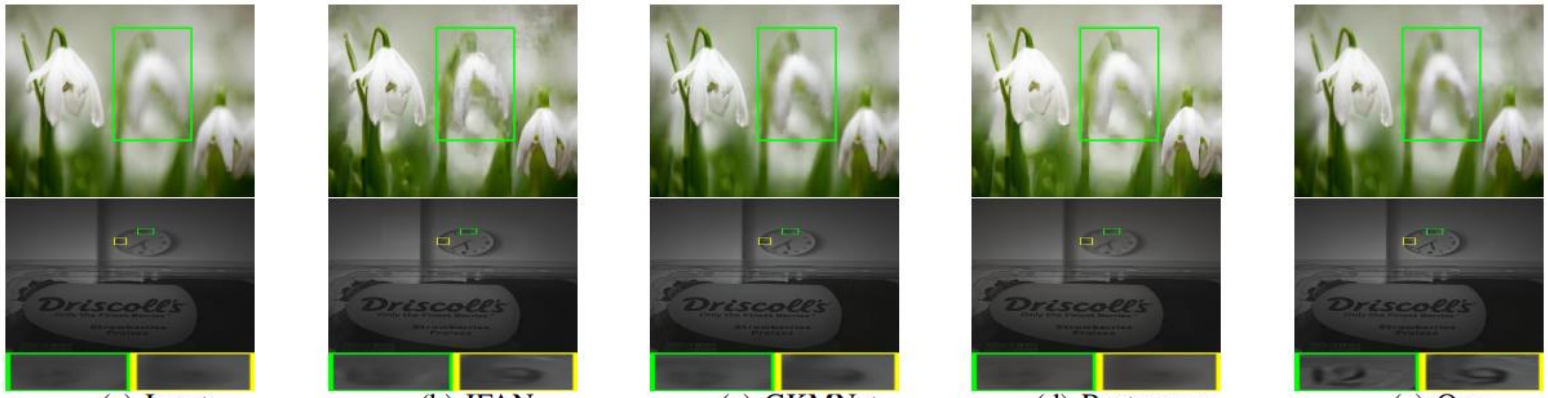
P²IKT				DPDD			RealDoF			LF-DOF			# Params (M)
IGKM	IKPM	SRT	SRAM										
			✓	24.27	0.766	0.229	22.97	0.739	0.290	25.25	0.768	0.244	3.58
✓	✓		✓	25.91	0.791	0.208	25.22	0.766	0.252	26.40	0.809	0.222	3.64
		✓		26.03	0.798	0.199	25.46	0.782	0.240	26.01	0.799	0.220	3.26
✓		✓		26.13	0.800	0.200	25.72	0.783	0.238	26.73	0.807	0.220	3.26
	✓	✓		26.12	0.799	0.203	25.67	0.782	0.249	26.69	0.806	0.227	3.32
✓	✓	✓		**26.29**	**0.807**	**0.191**	**25.78**	**0.787**	**0.235**	**26.90**	**0.821**	**0.220**	3.32

  - SRAM은 Scale Recurrent Attention Model 뜻하며 squeeze attention module임
    - Baseline에서 사용된 attention module

Experiment

- Qualitative Results

< Qualitative results on RealDoF dataset >



< Qualitative results on CUKH dataset >

Experiment

- Quantitative Results

Model	DPDD			# Params (M)
	PSNR	SSIM	LPIPs	
Blurry Input	23.89	0.725	0.349	-
JNB	23.69	0.707	0.442	-
EBDB	23.94	0.723	0.402	-
DMENet	23.90	0.720	0.410	26.94
DPDNet-S	24.03	0.735	0.279	35.25
KPAC	25.22	0.774	0.227	1.58
IFAN	25.37	0.789	0.217	10.48
GKMNet	25.36	0.774	0.276	1.41
DRBNet	25.47	0.787	0.246	-
Restormer	25.98	**0.811**	**0.178**	26.10
P^{2}IKT (Ours)	**26.29**	0.807	0.191	3.32

< Results on DPDD dataset >

Model	RealDoF			LF-DOF			DED			RTF		
	PSNR	SSIM	LPIPs	PSNR	SSIM	LPIPs	PSNR	SSIM	LPIPs	PSNR	SSIM	LPIPs
Blurry Input	22.54	0.636	0.498	25.87	0.779	0.316	**28.58**	0.884	0.164	24.18	0.739	0.364
IFAN	25.01	0.770	0.250	26.11	0.817	0.220	27.85	**0.890**	**0.111**	24.92	0.821	0.215
GKMNet	24.58	0.735	0.337	25.96	0.802	0.271	27.93	0.883	0.144	25.10	0.826	0.274
Restormer	25.43	**0.801**	**0.218**	26.44	**0.824**	**0.207**	26.95	0.884	0.113	24.21	0.822	0.204
P^{2}IKT (Ours)	**25.78**	0.787	0.235	**26.90**	0.821	0.220	28.29	0.888	0.123	**25.85**	**0.839**	**0.207**

< Results on other defocus datasets >

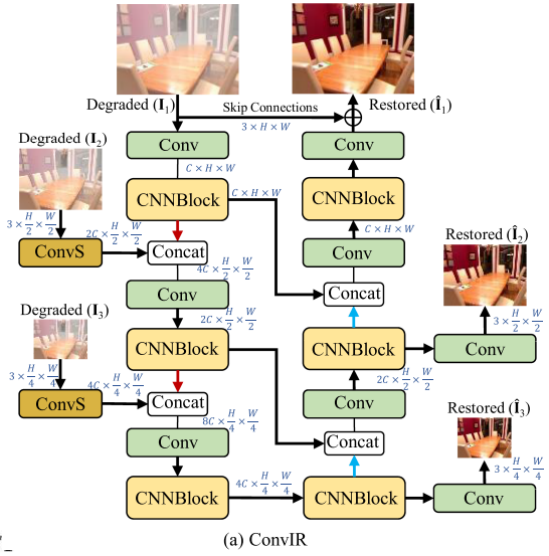
- Revitalizing Convolutional Network for Image Restoration [TPAMI 2024]

Introduction

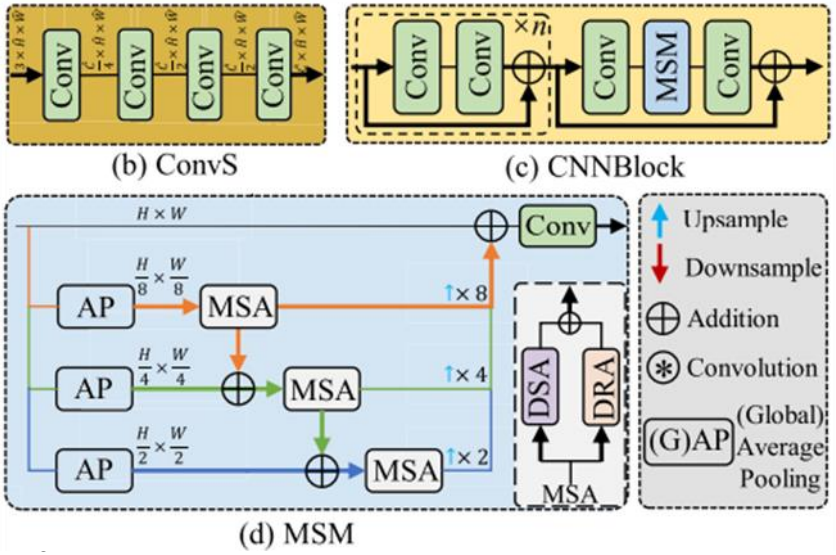
- 다양한 degradation을 제거하기 위하게 설계한 image restoration 모델
- 요즘 image restoration에서 transformer를 global 정보를 얻기 위해서 무조건 씀
- ConvIR는 CNN의 가능성을 보여주기 위해서 CNN로만 된 모델 제안
  - Transformer와 비슷하거나 더 높은 성능 보여줌
  - Self-attention의 complexity 줄이기 위해서 CNN만 쓰는 것이 선호
- 잘 되는 image restoration 모델은 꼭 고려해야 하는 것의 분석
  - Multi-scale representation learning
    - 다양한 사이즈의 blur를 더 잘 제거하기 위해서 필요함
  - Spatial attention
    - 중요한 부분에만 집중해서 spatially-varying blur를 없애기 위해서 필요함
  - Frequency modulation
    - Sharp한 이미지와 blurry 이미지의 frequency 차이를 줄이리 위해서 필요함
  - Low computational complexity
    - 이미지의 행상도가 높아도 efficient해야 함

Method

- 모델 구조
  - U-Net 구조
  - 3x3 convolution을 통해 shallow feature를 얻다가 CNNBlock 3개 사용
    - CNNBlock는 residual block 몇 개로 이루어졌으며 마지막 block에 MSM을 추가
    - CNNBlock 통해 channel을 늘면서 spatial resolution을 줄임
  - 다음은, 입력 이미지를 downsampling해서 main path와 합침
    - 다른 blur level에 대해 degradation을 해제 가능하게 함.
    - Downsampling한 이미지에 ConvS 적용해서 channel 수 main path와 맞게 함



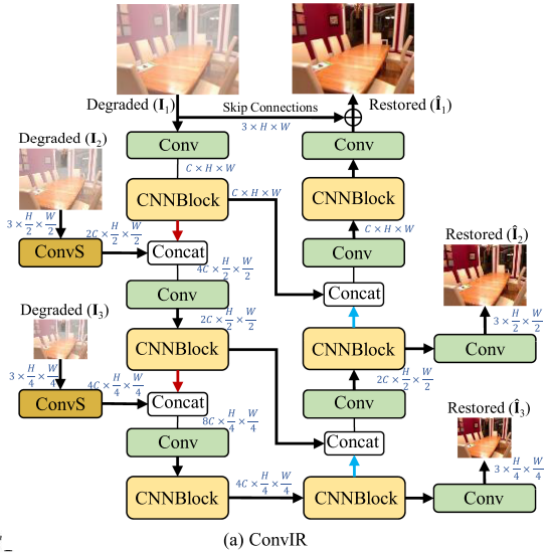
(a) ConvIR



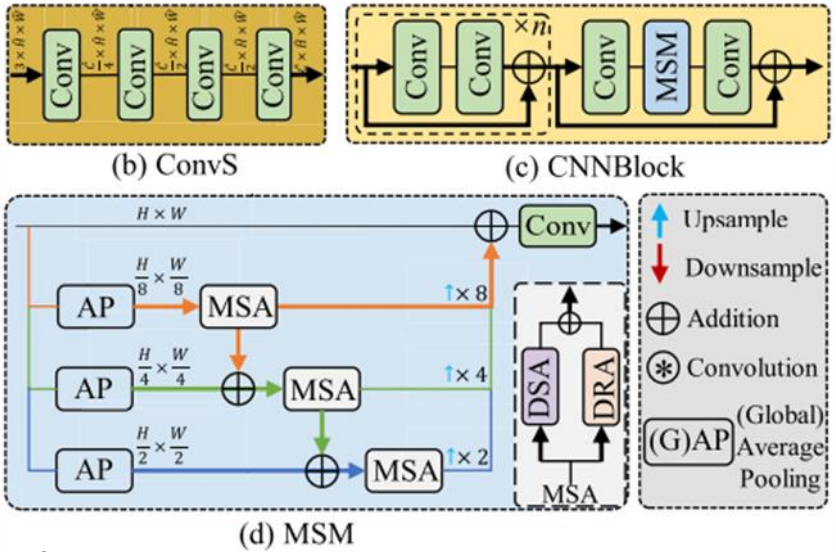
(d) MSM

Method

- 모델 구조
  - 이후에, downsampled 이미지의 feature를 main path에 추가
    - Conv로 channel 수 줄임
  - Decoder에 encode와 비슷하게 CNNBlock 3개 사용해서 high-resolution 이미지 복원
  - 학습 과정에 multi-output 기법 써서 low-res clean 이미지 출력됨
    - 일반 U-Net구조보다 성능 향상
  - Decoder에 encoder feature도 concat 되고 1x1 conv로 channel 수 반으로 줄임
  - 마지막에 3x3 conv 적용한 다음에 입력 이지미와 합친 이후에 clean 이미지를 출력



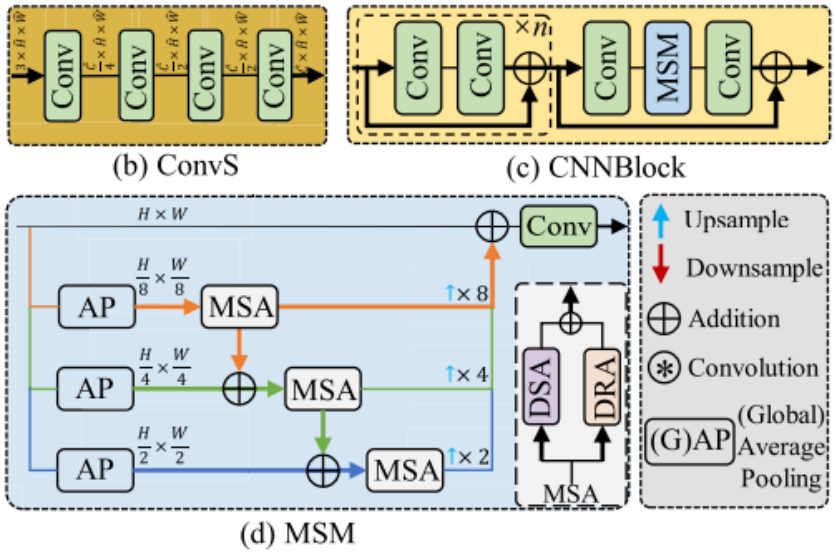
(a) ConvIR



(d) MSM

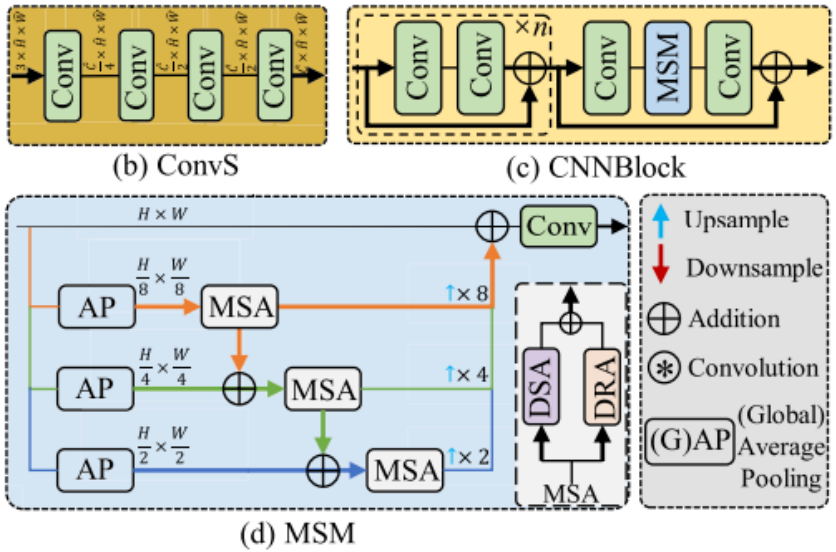
Method

- Multi-Scale Module (MSM)
  - Coarse-to-fine restoration을 위해서 multi-stage network을 따라함
  - Input tensor를 받고 average pooling과 Downsampling 통해 feature space로 변화
  - Branch마다 MSA로 생긴 resulting feature들이 다음 branch로 넘김
    - 이를 다양한 사이즈의 degradation을 progressively 잘 없앨 수 있음
  - 마지막에 downsample된 feature 다시 원래 사이즈로 upsample하고 합침



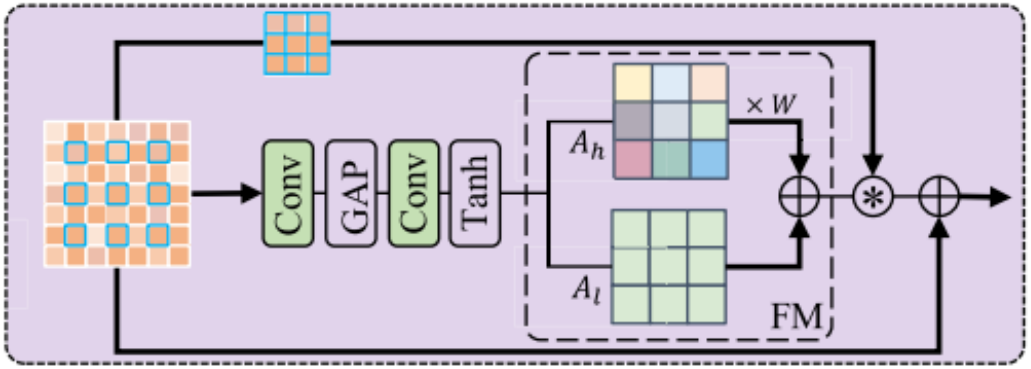
Method

- Multi-Shape Attention (MSA)
  - Multi-scale learning 더 efficient하게 사용하기 위해 모든 MSM branch에 MSA 추가
  - Transformer는 self-attention 통해 좋은 성능 보여줌
    - Quadratic complexity 때문에 image restoration에 큰 단점이 있음
    - 원래 convolution은 static 해서 spatially-varying degradation blur에 사용 불가능
  - MSA는 둘 강점을 가져가서 효율적이면서 뛰어난 성능 보여줌
  - Dilated Square Attention (DSA)와 Dilated Rectangle Attention (DRA)로 이루어짐



Method

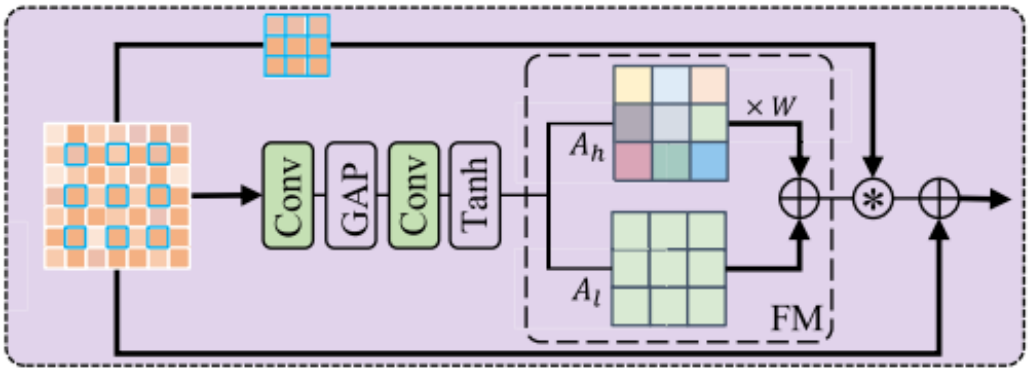
- Dilated Square Attention (DSA)
  - Convolution으로 입력 feature에 adaptive 하게 attention weight를 생성하고 convolution으로 aggregation 함
  - 보통 self-attention에 softmax로 weight normalization 함
    - 나온 weight가 low-pass filter의 kernel로 볼 수 있음
    - Image restoration에 sharp과 degraded 이미지의 차이가 거의 high-frequency에 존재
  - Softmax를 대신 tanh를 사용
    - Tanh로 이전에 존재한 low-pass filter의 limitation을 없앰
    - Tanh로 attention weight는 (-1, 1) range로 바꿔서 negative weight로 detrimental pixels 예방



(e) Dilated Square Attention (DSA)

Method

- Dilated Square Attention (DSA)
  - 입력을 그룹으로 나눠서 attention weights을 계산
    - Complexity/weight diversity의 trade off 때문에 모든 channel에 weight 만들지 않음
  - 이렇게 모든 feature 그룹 weights는 channel과 spatial dimension에 shared하게 사용
  - Filter modulation (FM)을 통해 high-pass filter을 설계
  - FM 이후에 계산 된 attention weigh에 input feature를 convo로 추가
    - Large receptive field을 보기 위해서 input feature의 pixel은 dilated 방식으로 선택



(e) Dilated Square Attention (DSA)

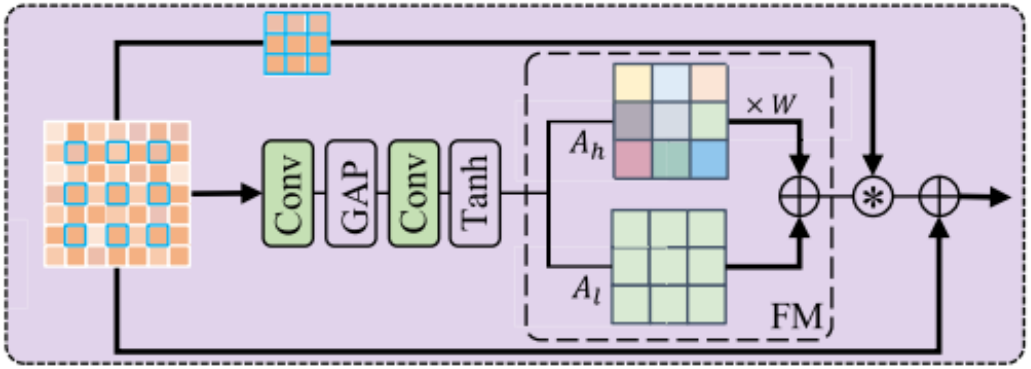
Method

- Filter modulation (FM)
  - Attention map A^{DSA}를 low/high-pass filter로 나누는 역할
    - Low-pass filter는 입력의 direct-current component만 유지하는 filter로 설계

      $\therefore A_l^{DSA} = \frac{1}{K^2} E$

      ✓ E는 A^{DSA}와 같은 shape이며 모든 value가 1
    - High-pass filter는 low-pass filter의 complement으로 계산

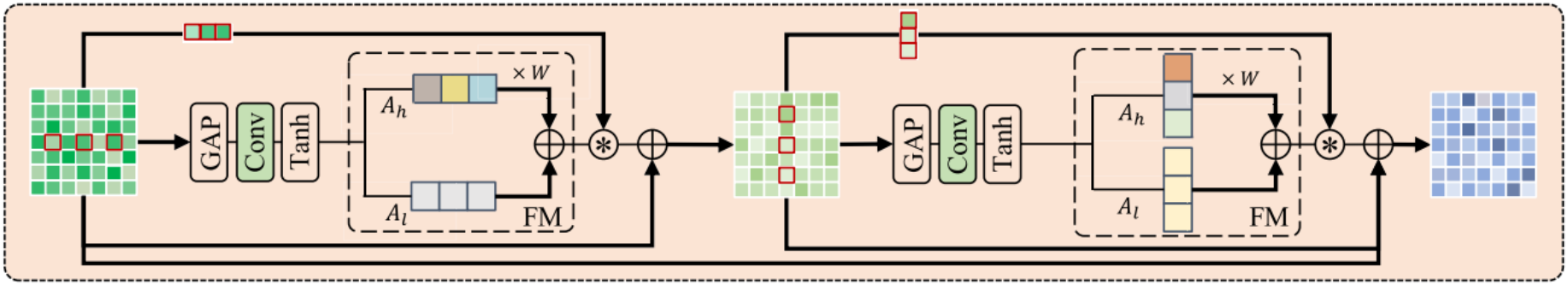
      $\therefore A_h^{DSA} = A^{DSA} - A_l^{DSA}$
  - High-pass filter의 attention map에 있는 영향을 learnable 파라미터 W로 reweight해서 중요한 frequency에 집중



(e) Dilated Square Attention (DSA)

Method

- Dilated Rectangle Attention (DRA)
  - DSA와 비슷한 구조
  - Multi-shape representation learning을 개선하기 위해서 rectangle 기반인 orthogonal attention 계산
    - 먼저 horizontal unit 실행한 다음에 vertical unit 실행
- Loss function
  - $\mathcal{L}_{spatial} = \sum_{i=1}^{3} \frac{1}{P_i} \left\| \hat{I}_i - Y_i \right\|_1$
  - $\mathcal{L}_{frequency} = \sum_{i=1}^{3} \frac{1}{S_i} \left\| [\mathcal{R}(\hat{I}_i), \mathcal{I}(\hat{I}_i)] - [\mathcal{R}(Y_i), \mathcal{I}(Y_i)] \right\|_1$



(f) Dilated Rectangle Attention (DRA)

Ablation

Methods	a	b	c	d	e	f
Baseline	✓	✓	✓	✓	✓	✓
MSM/Conv		✓	✓	✓	✓	✓
DSA/Conv			✓	✓	✓	✓
DSA/FM				✓	✓	✓
DSA/Dilation					✓	✓
DRA						✓
PSNR	31.23	31.46	31.53	31.64	31.76	31.92
Params/M	6.90	8.45	8.55	8.56	8.56	8.63
FLOPs/G	66.32	71.17	71.19	71.19	71.19	71.22
Time/s	0.134	0.152	0.165	0.166	0.170	0.206

< Break-down ablation on every model part >

2	4	8	PSNR	Params/M	FLOPs/G
			31.23	6.90	66.32
✓			31.45	7.74	70.91
✓	✓		31.62	8.18	71.15
✓	✓	✓	31.92	8.63	71.22

< Analysis on number of branches in MSM >

Pooling Types	Convolution	Max Pooling	Average Pooling
PSNR	31.75	31.65	31.92

Methods	Softmax	Linear	Sigmoid	Tanh
PSNR	31.75	31.79	31.83	31.92

< Analysis on pooling operator and functions >

Experiments

- Quantitative Results

Methods	Indoor Scenes				Outdoor Scenes				Combined				Params (M)	FLOPs (G)
	PSNR↑	SSIM↑	MAE↓	LPIPS↓	PSNR↑	SSIM↑	MAE↓	LPIPS↓	PSNR↑	SSIM↑	MAE↓	LPIPS↓		
DPDNet [29]	26.54	0.816	0.031	0.239	22.25	0.682	0.056	0.313	24.34	0.747	0.044	0.277	31.03	770
KPAC [7]	27.97	0.852	0.026	0.182	22.62	0.701	0.053	0.269	25.22	0.774	0.040	0.227	2.06	113
MDP [98]	28.02	0.841	0.027	-	22.82	0.690	0.052	-	25.35	0.763	0.040	-	46.86	1898
IFAN [6]	28.11	0.861	0.026	0.179	22.76	0.720	0.052	0.254	25.37	0.789	0.039	0.217	10.48	363
DRBNet [19]			-				-		25.73	0.791	-	0.183	11.69	693
FocalNet [46]	29.10	0.876	0.024	0.173	23.41	0.743	**0.049**	0.246	26.18	0.808	0.037	0.210	12.82	1376
Restormer [16]	28.87	0.882	0.025	0.145	23.24	0.743	0.050	0.209	25.98	0.811	0.038	0.178	26.16	1983
IRNeXt [34]	29.22	0.879	0.024	0.167	**23.53**	0.752	**0.049**	0.244	26.30	0.814	0.037	0.206	14.76	1778
ConvIR-S (Ours)	28.95	0.877	0.024	0.158	23.32	0.747	0.050	0.221	26.06	0.810	0.037	0.190	5.53	579
ConvIR-B (Ours)	29.06	0.879	0.024	0.156	23.42	0.752	**0.049**	0.219	26.16	0.814	0.037	0.188	8.63	979
ConvIR-L (Ours)	**29.37**	**0.887**	**0.023**	**0.143**	23.51	**0.757**	**0.049**	**0.203**	**26.36**	**0.820**	**0.036**	**0.174**	14.83	1778

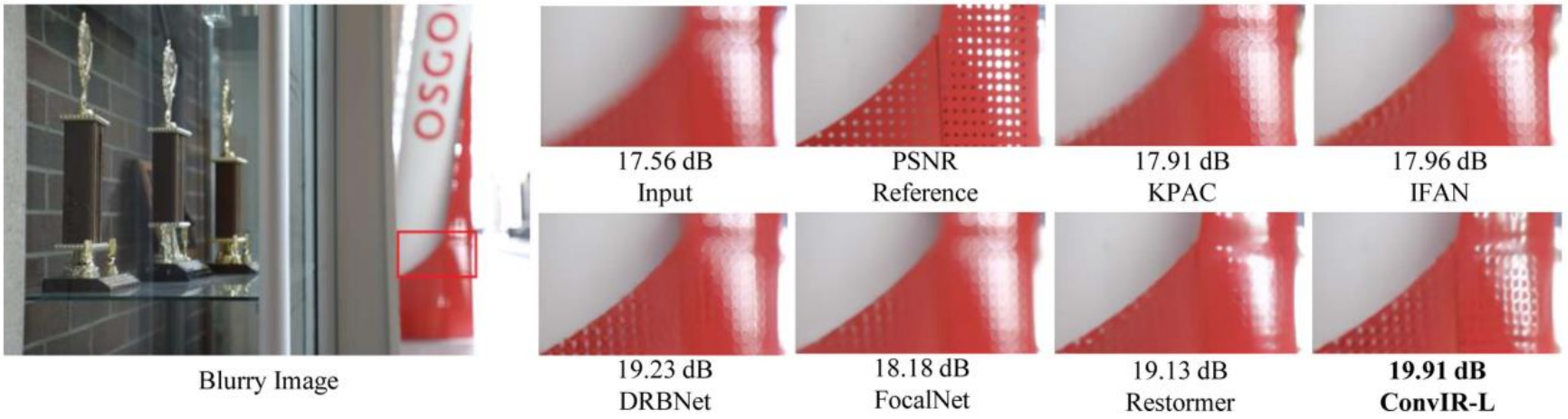
< Defocus deblur task>

Methods	PSNR	SSIM	FLOPs (G)	Params (M)	Time (s)	Memory (G)
MIMO-UNet++ [5]	32.68	0.959	617.64	16.1	1.277	10.395
MPRNet [12]	32.66	0.959	777.01	20.1	1.148	10.415
MAXIM-3S [47]	32.86	0.961	119.5	22.2	-	-
Restormer [16]	32.92	0.961	140.99	26.1	1.218	12.333
Stripformer [17]	33.08	0.962	170.46	20.0	1.054	12.149
PromptRestorer [32]	33.06	0.962	-	-	-	-
IRNeXt [34]	33.19	**0.963**	129.33	14.76	0.291	6.865
ConvIR-L (Ours)	**33.28**	**0.963**	129.34	14.83	0.323	6.867

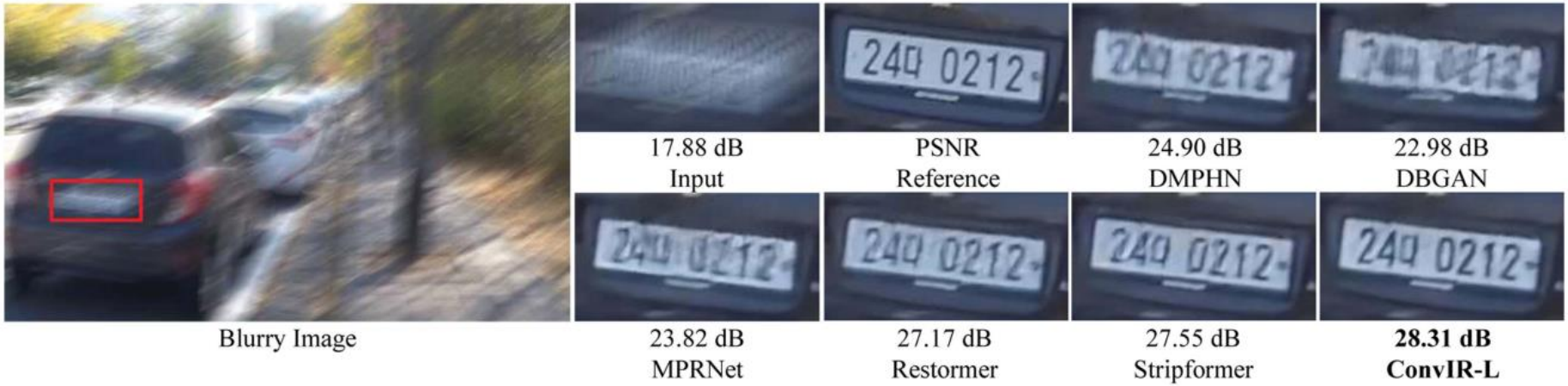
< Motion deblur task>

Experiments

- Qualitative Results



< Defocus deblur task>



< Motion deblur task>

Thank you!