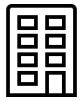
2022 하계 세미나

Image Deblurring in CVPR 2022



Sogang University
Vision & Display Systems Lab, Dept. of Electronic Engineering

Presented by
민성준

Outline

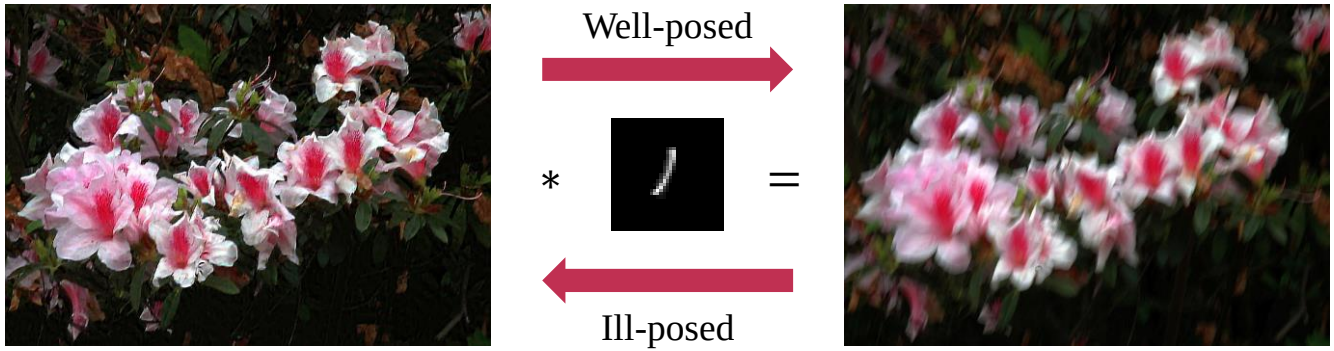
- Background
- Deblurring via Stochastic Refinement (CVPR 2022)
- XYDeblur: Divide and Conquer for Single Image Deblurring (CVPR 2022)

Background

- Image Deblurring
  - Problem formulation
    - $B = I \otimes k + n$
      - B: blurry image
      - I: clean image
      - k: blur kernel
      - ~~n: additive noise~~
    - Ill-posed problem

$$20 \times 5 = ?$$

$$? \times ? = 100$$



Background

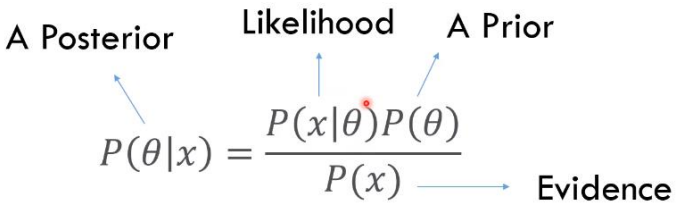
- Image Deblurring
  - Optimization & regularization
    - Maximum a Posteriori (MAP) formulation

$$? \times ? = 100$$

prior: 짝수

⁘ prior 활용하여 문제 해결

✓이미지들의 특성을 사전 정보로 알고 있다.



⁘ 더 좋은 prior를 어떻게 만들 것인가?

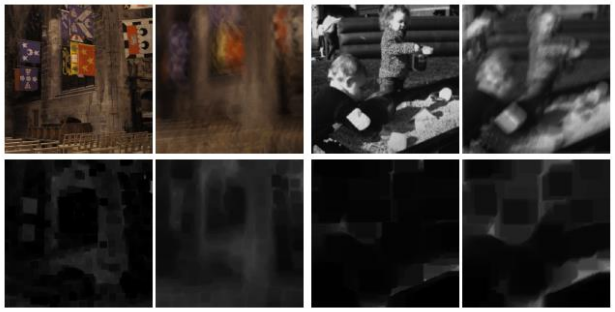
✓Dark channel prior

✓Gradient sparsity prior

✓Deep denoiser prior

Background

- Image Deblurring
  - Optimization & regularization
    - $\min_{x,k} D(x * k - y) + \gamma P_k(k) + \lambda P_x(x)$
    - Prior formulation
      - [1] Blind image deblurring using dark channel prior
        - ✓Prior: Blurry image의 dark channel은 sparse하게 구성된다
        - ✓Conventional approach → learning-based approach



(a) Clear (b) Blurred (c) Clear (d) Blurred

Figure 2. Blurred images have less sparse dark channels than clear images. The blur process (convolution) outputs a weighted average of pixels in a neighborhood and tends to increase the value of the minimum pixel. Top: images; bottom: corresponding dark channels computed with an image patch size of 35×35.

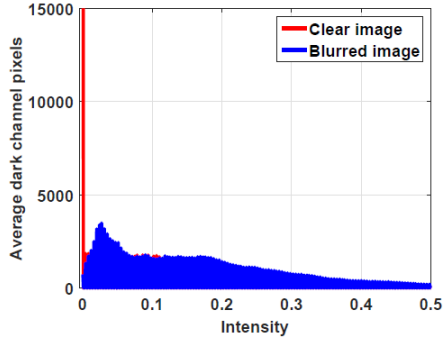


Figure 3. Intensity histograms for dark channels of both clear and blurred images in a dataset of 3,200 natural images. Blurred images have far fewer zero dark channel pixels than clear ones, confirming our analysis in the text. The dark channel of each image has been computed with an image patch size of 35×35.

Background

- Image Deblurring
  - Optimization & regularization
    - $\min_{x,k} D(x * k - y) + \gamma P_k(k) + \lambda P_x(x)$
    - Prior formulation
      - [1] Blind image deblurring using dark channel prior
        - ✓Prior: Blurry image의 dark channel은 sparse하게 구성된다
        - ✓Conventional approach → learning-based approach

< GoPro dataset blur/sharp image pair >

Background

- Image Deblurring
  - CVPR 2022 accepted list

Learning to Deblur Using Light Field Generated and Real Defocus Images
Deblur-NeRF: Neural Radiance Fields From Blurry Images
Pixel Screening Based Intermediate Correction for Blind Deblurring
Deblurring via Stochastic Refinement
XYDeblur: Divide and Conquer for Single Image Deblurring
Multi-Scale Memory-Based Video Deblurring
Unifying Motion Deblurring and Frame Interpolation With Events

Diffusion in Deblurring[1]

- Limitations in conventional supervised learning models
  - Conventional CNNs trained in sharp-blurry image pairs
    - Supervised learning by minimizing L_1 or L_2 pixel loss
      - PSNR ↑, but human perception..?
      - style loss, adversarial loss, content loss, contextual loss 등을 추가하여 보완
      - Single input – single output (deterministic) 형태로는 human perception을 못 따라감
        - ✓Image regression to the mean
        - ✓Single input – multi output을 내자!
          - Ill-posed problem을 well-posed problem으로 풀려고 한 문제
          - Generative model

Diffusion in Deblurring

- Predict-and-refine
  - Residual learning
    - Initial predictor g_θ
      - Residual learning을 위한 초기에 한번 학습 (deterministic)
    - Denoiser network f_θ
      - Residual image를 target output으로 하여 여러 번 학습 (stochastic)

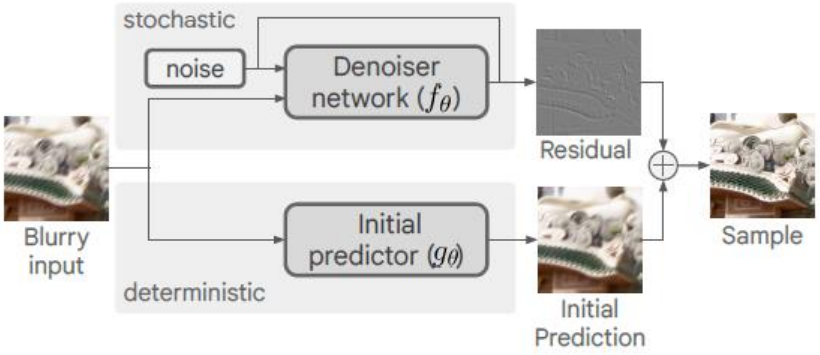
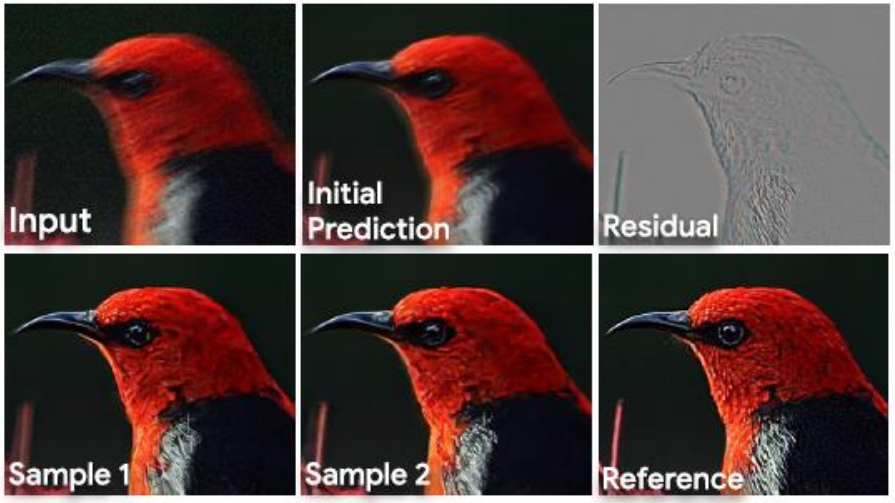


Figure 2. Diagram describing our dual-network architecture. The initial predictor produces the deterministic candidate for the denoiser network, which then models the residual.



Diffusion in Deblurring

- Conditional diffusion model
  - Forward diffusion step
    - Markov chain
    - 1회 noise 뿌리기

$$q(x_t \mid x_{t-1}) \triangleq \mathcal{N}(x_t; \sqrt{\alpha_t}x_{t-1}, (1-\alpha_t)\boldsymbol{I}_d),$$

    - Generalization

$$q(x_t \mid x_0) = \mathcal{N}(x_t; \sqrt{\bar{\alpha}_t}x_0, (1-\bar{\alpha}_t)\boldsymbol{I}_d)$$

$$q(x_{t-1} \mid x_t, x_0) = \mathcal{N}(x_{t-1}; \boldsymbol{\mu}_t(x_t, x_0), \beta_t \boldsymbol{I}_d),$$

$$\boldsymbol{\mu}_t(\boldsymbol{x}_t, \boldsymbol{x}_0) = \frac{\sqrt{\bar{\alpha}_{t-1}}(1-\alpha_t)}{1-\bar{\alpha}_t}\boldsymbol{x}_0 + \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}\boldsymbol{x}_t,$$

$$\beta_t = \frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}(1-\alpha_t).$$

$$\boldsymbol{a} \quad \boldsymbol{ar^1} \quad \dots \quad \boldsymbol{ar^{t-1}} \quad \underset{\div r}{\overset{q(x_t|x_{t-1}) \\ \times r}{\rightleftarrows}} \quad \boldsymbol{ar^t} \quad \dots \quad \boldsymbol{ar^{n-1}}$$

$$p(x_{t-1}|x_t)$$

Diffusion in Deblurring

- Conditional diffusion model
  - Forward diffusion step
    - Markov chain
    - 1회 noise 뿌리기

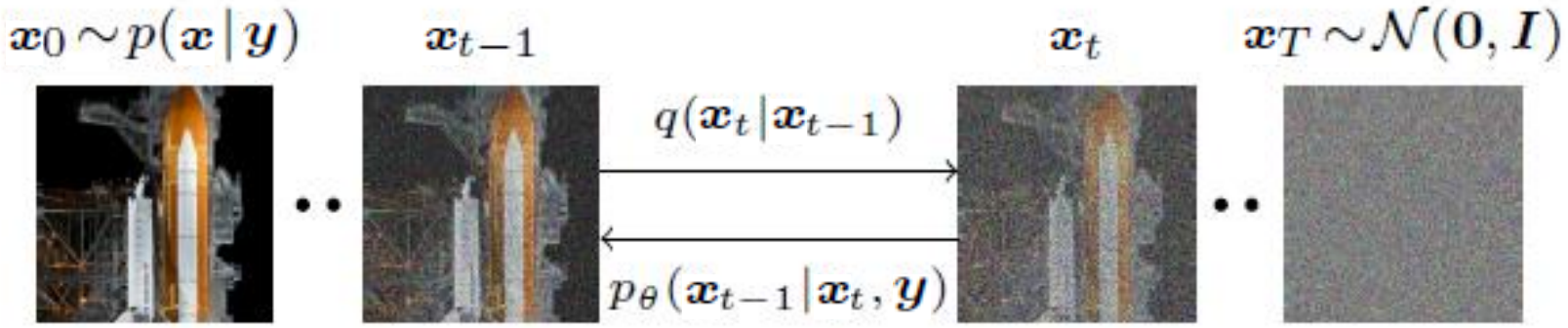
$$q(x_t \mid x_{t-1}) \triangleq \mathcal{N}(x_t; \sqrt{\alpha_t}x_{t-1}, (1-\alpha_t)\boldsymbol{I}_d),$$

    - Generalization

$$q(x_t \mid x_0) = \mathcal{N}(x_t; \sqrt{\bar{\alpha}_t}x_0, (1-\bar{\alpha}_t)\boldsymbol{I}_d)$$

$$q(x_{t-1} \mid x_t, x_0) = \mathcal{N}(x_{t-1}; \boldsymbol{\mu}_t(x_t, x_0), \beta_t \boldsymbol{I}_d),$$

$$\boldsymbol{\mu}_t(\boldsymbol{x}_t, \boldsymbol{x}_0) = \frac{\sqrt{\bar{\alpha}_{t-1}}(1-\alpha_t)}{1-\bar{\alpha}_t}\boldsymbol{x}_0 + \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}\boldsymbol{x}_t,$$
$$\beta_t = \frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}(1-\alpha_t).$$



Diffusion in Deblurring

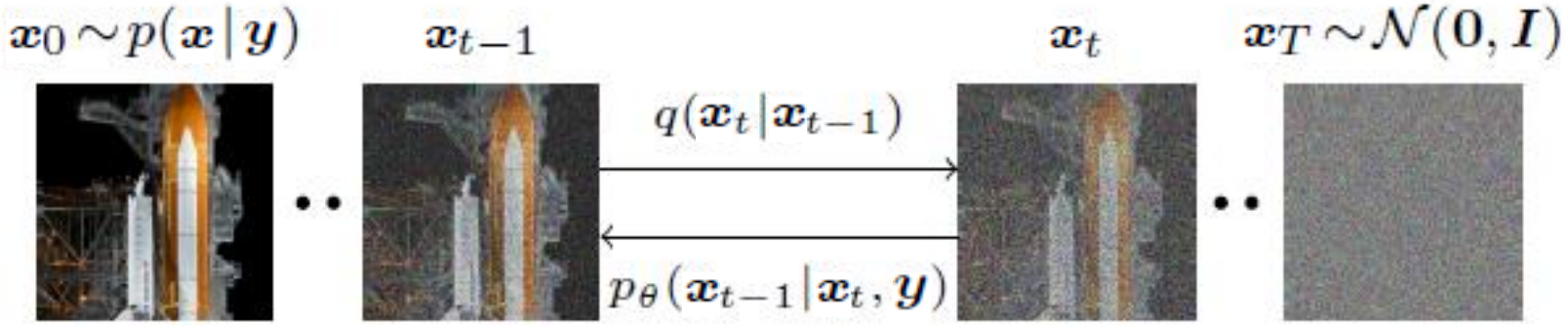
- Conditional diffusion model
  - Reverse diffusion step
    - Estimate x_0!
      - $x_0 = f_\theta(x_t, t)$ 가 되도록 f_θ 학습

$$p_\theta(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t) \triangleq q(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t, f_\theta(\boldsymbol{x}_t, t))$$

    - Continuous noise level (Noise scheduler)
      - Noise level $\bar{\alpha}_t$

$$\boldsymbol{x}_t = \sqrt{\bar{\alpha}_t}\boldsymbol{x}_0 + (1 - \bar{\alpha}_t)\boldsymbol{\epsilon}$$

$$\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I}_d)$$



Diffusion in Deblurring

- Conditional diffusion model
  - Loss
    - Conditional diffusion probabilistic model
      - Additional blurry input $\boldsymbol{y}$

$$L_{\text{Base}}(\theta) = \mathbb{E}\left\| \boldsymbol{\epsilon} - f_\theta(\sqrt{\bar{\alpha}}\boldsymbol{x}_0 + \sqrt{1-\bar{\alpha}}\boldsymbol{\epsilon}, \bar{\alpha}, \boldsymbol{y}) \right\|_1$$

    - Residual learning

$$\mathbb{E}\left\| \boldsymbol{\epsilon} - f_\theta\left(\sqrt{\bar{\alpha}}\underbrace{(\boldsymbol{x}_0 - g_\theta(\boldsymbol{x}_0))}_{\text{residual}} + \sqrt{1-\bar{\alpha}}\boldsymbol{\epsilon}, \bar{\alpha}, \boldsymbol{y}\right) \right\|_1$$

  - Overall algorithm

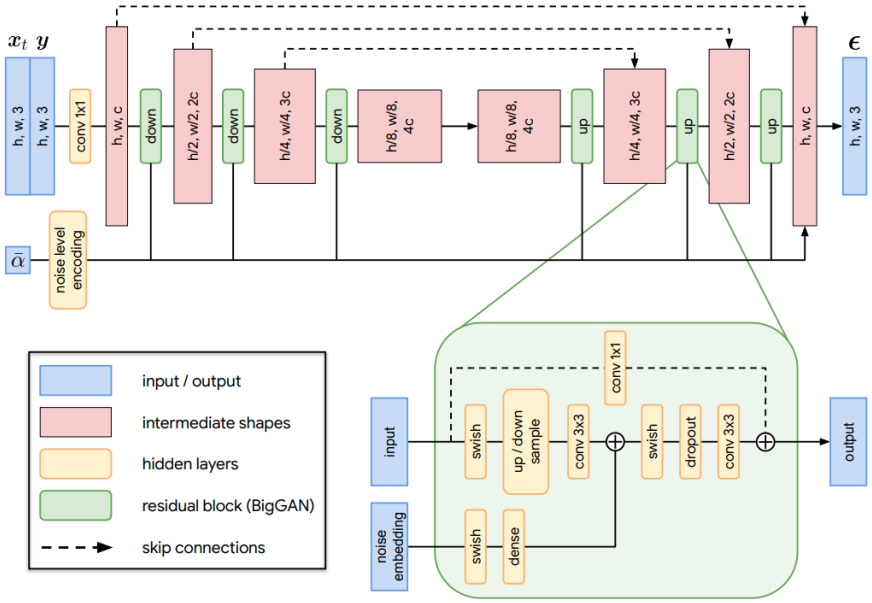
Algorithm 1 Predict-and-refine diffusion sampling.
The expressions for $\boldsymbol{\mu}_t, \bar{\alpha}_t, \beta_t$ can be found in Sec. 3.

Require: f_θ: Denoiser network, g_θ: Initial predictor, $\boldsymbol{y}$: Blurry input image, $\boldsymbol{\alpha}_{1:T}$: Noise schedule.

1: $\boldsymbol{x}_{\text{init}} \leftarrow g_\theta(\boldsymbol{y})$ ▷ Initial prediction
2: $\boldsymbol{z}_T \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}_d)$ ▷ Run diffusion sampling
3: **for** $t = T, \ldots, 1$ **do**
4: $\quad \boldsymbol{\epsilon}_t \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}_d)$
5: $\quad \boldsymbol{z}_{t-1} \leftarrow \boldsymbol{\mu}_t(\boldsymbol{z}_t, f_\theta(\boldsymbol{z}_t, \bar{\alpha}_t, \boldsymbol{y})) + \beta_t \boldsymbol{\epsilon}_t$ ▷ Reverse diffusion step; see Eq. (3)
6: **end for**
7: **return** $\boldsymbol{x}_{\text{init}} + \boldsymbol{z}_0$ ▷ Return the final restoration

Diffusion in Deblurring

- Architecture
  - Initial predictor network (network size ↑)
    - Channel multiplier {1, 2, 3, 4} with initial 64
  - Denoiser network (network size ↓)
    - Channel multiplier {1, 2, 3, 4} with initial 32



Diffusion in Deblurring

- Results

	Perceptual				Distortion	
	LPIPS↓	NIQE↓	FID↓	KID↓	PSNR↑	SSIM↑
Ground Truth	0.0	3.21	0.0	0.0	∞	1.000
HINet [11]	0.088	4.01	17.91	8.15	32.77	0.960
MPRNet [73]	0.089	4.09	20.18	9.10	32.66	0.959
MIMO-UNet+ [14]	0.091	4.03	18.05	8.17	32.45	0.957
SAPHNet [63]	0.101	3.99	19.06	8.48	31.89	0.953
SimpleNet [40]	0.108				31.52	0.950
DeblurGANv2 [35]	0.117	3.68	13.40	4.41	29.08	0.918
Ours	0.059	3.39	4.04	0.98	31.66	0.948
Ours-SA	0.078	4.07	17.46	8.03	33.23	0.963

< Qualitative result on GoPro dataset >

	Perceptual				Distortion	
	LPIPS↓	NIQE↓	FID↓	KID↓	PSNR↑	SSIM↑
Ground Truth	0.0	2.72	0.0	0.0	∞	1.000
HINet [11]	0.120	3.20	15.17	7.33	30.33	0.932
MIMO-UNet+ [14]	0.124	3.24	16.01	7.91	29.99	0.930
MPRNet [73]	0.114	3.46	16.58	8.35	30.96	0.939
SAPHNet [63]	0.128	3.21	16.77	8.39	29.99	0.930
DeblurGAN-v2 [35]	0.159	2.96	15.51	6.97	27.51	0.885
Ours	0.089	2.69	5.43	1.61	29.77	0.922
Ours-SA	0.092	2.93	6.37	2.40	30.07	0.928

< Qualitative result on HIDE dataset >

Diffusion in Deblurring

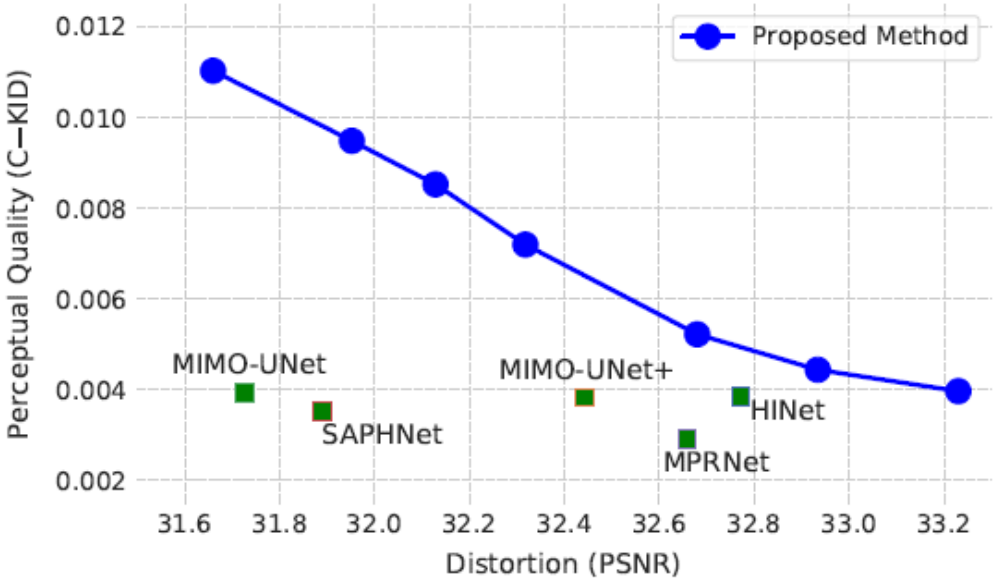
- Evaluation
  - Image regression to the mean
    - Ill-posed problem에 대해 하나의 정답을 유추하려면 여러 정답들의 평균을 따라간다

	a	b	$a \times b$
Case 1	10	10	100
Case 2	20	5	10
Case 3	25	4	100
Average	18.33	6.33	~~116.03~~

  - Different models
    - T Step ↑ with noise level ↓ = better perceptual quality
      - C-KID, LPIPS, NIQE, FID
      - "Ours" model, T = 500
    - T Step ↓ with noise level ↑ = lower distortion
      - PSNR, SSIM
      - "Ours-SA" model, T = 10 with sample averaging
    - No retraining or fine-tuning needed
      - Only changing inference hyperparameter

Diffusion in Deblurring

- Evaluation
  - "Ours" model
    - better perceptual quality (C-KID, LPIPS, NIQE, FID)
  - "Ours-SA" model
    - lower distortion (PSNR, SSIM)
    - 기존의 deterministic model들과 정량 지표 비교를 위해 제안



Diffusion in Deblurring

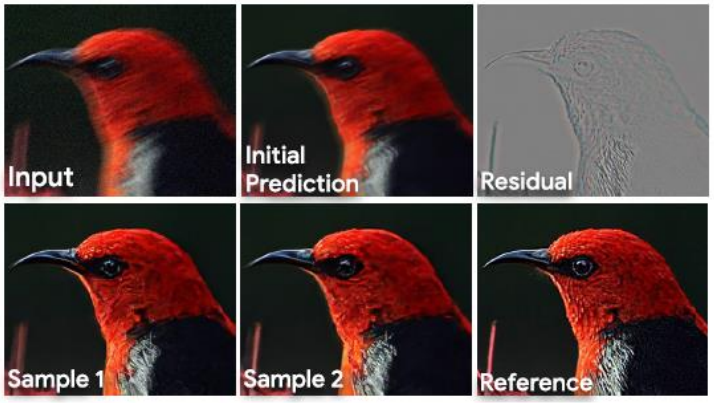
- Discussion
  - Human study for qualitative evaluation
    - Preference rate

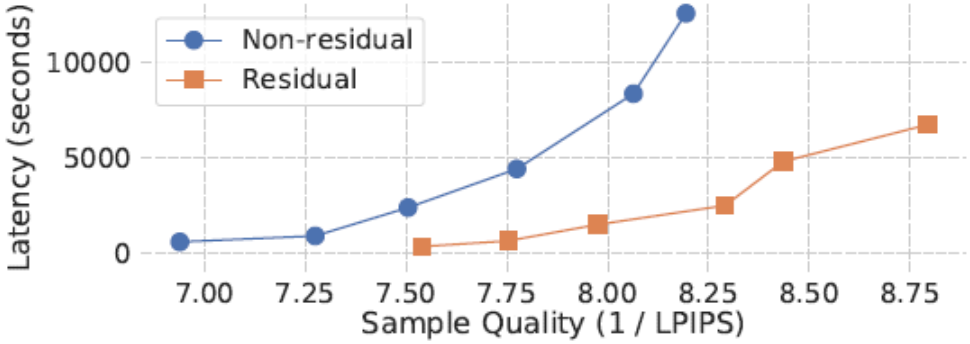
	HINet	MPRNet	Ours	Ours-SA	Reference
HINet [11]	-	54.9	29.1	31.0	14.5
MPRNet [73]	45.1	-	26.6	25.3	11.9
Ours	70.9	73.4	-	58.8	37.1
Ours-SA	69.0	74.7	41.2	-	26.7
Reference	85.5	88.1	62.9	73.3	-

    - Per-pixel standard deviation

Diffusion in Deblurring

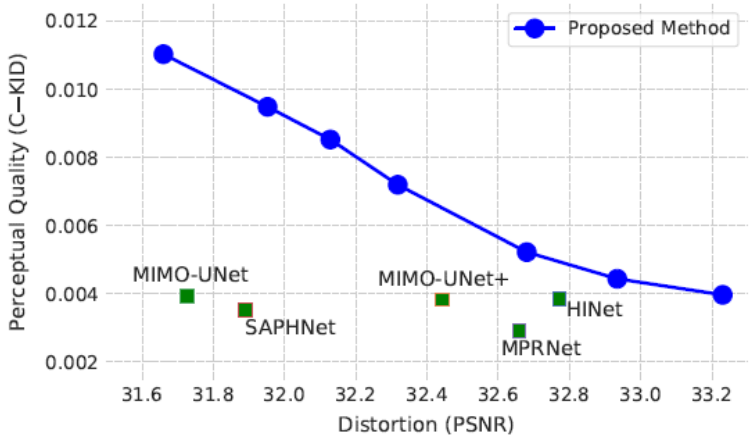
- Discussion
  - Benefits of residual modeling
    - Reduction in the computational cost of sampling
      - Diffusion sampling에 사용되는 denoising network는 여러 번 학습되어야 함
      - 최대한 cost가 낮도록 설계
        - ✓대신 initial predictor를 깊게 설계
          - Network params: initial predictor: 33M / residual model: 28M
    - Residual images are simpler to model





Diffusion in Deblurring

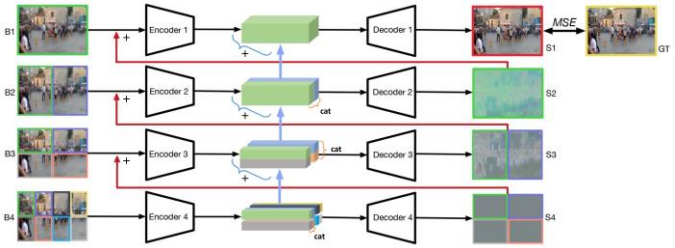
- Conclusion
  - Predict-and-refine
    - Initial predictor + diffusion denoising network
    - Residual learning을 하여 cost efficiency 향상
    - Ill-posed problem을 해결하기 위해 multiple output 생성
  - Perception-Distortion trade-off
    - 동일한 weight로 inference parameter를 설정하여 목적에 따라 다른 output 이미지 생성
      - Higher perception quality
      - Lower distortion

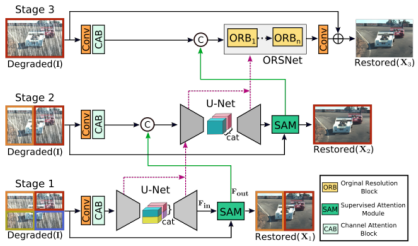


[1] Whang, Jay, et al. "Deblurring via stochastic refinement." *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2022.
[2] Zhang, Hongguang, et al. "Deep stacked hierarchical multi-patch network for image deblurring." Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2019.
Zamir, Syed Waqas, et al. "Multi-stage progressive image restoration." Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. 2021.
[4] Lin, Yen-Yu, Chung-Chi Tsai, and Chia-Wen Lin. "BANet: Blur-aware attention networks for dynamic scene deblurring." *arXiv preprint arXiv:2101.07518* (2021).
[5] Cho, Sung-Jin, et al. "Rethinking coarse-to-fine approach in single image deblurring." *Proceedings of the IEEE/CVF international conference on computer vision*. 2021.

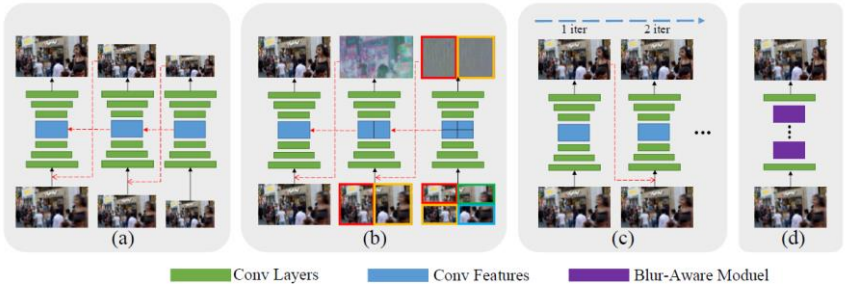
XYDeblur [1]

- Related Works
  - U-Net in deblurring
    - 기존의 image restoration task에서 U-Net 형태의 구조 자주 사용
      - DMPHN[2], MPRNet[3]





- 효율적인 U-Net 형태를 어떻게 구성할 수 있는지에 대한 연구도 많이 존재
  - BANet[4], MINO-Unet[5]



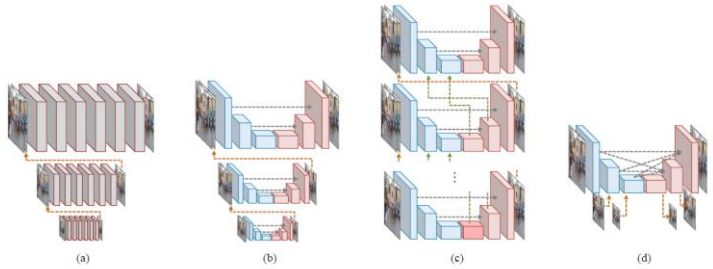
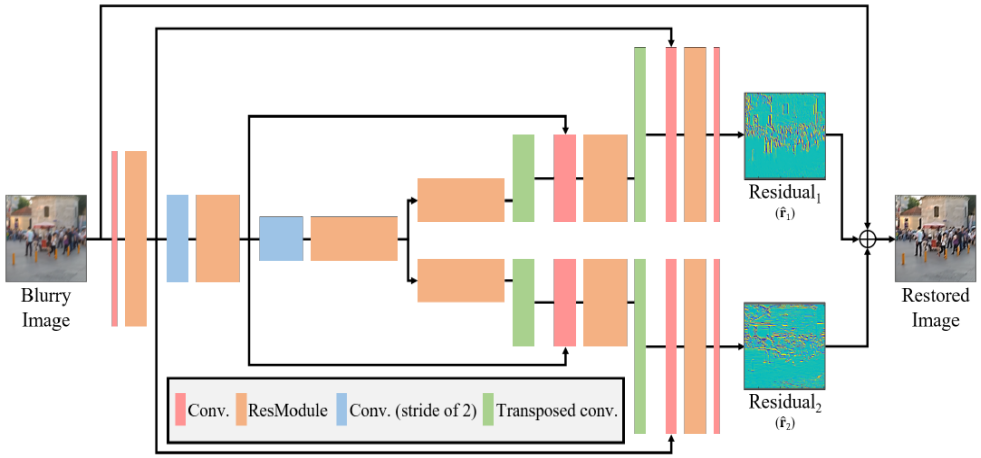


Figure 2. Comparison of coarse-to-fine image deblurring network architectures: (a) DeepDeblur, (b) PSS-NSC, (c) MT-RNN, and (d) proposed MIMO-UNet.

XYDeblur

- Proposed Method
  - U-Net의 새로운 구조 제안
    - One encoder – one decoder → one encoder – two decoder로 변경
      - 각각 horizontal / vertical 특성에 집중
        - ✓$\mathcal{D}_{hor}, \mathcal{D}_{ver}$ 두 가지 decoder로 구성
    - 여러 U-Net 기반 deblurring network에 적용 가능
      - Params 증가 없이 성능 향상



XYDeblur

- Proposed Method
  - 2D blur decoupling
    - Linear span theory
      - Multiple independent regression outputs a larger outer space
        - ✓Two separated decoder를 사용하자
          - $\mathcal{D}_{hor}, \mathcal{D}_{ver}$ 두 가지 decoder로 구성
    - Residual learning
      - Latent variable $\mathbf{z} = \mathcal{E}(\mathbf{x}; \Theta_e)$,
      - Residual image $\hat{\mathbf{r}} = \mathcal{D}_{\text{hor}}(\mathbf{z}; \Theta_{\text{hor}}) + \mathcal{D}_{\text{ver}}(\mathbf{z}; \Theta_{\text{ver}})$,

XYDeblur

- Spatial kernel rotation for parameter sharing
  - 과연 feature decoupling이 원하는 대로 될까…?
    - Naive U-Net^{2D} 적용
      - No! color shift 발생
    - XYDeblur 적용
      - Weight rotation

$$\hat{\mathbf{r}} = \mathcal{D}_{\text{hor}}(\mathbf{z}; \Theta_{\text{hor}}) + \mathcal{D}_{\text{ver}}(\mathbf{z}; \Theta_{\text{ver}}),$$

$$\hat{\mathbf{r}} = \mathcal{D}_{\text{hor}}(\mathbf{z}; \Theta_{\text{d}}) + \mathcal{D}_{\text{ver}}(\mathbf{z}; \Theta_{\text{rd}}),$$

      - Θ_{rd}: counterclockwise 90° rotated Θ_r
        - ✓Axial information을 제외하고 모든 특성 공유
        - ✓Parameter sharing하여 decoder에 추가 parameter 요구 X
      - 다른 U-Net 구조에도 적용 가능할까..?

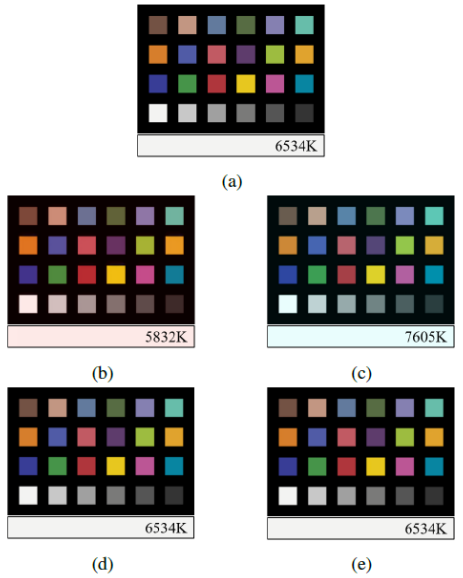


Figure 3. Input ColorChecker image and resultant images. The magnified color white of the chart and its color temperature are shown in the bottom row. (a) ColorChecker image; (b) input + $\hat{\mathbf{r}}_1$ of U-Net^{2D}; (c) input + $\hat{\mathbf{r}}_2$ of U-Net^{2D}; (d) input + $\hat{\mathbf{r}}_1$ of XYDeblur; (e) input + $\hat{\mathbf{r}}_2$ of XYDeblur.

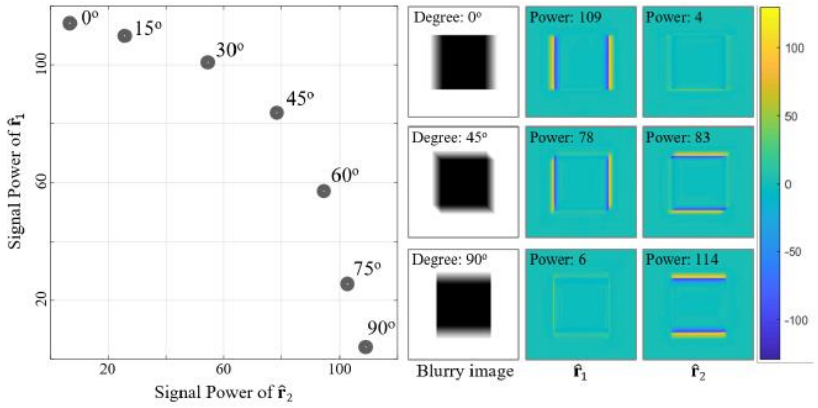
XYDeblur

- Proposed Method
  - 구현 코드

```
def forward(self, x):
    output = list()

    # Common encoder
    x_e1 = self.Encoder1(x)
    x_e2 = self.Encoder2(x_e1)
    x_decomp = self.Encoder3(x_e2)

    # Resultant image reconstruction
    x_decomp1 = self.Decoder1_1(x_decomp)
    x_decomp1 = self.Convs1_1(torch.cat([x_decomp1, x_e2], dim=1))
    x_decomp1 = self.Decoder1_2(x_decomp1)
    x_decomp1 = self.Convs1_2(torch.cat([x_decomp1, x_e1], dim=1))
    x_decomp1 = self.Decoder1_3(x_decomp1)
    x_decomp1 = self.Decoder1_4(x_decomp1)

    x_decomp_rot = x_decomp.transpose(2, 3).flip(2)
    x_e1_rot = x_e1.transpose(2, 3).flip(2)
    x_e2_rot = x_e2.transpose(2, 3).flip(2)

    x_decomp2 = self.Decoder1_1(x_decomp_rot)
    x_decomp2 = self.Convs1_1(torch.cat([x_decomp2, x_e2_rot], dim=1))
    x_decomp2 = self.Decoder1_2(x_decomp2)
    x_decomp2 = self.Convs1_2(torch.cat([x_decomp2, x_e1_rot], dim=1))
    x_decomp2 = self.Decoder1_3(x_decomp2)
    x_decomp2 = self.Decoder1_4(x_decomp2)

    x_decomp2 = x_decomp2.transpose(2, 3).flip(3)

    x_final = x_decomp1 + x_decomp2 + x

    output.append(x_decomp1)
    output.append(x_decomp2)
    output.append(x_final)

    return output
```

XYDeblur

- Experiments
  - Validation of proposed method
    - Synthetically blurred image를 통해 각각 decoder의 효용성 증명
      - $\mathcal{D}_{hor}, \mathcal{D}_{ver}$ 에 의해 생성된 decoupled feature
      - 인위적으로 x방향, y방향으로 blur된 영상에 대해 signal power 확인
        - ✓Signal power: $\frac{1}{M}\|s\|_2^2$
  - Number of decoders
    - 2 decoders is sufficient



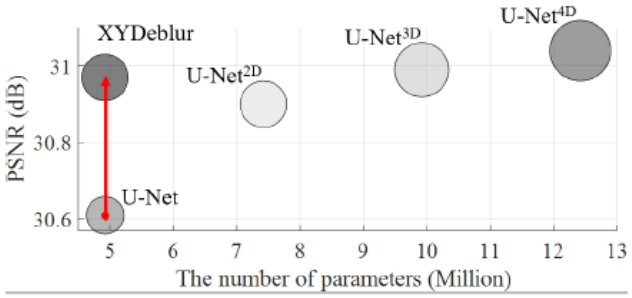
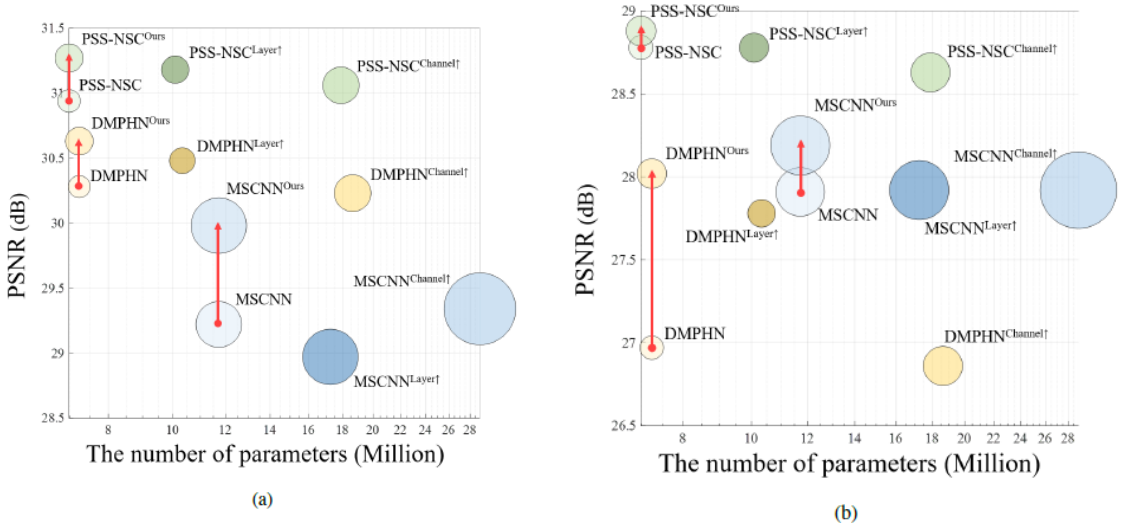


Figure 5. Various number of decoders and their performance. The radius of the circle indicates GMACs of the network.

XYDeblur

- Experiments
  - Generalization
    - 범용적으로 쓰이는 U-Net 구조에 적용 가능한지?
      - PSS-NSC, DMPHN, MSCNN과 같은 일반적인 U-Net 형태의 deblurring network에 적용
  - Deeper network
    - 성능 향상은 단순히 네트워크를 더 깊게 파서 아닐까?
      - $\text{Network}^{\text{Ours}}$, $\text{Network}^{\text{Channel}}$, $\text{Network}^{\text{Layer}}$



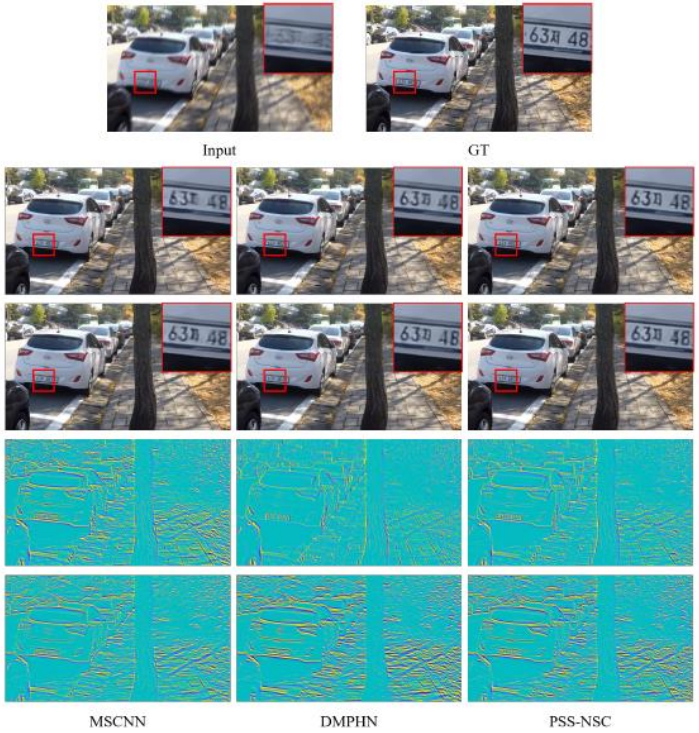
XYDeblur

- Results

Table 2. The proposed approach in the top-performing networks

Method	Variations	GoPro		RealBlur		Complexity†		
		PSNR	SSIM	PSNR	SSIM	Params	GMACs	VRAM
PSS-NSC	Baseline*	$30.94_{(30.92)}$	0.9494	28.78	0.8716	6.98	1167.24	3.79
	Channel†	31.06	0.9509	28.63	0.8667	17.89	3020.81	5.97
	Layer†	31.18	0.9524	28.78	0.8721	10.08	1703.95	5.55
	Ours	31.27	0.9531	28.88	0.8765	6.98	1784.39	5.68
DMPHN	Baseline*	$30.28_{(30.25)}$	0.9408	26.97	0.8170	7.23	1100.18	1.62
	Channel†	30.23	0.9414	26.86	0.8201	18.63	3060.28	2.52
	Layer†	30.48	0.9443	27.78	0.8366	10.33	1509.11	2.30
	Ours	30.63	0.9458	28.02	0.8459	7.23	1754.07	2.41
MSCNN	Baseline*	$29.22_{(29.08)}$	0.9273	27.91	0.8442	11.72	4728.69	3.21
	Channel†	29.34	0.9292	27.92	0.8432	28.94	11673.07	4.84
	Layer†	28.97	0.9250	27.92	0.8449	17.25	6960.32	4.71
	Ours	29.98	0.9382	28.19	0.8550	11.72	6966.13	4.78

* The PSNR in parentheses is the value reported by the authors in their original paper.
† The number of parameters is measured in million.
† GMACs is estimated for the input resolution of 720P (1280×720).
† The usage of VRAM is measured in GB with the input size of 256×256.
The increased and decreased PSNR and SSIM value compared to the baseline method are shown in red and blue, respectively.



XYDeblur

- Conclusion
  - Contribution
    - Feature decoupling을 통해 U-Net 형태의 deblurring network 성능 향상
      - Horizontal + vertical residual learning
      - Params 증가 X
      - Weight rotation (transpose)을 통한 간단한 구현
  - Limitation
    - Params 증가는 없지만 계산량 및 GPU 사용량 증가는 불가피
    - Residual learning 형태로 적용
      - Full image training 진행 시 복잡도 증가 및 성능 하락
    - 다른 형태의 decoder가 존재할 시 적용 어려움 예상
      - Transfer block 등이 존재할 시 어려움 존재