Deep Joint Unrolling for Deblurring and Low-Light Image Enhancement (JUDE) Supplementary

Tu Vo Chan Y. Park
KC Machine Learning Lab
{tuvv, chan.y.park}@kc-ml2.com

Abstract

This supplementary material accompanies the paper "Deep Joint Unrolling for Deblurring and Low-Light Image Enhancement (JUDE)" submitted to WACV 2024. We begin by detailing the derivation of our closed-form solution. Then we provide additional qualitative analysis on both synthetic data [7] and real-world data [5].

1. Derivation of closed-form solution

In this section, we will provide the complete closed-form solutions for each variable **Solution to the Optimization** described in **Section 3.2** of the submitted paper.

Update $\mathcal{P}_{t+1}$: From the original formulation of $\mathcal{P}$:

$$\mathcal{P}_{t+1} = \underset{\mathcal{V}}{\operatorname{argmin}}\, g_P(\mathcal{P}) + \frac{\lambda_2}{2}\|\mathcal{Z}_t - \mathcal{P} \odot \mathcal{Q}_t\|_2^2 + \langle \mathbf{\Gamma}_t, \mathcal{R}_t - \mathcal{P}\rangle + \frac{\lambda_3}{2}\|\mathcal{R}_t - \mathcal{P}\|_2^2$$

$$= \underset{\mathcal{P}}{\operatorname{argmin}}\, g_P(\mathcal{P}) + Tr\left(-\mathbf{\Gamma}_t \mathcal{P}^T\right) + \frac{\lambda_3}{2} Tr(\mathcal{P}\mathcal{P}^T - 2\mathcal{R}_k\mathcal{P}^T) + \frac{\lambda_2}{2} Tr\left((\mathcal{Q}_t \odot \mathcal{P})(\mathcal{Q}_t \odot \mathcal{P})^T - 2\mathcal{Z}_t(\mathcal{P} \odot \mathcal{Q}_t)^T\right)$$

$$= \underset{\mathcal{P}}{\operatorname{argmin}}\, g_P(\mathcal{P}) + Tr\left(-\mathbf{\Gamma}_t \mathcal{P}^T - \lambda_2 \mathcal{Z}_t \odot \mathcal{Q}_t \mathcal{P}^T - \lambda_3 \mathcal{R}_k \mathcal{P}^T\right) + \frac{\mathcal{Q}_t^2\lambda_2 + \lambda_3}{2}||\mathcal{P}||_F^2$$

$$= \underset{\mathcal{P}}{\operatorname{argmin}}\, g_P(\mathcal{P}) + \langle -\mathbf{\Gamma}_t - \lambda_2 \mathcal{Z}_t \odot \mathcal{Q}_t - \lambda_3 \mathcal{R}_k, \mathcal{P}\rangle + \frac{\mathcal{Q}_t^2\lambda_2 + \lambda_3}{2}||\mathcal{P}||_F^2$$

$$= \underset{\mathcal{P}}{\operatorname{argmin}}\, g_P(\mathcal{P}_t) + \frac{\mathcal{Q}_t^2\lambda_2 + \lambda_3}{2}\left(\mathcal{P}_t - \frac{\lambda_2 \mathcal{Z}_t\mathcal{Q}_t + \lambda_3\mathcal{R}_t + \mathbf{\Gamma}_t}{\mathcal{Q}_t^2\lambda_2 + \lambda_3}\right)^2 \quad (1)$$

$$= \mathcal{D}_{g_P}\left(\mathcal{P} - \frac{\mathbf{\Psi}_t}{\mathcal{Q}_t^2\lambda_2 + \lambda_3}\right) \quad (2)$$

where: $\mathbf{\Psi}_t = \lambda_2 \mathcal{Z}_t\mathcal{Q}_t + \lambda_3\mathcal{R}_t + \mathbf{\Gamma}_t$.

The operator $\mathcal{D}_{g_P}$ is associated with regularization functions $g(\cdot)$. As stated in the paper, we use ResUNet [3] to implement the Data Operator $\mathcal{D}$ to learn from training data and effectively reconstruct intricate and diverse visual features.

The same expansion can be applied when solving $\mathcal{Q}_{t+1}$ and $\mathcal{Z}_{t+1}$, resulting in the solutions in Equation 13, and Equation 18 in the submitted paper.

Update $\mathcal{R}_{t+1}$:

$$\mathcal{R}_{t+1} = \underset{\mathcal{R}}{\operatorname{argmin}} \langle \mathbf{\Gamma}_t, \mathcal{R} - \mathcal{P}_{t+1}\rangle + \frac{\lambda_3}{2}\|\mathcal{R} - \mathcal{Q}_t\|_2^2 \quad (3)$$

By differentiating Equation 3 with respect to $\mathcal{R}$ and setting derivative to 0, we have

$$\frac{\partial\left(\langle \mathbf{\Gamma}_t, \mathcal{R} - \mathcal{P}_{t+1}\rangle + \frac{\lambda_3}{2}\|\mathcal{R} - \mathcal{Q}_t\|_2^2\right)}{\partial \mathcal{R}} = 0 \quad (4)$$

$$\Leftrightarrow -\mathbf{\Gamma}_t - \lambda_3\mathcal{P}_{t+1}\mathcal{R} + \lambda_3\mathcal{R} = 0 \quad (5)$$

Then the closed-form solution for $\mathcal{R}$ can be derived as:

$$\Leftrightarrow \mathcal{R} = \frac{\lambda_3\mathcal{P}_{t+1} + \mathbf{\Gamma}_t}{\lambda_3} \quad (6)$$

Similarly, we can find the solution for $\mathcal{L}_{t+1}$ by applying the derivative and setting it to 0.

Update $\mathcal{I}_{t+1}$:

$$\mathcal{I}_{t+1} = \underset{\mathcal{I}}{\operatorname{argmin}} \frac{\lambda_1}{2}\|\mathcal{X} - \mathcal{H} \otimes \mathcal{I}_t\|_2^2 + \langle \mathbf{\Delta}, \mathcal{I}_t - \mathcal{Z}_{t+1} \rangle + \frac{\lambda_5}{2}\|\mathcal{I}_t - \mathcal{Z}_{t+1}\|_2^2 \tag{7}$$

$$\Leftrightarrow \lambda_1(\mathcal{H}\mathcal{H}^T\mathcal{I}_t - \mathcal{X}\mathcal{H}^T) + \lambda_5(\mathcal{I}_t - \mathcal{Z}_{t+1}) + \mathbf{\Delta}_t = 0 \tag{8}$$

By applying Fast Fourier transform (FFT) $\mathcal{I}_{t+1}$ has the following solution:

$$\mathcal{I}_{t+1} = \mathcal{F}^{-1}\left\{\frac{\mathcal{F}(\lambda_1\mathcal{H}^T\mathcal{X} + \lambda_5\mathcal{Z}_{t+1} - \mathbf{\Omega}_t)}{\lambda_1\mathcal{F}(\mathcal{H})^2 + \lambda_5}\right\} \tag{9}$$

2. Ablation Study

2.1. Impact of choosing different components

In this section, we evaluate the impact of selecting different CNN architectures for the data modules and the integration of the Reflectance Denoiser. For the data modules, we experimented with IRCNN [6] and ResUNet [3], both of which are common baselines for data priors. As shown in Table 1, ResUNet outperforms IRCNN when being used as the data priors. In addition, adding the Reflectance Denoiser before merging significantly boosts the performance, and delivers the highest image quality, demonstrating the effectiveness of our configuration.

Components			Metrics		
IRCNN	ResUNet	Reflect. Denoiser	**PSNR ↑**	**SSIM ↑**	**LPIPS ↓**
✓			25.644	0.928	0.134
	✓		26.486	0.929	**0.127**
	✓	✓	**26.884**	**0.932**	**0.127**

Table 1. Ablation study on different components of JUDE. Employing ResUNet as the data module greatly enhances the performance. In addition, the combination of ResUNet as the data prior and the reflectance denoiser added provides the best performance.

2.2. Impact of choosing different K

We evaluate the impact of the number of unrolled iterations K on network performance and report it in Table 2. Specifically, we experimented with K values of 3, 5, and 6. As presented in Table 2, with three iterations, the number of learned features and operations was insufficient for accurate restoration, resulting in the poorest performance. As K increases, restoration performance improves due to the greater capacity to learn features. However, further increasing K (such as 6) leads to higher computational complexity and memory usage. This occurs because a larger K proportionally increases the number of operations and parameters, making convergence more difficult and requiring more training samples. Therefore, we selected $K = 5$ in our work to achieve a balance between performance and computational efficiency.

K	**PSNR ↑**	**SSIM ↑**	**LPIPS ↓**
3	26.019	0.923	0.134
5	**26.884**	**0.932**	**0.127**
6	26.393	0.929	0.131

Table 2. Ablation study on different numbers of unrolled iteration K.

2.3. Impact of choosing different loss functions

Finally, we show the advantages of our loss function over simple Mean Absolute Error (MAE) loss by taking out the Fast Fourier Transform (FFT) loss. Table 3 demonstrates the results. The performance of the model when training with MAE alone is worse than the model trained with the combination of MAE and FFT losses. This is because FFT helps capture both global structures and fine details by analyzing the frequency components of an image, preserving important features like edges and textures. Additionally, it is more resilient to large-scale intensity changes, maintaining structural integrity across different scales.

Losses	**PSNR ↑**	**SSIM ↑**	**LPIPS ↓**
MAELoss	26.731	0.922	0.138
MAELoss + FFTLoss	**26.884**	**0.932**	**0.127**

Table 3. Ablation study on different loss functions used to train JUDE.

3. Qualitative Results

In our submitted paper, we have provided sufficient quantitative results (Table 1) and parts of visual comparisons (Figs. 5-6) due to the limit of space. Here, we provide more qualitative analysis results on both the LOL-Blur [7] and Real-LOL-Blur [5] datasets.

3.1. LOL-Blur Dataset Evaluation

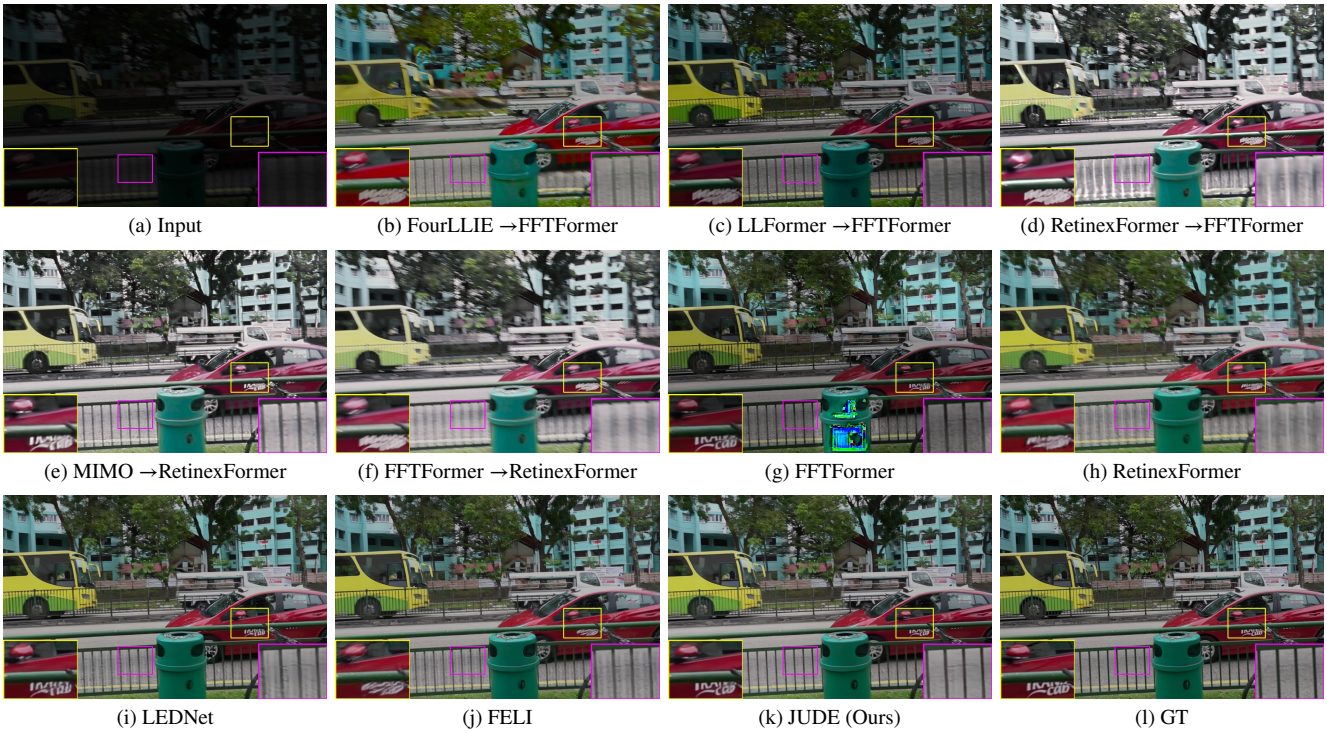
Figures 1, 2, and 3 illustrate JUDE's efficacy compared to other methods on various samples from the LOL-Blur dataset [7]. Separate low-light enhancement and deblurring techniques often struggle with color fidelity and detail preservation. Methods like RetinexFormer [1], FFTFormer [2], LEDNet [7], and FELI [4] exhibit noticeable distortions and fail to reconstruct details in areas with significant motion, even after retraining on the LOL-Blur dataset [7]. In contrast, JUDE accurately recovers details and enhances images to the proper brightness level.

(a) Input (b) FourLLIE →FFTFormer (c) LLFormer →FFTFormer (d) RetinexFormer →FFTFormer

(e) MIMO →RetinexFormer (f) FFTFormer →RetinexFormer (g) FFTFormer (h) RetinexFormer

(i) LEDNet (j) FELI (k) JUDE (Ours) (l) GT

Figure 1. Visual comparison on LOL-Blur dataset #0082. The yellow box indicates obvious differences and is shown at the bottom-left of each result image. (Zoom in for better visualization.)



(a) Input (b) FourLLIE →FFTFormer (c) LLFormer →FFTFormer (d) RetinexFormer →FFTFormer

(e) MIMO →RetinexFormer (f) FFTFormer →RetinexFormer (g) FFTFormer (h) RetinexFormer

(i) LEDNet (j) FELI (k) JUDE (Ours) (l) GT

Figure 2. Visual comparison on LOL-Blur dataset #0118. The yellow and purple boxes indicate obvious differences and are shown at the bottom of each result image. (Zoom in for better visualization.)

(a) Input (b) FourLLIE →FFTFormer (c) LLFormer →FFTFormer (d) RetinexFormer →FFTFormer

(e) MIMO →RetinexFormer (f) FFTFormer →RetinexFormer (g) FFTFormer (h) RetinexFormer

(i) LEDNet (j) FELI (k) JUDE (Ours) (l) GT

Figure 3. Visual comparison on LOL-Blur dataset #0232. The yellow and purple boxes indicate obvious differences and are shown at the bottom of each result image. (Zoom in for better visualization.)

3.2. Real-World Data Evaluation

Figures 4, 5, 6, 7, 8, and 9 provide a comparative analysis of JUDE's performance against other methods on the Real-LOL-Blur dataset [5]. Various combinations of low-light enhancement and deblurring techniques consistently produce images that are either color-distorted or remain blurry. Moreover, methods trained or retrained on the LOL-Blur dataset [7] struggle with generalization, failing to accurately reconstruct details and faithfully represent the original scenes. JUDE, however, demonstrates robust generalization capabilities, effectively recovering intricate details and enhancing images to a visually pleasing brightness level, even in challenging real-world conditions.

References

[1] Yuanhao Cai, Hao Bian, Jing Lin, Haoqian Wang, Radu Timofte, and Yulun Zhang. Retinexformer: One-stage retinex-based transformer for low-light image enhancement. In *Int. Conf. Comput. Vis.*, 2023. 2

[2] Zheng Chen, Yulun Zhang, Ding Liu, Jinjin Gu, Linghe Kong, Xin Yuan, et al. Hierarchical integration diffusion model for realistic image deblurring. *Adv. Neural Inform. Process. Syst.*, 36, 2024. 2

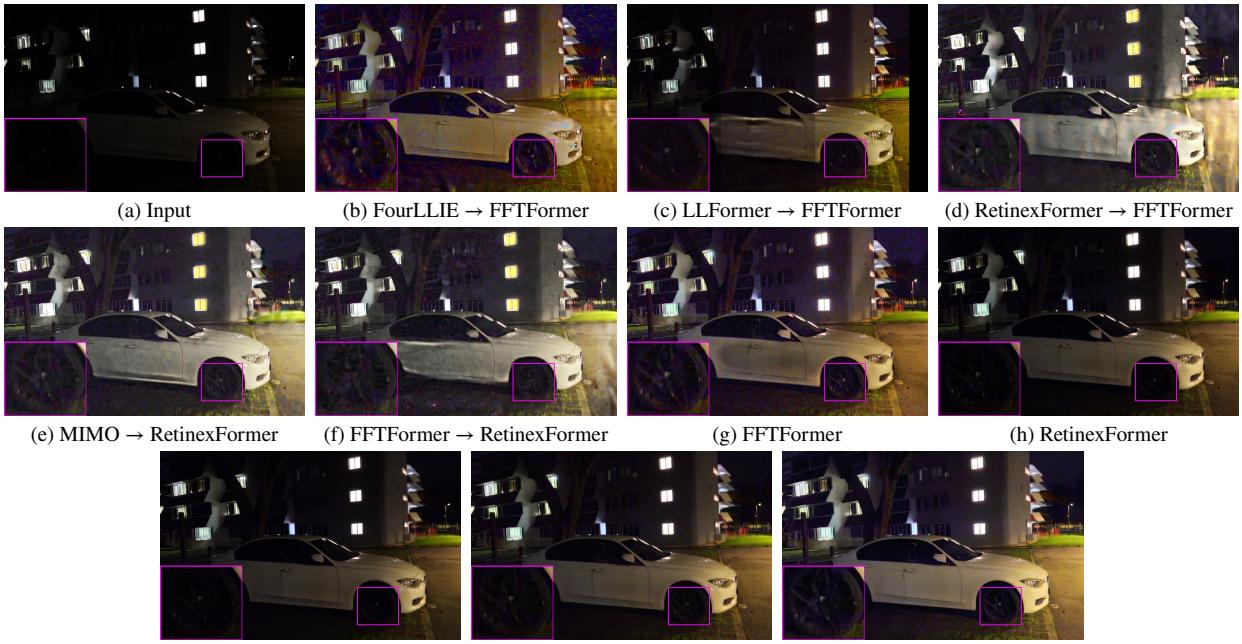
[3] Foivos I Diakogiannis, François Waldner, Peter Caccetta, and Chen Wu. Resunet-a: A deep learning framework for semantic segmentation of remotely sensed data. *ISPRS Journal of Photogrammetry and Remote Sensing*, 162:94–114, 2020. 1, 2

[4] Trung Hoang, Jon McElvain, and Vishal Monga. Fast and physically enriched deep network for joint low-light enhancement and image deblurring. In *ICASSP*, 2024. 2

[5] Jaesung Rim, Haeyun Lee, Jucheol Won, and Sunghyun Cho. Real-world blur dataset for learning and benchmarking deblurring algorithms. In *Eur. Conf. Comput. Vis.*, pages 184–201, 2020. 1, 2, 4, 5, 6, 7, 8, 9

[6] Kai Zhang, Wangmeng Zuo, Shuhang Gu, and Lei Zhang. Learning deep cnn denoiser prior for image restoration. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 3929–3938, 2017. 2

[7] Shangchen Zhou, Chongyi Li, and Chen Change Loy. Lednet: Joint low-light enhancement and deblurring in the dark. In *Eur. Conf. Comput. Vis.*, 2022. 1, 2, 4



(a) Input (b) FourLLIE → FFTFormer (c) LLFormer → FFTFormer (d) RetinexFormer → FFTFormer

(e) MIMO → RetinexFormer (f) FFTFormer → RetinexFormer (g) FFTFormer (h) RetinexFormer

(i) LEDNet (j) FELI (k) JUDE (Ours)

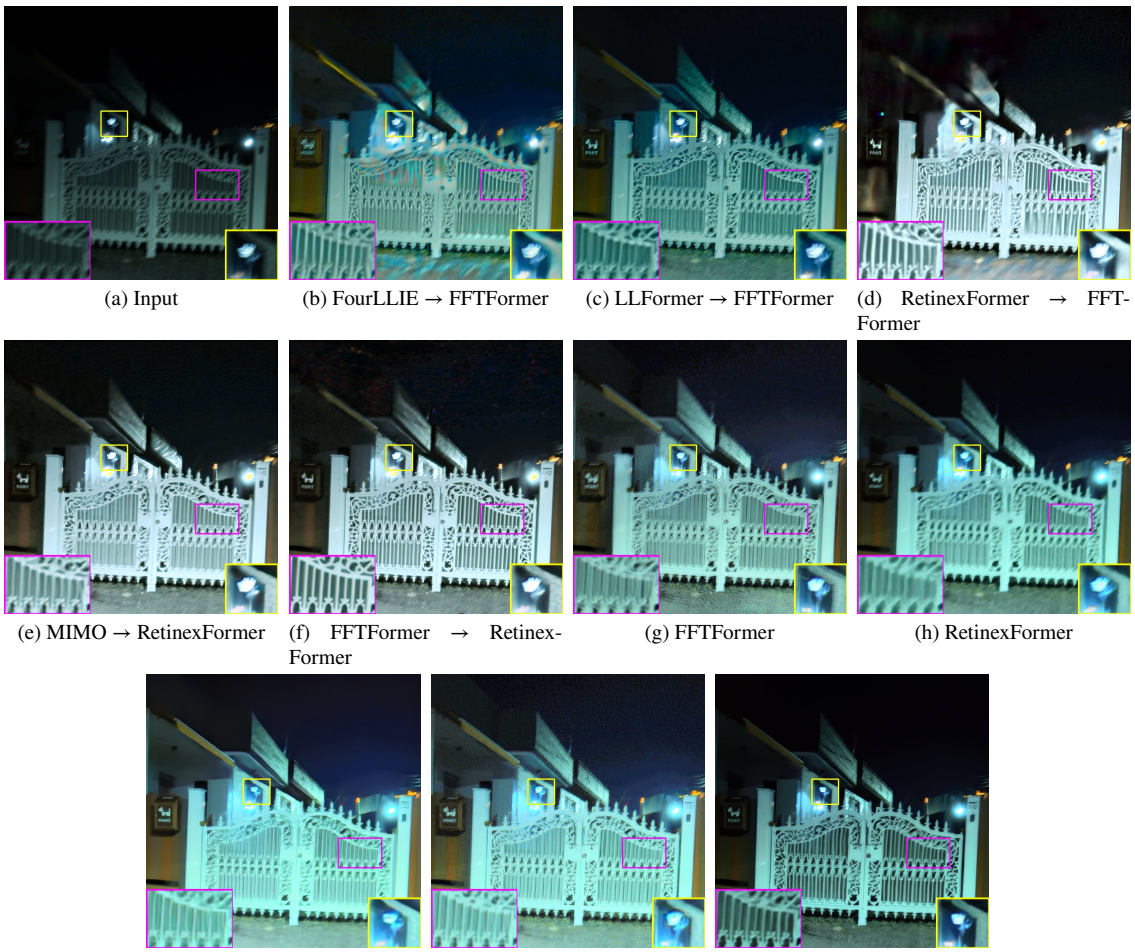
Figure 4. Visual comparison Real-LOL-Blur [5] dataset #C0335. The purple box indicates obvious differences and is shown at the bottom-left of each result image. (Zoom in for better visualization.)

(a) Input (b) FourLLIE → FFTFormer (c) LLFormer → FFTFormer (d) RetinexFormer → FFTFormer

(e) MIMO → RetinexFormer (f) FFTFormer → RetinexFormer (g) FFTFormer (h) RetinexFormer

(i) LEDNet (j) FELI (k) JUDE (Ours)

Figure 5. Visual comparison Real-LOL-Blur [5] dataset #C0326. The yellow box indicates obvious differences and is shown at the bottom-left of each result image. (Zoom in for better visualization.)



(a) Input (b) FourLLIE → FFTFormer (c) LLFormer → FFTFormer (d) RetinexFormer → FFTFormer

(e) MIMO → RetinexFormer (f) FFTFormer → RetinexFormer (g) FFTFormer (h) RetinexFormer

(i) LEDNet (j) FELI (k) JUDE (Ours)

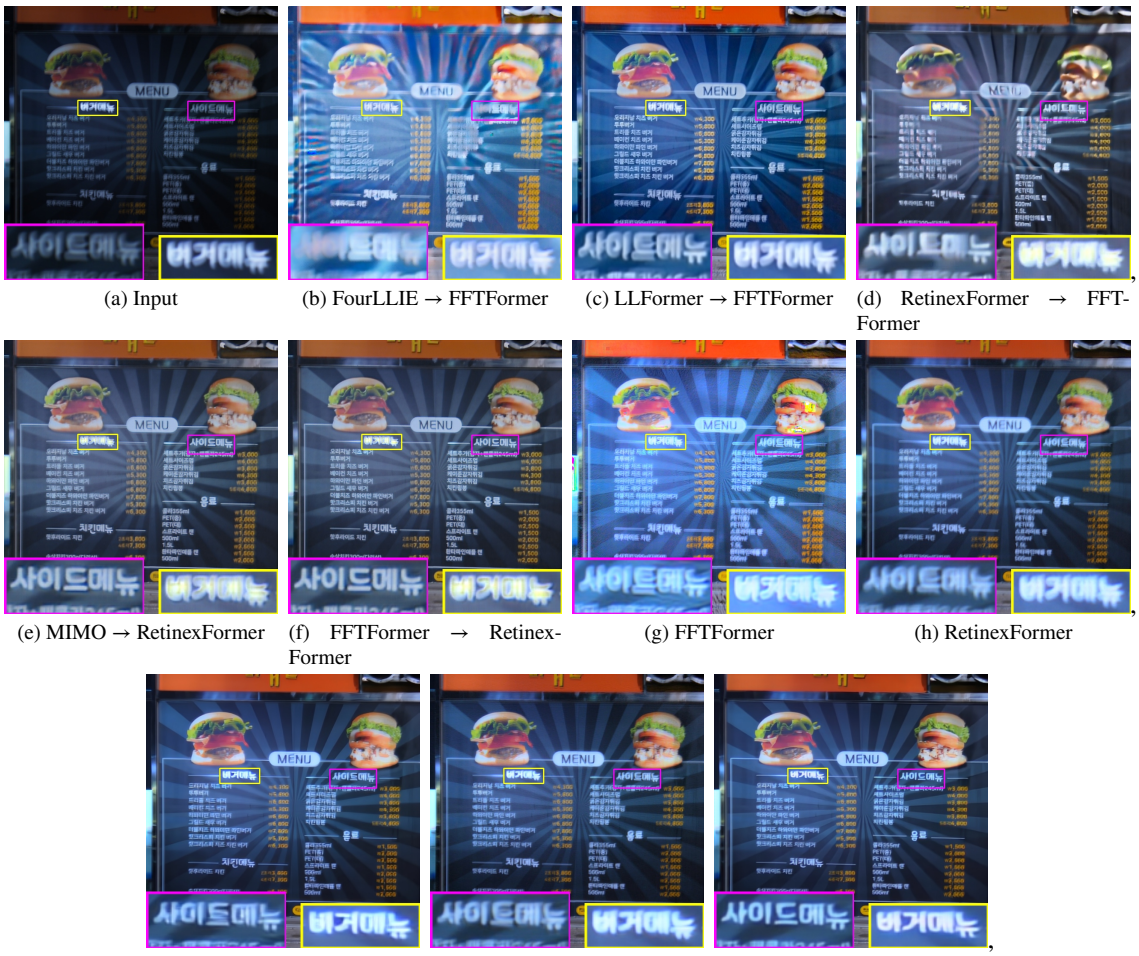
Figure 6. Visual comparison Real-LOL-Blur [5] dataset #232. The yellow and purple boxes indicate obvious differences and are shown at the bottom of each result image. (Zoom in for better visualization.)

(a) Input (b) FourLLIE → FFTFormer (c) LLFormer → FFTFormer (d) RetinexFormer → FFTFormer

(e) MIMO → RetinexFormer (f) FFTFormer → RetinexFormer (g) FFTFormer (h) RetinexFormer

(i) LEDNet (j) FELI (k) JUDE (Ours)

Figure 7. Visual comparison Real-LOL-Blur [5] dataset #207. The purple box indicates obvious differences and is shown at the bottom-right of each result image. (Zoom in for better visualization.)

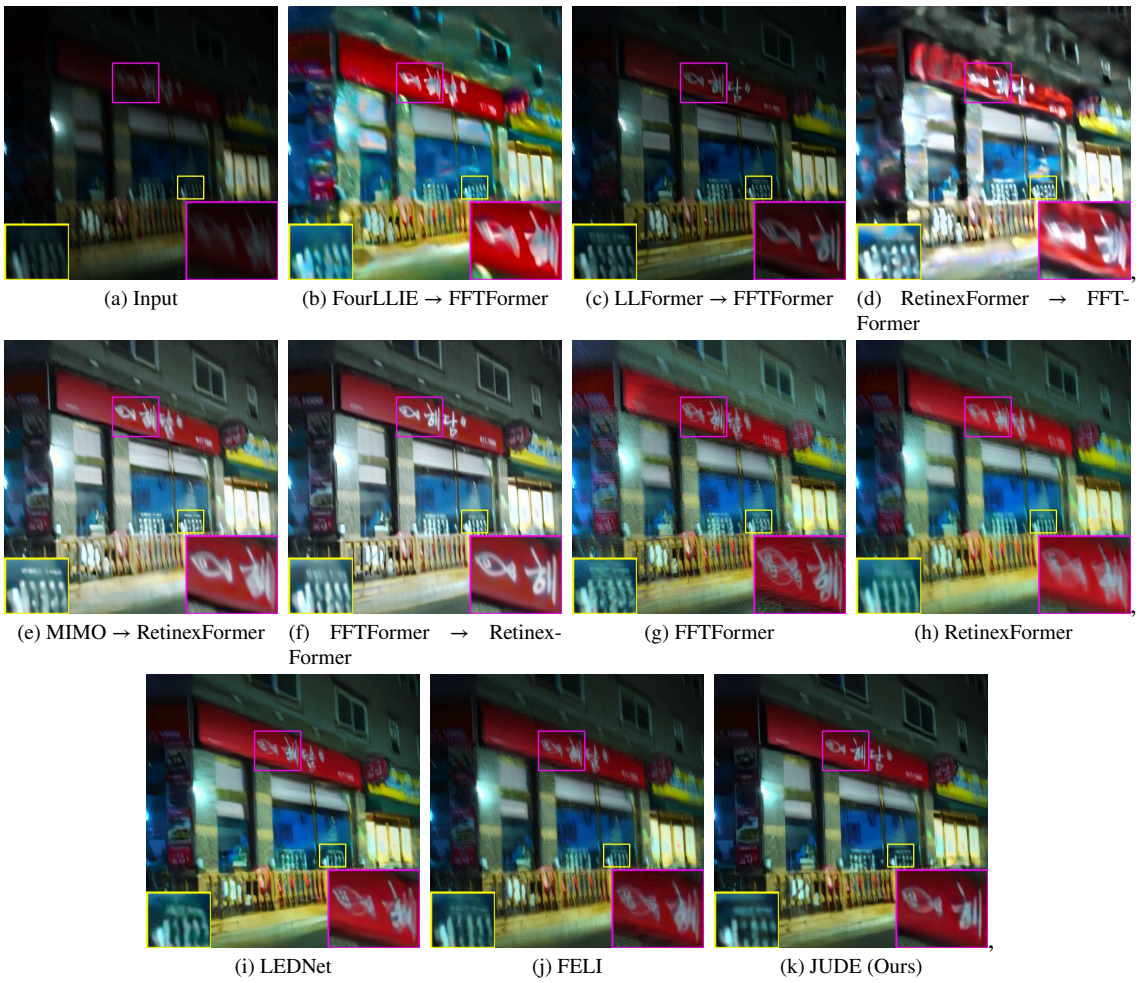


(a) Input (b) FourLLIE → FFTFormer (c) LLFormer → FFTFormer (d) RetinexFormer → FFTFormer

(e) MIMO → RetinexFormer (f) FFTFormer → RetinexFormer (g) FFTFormer (h) RetinexFormer

(i) LEDNet (j) FELI (k) JUDE (Ours)

Figure 8. Visual comparison Real-LOL-Blur [5] dataset #205. The yellow and purple boxes indicate obvious differences and are shown at the bottom of each result image. (Zoom in for better visualization.)



(a) Input (b) FourLLIE → FFTFormer (c) LLFormer → FFTFormer (d) RetinexFormer → FFTFormer

(e) MIMO → RetinexFormer (f) FFTFormer → RetinexFormer (g) FFTFormer (h) RetinexFormer

(i) LEDNet (j) FELI (k) JUDE (Ours)

Figure 9. Visual comparison Real-LOL-Blur [5] dataset #188. The purple box indicates obvious differences and is shown at the bottom-right of each result image. (Zoom in for better visualization.)