Exposure-slot: Exposure-centric representations learning with Slot-in-Slot Attention for Region-aware Exposure Correction -Supplementary Material-

Anonymous CVPR submission

Paper ID 10942

A. Encoder and Decoders

We use an Image Encoder (**Enc**), an Image Decoder ($\mathbf{Dec}_{enhance}$) , and a Slot Decoder ($\mathbf{Dec}_{slot}$) for slot reconstruction as the backbone network of Exposure-slot.

Table S1 and S2 presents the detailed architecture of the **Enc**, $\mathbf{Dec}_{enhance}$ and $\mathbf{Dec}_{slot}$. The Conv-block includes a convolution operation with a stride of 1 and padding of 1, followed by the GeLU [4] activation function.

Stage	Operations	Outputs
Enc-1	Conv-block, 3×3	$h \times w \times 32$
	Conv-block, 3×3	$h \times w \times 32$
	batchnorm2d(32)	$h \times w \times 32$
Enc-2	Conv-block, 3×3	$h \times w \times 16$
	PixelShuffle(2)	$h/2 \times w/2 \times 64$
	Conv-block, 3×3	$h/2 \times w/2 \times 64$
	batchnorm2d(64)	$h/2 \times w/2 \times 64$
Enc-3	Conv-block, 3×3	$h/2 \times w/2 \times 32$
	PixelShuffle(2)	$h/4 \times w/4 \times 128$
	Conv-block, 3×3	$h/4 \times w/4 \times 128$
	batchnorm2d(128)	$h/4 \times w/4 \times 128$
$\mathbf{Dec}_{enhance}$-1	Conv-block, 3×3	$h/4 \times w/4 \times 128$
	PixelUnshuffle(2)	$h/2 \times w/2 \times 64$
-	*Skip-connection* with Enc-2	$h/2 \times w/2 \times 128$
$\mathbf{Dec}_{enhance}$-2	Conv-block, 3×3	$h/2 \times w/2 \times 128$
	PixelUnshuffle(2)	$h \times w \times 32$
	Conv-block, 3×3	$h \times w \times 32$
	Conv-block, 3×3	$h \times w \times 32$
-	*Skip-connection* with Enc-1	$h \times w \times 64$
$\mathbf{Dec}_{enhance}$-3	Conv-block, 3×3	$h \times w \times 32$
	Conv-block, 3×3	$h \times w \times 32$
	Conv-block, 1×1	$h \times w \times 3$

Table S1. Specification of Image Encoder (**Enc**) and Decoder ($\mathbf{Dec}_{enhance}$) architecture.

Stage	Operations	Outputs
$\mathbf{Dec}_{slot}$-1	PixelUnshuffle(2)	$h/2 \times w/2 \times 64$
	Conv-block, 3×3	$h/2 \times w/2 \times 64$
	Conv-block, 3×3	$h/2 \times w/2 \times 64$
$\mathbf{Dec}_{slot}$-2	PixelUnshuffle(2)	$h \times w \times 32$
	Conv-block, 3×3	$h \times w \times 32$
	Conv-block, 3×3	$h \times w \times 16$
$\mathbf{Dec}_{slot}$-3	Conv-block, 3×3	$h \times w \times 3$

Table S2. Specification of Slot Decoder ($\mathbf{Dec}_{slot}$) architecture.

B. More Results of Each Structural Levels

In the main manuscript, we set the default configuration of the SSAB block to 2-level. Additionally, in Sec.4.3 of the main manuscript, Table 5, we validated the effectiveness of different structural levels on the SICE [3] dataset. Furthermore, in Table S3, we provide additional results for 1 and 3-level SSAB on the MSEC [1] and LCDPNet [8] datasets.

Dataset	Model	K^{main}	$K^{sub\text{-}1}$	$K^{sub\text{-}2}$	PSNR↑	SSIM↑
SICE [3]	1-level	3	-	-	22.02	0.7131
	2-level	3	7	-	22.81	0.7236
	3-level	3	7	10	**23.06**	**0.7306**
MSEC [1]	1-level	3	-	-	23.04	0.8668
	2-level	3	7	-	23.18	0.8697
	3-level	3	7	10	**23.25**	**0.8700**
LCDP [8]	1-level	3	-	-	23.81	0.8596
	2-level	3	7	-	24.03	0.8592
	3-level	3	7	10	**24.13**	**0.8629**

Table S3. Extended version of Table 5 in the main manuscript. Ablation studies on not only SICE [3] dataset but also MSEC [1] and LCDP [8] dataset is additionally provided.

As the level increases, the performance metric values for PSNR and SSIM improve across all benchmark

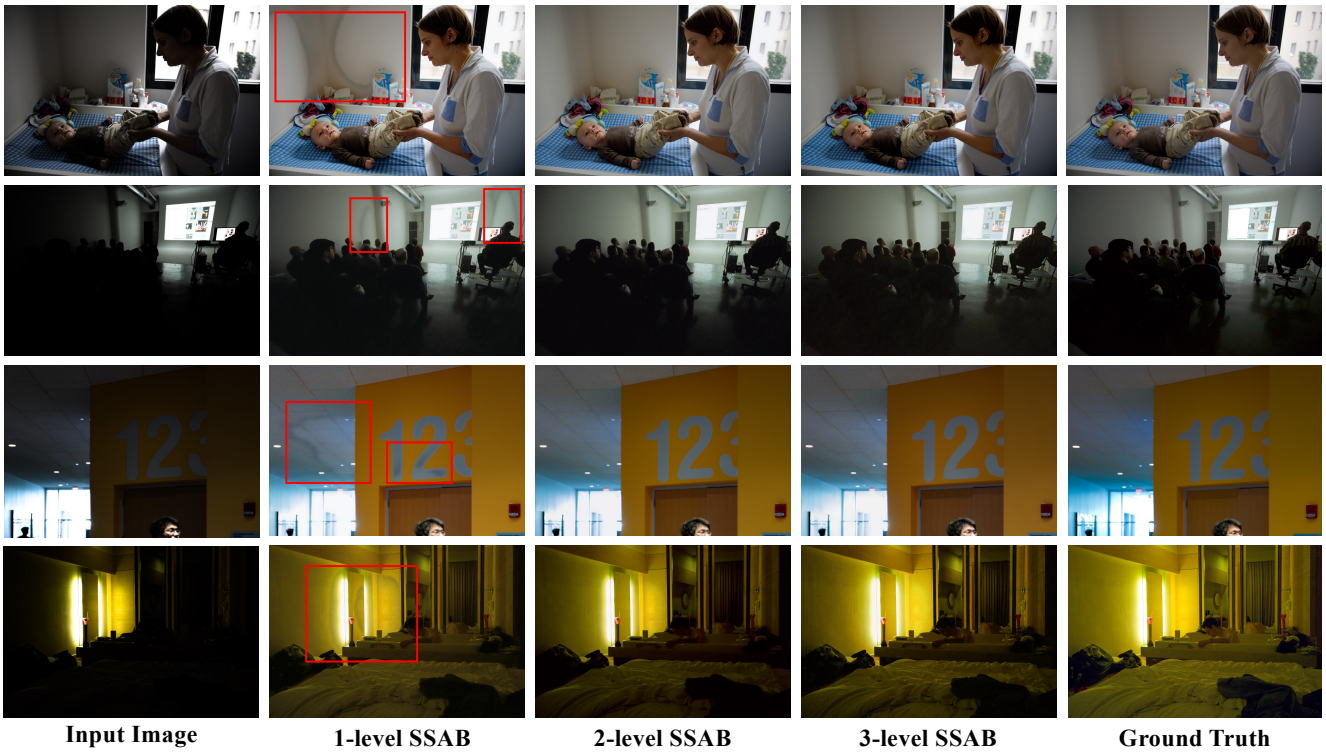

Figure S1. Visual comparisons across different n-level SSAB structures. ($n = 1, 2, 3$)

datasets [1, 3, 8]. Notably, the 3-level SSAB achieves a 0.07 performance gain in SSIM on the SICE dataset compared to the 2-level SSAB. Depending on computer resources, users can set SSAB levels beyond 2-levels to achieve better results.

Model	Params (M)	FLOPs (G)	Time (S)
1-level	1.079	14.064	0.0644
2-level	1.229	14.175	0.0676
3-level	1.465	14.618	0.0938

Table S4. Computational cost across different levels of SSAB.

In Table S4, we present the computational cost of different levels of SSAB in terms of parameters, FLOPs (Floating point operations per second), and execution time. The Flops is calculated on $3 \times 256 \times 256$ input, and execution time measurements taken on a single 844×1500 RGB image using an NVIDIA RTX 4090 GPU. The 1-level and the 2-level SSAB, which we used as the default in the main manuscript, show minimal differences in computational cost. However, the 3-level configuration requires slightly more runtime, with an execution time increase of approximately 0.1 seconds.

Additionally, Fig. S1 presents the visual output results corresponding to each level. The 1-level SSAB tends to produce artifacts such as blotches caused by lighting (red boxes), whereas such issues are absent with configurations of 2 and 3-level SSAB. To aid understanding, Fig. S2 visualizes the slot attention maps for each level. In the 1-level SSAB, the attention maps show stark partitioning based on light sources, whereas this issue is resolved in configurations of 2-level SSAB or higher.

C. More Quantitative results on Perceptual Quality

In Table S5, we also provide a perceptual comparison of the results with other methods. The evaluation is conducted on SICE [3] dataset. To measure the perceptual quality, we adopt Learned Perceptual Image Patch Similarity (LPIPS) [9] and Perception Index (PI) [2].

Model	#Params	Time (S)	LPIPS↓	PI↓
ECLNet [6]	0.018	0.1328	0.268	3.346
FECNet [5]	0.150	0.0746	0.298	3.679
CSEC [7]	1.364	2.3633	0.208	2.993
Exposure-slot	1.229	**0.0676**	**0.161**	**2.949**

Table S5. Results of Perceptual Quality

CVPR #10942

Figure S2. Visual comparisons of slot attention maps across different n-level SSAB structures ($n = 1, 2, 3$).

The execution time measurements taken on a single 844×1500 RGB image using an NVIDIA RTX 4090 GPU. Exposure-slot delivers faster processing speeds compared to existing approaches while maintaining high perceptual quality.

D. More Qualitative results.

Fig.S3,S4 and S5 present additional visual results on the LCDP [8] dataset. For comparison, we include LCDPNet [8] and CSEC [7] as baseline methods. LCDPNet shows more robust results against light source diffusion compared to CSEC but struggles with accurate color correction, whereas CSEC excels at color correction but generates artifacts due to light source diffusion. In contrast, our proposed method, Exposure-slot achieves robust correction results, effectively addressing both challenges.

E. Visualization of Slot Attention Maps

We visualize the progressive refinement of each slot attention map. In Fig. S6 – S18, we provide attention maps of main- and sub-slots at each iteration. As shown in figures, attention maps progressively evolve into features optimized for exposure correction, reflecting gradual feature improvement and smoothing with each iteration.

F. Visualization of t-SNE.

This section provides details regarding the t-SNE plots shown in Fig. 5 of the main manuscript. In Fig. 5, t-SNE is visualized based on the feature vectors from $\mathbf{V}$ in Eq. 2. The left side of the figure illustrates the t-SNE plot of $\mathbf{V}$ before applying prompts, while the right side displays the t-SNE plot of $\mathbf{V} \cdot \mathbf{P}^{final}$, representing the prompt applied version.

For further information, in this supplementary material, we provide t-SNE plots for the 1, 2, and 3-level SSAB structure in Fig. S19, S20, and S21. For each figure, the plots from left to right represent features before prompts, features after prompts, prompts, and slots. Features before and after prompts are visualized using the method employed in Fig. 5, while $\mathbf{S}^{\text{final}}$ and $\mathbf{P}^{\text{final}}$ from Eq. 9 are used for slots and prompts, respectively. Additionally, from top to bottom, each row corresponds to plots clustered at the main-slot, sub-slot, and sub2-slot levels. We observe that not only the prompts and slots exhibit distinct distributions and relationships, but the features after prompts also display unique patterns for each level of the SSAB structure. This indicates that as the level increases, the SSAB structure becomes progressively more adept at performing sophisticated exposure correction.

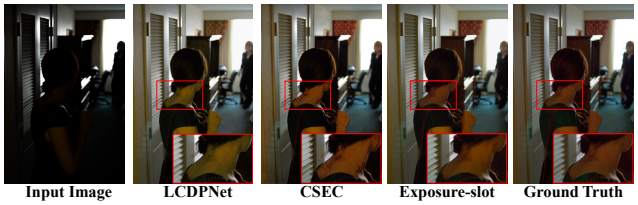

Figure S3. Visual results from LCDP [8] dataset. We utilize LCDPNet [8] and CSEC [7] as comparison methods.

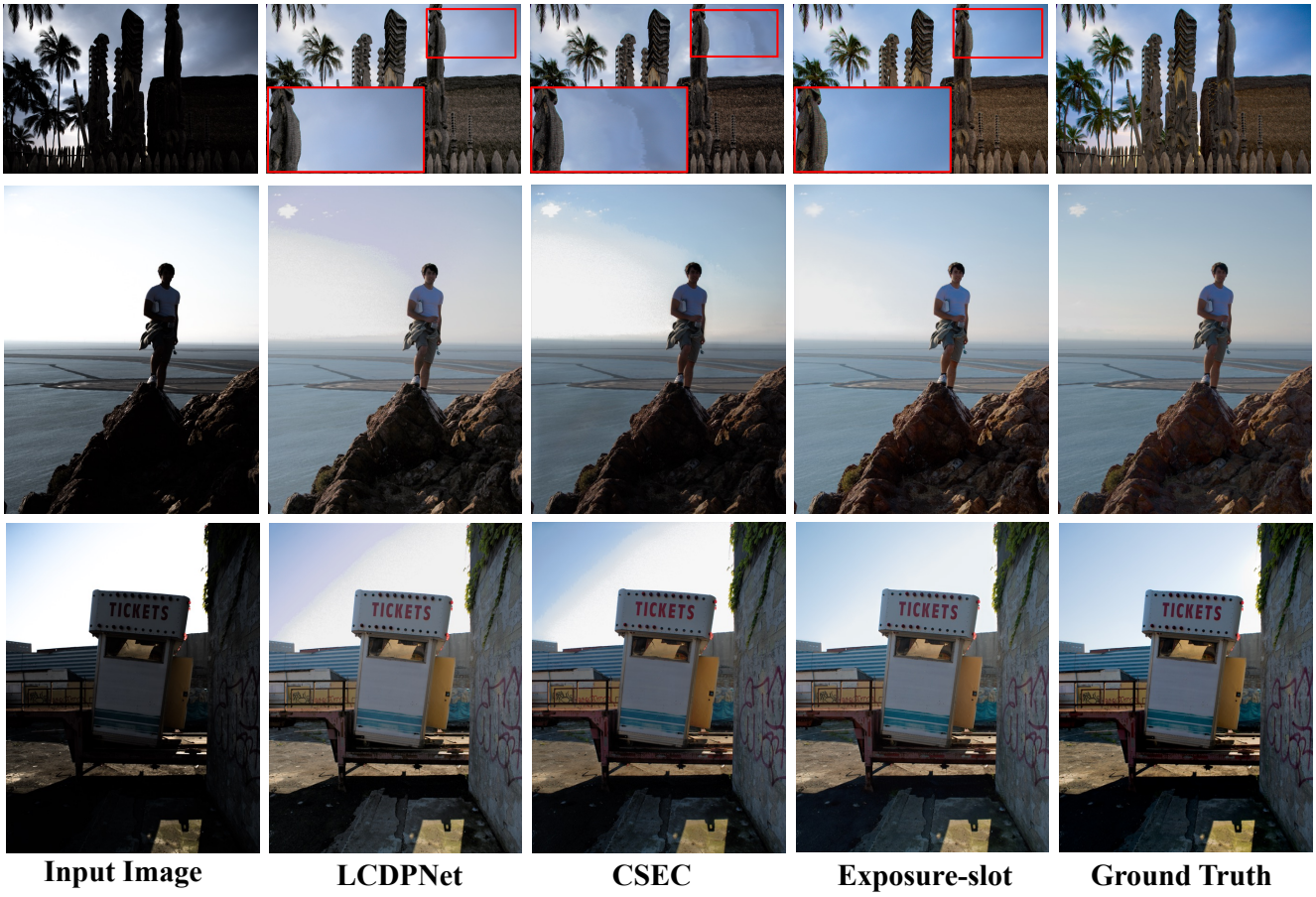

Figure S4. Visual results from LCDP [8] dataset. We utilize LCDPNet [8] and CSEC [7] as comparison methods.

Figure S5. Visual results from LCDP [8] dataset. We utilize LCDPNet [8] and CSEC [7] as comparison methods.

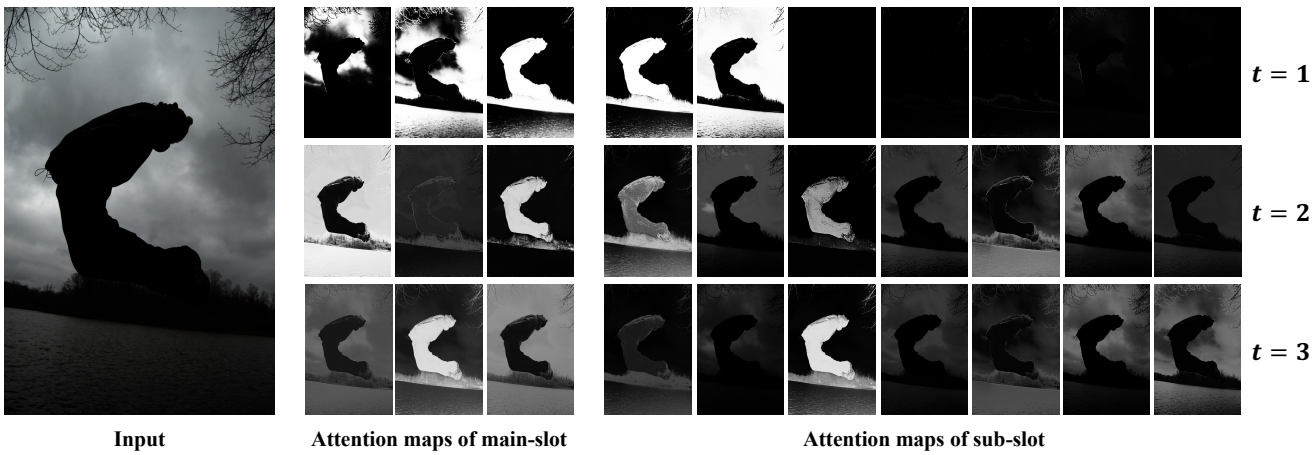

Figure S6. Visual result of attention maps from LCDP [8] dataset. From top to bottom, we present the attention maps of the main-slot and sub-slot at each iteration ($t = 1, 2, 3$).

CVPR
#10942

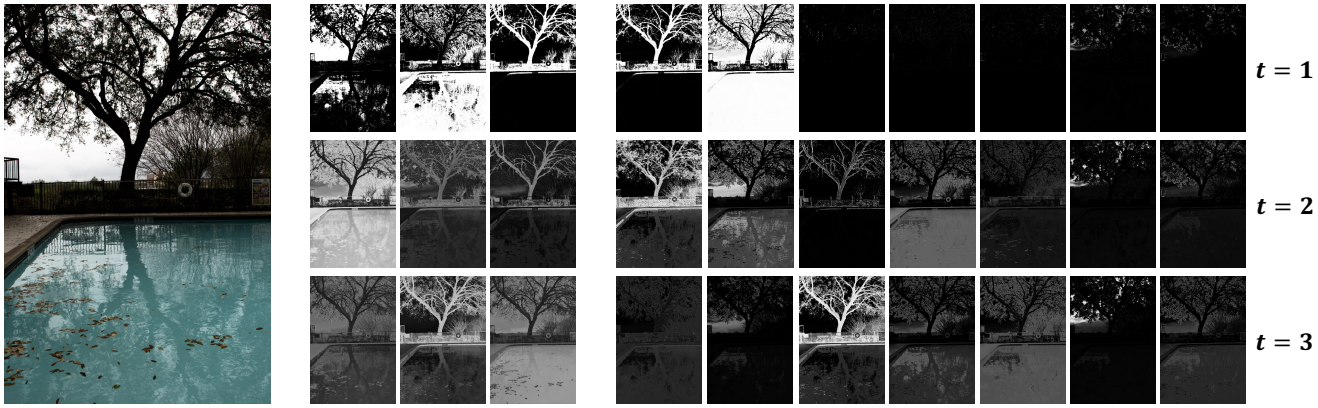

Figure S7. Visual result of attention maps from LCDP [8] dataset. From top to bottom, we present the attention maps of the main-slot and sub-slot at each iteration ($t = 1, 2, 3$).

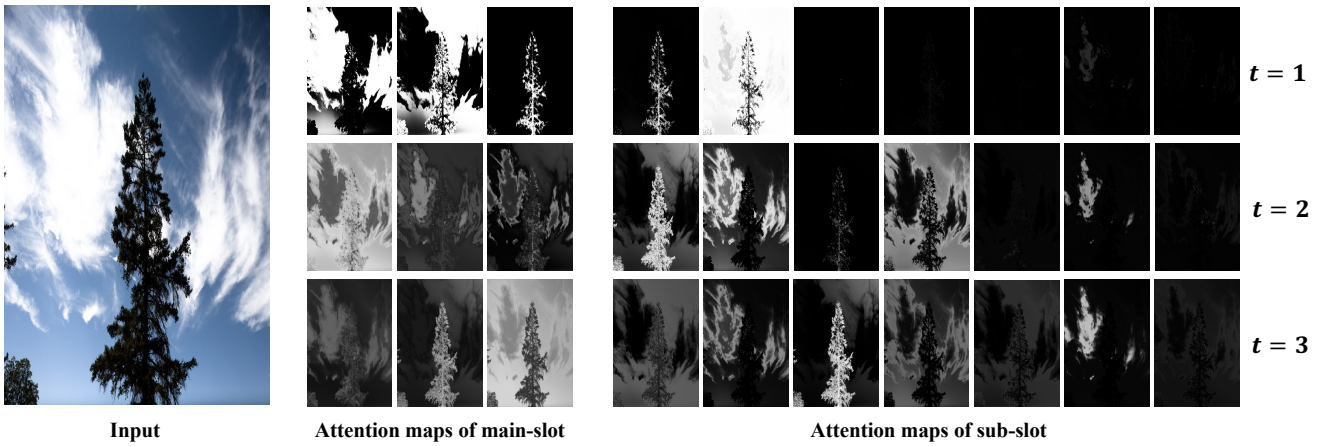

Figure S8. Visual result of attention maps from LCDP [8] dataset. From top to bottom, we present the attention maps of the main-slot and sub-slot at each iteration ($t = 1, 2, 3$).

CVPR
#10942

Figure S9. Visual result of attention maps from LCDP [8] dataset. From top to bottom, we present the attention maps of the main-slot and sub-slot at each iteration ($t = 1, 2, 3$).

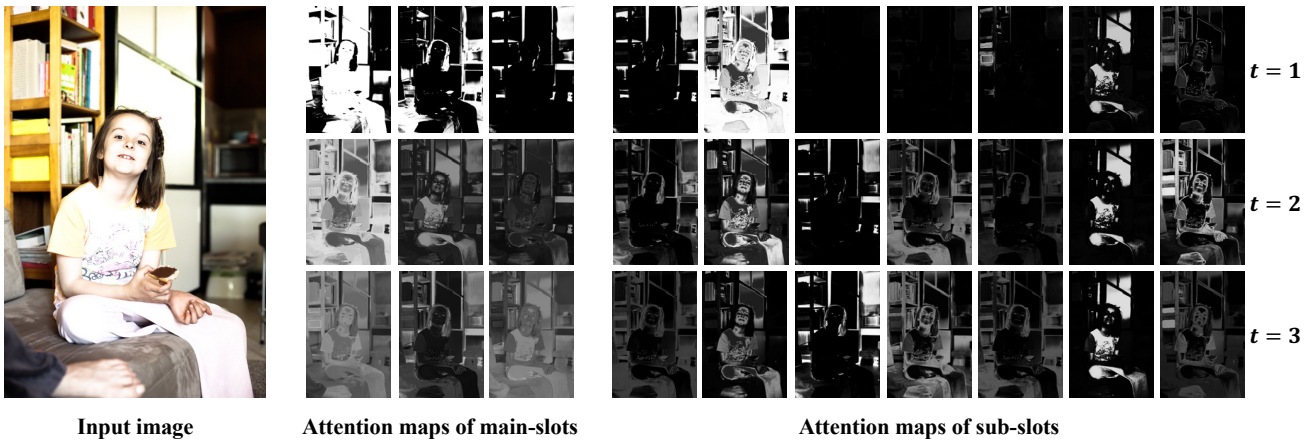

Figure S10. Visual result of attention maps from LCDP [8] dataset. From top to bottom, we present the attention maps of the main-slot and sub-slot at each iteration ($t = 1, 2, 3$).

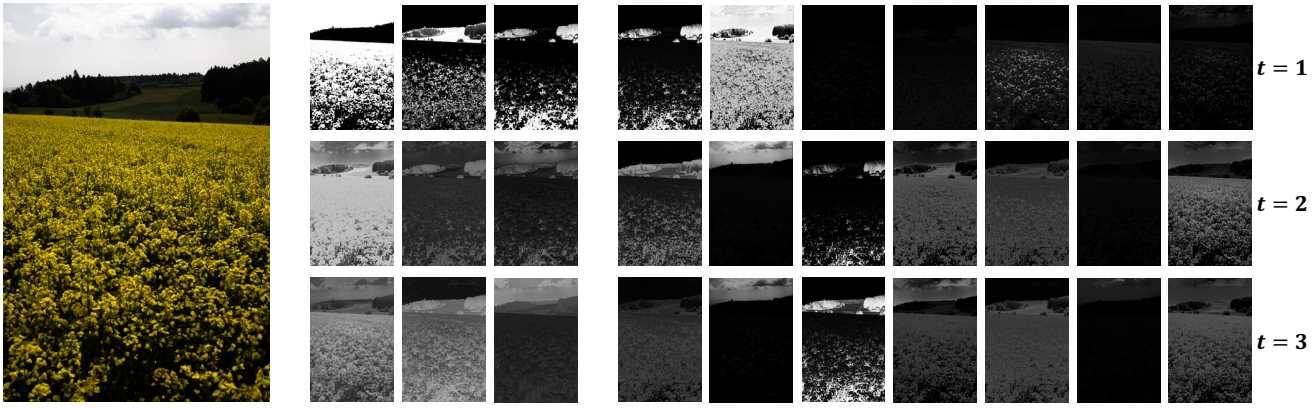

Figure S11. Visual result of attention maps from LCDP [8] dataset. From top to bottom, we present the attention maps of the main-slot and sub-slot at each iteration ($t = 1, 2, 3$).

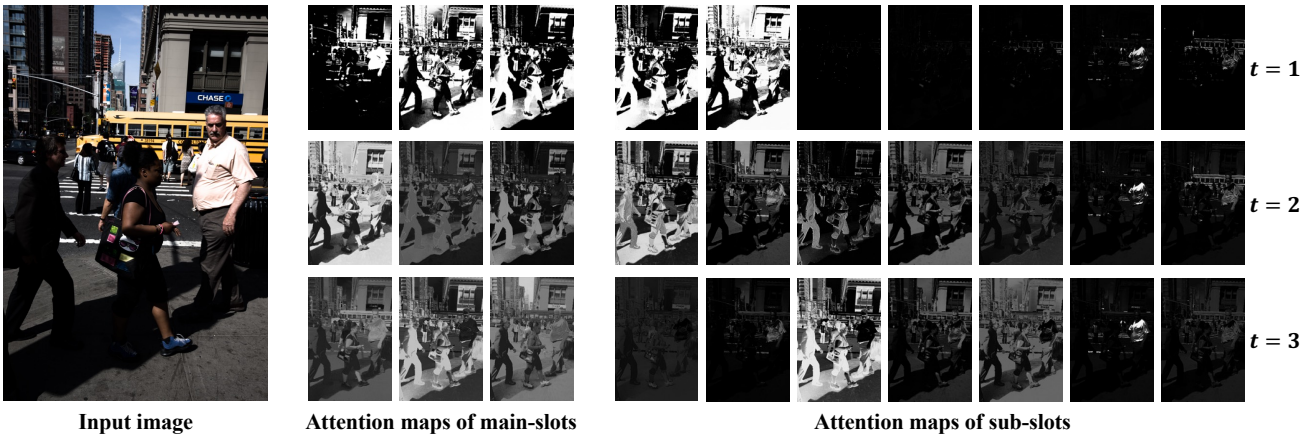

Figure S12. Visual result of attention maps from LCDP [8] dataset. From top to bottom, we present the attention maps of the main-slot and sub-slot at each iteration ($t = 1, 2, 3$).

CVPR
#10942

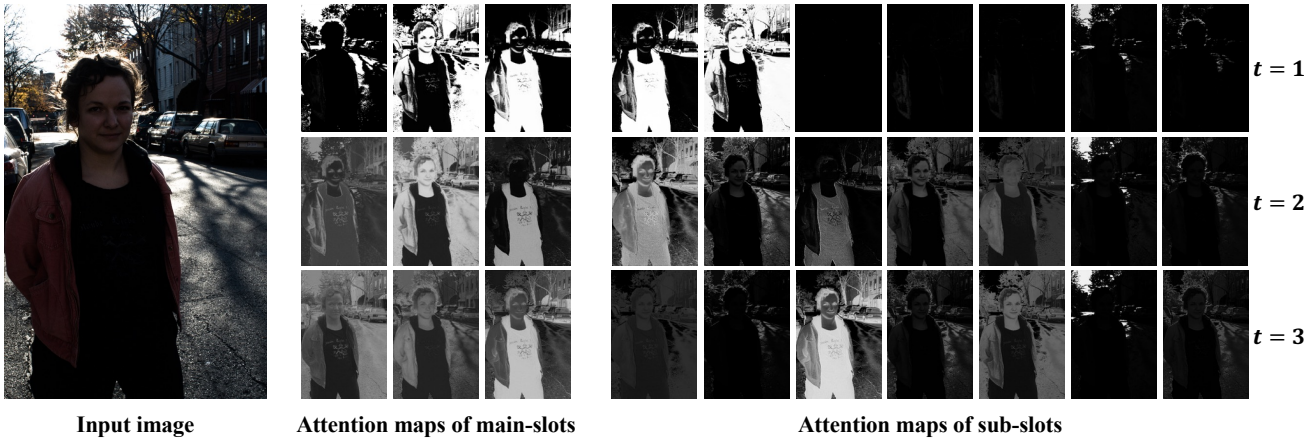

Figure S13. Visual result of attention maps from LCDP [8] dataset. From top to bottom, we present the attention maps of the main-slot and sub-slot at each iteration ($t = 1, 2, 3$).

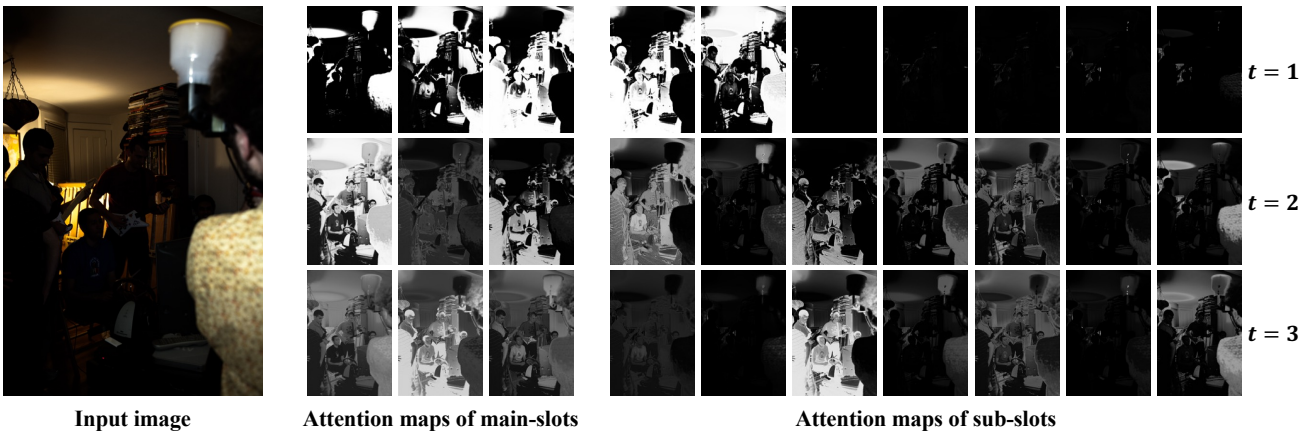

Figure S14. Visual result of attention maps from LCDP [8] dataset. From top to bottom, we present the attention maps of the main-slot and sub-slot at each iteration ($t = 1, 2, 3$).

Figure S15. Visual result of attention maps from LCDP [8] dataset. From top to bottom, we present the attention maps of the main-slot and sub-slot at each iteration ($t = 1, 2, 3$).

CVPR
#10942

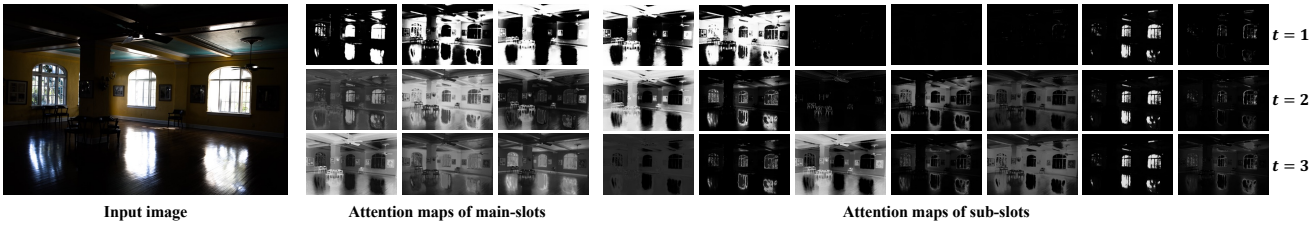

Figure S16. Visual result of attention maps from LCDP [8] dataset. From top to bottom, we present the attention maps of the main-slot and sub-slot at each iteration ($t = 1, 2, 3$).

Figure S17. Visual result of attention maps from LCDP [8] dataset. From top to bottom, we present the attention maps of the main-slot and sub-slot at each iteration ($t = 1, 2, 3$).

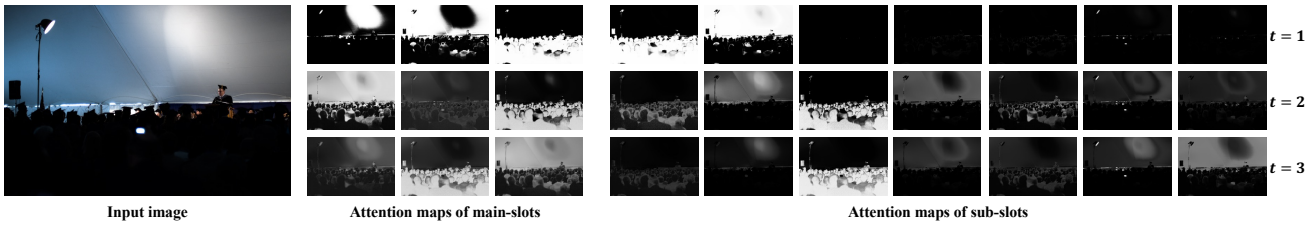

Figure S18. Visual result of attention maps from LCDP [8] dataset. From top to bottom, we present the attention maps of the main-slot and sub-slot at each iteration ($t = 1, 2, 3$).

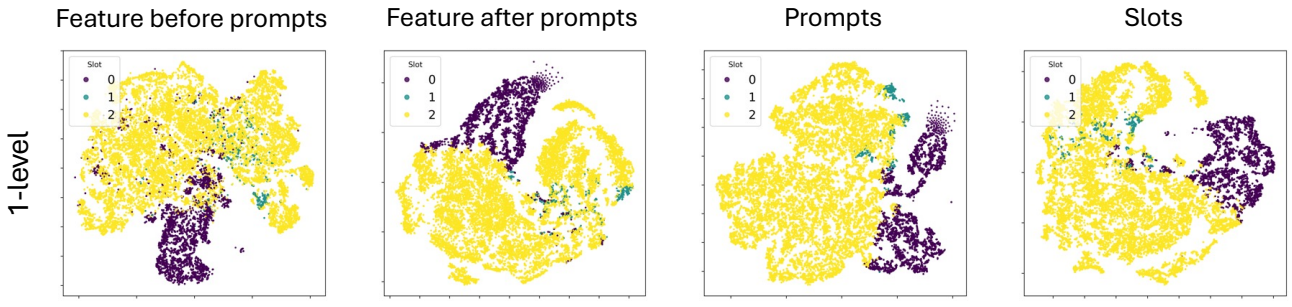

Figure S19. The t-SNE plots for the 1-level SSAB structure are presented. From left to right, the plots represent features before prompts ($\mathbf{V}$), features after prompts ($\mathbf{V} \cdot \mathbf{P}^{\text{final}}$) as defined in Eq. 2, prompts ($\mathbf{P}^{\text{final}}$), and slots ($\mathbf{S}^{\text{final}}$) as defined in Eq. 9, respectively.

CVPR
#10942

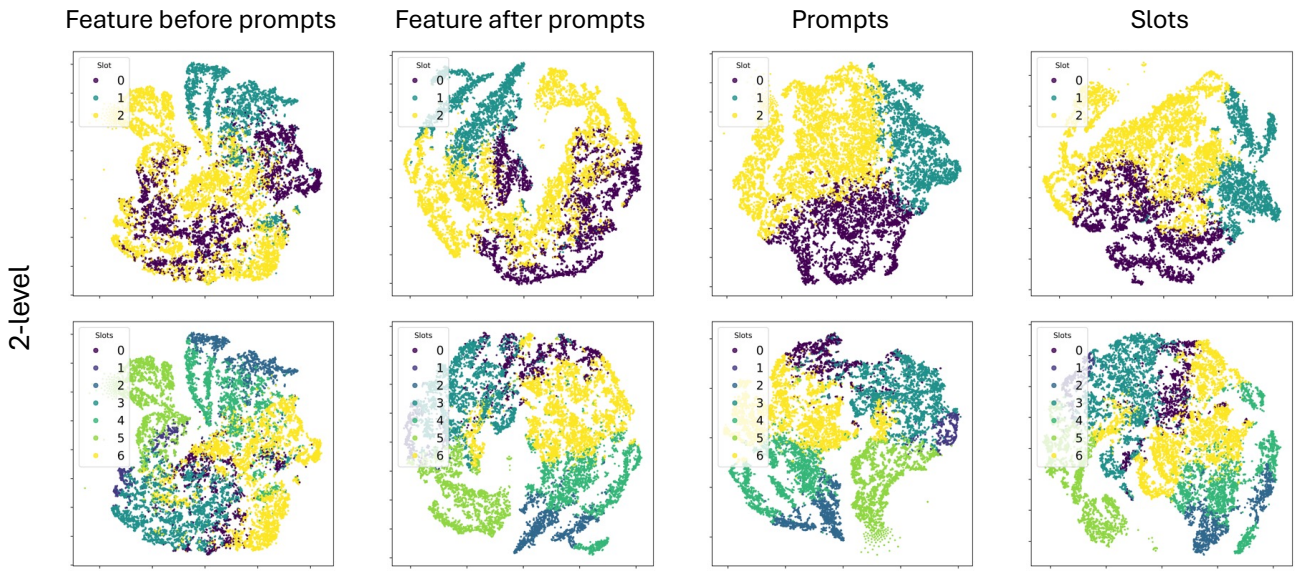

Figure S20. The t-SNE plots for the 2-level SSAB structure are presented. From left to right, the plots represent features before prompts ($\mathbf{V}$), features after prompts ($\mathbf{V} \cdot \mathbf{P}^{\text{final}}$) as defined in Eq. 2, prompts ($\mathbf{P}^{\text{final}}$), and slots ($\mathbf{S}^{\text{final}}$) as defined in Eq. 9, respectively. Additionally, from top to bottom, each row corresponds to plots clustered at the main-slot and sub-slot, respectively.

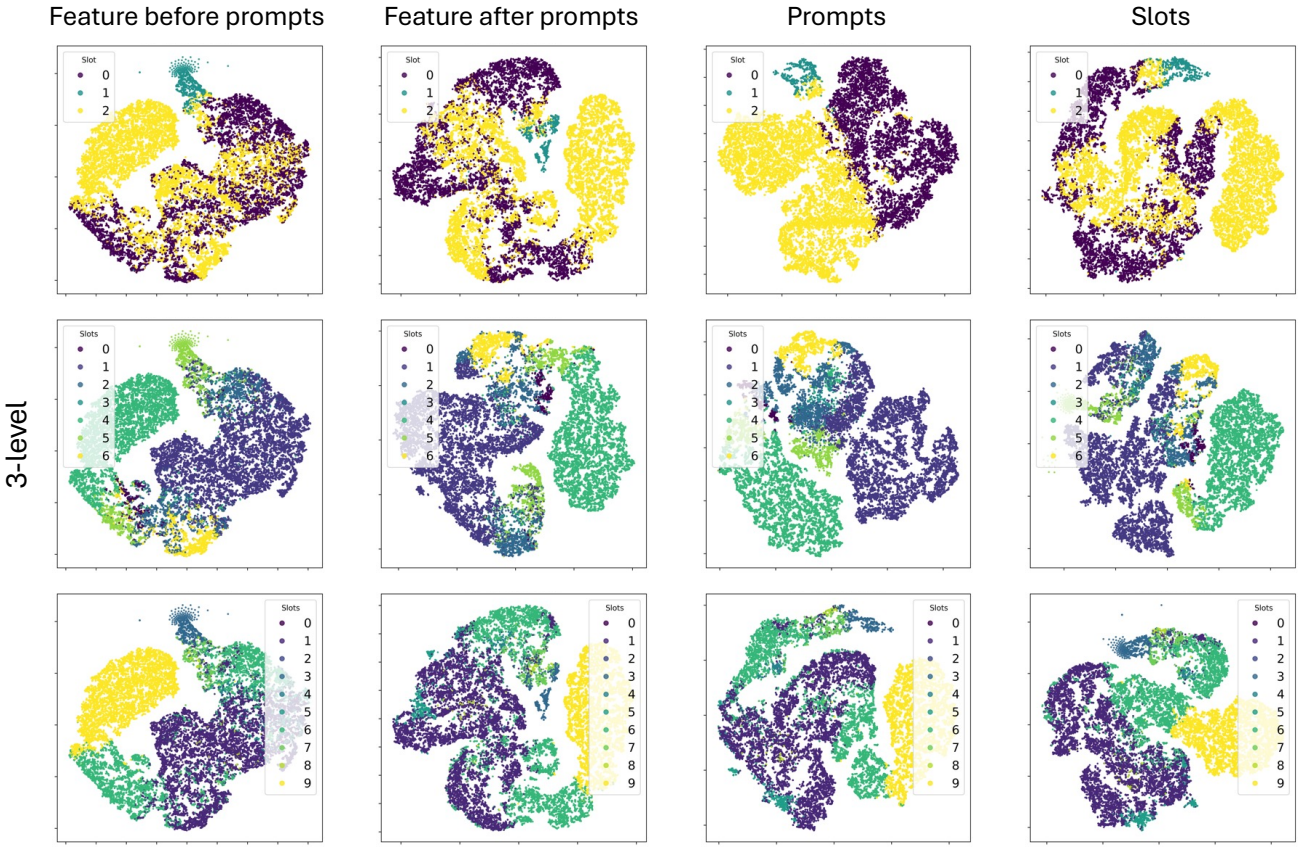

Figure S21. The t-SNE plots for the 2-level SSAB structure are presented. From left to right, the plots represent features before prompts ($\mathbf{V}$), features after prompts ($\mathbf{V} \cdot \mathbf{P}^{\text{final}}$) as defined in Eq. 2, prompts ($\mathbf{P}^{\text{final}}$), and slots ($\mathbf{S}^{\text{final}}$) as defined in Eq. 9, respectively. Additionally, from top to bottom, each row corresponds to plots clustered at the main-slot, sub-slot, and sub2-slot, respectively.

References

[1] Mahmoud Afifi, Konstantinos G Derpanis, Bjorn Ommer, and Michael S Brown. Learning multi-scale photo exposure correction. In *CVPR*, 2021. 1, 2

[2] Yochai Blau, Roey Mechrez, Radu Timofte, Tomer Michaeli, and Lihi Zelnik-Manor. The 2018 pirm challenge on perceptual image super-resolution. In *ECCVW*, 2018. 2

[3] Jianrui Cai, Shuhang Gu, and Lei Zhang. Learning a deep single image contrast enhancer from multi-exposure images. *IEEE Transactions on Image Processing*, 27, 2018. 1, 2

[4] Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*, 2016. 1

[5] Jie Huang, Yajing Liu, Feng Zhao, Keyu Yan, Jinghao Zhang, Yukun Huang, Man Zhou, and Zhiwei Xiong. Deep fourier-based exposure correction network with spatial-frequency interaction. In *ECCV*, 2022. 2

[6] Jie Huang, Man Zhou, Yajing Liu, Mingde Yao, Feng Zhao, and Zhiwei Xiong. Exposure-consistency representation learning for exposure correction. In *ACMMM*, 2022. 2

[7] Yiyu Li, Ke Xu, Gerhard Petrus Hancke, and Rynson WH Lau. Color shift estimation-and-correction for image enhancement. In *CVPR*, 2024. 2, 3, 4, 5

[8] Haoyuan Wang, Ke Xu, and Rynson WH Lau. Local color distributions prior for image enhancement. In *ECCV*. Springer, 2022. 1, 2, 3, 4, 5, 6, 7, 8, 9, 10

[9] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 586–595, 2018. 2