Unsupervised Exposure Correction Supplementary Material

Ruodai Cui[1], Li Niu[2⋆], and Guosheng Hu[3]

[1] Qualcomm Technologies, Inc.
ruodcui@qti.qualcomm.com
[2] Department of Computer Science and Engineering, MoE Key Lab of Artificial Intelligence, Shanghai Jiao Tong University
ustcnewly@sjtu.edu.cn
[3] University of Bristol

1 Introduction

In this Supplementary Material, we provide detailed information and additional results across the following sections:

1. **Visual Comparison of Datasets in Sec. 3:** A comparative analysis between our Radiometry Correction Dataset and Afifi's MSEC Dataset [1], highlighting the features of our dataset.
2. **Training Implementation Details in Sec. 4:** Here, we delve into the specifics of the training implementation for our Unsupervised Exposure Correction (UEC) Method, outlining the techniques and parameters that facilitated the learning process.
3. **Additional Experimental Results in Sec. 5:** Further experimental outcomes are presented in this section, which includes:
   (a) **Varying Exposure Manipulation.** Our method can adjust exposure within a certain range rather than a specific value. We present the visualized results.
   (b) **Edge Detection Results:** We compare the performance of our UEC Model with the ECM [4] in terms of edge detection capabilities, showcasing the effectiveness of our model in preserving low-level features.
   (c) **Visual Comparisons:** An illustrative comparison of the results from our UEC Model against the ECM (Exposure Correction Model) [4], demonstrating the advancements our model brings over the current SOTA supervised model.
   (d) **Edge Detection Results:** We compare the performance of our UEC Model with the ECM [4] in terms of edge detection capabilities, showcasing the effectiveness of our model in preserving low-level features.
   (e) **Consumption Cost Analysis:** An evaluation of the time and memory consumption by our model at different resolutions.

⋆ Corresponding author.

2 Network Details

Exposure Feature Encoder. In this section, we aim to implement the function e. Our goal is to extract exposure features from the input images and map them into a latent space. Empirically, we find the image encoders for image enhancement, e.g. Neural Color Operators [8], can work well for our task since they can encode the pixel-wise image information. Specifically, our encoder consists of three layers: two convolutional layers and a global pooling layer. The initial two convolutional layers are responsible for the extraction of 32-channel feature maps. Subsequently, we employ three different pooling functions, each dedicated to computing channel-wise maximum, average, and standard deviation, respectively. This process results in the generation of three 32D vectors, which are then concatenated to produce the final 96D representation.

Parameter Predictor. Having extracted the exposure features, we now quantify the exposure disparity between two images. Given the implicit nature of exposure difference, we opt to output the parameters for exposure correction rather than the direct difference. This is accomplished by concatenating the exposure features and applying two linear layers to produce a numerical value λ in Eq. (1). This resulting scalar serves as the parameter for Exposure Corrector.

Exposure Corrector. UEC employs a color transformation approach for exposure correction. The color transformation using deep learning can generally be denoted as $g(\phi, pixel)$, where ϕ represents the hyper-parameters from features of the input image, and $pixel$ denotes the RGB value of a single pixel within the image. We realized that the enhancement of exposure shares some similarity with brightness enhancement, which is a linear transformation, so our specific form of $g(\cdot,\cdot)$ is a combination of linear and non-linear transformation:

$$I_{out}(x,y) = \lambda \cdot I_{in}(x,y) + (1-\lambda) \cdot h(I_{in}(x,y)), \tag{1}$$

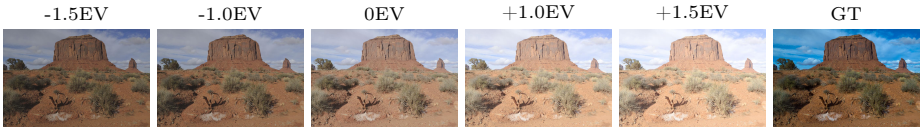
where λ, predicted by Parameter Predictor, modulates the transformed between the original input image I_{in} and its corresponding output I_{out}. The non-linear transformation function $h(\cdot)$ is implemented using a Multi-Layer Perceptron (MLP). This procedure is iterative, meaning the output from one cycle can serve as the input for the subsequent iteration. We repeat this cycle three times to enhance performance.

3 Dataset Comparision

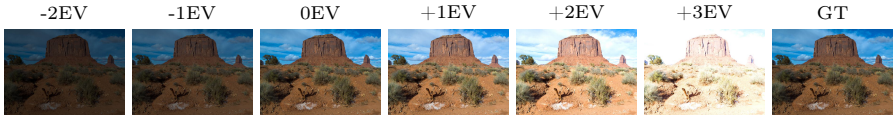
Afifi et al. [1] constructed their dataset using a forward exposure adjustment approach, which involves modifying only the exposure while retaining other parameters from a camera's image signal processor (ISP). They designated the results of all experts tuning as the ground truth. However, this approach leads to a confusing scenario where the ground truth for a given input image varies.

Such manual adjustments inherently include individual aesthetic preferences, which pose challenges to the stability and generalizability of the results. In contrast, our methodology employs a reverse strategy, known as backward exposure adjustment. This entails adjusting the exposure of the ground truth image to generate synthetic, poorly-exposed images. We hypothesize that individual stylistic elements can be encapsulated by all other ISP parameters derived from Expert C's ground truth.

The distinctions between our dataset and the MSEC Dataset are illustrated in Fig. 1. A key observation in the MSEC Dataset is the apparent discrepancy between the input images and their corresponding ground truths, particularly in terms of colorimetry. For example, the input images feature a sky with a purplish hue, whereas the ground truth images depict the sky in a pure blue color, which we attribute to the photographer's unique stylistic choices. In contrast, within our Radiometry Correction Dataset, the variations observed across different images—whether between input images or between input images and their ground truths—are exclusively attributed to changes in exposure levels.



(a) Afifi et al's MSEC Dataset



(b) Our Radiometry Correction Dataset

Fig. 1: Comparative Analysis of Datasets: (a) Afifi et al's MSEC Dataset vs. (b) Our Radiometry Correction Dataset.

4 Implementation Details

In our experimental setup, we adopt a batch size of 1, allowing for fine-grained updates during the optimization process. We employ the Adam optimizer with a learning rate of 0.0001 and set the β_1 value to 0.9 and β_2 to 0.999. To ensure uniformity in the input dimensions, all images are resized to 448×448 pixels.

Our learning rate schedule is characterized by a step-wise decay strategy. Initially, the learning rate is set at 0.0001 for the first 100 iterations. Subsequently, we apply a slower decay rate for the following 100 iterations, eventually reaching its minimum value at 6,574,200 iterations.

During the testing phase, all images are consistently resized to dimensions of 256×256 pixels. To fulfill the reference image requirement of our method, we

select the first image from the dataset, specifically the ground truth image found in the first row of Fig. 1, to serve as the reference image.

5 Experiment Results

5.1 Results of Varying Exposure Manipulation.

Figure 2 demonstrates the visualized results of exposure manipulation on the MSEC dataset [1].

5.2 Comparison of Results on MSEC Dataset

We conducted experiments using Afifi et al.'s MSEC Dataset [1]. Detailed data can be found in Tab. 1. Notably, this dataset encompasses five distinct ground truth images, each corresponding to a specific input. These ground truth images are individually labeled by the author as Expert A, Expert B, Expert C, Expert D, and Expert E. Consequently, we present results separately for each of these five experts and also compute their ensemble average outcomes.

In Fig. 3 4 and 5, we present various test images, our results, ground truth images, and the outcomes of ECM [4]. Although we may not exhibit a distinct advantage in terms of PSNR calculations for 256×256 resolution images, it is evident that our images boast superior detail quality, a more natural aesthetic, and fewer color casts.

Method	Expert A		Expert B		Expert C		Expert D		Expert E		Avg.	
	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
WVM [5]	14.488	0.665	15.803	0.699	15.117	0.678	15.863	0.693	16.469	0.704	15.548	0.688
HDRCNN w/PS [3]	15.812	0.667	16.970	0.699	16.428	0.681	17.301	0.687	18.650	0.702	17.032	0.687
DPED (iPhone) [7]	15.134	0.609	16.505	0.636	15.907	0.622	16.571	0.627	17.251	0.649	16.274	0.629
DPED (BlackBerry) [7]	16.910	0.642	18.649	0.713	17.606	0.653	18.070	0.679	18.217	0.668	17.890	0.671
DPE(HDR) [2]	15.690	0.614	16.548	0.626	16.305	0.626	16.147	0.615	16.341	0.633	16.206	0.623
DPE(S-FiveK) [2]	16.933	0.678	17.701	0.668	17.741	0.696	17.572	0.674	17.601	0.670	17.510	0.677
Zero-DCE [6]	11.643	0.536	12.555	0.539	12.058	0.544	12.964	0.548	13.769	0.580	12.597	0.549
Afifi et al. w/o Ladv [1]	19.158	0.746	20.096	0.734	20.205	0.769	18.975	0.719	18.983	0.727	19.483	0.739
Afifi et al. w/ Ladv [1]	19.114	0.743	19.960	0.723	20.080	0.763	18.868	0.709	18.864	0.719	19.377	0.731
ECM [4]	20.443	0.860	21.773	0.887	22.332	0.897	19.984	0.870	19.840	0.875	20.874	0.877
Ours	18.445	0.794	19.667	0.832	18.955	0.810	18.509	0.805	18.204	0.820	18.756	0.812

Table 1: Comparison of Results on MSEC Dataset.

5.3 Results on Edge Detection

We investigate the broader application of exposure correction on downstream tasks, such as edge detection, emphasizing its potential to mitigate the adverse effects of non-ideal exposure on low-level image features. The results are shown on Fig. 6.

5.4 Time and Memory Consumpation

The main text has already detailed the experiments concerning time and memory consumption. Additionally, we conducted supplementary experiments where we evaluated our model's performance at various resolutions. Recognizing the limitations of ECM [4], which is confined to handling only 256-pixel resolution, we exclusively examined our algorithm's capabilities across different resolutions. The results are presented in Tab. 2.

Resolution	Width	Height	Time (GPU)	Time (CPU)
4K	4096	2160	23.32ms	N/A
2K	2560	1440	9.40ms	N/A
1080P	1920	1080	5.49ms	476.08ms
720P	1280	720	2.72ms	212.08ms
480P	720	480	1.44ms	81.11ms

Table 2: Comparison on different resolutions for our method

References

1. Afifi, M., Derpanis, K.G., Ommer, B., Brown, M.S.: Learning Multi-Scale Photo Exposure Correction. Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition pp. 9153–9163 (2021)
2. Chen, Y.S., Wang, Y.C., Kao, M.H., Chuang, Y.Y.: Deep photo enhancer: Unpaired learning for image enhancement from photographs with gans. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 6306–6314 (2018)
3. Dayley, L.D., Dayley, B.: Photoshop CS5 Bible. John Wiley & Sons (2010)
4. Eyiokur, F.I., Yaman, D., Ekenel, H.K., Waibel, A.: Exposure Correction Model to Enhance Image Quality. IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops **2022-June**, 675–685 (2022)
5. Fu, X., Zeng, D., Huang, Y., Zhang, X.P., Ding, X.: A weighted variational model for simultaneous reflectance and illumination estimation. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 2782–2790 (2016)
6. Guo, C., Li, C., Guo, J., Loy, C., Hou, J., Kwong, S., Cong, R.: Zero-reference deep curve estimation for low-light image enhancement. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 1780–1789 (2020)
7. Ignatov, A., Kobyshev, N., Timofte, R., Vanhoey, K., Van Gool, L.: Dslr-quality photos on mobile devices with deep convolutional networks. In: Proceedings of the IEEE international conference on computer vision. pp. 3277–3285 (2017)
8. Wang, Y., Li, X., Xu, K., He, D., Zhang, Q., Li, F., Ding, E.: Neural color operators for sequential image retouching. In: European Conference on Computer Vision. pp. 38–55. Springer (2022)

(a) Exposure control example 1.

(b) Exposure control example 2.

Fig. 2: Examples of exposure control. The first column of each row represents the input, while the subsequent columns display the results derived from this input.

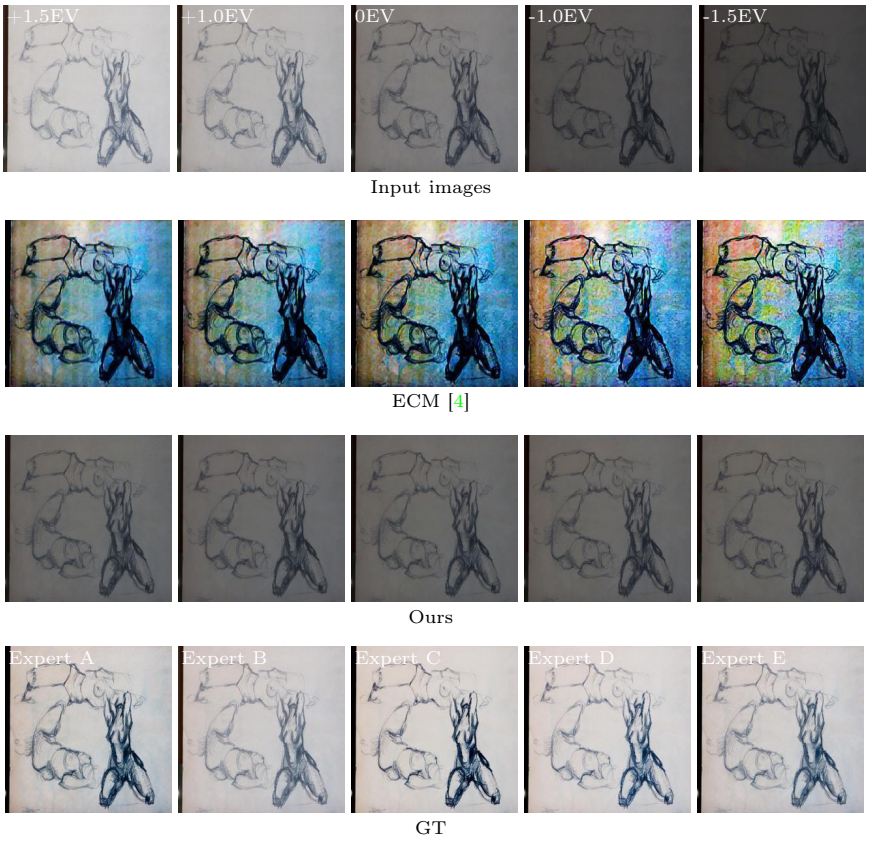


Fig. 3: Visual Comparison: ECM [4] vs. Our UEC method.

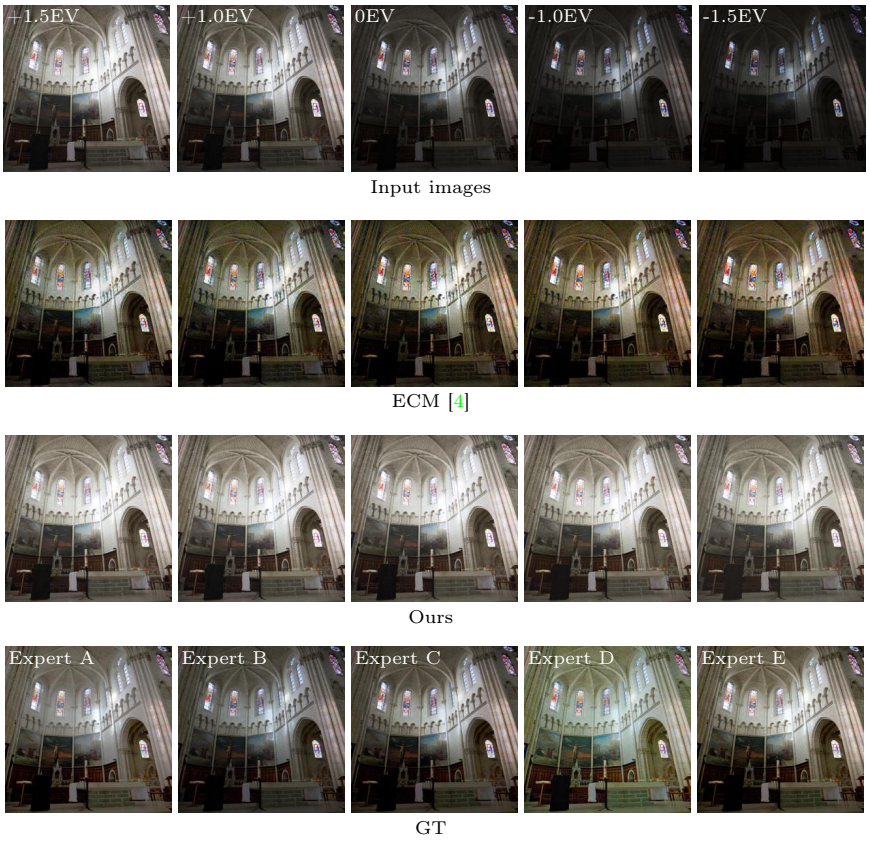


Fig. 4: Visual Comparison: ECM [4] vs. Our UEC method.

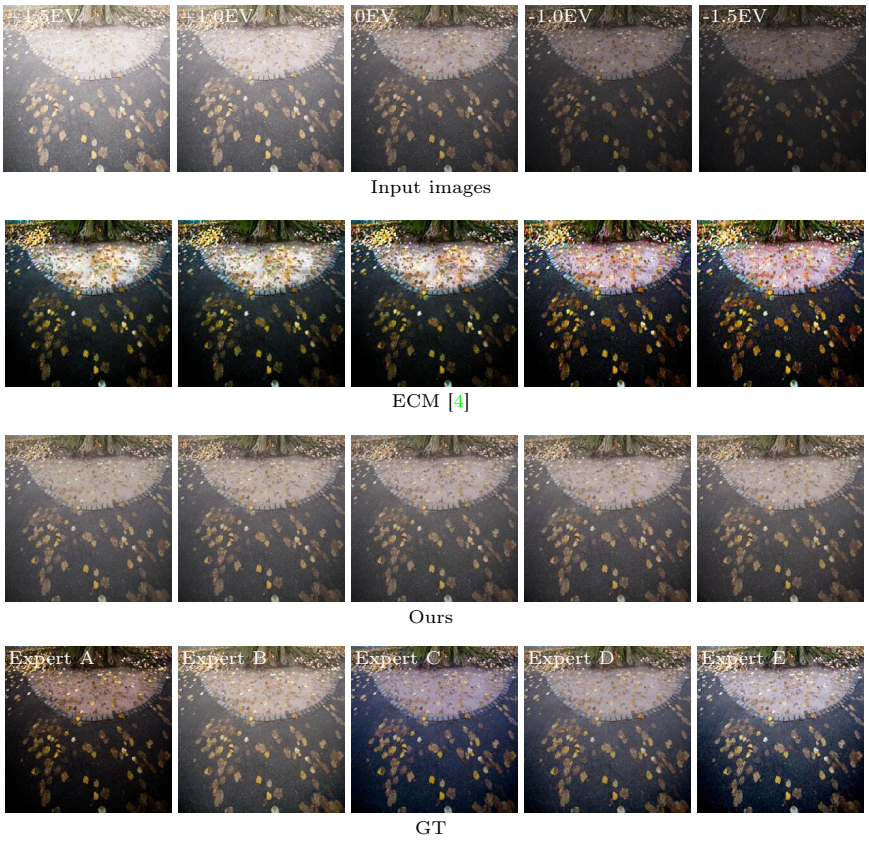


Fig. 5: Visual Comparison: ECM [4] vs. Our UEC method.

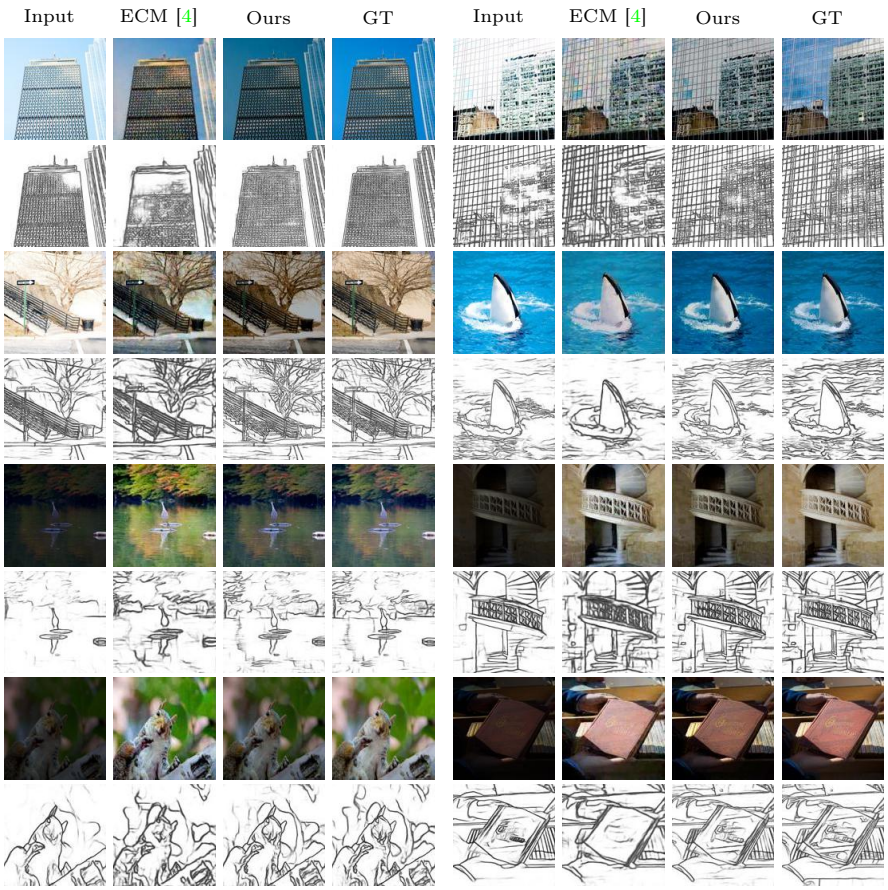


Fig. 6: Illustration of the edge detection outputs.