This CVPR 2021 paper is the Open Access version, provided by the Computer Vision Foundation.
Except for this watermark, it is identical to the accepted version;
the final published version of the proceedings is available on IEEE Xplore.

Restoring Extremely Dark Images in Real Time

Mohit Lamba
Indian Institute of Technology Madras
ee18d009@smail.iitm.ac.in

Kaushik Mitra
Indian Institute of Technology Madras
kmitra@ee.iitm.ac.in

Abstract

A practical low-light enhancement solution must be computationally fast, memory-efficient, and achieve a visually appealing restoration. Most of the existing methods target restoration quality and thus compromise on speed and memory requirements, raising concerns about their real-world deployability. We propose a new deep learning architecture for extreme low-light single image restoration, which despite its fast & lightweight inference, produces a restoration that is perceptually at par with state-of-the-art computationally intense models. To achieve this, we do most of the processing in the higher scale-spaces, skipping the intermediate-scales wherever possible. Also unique to our model is the potential to process all the scale-spaces concurrently, offering an additional 30% *speedup without compromising the restoration quality. Pre-amplification of the dark raw-image is an important step in extreme low-light image enhancement. Most of the existing state of the art methods need GT exposure value to estimate the pre-amplification factor, which is not practically feasible. Thus, we propose an amplifier module that estimates the amplification factor using only the input raw image and can be used "off-the-shelf" with pre-trained models without any fine-tuning. We show that our model can restore an ultra-high-definition* $4K$ *resolution image in just* 1 $sec.$ *on a CPU and at* 32 fps *on a GPU and yet maintain a competitive restoration quality. We also show that our proposed model, without any fine-tuning, generalizes well to cameras not seen during training and to subsequent tasks such as object detection.*

1. Introduction

The Computer Vision community has witnessed excellent methods in the last two decades for low-light enhancement [32, 62, 20, 42, 10, 63, 61]. Especially noteworthy is SID's [10] recent success in restoring extreme low-light images captured in *near zero lux* conditions ($0.1-5$ lux). Since then several deep learning architectures have been proposed for enhancing dark images [44, 18, 34, 65].

Dark input: $20\times$ amplified for visualization Restored video + Object Detection

Figure 1. *(One frame of the restored video)* We can restore ultra-high-definition $4K$ resolution night-time images at 32 fps on a GPU. This enables real-time visualization and subsequent inference such as object detection.

In spite of these advances, such solutions may not be appropriate for real-world deployment due to their prohibitively high computational cost. In fact, a practical solution must have low network latency, less memory footprint, fewer model parameters, smaller operations count and yet maintain a pleasing restoration. Conventionally, however, these qualities are mutually contradictory. This is especially true for extreme low-light restoration where colors are hard to recover and noise suppression is significantly challenging. Thus, the predominant trend is to forsake model speed and computational efficiency for better restoration, raising concerns for real-world deployment [9, 24, 26, 75, 60]. For example, two recent methods, SID [10] and SGN [18], require 562 GMAC (Giga multiply-accumulate) and 2474 GMAC floating-point operations, respectively, to restore a single $4K$ resolution raw image. This demands staggering levels of computations, which is unlikely to be available with the deployed edge devices and will cause enormous network latency. This may be frustrating for a casual user and impractical for critical tasks. Thus, our goal is to design a network which achieves similar restoration quality but with drastically low operations.

We propose a deep learning architecture that extracts most of the information required for restoration from higher scale-spaces (which operate at lower resolution) and skip intermediate scales, as shown in Fig. 2. At the highest scale, we use the widely used Residual Dense Block (RDB) [72],

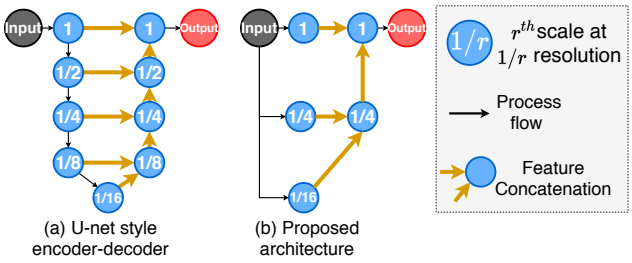


Figure 2. (a) Almost all methods rely on sequential processing. (b) We propose a parallel architecture for high inference speed.

which uses non-linear rectification after each convolutional layer. But the excessive use of non-linear rectification has been criticized in recent works for decreasing the accuracy, and the naive approach of limiting their usage has worked successfully [21, 56, 67, 71]. However, ours is already a lightweight model with comparatively less number of convolutions and non-linear activations. Thus to throw away some of the non-linearity from our design deprives it to model complex functions and the restoration degrades. Consequently, we modify the RDB to reduce the side effects of negative clipping while maintaining sufficient non-linearity.

Most architectures rely on sequential data processing by making the previous convolution block's output as the input to the current convolution block. By this approach, the only way to limit network latency is to either accelerate hardware operations — a technology that has started to plateau, or reduce the network operations and compromise with the restoration quality. To break this conundrum, we imbue our design with *architectural parallelism* that allows concurrent processing of various scale-spaces (see Fig. 2), offering an additional 30% speedup but with no effect on the restoration quality.

SID [10], a landmark work on extreme low-light restoration, pre-amplified raw images for a successful recovery. But, the pre-amplification required the knowledge of Ground-Truth (GT) exposure even during inference (see Eq. (3)). Although this is now a standard practice to recover dark raw images [34, 18, 44, 64], a real-world solution will be benefited if the amplification is computed using only the input raw image during inference. To this end, we propose a new amplifier module that directly uses the intensity values of the dark raw image to estimate the amplification. Thus, our amplifier can automatically adapt to varying light levels, allowing it sometimes to achieve a perceptual score even better than the GT image. Finally, we show that our amplifier can be augmented *as-it-is* to existing pre-trained models without any fine-tuning for a successful restoration.

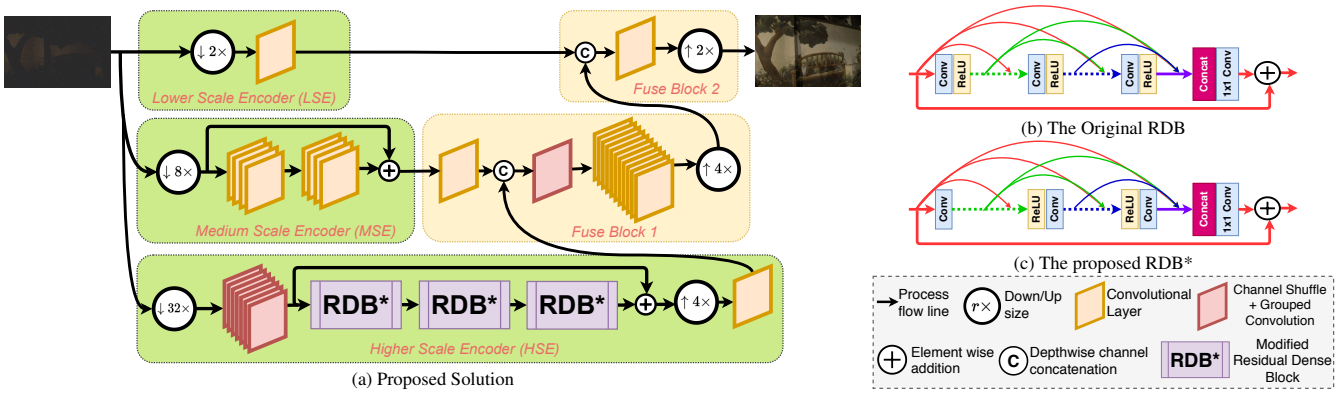
Our contributions. In summary, the contributions of this work are: (1) A new deep learning architecture for extreme low-light single image restoration, which compared to state-of-the-art [10, 18, 65] is $5-100\times$ faster, $6-20\times$ computationally cheaper, uses $3-11\times$ fewer model parameters and has MAC operations lower by an order. (2) A systematic strategy to enable architectural parallelism for additional speedup with no effect on restoration. (3) A modification to the popular Residual Dense Block for better restoration. (4) A novel amplifier module useful in a real-world scenario where the amplification factor is estimated only from the input image. It can be used directly with pre-trained models with no fine-tuning. (5) Our model generalizes well to cameras not seen during training and also to subsequent tasks such as object detection without any fine-tuning. Our code is available at the *mohitlamba94.github.io/Restoring-Extremely-Dark-Images-In-Real-Time*.

2. Related work

Low-light enhancement. Initial approaches used histogram equalization and its variants to increase the dynamic range [31, 51, 59, 11, 27, 37]. Gradually, it was observed that a prior decomposition into illumination and reflectance components using the Retinex theory [36, 35] favored better recovery [62, 14, 15, 39, 20, 30]. Recent works, however, use neural networks and occasionally use the Retinex theory for better results [19, 66, 61, 42, 63, 29, 38, 54, 73].

Extreme low-light restoration. Chen *et al*. [10] proposed the SID dataset for restoring extremely dark nighttime images having very poor colors and a large amount of noise. This work has since then spurred several works to restore extreme low-light images [44, 18, 65, 4, 3]. Many of them use the U-Net style encoder-decoder for restoration and have a huge computational overhead. Recently, Wei *et al*. [64], proposed a noise formation framework for CMOS sensors to synthesize dark raw images. Complementing the real data, these synthetic images can be calibrated to different cameras and utilized to train existing networks. Previous works have also used burst photography [22, 41, 46]. Burst shots requires some treatment for proper alignment which is difficult for dark noisy images. Thus, the restorations are susceptible to ghosting artifacts [10, 64], and so we focus on restoring single-frame images.

Lightweight models. DCE [19] and LLPackNet [34] are two most recent works on lightweight low-light enhancement. DCE does not use too many convolutional layers to limit the MAC operations but still has considerable network latency because much of the processing occurs at lower scales. LLPacknet takes the exact opposite approach by having only one scale-space operating at $16\times$ lower resolution but with significantly blurry results. We take a balanced approach and carefully fuse the details from different scales such that the network latency decreases, but the restoration quality improves. Apart from this, architectural parallelism for speedup, modifying RDB for superior results and an amplifier that can be augmented to pre-trained networks *without any fine-tuning* are contributions exclusive to this work.



(a) Proposed Solution

(b) The Original RDB

(c) The proposed RDB*

Figure 3. Architectural details of the proposed model. Depth of convolutional layer is roughly proportional to number of o/p features.

3. Fast & lightweight dark image restoration

3.1. Network architecture

The success of a practical low-light enhancement solution is not just limited to restoration quality but also inference speed and computational feasibility. However, most restoration networks use U-net style encoder-decoder wherein processing at lower scales causes significant latency and computational overhead. Thus, as shown in Fig. 3, we jump over intermediate scales and operate at just three scales — Lower Scale Encoder (LSE) at $1/2$ resolution, Medium Scale Encoder (MSE) at $1/8$ resolution, and Higher Scale Encoder (HSE) at $1/32$ resolution. As HSE operates at the lowest resolution, most of the convolutional layers are allocated to HSE. HSE's basic building block is the Residual Dense Block (RDB), to which we suggest a simple modification for better restoration while maintaining the same time-memory complexity. All the encoder scales operate directly on the input image and do not have any inter-dependencies. The execution of all scales can thus be parallelized, offering additional speedup. Finally, Fuse Block 1 (FB 1) and Fuse Block 2 (FB 2) fuses details from all the scales to generate the restored image.

We provide an end-to-end solution by accepting a raw image as input and generating a restored RGB image as output. We choose to work on raw image because: (1) the noise in raw image is much simpler to model and hence easier for CNNs to remove [64, 1, 76], (2) the captured low-light data has very small intensity values that would be irreversibly lost if subjected to camera's compression and quantization routines, and (3) end-to-end solutions for raw image input tend to be much more accurate and faster [10, 2, 76].

The raw image consists of a single channel with mosaic colors that need to be decoupled into separate channels for subsequent operations. This decoupling happens when we downsample the raw image using the Pixel-Shuffle operation [58, 69]. The only condition is that the downsampling factor must be a multiple of the mosaicing pattern's size. As almost all cameras use 2×2 Bayer pattern for mosaicing, we choose a downsampling factor of $2, 8$ and 32 for LSE, MSE and HSE, respectively. The details of LSE/MSE/HSE are deferred until Sec. 3.1.2.

Fuse Block 1 fuses the low and medium frequency details. For this, the HSE output is upsampled by $4\times$ using Pixel-Shuffle and is depth-wise concatenated to MSE's output. To save downstream computations, we reduce the channel dimension using grouped convolution [28]. Grouped convolution is a much lighter operation than normal convolution because it does not consider the full feature depth. Thus if we use grouped convolution after depth-wise concatenation, most of the kernels will be exposed to either HSE's or MSE's output and fail to capture the interdependencies. To check this, we interleave the channels obtained from MSE and HSE via a simple channel shuffling operation before performing grouped convolution. This is followed by normal convolution for additional processing. Finally, the Fuse Block 1 output is $4\times$ upsampled and depth-wise concatenated to LSE's output. A convolutional layer in the Fuse Block 2 then generates the restored RGB image.

To train the network, we compute L1 loss and MS-SSIM loss [74] between the restored RGB image and the GT with a weightage of 0.8 and 0.2, respectively. Also, as suggested by SID [10], we use LeakyReLU non-linearity instead of ReLU with a negative slope of 0.2.

3.1.1 Modifying RDB to RDB*

The canonical RDB [72] does non-linear rectification [23] after each convolutional layer, see Fig. 3 b). But recent works have objected the unconstrained use of non-linear rectifiers such as ReLU [21, 56, 67, 71, 50]. Such rectifiers clip the negative values of feature maps that are irreversibly lost. Nevertheless, they are necessary to infuse the model with sufficient non-linearity. We thus propose a modification to RDB and call it RDB*, as shown in Fig. 3 c). As in the RDB, each convolutional layer in RDB* passes a rectified output to subsequent convolutional layer, guaranteeing sufficient non-linearity in RDB*. But different from RDB,

the last pointwise 1×1 convolutional layer in RDB* sees non-rectified output of all the previous layers. Thus unlike RDB, our RDB* is *simultaneously processing both rectified and non-rectified output of each convolutional layer*. Consequently, RDB* avoids the problem of complete loss of information due to non-linear rectification.

3.1.2 Achieving effective parallelism

LSE, MSE and HSE directly operate on the input image. A partial motivation for this comes from better anti-aliasing reported by Burt and Adelson on adopting a similar strategy for their classical Laplacian pyramid [7, 8]. But our main motivation is to enable concurrent processing of LSE, MSE and HSE and thus achieve additional speedup.

Parallelism is best utilized if the concurrent tasks take nearly the same time to execute so that the idle time is minimized. Thus an important question now is, how many convolutions to perform in LSE, MSE and HSE for them to have the same execution speed? To answer this, we make a simplifying assumption that this is equivalent to saying that LSE, MSE and HSE should roughly have the same number of floating-point operations. Limitations of this assumption is that this does not account for shuffling operations and for hardware specific requirements. Still, the following analysis helps us in making an educated guess on the number of convolutional layers to use.

Let the size of the raw image and the convolutional kernels be $H \times W$ and $k_\gamma \times k_\gamma$, respectively, where $\gamma \in \{LSE, MSE, HSE\}$. Also, let C^i_γ and C^o_γ denote the number of input and output channels, n_γ the number of convolutional layers and r_γ the downsampling factor. For all convolutions, the input and output have the same spatial resolution. Thus, number of operations is given by:

$$\frac{H}{r_\gamma} \times \frac{W}{r_\gamma} \times k^2_\gamma \times C^i_\gamma \times C^o_\gamma \times n_\gamma. \tag{1}$$

LSE operates at the highest image resolution, and so is computationally expensive. We thus use a single convolutional layer, $n_{LSE} = 1$, for LSE that accepts $C^i_{LSE} = 4$ channels because of downsampling by a factor of $r_{LSE} = 2$ and outputs $C^o_{LSE} = 12$ channels. For MSE and HSE, we use 3×3 convolutional kernels, but for LSE we use a larger kernel to compensate for doing a single lightweight convolution. Thus, we have $k_{LSE} = 7$ and $k_{MSE} = k_{HSE} = 3$.

In MSE, we downsample the raw image using PixelShuffle by a factor of $r_{MSE} = 8$, which results in $8^2 = 64$ channels. We use residual block for MSE and thus, we set $C^i_{MSE} = C^o_{MSE} = 64$. If we now equate the number of operations in LSE and MSE, we get $1 < n_{MSE} < 2$. Since we are using residual block, we choose $n_{MSE} = 2$.

HSE can accommodate a lot of convolutional layers for the same operations count of MSE. So we use n_{HSE} number of RDB*s, each having $\eta = 5$ convolutional layers and a growth factor of $C_{gr} = 32$ [72]. Also, in HSE we downsample the raw image by a factor of $r_{HSE} = 32$, which results in $32^2 = 1024$ channels. To minimize the downstream computations, the channel width is reduced to $C^i_{HSE} = 64$ using grouped convolution. But as in Fuse Block 1, we first shuffle the channels to ensure that each group sees channels corresponding to each Bayer color in equal proportion. Thus, the net operations count is:

$$\frac{H}{r_{HSE}} \frac{W}{r_{HSE}} \Bigg(n_{HSE}\, k^2_{HSE}\, C_{gr} \sum_{j=1}^{\eta} [C^i_{HSE} + (j-1) C_{gr}] + n_{HSE} [C^i_{HSE} + \eta \cdot C_{gr}] C^i_{HSE} \; + \; k^2_{HSE}\, r^2_{HSE} \Bigg). \tag{2}$$

If we now equate MSE's and HSE's floating-point operations count, we find that we can use $n_{HSE} = 3$ RDB*s. More details about our network architecture can be found in the supplementary.

3.2. Amplifier module

The captured extreme low-light raw images have very small intensity-values. Thus, it is essential to pre-amplify them for a good restoration. Recent methods [10, 64, 18, 34] provide this necessary amplification using the GT exposure setting as shown below,

$$Amplification\ (Amp.) = \frac{GT\ exposure}{i/p\ Image\ exposure}. \tag{3}$$

The GT exposure is, however, unlikely to be available in real-world setting. We thus propose a very simple amplifier that can be used when the GT exposure or any other image meta-data is not available.

The raw images in the SID [10] dataset (normalized to the range $0 - 1$) have very low average intensity values $< 10^{-2}$. A very simple pre-amplifier would be to multiply the raw image by the factor:

$$Amp. = m \cdot \left(\frac{\sum_{i,j} x_{i,j}}{H \cdot W} \right)^{-1}, \tag{4}$$

where $x_{i,j}$ is the intensity of the (i, j)-th pixel in a $H \times W$ low-light raw image and m is a hyperparameter that governs the overall brightness of the final restoration. A suitable value for m can be 0.5 because images are normalized in the range $0 - 1$. Alternatively, m can be adjusted to the user's subjective liking.

However, low-light images generally have saturated highlights [13] which tend to introduce halo artifacts [13, 62, 45]. These spurious light sources do not enhance the scene visibility but cause Eq. (4) to wrongly estimate a much lower amplification. Thus, instead of using Eq. (4), we use weighted intensity average value for estimating the amplification factor:

$$Amp. = m \cdot \left(\frac{\sum_{i,j} x_{i,j} \cdot w_{i,j}}{\sum_{i,j} w_{i,j}} \right)^{-1}, \tag{5}$$

		SID [10]	DID [44]	SGN [18]	LLPackNet [34]	DCE [19]	LDC [65]	Ours
Parameters *(in million* ↓)		7.7	2.5	3.5	1.16	0.79	8.6	**0.78**
GMACs	**Bayer**	562.06	>2000	>2000	83.46	361	>2000	**59.8**
(↓)	**X-Trans**	1118.8	>2000	>2000	166.12	719.26	>2000	**119.03**
Memory	**Bayer**	6.184	10.72	9.3	2.23	3.58	24.40	**1.16**
***(in GB* ↓)**	**X-Trans**	11.27	19.12	18.21	4.94	8.13	>30	**2.48**
GPU inference	**Bayer**	156.85	>1000	867.50	40.17	116.21	>1000	**30.37**
time *(in ms* ↓)	**X-Trans**	303.64	>1000	>1000	78.94	225.13	>1000	**59.42**
CPU inference	**Bayer**	9.35	110.29	35.20	2.75	6.20	>200	**1.17**
time *(in sec.* ↓)	**X-Trans**	18.59	>200	68.94	4.79	12.32	>200	**2.38**
PSNR / SSIM	**Bayer**	28.88/0.787	28.41/0.780	28.91/0.789	27.83/0.75	26.53/0.73	**29.56/0.799**	28.66/0.790
***(in dB* ↑ / ↑)**	**X-Trans**	26.61/0.680	–	**26.90/0.683**	24.13/0.59	–	26.70/0.681	26.60/0.682
NIQE / Ma	**Bayer**	**4.39 / 6.93**	4.52/ 6.85	4.40 / 6.92	5.12/ 5.98	4.64 /6.09	4.40 / 6.91	4.41/6.92
(↓ / ↑)	**X-Trans**	**4.45 / 6.78**	–	4.46 / **6.78**	5.20 / 5.97	–	4.47 / 6.77	4.46 / 6.77

Table 1. Compared to state-of-the-art, we achieve a drastic improvement in inference speed and computational efficiency while maintaining a competitive restoration. The best result is in **bold** and the second best is underlined. GT exposure has been used for pre-amplification.

where $w_{i,j}$ is the weight assigned to the (i,j)-th pixel. We choose the weights such that $w_{i,j} \approx 1$ for small intensity values and it exponentially reduces to 0 for large intensity values. To define these weights, we quantize the intensity range $0-1$ into n bins. Since most of the intensities $x_{i,j}$ will be low valued, we allocate finer bins for lower intensities and coarser bins for larger intensities. We do this by making the bin edges b_k equidistant on a logarithmic scale:

$$b_k = 2^{\frac{k \cdot 8}{n}}/2^8 \; \forall \; k \in [1, n] \text{ and } b_0 = 0. \tag{6}$$

Here we choose 2^8 as the normalizing factor as most camera sensors by default are 8-bit sensors. But even for other cases, the quantization defined in Eq. (6) is valid as long as the intensities are in the range $0-1$. Finally, we choose the weights $w_{i,j}$ as:

$$w_{i,j} = 2^{\frac{(n-k+1)\cdot 8}{n}}/2^8 \text{ if } b_{k-1} < x_{i,j} \leq b_k. \tag{7}$$

4. Experiments

4.1. Experimental settings

We used PyTorch [49], running on Intel Xeon E5-1620V4 CPU with 64GB RAM and GTX 1080Ti GPU to implement our network. We trained the network using ADAM optimizer [33] with default parameters for $1000K$ iterations. For the first $500K$ iterations the learning rate was set to 10^{-4} and thereupon reduced to 10^{-5}. We initialized all convolutional layers with MSRA [23] initialization. For training, we used randomly cropped 512×512 patches with horizontal and vertical flipping and a batch size of 8. Additionally, we conditioned all convolutional layers with *weight normalization* [55]. During testing, we used full image resolution with no augmentation or weight normalization. For our amplifier, we fix $n = 128$ in Eq. (5).

We use the extreme low-light SID dataset [10] containing short-exposure raw images with corresponding long-exposure GT images for benchmarking. The dataset was captured using two camera sensors: Sony α7S II Bayer sensor with image resolution 4256×2848 and Fujifilm X-Trans sensor with image resolution 6032×4032. We report results for both sensors but, like most previous works [64, 1, 2, 6, 17, 76], focus on the widely used Bayer sensor. The training and testing split mentioned in the SID dataset is used for all comparisons.

We compare with six recent low-light enhancement solutions, namely, SID [10], DID [44], SGN [18], LLPackNet [34], DCE [19] and LDC [65] with publicly available codes. Of them, LLPackNet is especially lightweight. DCE is also comparatively lightweight but was only tested for low-light images with relatively negligible noise and good color representation. Consequently, the pre-trained DCE performed badly for extremely dark images in the SID dataset with less than 15 dB PSNR. Therefore, we retrained it on the SID dataset and got around 7 dB improvement for all images and use this version in all comparisons.

4.2. Quantitative and perceptual comparisons

In Table 1 we assess methods on multiple criteria related to real-life deployability such as: number of model parameters, MAC operations, peak RAM utilization, inference speed, and restoration quality. For peak RAM utilization and inference speed, we report values averaged over a hundred trials. For a fair comparison across all methods, we use the knowledge of the GT exposure mentioned in the SID dataset (Eq. (3)) for pre-amplification and disable parallelism in our network (i.e. LSE/HSE/MSE execute one after another) as existing methods use PyTorch's serial scheduler. Most methods report evaluations on high-end GPUs but in reality, the target device will be resource-constrained, having limited computational capabilities. Thus, along with GPU inference speed, we also report CPU inference speed and GMACs consumed.

It is evident from Table 1, that we achieve notable improvement on several metrics. Specifically, we outperform lightweight solutions DCE and LLPackNet in all respects.

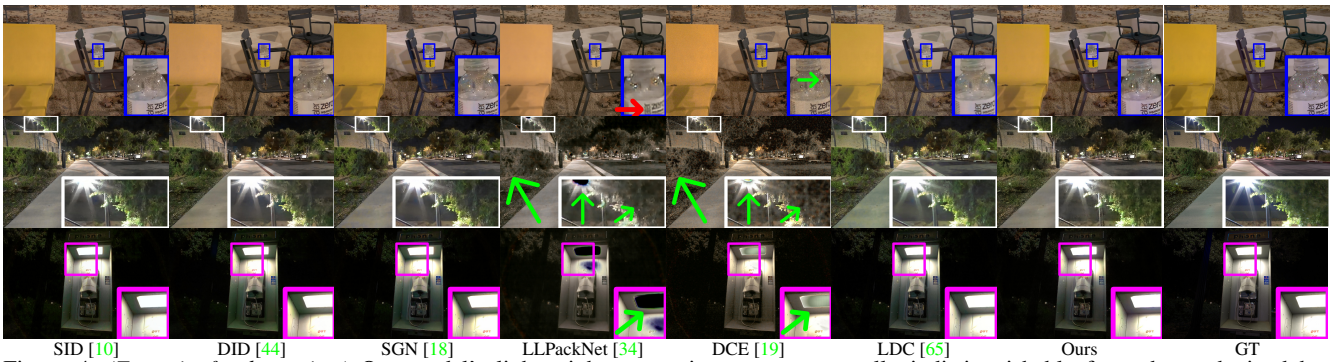


Figure 4. *(Zoom-in for best view)* Our model's lightweight restorations are perceptually indistinguishable from those obtained by computationally-intense models. The existing lightweight solutions, DCE [19] and LLPackNet [34] do not exhibit good color restoration (green arrows). LLPackNet also struggles to recover high-frequency details (red arrow where "*zero water*" is not readable).

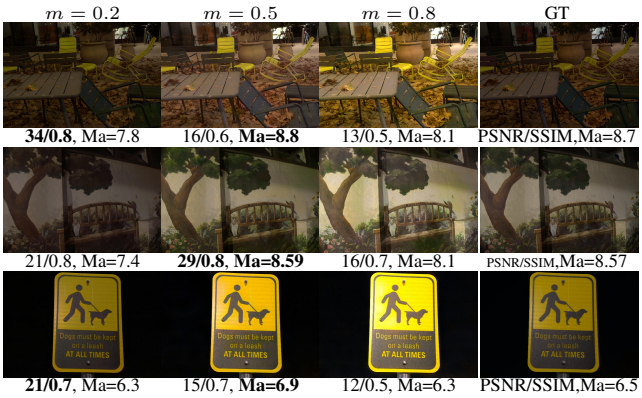


Figure 5. Images restored by our model using our amplifier. All the restorations are perceptually as good as the GT, with sometimes better perceptual scores (Ma values). Consequently, our model gives the user the flexibility of choosing an appropriate brightness for the restored image by controlling the amplifier hyperparameter m (Eq. (5)). As our model is especially fast, it can generate multiple outputs for the user to choose from in a very short interval.

Figure 6. The MLP based amplifier [34], even if jointly retrained with the SID restoration model, causes several artifacts. In comparison, our amplifier module helps achieve much superior restoration without any retraining.

Compared to the state-of-the-art resource intensive methods [10, 18, 65], we are $5-100\times$ faster, $6-20\times$ computationally cheaper, use $3-11\times$ fewer model parameters and have MAC operations lower by at least an order. Compared to them, our PSNR/SSIM values are lower by only a small margin, which should be acceptable in the light of many previous methods aimed at feasible real-time performance [68, 16, 75, 25, 52, 24, 34, 9]. Moreover, in Sec. 4.4 we shall demonstrate that our model exhibits the best generalizability over the state-of-the-art.

PSNR/SSIM are known to correlate less with human visual perception [5] as compared to recent perceptual quality metrics *Ma* [43] and *NIQE* [47]. In fact a restoration with high PSNR/SSIM can have very poor perceptual quality [70, 5]. Thus as far as the end goal of enhancement is concerned, which is to have a pleasing and visually consistent image, we are at par with state-of-the-art. This is substantiated by our having the same perceptual score, Ma/NIQE, as the top-performing models. Fig. 4 further confirms this wherein the restorations achieved by our lightweight model are hard to distinguish from those reached by heavy-duty models, which may overwhelm the target devices.

4.3. Amplifier module

We augment our model trained on the SID Sony dataset with the proposed amplifier, which automatically estimates the amplification factor from the raw image. Fig. 5 shows the restored images for different values of the amplification hyperparameter m. We observe that all the restorations are quite good with nearly the same perceptual score (Ma score) as that of the GT. These restorations only differ in overall brightness with no perceived distortions. Yet the PSNR/SSIM vary a lot. In fact, in the first row, the restoration having a PSNR of 34dB is perceptually no superior than the restoration having just 16dB PSNR. This happens because PSNR/SSIM is measured with respect to a particular GT. But for image enhancement, the choice of GT is quite subjective. For example, for an image captured late in the night, one can potentially select any image captured with randomly higher exposure as the GT. Thus, no-reference perceptual quality metrics such as Ma and NIQE are more appropriate for benchmarking image enhancement. We also note that amongst all restorations, the one with $m = 0.5$ generally has the highest Ma score, which is sometimes even higher than the GT's perceptual score. In fact, the visibility in the images restored with $m = 0.5$ is often better than the GT. We thus fix $m = 0.5$ as the default.

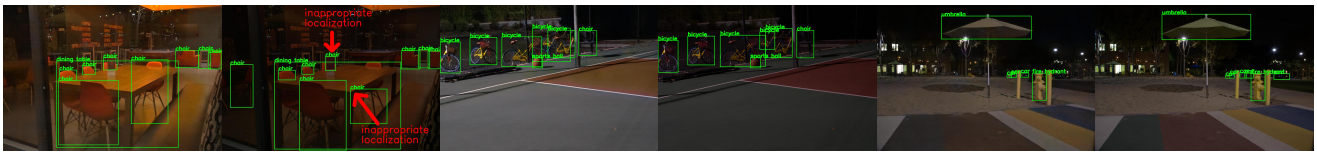


Figure 7. Visualization of the objects detected by Faster RCNN from the dark images restored by our model. The proposed amplifier enables better visibility and occasionally favors better localization.

Methods NR: No Retrain, R: Retrain	PSNR ↑ / SSIM ↑ GT	 GT-GC	Ma ↑ / NIQE ↓ No reference
LLPackNet + [34] amp. + R	23.27/0.69	23.43/0.71	5.83 / 5.50
LLPackNet + Our amp. + NR	**23.35/0.71**	**24.34/0.74**	**5.97 / 5.12**
SID + [34] amp. + R	22.98/0.71	23.17/0.72	6.21 / 4.90
SID + Our amp. + NR	**23.84/0.74**	**26.13/0.78**	**6.94 / 4.39**
Ours + [34] amp. + R	23.30/0.71	23.80/0.72	6.11 / 4.62
Ours + Our amp. + NR	**23.85/0.75**	**26.08/0.79**	**6.93 / 4.40**
DID + Our amp. + NR	23.29/0.71	25.64/0.74	6.84 / 4.49
SGN + Our amp. + NR	23.90/0.73	26.12/0.77	6.94 / 4.40
DCE + Our amp. + NR	22.72/0.70	25.13/0.72	6.08 / 4.65
LDC + Our amp. + NR	23.86/0.75	26.10/0.79	6.92 / 4.40

Table 2. Quantitative superiority of our amplifier module over the MLP based amplifier [34].

		SID [10]	SGN [18]	LDC [65]	Ours
Canon EOS 70D	**PSNR/SSIM**	24.80/0.51	23.75/0.45	22.65/0.40	**24.90/0.53**
	Ma/NIQE	6.84/4.32	6.70/4.35	6.07/4.93	**6.85/4.31**
	GMACs	925.64	>3000	>3000	**98.48**
Canon EOS 700D	**PSNR/SSIM**	25.32/0.51	25.09/0.48	24.21/0.41	**25.33/0.52**
	Ma/NIQE	6.85/4.25	6.72/4.28	6.74/4.31	**6.87/4.23**
	GMACs	830.77	>3000	>3000	**88.38**
Nikon D850	**PSNR/SSIM**	26.01/0.60	25.88/0.57	25.18/0.51	**26.27/0.63**
	Ma/NIQE	6.88/4.61	6.85/4.63	6.79/4.68	**6.90/4.60**
	GMACs	912.16	>3000	>3000	**97.04**

Table 3. Training on Sony α7S II camera and testing on images captured using different cameras.

In Table 2, we compare the proposed amplifier with an MLP based amplifier [34], which can also automatically estimate the amplification using the raw image's histogram. The MLP based amplifier had to be *independently re-trained with each restoration model, while our amplifier was augmented without any fine-tuning*. Because of this difficulty with MLP based amplifier, we report results with it for only three methods. We observe that for each restoration model, our amplifier achieves better quantitative scores. This is because the MLP based amplifier estimates quite low amplification factors in the range $4-5$ exhibiting poor sensitivity. In contrast, our amplifier's had estimates anywhere from $20-300$. The superiority of our amplifier is also substantiated by the visual results shown in Fig. 6, wherein we compare both amplifiers using the SID model as the main restoration method.

Given that there could be several acceptable GT for extreme low-light images, we generate multiple GT images by gamma correcting the reference images in a small window of 0.7 to 1.3 with a step size of 0.03. The maximum PSNR/SSIM values are reported in the column 'GT-GC' of Table. 2. We immediately note a $2-3dB$ improvement for our amplifier. This confirms that our amplifier does not introduce any artifacts in the restoration but only induces

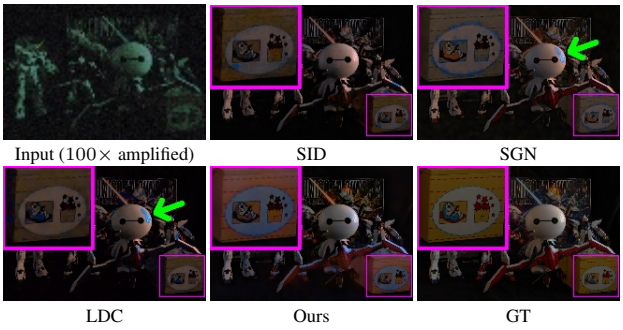


Figure 8. Training on Sony α7S II camera and testing on Nikon D850.

a different global brightness. However, a similar improvement is not seen for the MLP based amplifier because it actually causes several distortions in the restored image, as exemplified in Fig. 6. More results can be found in the supplementary.

4.4. Generalizability

From Sec. 4.2, it is evident that our network's fast inference speed and computational efficiency is not at the cost of restoration quality. To further substantiate this observation, we conducted additional experiments wherein we do not assume the knowledge of GT exposure and augment our amplifier to all the models.

Cameras of different make and model. The existing state-of-the-art restoration models exert a high model and computational complexity for a disproportionately small increase in reconstruction quality. This makes them susceptible to overfitting on a particular dataset, and the performance may not translate to anonymous images. To investigate this, we employed the models trained on the SID Sony camera dataset to restore the test images captured by other cameras present in the ELD dataset [64]. Table 3 reports the average results. Our model achieves the best PSNR / SSIM / Ma / NIQE score with much less computation.

Object detection. We ran Faster RCNN [53] trained on MS COCO [40] over the SID Sony dataset's low-light images restored by different methods. To compute the mean Average Precision (mAP) of the detected objects, we acquired the GT objects by running Faster RCNN over the GT images with confidence greater than 70%. A similar strategy was adopted by Sasagawa and Nagahara [57]. The results are shown in Table 4, wherein our model achieves

Method	mAP % ↑	GMAC ↓
SID + Our amplifier + Faster RCNN	53.15	12.15 + 124.63
SGN + Our amplifier + Faster RCNN	54.57	53.51 + 124.63
Our model + Our amplifier + Faster RCNN	**54.67**	**1.29** + 124.63

Table 4. mean Average Precision (mAP) with an IoU threshold of 50% for object detection on the SID Sony dataset.

Block	Percentage of total time CPU	GPU
Lower Scale Encoder (LSE)	14.58 %	15.05 %
Medium Scale Encoder (MSE)	16.23 %	15.14 %
Higher Scale Encoder (HSE)	15.36 %	16.90 %
Fuse Block 1 (FB 1)	17.94 %	19.61 %
Fuse Block 2 (FB 2)	35.89 %	33.30 %

Table 5. Run-time for each block in our network. Parallelizing LSE/MSE/HSE achieves $\approx 30\%$ speedup.

better mAP than the competing methods.

Table 4 also report GMACs for a 512×512 image. Our method is at least $10\times$ lighter, consuming less than 2% of the total computation required for the joint task of enhancement and detection. Our model can enhance 512×512 images on a GPU at 200 fps; but when Faster RCNN follows our model, the speed reduces to 22 fps and with YOLO [52] it is close to 38 fps.

Recently, [57] designed and trained a network specifically for object detection (and not enhancement) on the SID dataset. They achieved a 55% mAP@IoU=0.5. On the other hand, we created our model with the sole intention of enhancement and still manage to achieve a 54% mAP@IoU=0.5 without any fine-tuning. Note that this is not a strict one-to-one comparison because we did not have access to their exact dataset or code; still, we mention these results just to put things in perspective. Lastly, it would be best to have manually annotated labels because the low contrast of the GT images sometimes causes improper localization. Interestingly, our amplifier offers better scene visibility, which occasionally improves the localization, see Fig. 7.

4.5. Architectural parallelism

Table 5 reports the run-time for each block in our network. Coherent with our design objective discussed in Sec. 3.1.2, LSE/MSE/HSE all took the same time to execute, which is roughly 15% of the total execution time. We then assigned three different computational nodes (warps [12, 48] in GPU or threads in CPU) to process the LSE, MSE and HSE blocks independently, and this reduced the encoder's net run-time roughly from 45% to 15% of the total run-time. Note that this architectural parallelism was possible because we explicitly designed our encoder such that there are no dependencies amongst three scale-spaces. In contrast, the higher scales in existing methods operate on the output of previous scale-spaces, and so parallelizing them is not straightforward, if not impossible.

Our architecture incurs only two additional communication overheads — one at the beginning of FB 1 and the other at the start of FB 2. Moreover, the outputs of LSE/MSE/HSE were only $30MB - 120MB$ at single precision, and so the communication overhead was small because modern CPUs and GPUs offer a data-bandwidth of $20 - 50$ GB/s and $750 - 1000$ GB/s, respectively [12, 48]. In our pre-preliminary experiment, architectural parallelism boosted SID Sony images' inference speed from 32 fps to 40 fps on our GPU. Given our limited experience in writing CUDA codes, we are hopeful that efficient implementations in the future can further raise it to 45 fps or more.

w/o masking (Eq. (4)) proposed (Eq. (5)) GT

Figure 9. Ablation study on the proposed amplifier module.

4.6. Ablation study

Model architecture. We retrained our network on the SID Sony dataset after replacing the proposed RDB* with the canonical RDB using GT exposure setting for pre-amplification. Both used LeakyReLU non-linearity, as suggested by SID. This caused the PSNR/SSIM to drop from $28.66dB/0.79$ to $27.96dB/0.77$.

Amplifier module. We disable masking highlights and use Eq. (4) instead of the proposed Eq. (5) for amplification. We then augmented it to our pre-trained Sony model and observed that often the restoration had a greenish color cast due to inappropriate amplification, as shown in Fig. 9. Quantitatively, PSNR/SSIM reduced by $0.5dB/0.01$.

5. Conclusion

We proposed a fast and lightweight deep learning based solution for extreme low-light restoration. Specifically, compared to top-performing methods, we were $5 - 100\times$ faster while having similar qualitative results and quantitative perceptual scores. We also devised a mechanism to enable architectural parallelism to further boost the inference speed for a $4K$ resolution image from 32 fps to 40 fps on GPU. The proposed amplifier's effectiveness was also demonstrated by augmenting it to all seven restoration methods without any fine-tuning and yet achieve good restorations, having perceptual scores occasionally better than GT images. Besides being fast, our method generalizes better than the state-of-the-art in restoring images captured from a camera not seen during training and for the task of object detection. Finally, the ablation study confirmed the efficacy of the proposed RDB* over the RDB in restricting the negative impact of non-linear rectifications.

References

[1] Abdelrahman Abdelhamed, Marcus A Brubaker, and Michael S Brown. Noise flow: Noise modeling with conditional normalizing flows. In *ICCV*, 2019.

[2] Abdelrahman Abdelhamed, Stephen Lin, and Michael S Brown. A high-quality denoising dataset for smartphone cameras. In *CVPR*, 2018.

[3] Sophy Ai and Jangwoo Kwon. Extreme low-light image enhancement for surveillance cameras using attention u-net. *Sensors*, 20(2):495, 2020.

[4] Yousef Atoum, Mao Ye, Liu Ren, Ying Tai, and Xiaoming Liu. Color-wise attention network for low-light image enhancement. In *CVPR Workshop*, 2020.

[5] Yochai Blau and Tomer Michaeli. The perception-distortion tradeoff. In *CVPR*, 2018.

[6] Tim Brooks, Ben Mildenhall, Tianfan Xue, Jiawen Chen, Dillon Sharlet, and Jonathan T Barron. Unprocessing images for learned raw denoising. In *CVPR*, 2019.

[7] Peter Burt and Edward Adelson. The laplacian pyramid as a compact image code. *IEEE Transactions on communications*, 31(4):532–540, 1983.

[8] Peter J Burt and Edward H Adelson. A multiresolution spline with application to image mosaics. *ACM Transactions on Graphics (TOG)*, 2(4):217–236, 1983.

[9] Alfredo Canziani, Adam Paszke, and Eugenio Culurciello. An analysis of deep neural network models for practical applications. *arXiv:1605.07678*, 2016.

[10] Chen Chen, Qifeng Chen, Jia Xu, and Vladlen Koltun. Learning to see in the dark. In *CVPR*, 2018.

[11] Dinu Coltuc, Philippe Bolon, and J-M Chassery. Exact histogram specification. *IEEE Transactions on Image Processing*, 15(5):1143–1152, 2006.

[12] Tim Dettmers. Making deep learning accessible. In *https://timdettmers.com/category/hardware*, Accessed, Nov. 2020.

[13] Gabriel Eilertsen, Joel Kronander, Gyorgy Denes, Rafał K Mantiuk, and Jonas Unger. Hdr image reconstruction from a single exposure using deep cnns. *ACM transactions on graphics (TOG)*, 36(6):1–15, 2017.

[14] Xueyang Fu, Delu Zeng, Yue Huang, Yinghao Liao, Xinghao Ding, and John Paisley. A fusion-based enhancing method for weakly illuminated images. *Signal Processing*, 129:82–96, 2016.

[15] Xueyang Fu, Delu Zeng, Yue Huang, Xiao-Ping Zhang, and Xinghao Ding. A weighted variational model for simultaneous reflectance and illumination estimation. In *CVPR*, 2016.

[16] Hongyang Gao, Zhengyang Wang, and Shuiwang Ji. Channelnets: Compact and efficient convolutional neural networks via channel-wise convolutions. In *NIPS*, 2018.

[17] Abhiram Gnanasambandam and Stanley H Chan. Image classification in the dark using quanta image sensors. *ECCV*, 2020.

[18] Shuhang Gu, Yawei Li, Luc Van Gool, and Radu Timofte. Self-guided network for fast image denoising. In *ICCV*, 2019.

[19] Chunle Guo, Chongyi Li, Jichang Guo, Chen Change Loy, Junhui Hou, Sam Kwong, and Runmin Cong. Zero-reference deep curve estimation for low-light image enhancement. In *CVPR*, 2020.

[20] Xiaojie Guo, Yu Li, and Haibin Ling. Lime: Low-light image enhancement via illumination map estimation. *IEEE Transactions on image processing*, 26(2):982–993, 2016.

[21] Dongyoon Han, Jiwhan Kim, and Junmo Kim. Deep pyramidal residual networks. In *CVPR*, 2017.

[22] Samuel W Hasinoff, Dillon Sharlet, Ryan Geiss, Andrew Adams, Jonathan T Barron, Florian Kainz, Jiawen Chen, and Marc Levoy. Burst photography for high dynamic range and low-light imaging on mobile cameras. *ACM Transactions on Graphics (TOG)*, 35(6):1–12, 2016.

[23] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *ICCV*, 2015.

[24] Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv:1704.04861*, 2017.

[25] Gao Huang, Shichen Liu, Laurens Van der Maaten, and Kilian Q Weinberger. Condensenet: An efficient densenet using learned group convolutions. In *CVPR*, 2018.

[26] Forrest N Iandola, Song Han, Matthew W Moskewicz, Khalid Ashraf, William J Dally, and Kurt Keutzer. Squeezenet: Alexnet-level accuracy with 50$\times$ fewer parameters and $<$ 0.5 mb model size. *arXiv:1602.07360*, 2016.

[27] Haidi Ibrahim and Nicholas Sia Pik Kong. Brightness preserving dynamic histogram equalization for image contrast enhancement. *IEEE Transactions on Consumer Electronics*, 53(4):1752–1758, 2007.

[28] Yani Ioannou, Duncan Robertson, Roberto Cipolla, and Antonio Criminisi. Deep roots: Improving cnn efficiency with hierarchical filter groups. In *CVPR*, 2017.

[29] Yifan Jiang, Xinyu Gong, Ding Liu, Yu Cheng, Chen Fang, Xiaohui Shen, Jianchao Yang, Pan Zhou, and Zhangyang Wang. Enlightengan: Deep light enhancement without paired supervision. *arXiv:1906.06972*, 2019.

[30] Daniel J Jobson, Zia-ur Rahman, and Glenn A Woodell. A multiscale retinex for bridging the gap between color images and the human observation of scenes. *IEEE Transactions on Image processing*, 6(7):965–976, 1997.

[31] Yeong-Taeg Kim. Contrast enhancement using brightness preserving bi-histogram equalization. *IEEE transactions on Consumer Electronics*, 43(1):1–8, 1997.

[32] Ron Kimmel, Michael Elad, Doron Shaked, Renato Keshet, and Irwin Sobel. A variational framework for retinex. *International Journal of computer vision*, 52(1):7–23, 2003.

[33] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *ICLR*, 2015.

[34] Mohit Lamba, Atul Balaji, and Kaushik Mitra. Towards fast and light-weight restoration of dark images. In *BMVC*, 2020.

[35] Edwin H Land. The retinex theory of color vision. *Scientific american*, 237(6):108–129, 1977.

[36] Edwin H Land and John J McCann. Lightness and retinex theory. *Josa*, 61(1):1–11, 1971.

[37] Chulwoo Lee, Chul Lee, and Chang-Su Kim. Contrast enhancement based on layered difference representation of 2d histograms. *IEEE transactions on image processing*, 22(12):5372–5384, 2013.

[38] Chongyi Li, Jichang Guo, Fatih Porikli, and Yanwei Pang. Lightennet: a convolutional neural network for weakly illuminated image enhancement. *Pattern Recognition Letters*, 104:15–22, 2018.

[39] Mading Li, Jiaying Liu, Wenhan Yang, Xiaoyan Sun, and Zongming Guo. Structure-revealing low-light image enhancement via robust retinex model. *IEEE Transactions on Image Processing*, 27(6):2828–2841, 2018.

[40] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *ECCV*, 2014.

[41] Ziwei Liu, Lu Yuan, Xiaoou Tang, Matt Uyttendaele, and Jian Sun. Fast burst images denoising. *ACM Transactions on Graphics (TOG)*, 33(6):1–9, 2014.

[42] Kin Gwn Lore, Adedotun Akintayo, and Soumik Sarkar. Llnet: A deep autoencoder approach to natural low-light image enhancement. *Pattern Recognition*, 61:650–662, 2017.

[43] Chao Ma, Chih-Yuan Yang, Xiaokang Yang, and Ming-Hsuan Yang. Learning a no-reference quality metric for single-image super-resolution. *Computer Vision and Image Understanding*, 158:1–16, 2017.

[44] Paras Maharjan, Li Li, Zhu Li, Ning Xu, Chongyang Ma, and Yue Li. Improving extreme low-light image denoising via residual learning. In *Int. Conf. Multimedia and Expo (ICME)*, 2019.

[45] Laurence Meylan and Sabine Susstrunk. High dynamic range image rendering with a retinex-based adaptive filter. *IEEE Transactions on image processing*, 15(9):2820–2830, 2006.

[46] Ben Mildenhall, Jonathan T Barron, Jiawen Chen, Dillon Sharlet, Ren Ng, and Robert Carroll. Burst denoising with kernel prediction networks. In *CVPR*, 2018.

[47] Anish Mittal, Rajiv Soundararajan, and Alan C Bovik. Making a "completely blind" image quality analyzer. *IEEE Signal processing letters*, 20(3):209–212, 2012.

[48] NVIDIA. Cuda c++ best practices guide. In https://docs.nvidia.com/cuda/pdf/CUDA_C_Best_Practices_Guide.pdf, Accessed, Nov. 2020.

[49] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In *NIPS*. 2019.

[50] Chao Peng, Xiangyu Zhang, Gang Yu, Guiming Luo, and Jian Sun. Large kernel matters–improve semantic segmentation by global convolutional network. In *CVPR*, 2017.

[51] Stephen M Pizer, E Philip Amburn, John D Austin, Robert Cromartie, Ari Geselowitz, Trey Greer, Bart ter Haar Romeny, John B Zimmerman, and Karel Zuiderveld. Adaptive histogram equalization and its variations. *Computer vision, graphics, and image processing*, 39(3):355–368, 1987.

[52] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *CVPR*, 2016.

[53] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *NIPS*, 2015.

[54] Wenqi Ren, Sifei Liu, Lin Ma, Qianqian Xu, Xiangyu Xu, Xiaochun Cao, Junping Du, and Ming-Hsuan Yang. Low-light image enhancement via a deep hybrid network. *IEEE Transactions on Image Processing*, 28(9):4364–4375, 2019.

[55] Tim Salimans and Durk P Kingma. Weight normalization: A simple reparameterization to accelerate training of deep neural networks. In *NIPS*, 2016.

[56] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *CVPR*, 2018.

[57] Yukihiro Sasagawa and Hajime Nagahara. Yolo in the dark-domain adaptation method for merging multiple models. In *ECCV*, 2020.

[58] Wenzhe Shi, Jose Caballero, Ferenc Huszár, Johannes Totz, Andrew P Aitken, Rob Bishop, Daniel Rueckert, and Zehan Wang. Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network. In *CVPR*, 2016.

[59] J Alex Stark. Adaptive image contrast enhancement using generalizations of histogram equalization. *IEEE Transactions on image processing*, 9(5):889–896, 2000.

[60] Deqing Sun, Xiaodong Yang, Ming-Yu Liu, and Jan Kautz. Pwc-net: Cnns for optical flow using pyramid, warping, and cost volume. In *CVPR*, 2018.

[61] Ruixing Wang, Qing Zhang, Chi-Wing Fu, Xiaoyong Shen, Wei-Shi Zheng, and Jiaya Jia. Underexposed photo enhancement using deep illumination estimation. In *CVPR*, 2019.

[62] Shuhang Wang, Jin Zheng, Hai-Miao Hu, and Bo Li. Naturalness preserved enhancement algorithm for non-uniform illumination images. *IEEE Transactions on Image Processing*, 22(9):3538–3548, 2013.

[63] Chen Wei, Wenjing Wang, Wenhan Yang, and Jiaying Liu. Deep retinex decomposition for low-light enhancement. In *BMVC*, 2018.

[64] Kaixuan Wei, Ying Fu, Jiaolong Yang, and Hua Huang. A physics-based noise formation model for extreme low-light raw denoising. In *CVPR*, 2020.

[65] Ke Xu, Xin Yang, Baocai Yin, and Rynson WH Lau. Learning to restore low-light images via decomposition-and-enhancement. In *CVPR*, 2020.

[66] Wenhan Yang, Shiqi Wang, Yuming Fang, Yue Wang, and Jiaying Liu. From fidelity to perceptual quality: A semi-supervised approach for low-light image enhancement. In *CVPR*, 2020.

[67] Jiahui Yu, Yuchen Fan, Jianchao Yang, Ning Xu, Zhaowen Wang, Xinchao Wang, and Thomas Huang. Wide activation for efficient and accurate image super-resolution. *arXiv:1808.08718*, 2018.

[68] Jiuwei Li Yuxi Li, Jianguo Li and Weiyao Lin. Tiny-DSOD: Lightweight object detection for resource-restricted usage. In *BMVC*, 2018.

[69] Kai Zhang, Wangmeng Zuo, and Lei Zhang. Ffdnet: Toward a fast and flexible solution for cnn-based image denoising. *IEEE Transactions on Image Processing*, 27(9):4608–4622, 2018.

[70] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *CVPR*, 2018.

[71] Ting Zhang, Guo-Jun Qi, Bin Xiao, and Jingdong Wang. Interleaved group convolutions. In *ICCV*, 2017.

[72] Yulun Zhang, Yapeng Tian, Yu Kong, Bineng Zhong, and Yun Fu. Residual dense network for image super-resolution. In *CVPR*, 2018.

[73] Yonghua Zhang, Jiawan Zhang, and Xiaojie Guo. Kindling the darkness: A practical low-light image enhancer. In *ACM International Conference on Multimedia*, 2019.

[74] Hang Zhao, Orazio Gallo, Iuri Frosio, and Jan Kautz. Loss functions for image restoration with neural networks. *IEEE Transactions on computational imaging*, 3(1):47–57, 2016.

[75] Hengshuang Zhao, Xiaojuan Qi, Xiaoyong Shen, Jianping Shi, and Jiaya Jia. Icnet for real-time semantic segmentation on high-resolution images. In *ECCV*, 2018.

[76] Yinqiang Zheng, Mingfang Zhang, and Feng Lu. Optical flow in the dark. In *CVPR*, 2020.