IMAGE PROCESSING AND SYNTHESIS: FROM HAND-CRAFTED TO DATA-DRIVEN MODELING

BY

CHEN CHEN

DISSERTATION

Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy in Electrical and Computer Engineering
in the Graduate College of the
University of Illinois at Urbana-Champaign, 2018

Urbana, Illinois

Doctoral Committee:

Professor Minh N. Do, Chair
Professor David A. Forsyth
Professor John C. Hart
Dr. Vladlen Koltun, Senior Principal Researcher at Intel Labs
Assistant Professor Alexander Schwing

ABSTRACT

This work investigates image and video restoration problems using effective optimization algorithms. First, we study the problem of single image dehazing to suppress artifacts in compressed or noisy images and videos. Our method is based on the linear haze model and minimizes the gradient residual between the input and output images. This successfully suppresses any new artifacts that are not obvious in the input images. Second, we propose a new method for image inpainting using deep neural networks. Given a set of training data, deep generate models can generate high-quality natural images following the same distribution. We search the nearest neighbor in the latent space of the deep generate models using a weighted context loss and prior loss. This code is then converted to the clean and uncorrupted image of the input. Third, we study the problem of recovering high-quality images from very noisy raw data captured in low-light conditions with short exposures. We build deep neural networks to learn the camera processing pipeline specifically for low-light raw data with an extremely low signal-to-noise ratio (SNR). To train the networks, we capture a new dataset of more than five thousand images with short-exposed and long-exposed pairs. Promising results are obtained compared with the traditional image processing pipeline. Finally, we propose a new method for extreme-low light video processing. The raw video frames are pre-processed using spatial-temporal denoising. A neural network is trained to move the error in the pre-processed data, learning to perform the image processing pipeline and encourage temporal smoothness of the output. Both quantitative and qualitative results demonstrate the proposed method significantly outperform the existing methods. It also paves the way for future research on this area.

To my wife and my parents, for their love and support.

ACKNOWLEDGMENTS

I would like to thank my advisor Professor Minh N. Do for his guidance and support of my research. His wisdom inspired many ideas related to this work. I would also like to thank my thesis committee, Professor David A. Forsyth, Professor John Hart, and Professor Alexander Schwing and Dr. Vladlen Koltun, for their valuable comments of my work. I also thank Dr. Jue Wang, Dr. Oliver Wang, Dr. Eli Shechtman, Dr. David Salesin, Dr. Qifeng Chen, Dr. Jia Xu and Dr. Vladlen Koltun for the internship opportunities and their advice during the internships. Additionally, I would like to thank my fellow labmates for their help and many enlightening discussions, especially Raymond Yeh and Teckyian Lim for their collaborations to make the third chapter of this work. Thank Peggy Wells, Trong N. Nguyen, Mona Zehni, Vaishnavi Subramanian and Khoi-Nguyen Mac for allowing me to use their likenesses in the paper and presentations. I would like to recognize my family and friends for all of their love and encouragement. Finally, I would like to express my love and appreciation to my wife Jingcao, for her companion and support during the entire study.

TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION

1.1 Motivation

Given a corrupt, noisy or degraded input, image restorations aims to estimate the clean, original image. This inverse process is often ill-posed [1] and the level of noise corruption is also unknown. In order to seek a meaningful solution, it is often required to exploit some prior knowledge to counter noise or error in the data. Early approaches assume that the image data approximately follow the Gaussian distribution, such as the Tikhonov regularization and Gaussian mixture model. Later methods prefer a sparse prior to recover the missing, unknown or corrupted image information. This is because images are natural high-dimensional data but can be sparsely represented with respect to fixed bases or learned dictionaries [2]. Variations and extensions of the sparsity-based methods have led to promising results in a lot of applications, e.g., [3, 4].

The major part of our previous research is designing task-specific sparse models to low-level image processing problems by assuming the fixed bases or dictionaries are given [5, 6, 7, 8]. Despite the great success of these novel hand-crafted sparse models, there are still many limitations. First, it is not trivial to craft effective sparse models in some vision tasks. Another disadvantage is that the well-designed models for one task are often not easily adapted to other tasks. Moreover, the existing hand-crafted sparse models are often based on task-dependent observations without explicitly using the training data. Although dictionary learning could alleviate this, the off-the-shelf solvers are generally inefficient on large-scale dataset. Finally, even with the advanced optimization techniques, the speed of the sparsity-based methods is still hard to meet the requirement of real-time applications. These limitations motivate us to investigate a new learning framework to exploit image priors. In particular, we are interested in how to incorporate the large amount of available data into learning image priors.

1.2 Overview

In Chapter 2, we study the problem of image restoration from hazy images [9], which is called image dehazing in the literature. Most existing image dehazing methods tend to boost local image contrast for regions with heavy haze. Without special treatment, these methods may significantly amplify existing image artifacts such as noise, color aliasing and blocking, which are mostly invisible in the input images but are visually intruding in the results. This is especially the case for low-quality cellphone shots or compressed video frames. The recent work of Li et al. [10] addresses blocking artifacts for dehazing, but is insufficient to handle other artifacts. In this chapter, we propose a new method for reliable suppression of different types of visual artifacts in image and video dehazing. Our method makes contributions in both the haze estimation step and the image recovery step. Firstly, an image-guided, depth-edge-aware smoothing algorithm is proposed to refine the initial atmosphere transmission map generated by local priors. In the image recovery process, we propose Gradient Residual Minimization (GRM) for jointly recovering the haze-free image while explicitly minimizing possible visual artifacts in it. Our evaluation suggests that the proposed method can generate results with many fewer visual artifacts than previous approaches for lower-quality inputs such as compressed video clips.

In Chapter 3, we investage image restoration from corrupted images [11]. A large block is missing in the input images, and we need to recover the missing structure based on the context. This new problem is called semantic image inpainting, which differs from previous methods that only fill a smooth region or textures in the missing portion of the images. Semantic image inpainting is a challenging task where large missing regions have to be filled based on the available visual data. Existing methods which extract information from only a single image generally produce unsatisfactory results due to the lack of high-level context. In this chapter, we propose a novel method for semantic image inpainting, which generates the missing content by conditioning on the available data. Given a trained generative model, we search for the closest encoding of the corrupted image in the latent image manifold using our context and prior losses. This encoding is then passed through the generative model to infer the missing content. In our method, inference is possible irrespective of how the missing content is structured, while the state-of-the-art learning-based method requires specific information about the holes in the training phase. Experiments on three datasets show that our method successfully predicts information in large missing regions and achieves pixel-level photorealism, significantly outperforming the state-of-the-art methods.

In Chapter 4, a new pipeline is exploited to recover high-quality images from extreme low signal-to-noise ratio (SNR) images captured on fast low-light conditions [12]. Imaging in low light is challenging due to low photon count and low SNR. Short-exposure images suffer from noise, while long exposure can lead to blurry images and is often impractical. A variety of denoising, deblurring, and enhancement techniques have been proposed, but their effectiveness is limited in extreme conditions, such as video-rate imaging at night. To support the development of learning-based pipelines for low-light image processing, we introduce a dataset of raw short-exposure nighttime images, with corresponding long-exposure reference images. Using the presented dataset, we develop a pipeline for processing low-light images, based on end-to-end training of a fully convolutional network. The network operates directly on raw sensor data and replaces much of the traditional image processing pipeline, which tends to perform poorly on such data. We report promising results on the new dataset, analyze factors that affect performance, and highlight opportunities for future work.

In Chapter 5, we consider deep processing of very dark video: on the order of one lux of illuminance. At this level of darkness, the SNR is extremely low. (Negative if measured in dB.) We train deep networks on raw data directly. To support this line of work, we collect a new dataset of raw low-light video, in which high-resolution (20 MP) raw data is captured at video rate. We introduce a loss that can be used to train a network on videos of static scenes, for which ground truth is available, such that the network generalizes to videos of dynamic scenes at test time. Experimental results demonstrate that the presented approach outperforms state-of-the-art models for burst processing, per-frame processing, and blind temporal consistency.

CHAPTER 2

IMAGE AND VIDEO DEHAZING

2.1 Introduction

[1] Due to atmospheric absorption and scattering, outdoor images and videos are often degraded to have low contrast and visibility. In addition to the deterioration of visual quality, heavy haze also makes many computer vision tasks more difficult, such as stereo estimation, object tracking and detection etc. Therefore, removing haze from images and video becomes an important component in a post-processing pipeline. Conventional global contrast enhancement methods often do not perform well because the degradation is spatially varying. In general, accurate haze estimation and removal from a single image is a challenging task due to its ill-posed nature.

Haze removal has been extensively studied in the literature. Early approaches focus on using multiple images or extra information [13, 14, 15, 16] for dehazing. Recently dehazing from a single image has gained considerable attention, and can be broadly classified into two groups: methods based on transmission estimation [17, 18, 19, 20] and ones based on adaptive contrast enhancement [21, 22, 23]. Techniques in the latter group do not rely on any physical haze model, thus often suffer from visual artifacts such as strong color shift. The state-of-the-art methods often depend on a physical haze model for more accurate haze removal. They first estimate the atmosphere transmission map along with the haze color based on local image priors such as the dark channel prior [17] and the color-line prior [19]. The latent, haze-free image is then computed by directly removing the haze component in each pixel's color. Some methods are proposed to deal with special cases. For example, planar constraints can be utilized in road images [24]. Li et

[1] Chapter 2 has been published in *Proceedings of the 14th European Conference on Computer Vision*, 2016 [9]. Copyright Springer.

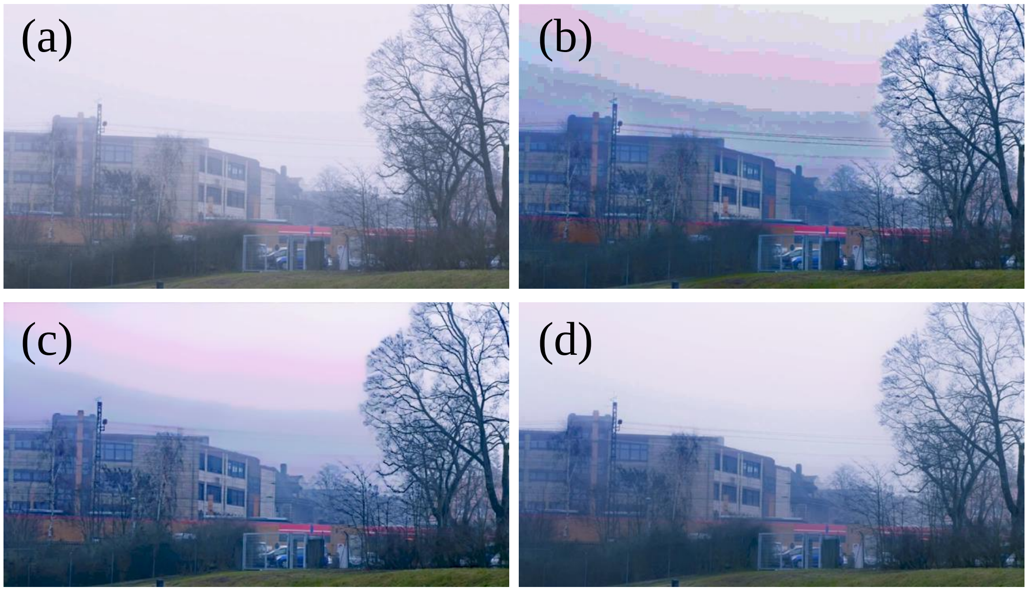


Figure 2.1: Dehaze one video frame. (a) Input image. (b) Result of He et al. [17]. (c) Result of Li et al. [10]. (d) Ours. Note the strong banding and color shifting artifacts in the sky region in (b) and (c).

al. proposed a method to dehaze videos when the coarse depth maps can be estimated by multi-view stereo [25].

The state-of-the-art methods usually can generate satisfactory results on high-quality input images. For lower-quality inputs, such as images captured and processed by mobile phones, or compressed video clips, most existing dehazing methods will significantly amplify image artifacts that are visual unnoticeable in the input, especially in heavy haze regions. An example is shown in Fig. 2.1, where the input image is one video frame extracted from a sequence captured by a cellphone camera. After dehazing using previous methods [17, 10], strong visible artifacts appear in the sky region of the results. These artifacts cannot be easily removed using post-processing filters without hampering the image content of other regions. Similarly, removing the original artifacts completely without destroying useful image details is also non-trivial as a pre-processing step.

Li et al. [10] were the first to consider the problem of artifact suppression in dehazing. Their approach is designed to remove only the blocking artifacts that are usually caused by compression. In this method, the input image is first decomposed into a structure layer and a texture layer, and dehazing is performed on the structure layer and deblocking is applied on the texture layer. The final output image is produced by re-combining the two layers. This method however often does not work well for other artifacts that commonly coexist in lower-quality inputs, e.g., the color banding artifact in Fig. 2.1 and color aliasing in later examples. In addition, their final results tend to be over-smoothed with missing fine image details, as we will show in our experimental results. This suggests that independent dehazing and deblocking on two separate layers is sub-optimal.

In this work, we propose a new method for image and video dehazing with an emphasis on preventing different types of visual artifacts in the output. Our method follows the general two-step framework and makes contributions in each step: estimating the atmosphere transmission map first, then recover the latent image. In the first step, after initializing the transmission map using existing local priors such as the dark channel prior [17], we refine it using a global method based on image guided Total Generalized Variation (TGV) [26] regularization. Compared with other commonly used refinement approaches, our method tends to produce transmission maps that are physically more correct: it produces very smooth regions within the surfaces/objects, while it generates strong edges at depth discontinuities. Observing that the boosted visual artifacts by existing methods are often not visible in the input image, in the second stage, we propose a novel way to recover the latent image by minimizing the gradient residual between the output and input images. It suppresses new edges which do not exist in the input image (often are artifacts), but has few effects on the edges that already exist, which are ideal properties for the dehazing task. Considering the existence of artifacts, the linear haze model may not hold on every pixel. We then explicitly introduce an “error” layer in the optimization, which could separate out the large artifacts that violate the linear haze model. Both quantitative and qualitative experimental results show that our method generates more accurate and more natural-looking results than the state-of-the-art methods on compressed inputs. In particular, our method shows significant improvement on video dehazing, which can suppress both spatial and temporal artifacts.

2.2 Overview of Transmission Map Initialization

The transmission map in our framework is required to be initialized by existing local priors, e.g., the widely used dark channel prior [17]. Here we provide a quick overview of the basic image formation model and this method. Note that our main contributions, transmission map refinement and image recovery, are orthogonal to the specific method that one could choose for initializing the transmission map.

Koschmieder et al. [27] proposed a physical haze model as:

$$\mathbf{I}(x) = \mathbf{J}(x)t(x) + \mathbf{A}(1 - t(x)) \tag{2.1}$$

where $\mathbf{I}$ is the hazy image, $\mathbf{J}$ is the scene radiance, $\mathbf{A}$ is the atmospheric light and assumed to be constant over the whole image, t is the medium transmission and x denotes the image coordinates. The transmission describes the portion of the light reaches to the camera without scattered. The task of dehazing is to estimate $\mathbf{J}$ (with $\mathbf{A}$ and t as by-products) from the input image $\mathbf{I}$, which is a severely ill-posed problem.

The dark channel prior, proposed by He et al. [17], is a simple yet efficient local image prior for estimating a coarse transmission map. The dark channel is defined as:

$$J^{dark}(x) = \min_{y \in \Omega(x)} (\min_{c \in \{r,g,b\}} J^c(y)) \tag{2.2}$$

where c denotes the color channel and $\Omega(x)$ is a local patch around x. Natural image statistics show that J^{dark} tends to be zero. We can rewrite Eq. (2.1) and take the minimum operations on both sides to get:

$$\min_{y \in \Omega(x)} (\min_{c} \frac{I^c(y)}{A^c}) = \min_{y \in \Omega(x)} (\min_{c} \frac{J(y)}{A^c} t(x)) + 1 - t(x) \tag{2.3}$$

By assuming the transmission map is constant in each small local patch, we can eliminate J^{dark} to obtain the coarse transmission map:

$$\tilde{t}(x) = 1 - \min_{y \in \Omega(x)} (\min_{c \in \{r,g,b\}} \frac{I^c(y)}{A^c}) \tag{2.4}$$

where the atmospheric light color $\mathbf{A}$ can be estimated as the brightest pixel color in the dark channel. This coarse transmission map is computed locally, thus often need to be refined. In practice it is often refined by soft matting [28] or guided image filtering [29]. Finally, the scene radiance is recovered by:

$$\mathbf{J}(x) = (\mathbf{I}(x) - \mathbf{A})/t(x) + \mathbf{A} \tag{2.5}$$

The dark channel prior described above is an elegant solution and often achieves high-quality results for high-quality inputs. However, as observed by by Li et al. [10], image artifacts, such as noise or blocking, can affect both dark channel computation and transmission map smoothing. The original dark channel approach often cannot generate high-quality results for images with artifacts.

2.3 TGV-Based Transmission Refinement

In the He et al. method, the transmission map is refined by soft matting [28] or guided image filtering [29]. Both methods are edge-aware operations. They work well with objects that have flat appearances. However, for objects/regions with strong textures, the refined transmission map using these methods tend to have false variations that are correlated with such textures. This is contradictory to the haze model, as the amount of haze in each pixel is only related to its depth, not its texture or color. Therefore, we expect the refined transmission map to be smooth inside the same object/surface, and only has discontinuities along depth edges. We thus propose a new transmission refinement method to try to achieve this goal without recovering the 3D scene.

We formulate the transmission refinement as a global optimization problem, consisting of a data fidelity term and regularization terms. Note that the transmission values of white objects are often underestimated by the dark channel method. We need a model that is robust to such outliers or errors. Instead of the commonly used ℓ_2 norm data term, we use the ℓ_1 norm to somewhat tolerate outliers and errors. The second-order Total Generalized Variation (TGV) [26, 30, 31, 32] with a guided image is adopted for regularization. Compared with conventional Total Variation (TV) regularization that encourages *piecewise constant* images and often suffers from undesired staircasing artifacts, TGV prefers *piecewise smooth* images. This is a desired property for the transmission, as we may have a slanted plane (e.g., road, bridge) whose transmission varies smoothly along with the change of depth.

Given the initial transmission $\tilde{t}$ and a guided image I, the optimization problem with TGV regularization is:

$$\min_{t,w}\{\alpha_1 \int |D^{1/2}(\nabla t - w)| \, \mathrm{d}x + \alpha_0 \int |\nabla w| \, \mathrm{d}x + \int |t - \tilde{t}| \, \mathrm{d}x\} \tag{2.6}$$

where $D^{1/2}$ is the anisotropic diffusion tensor [30] defined as:

$$D^{1/2} = \exp(-\gamma |\nabla I|^{\beta}) n n^T + n^{\perp} n^{\perp T} \tag{2.7}$$

where n is the direction of the gradient of the guided image $n = \frac{\nabla I}{|\nabla I|}$ and $n^{\perp}$ is the perpendicular direction, γ, β are parameters to adjust the sharpness and magnitude of the tensor, w is an auxiliary variable. Our experiments show that the sharp depth edges cannot be preserved without the guided image when using the TGV regularization. Unlike the previous local refinement methods, TGV performs globally and is less sensitive to the local textures.

To solve this problem, we apply the prime-dual minimization algorithm [33] with the Legendre Fenchel transform. The transformed primal-dual problem is given by:

$$\min_{t,w} \max_{p \in P, q \in Q} \{\alpha_1 \left\langle D^{1/2}(\nabla t - w), p \right\rangle + \alpha_0 \left\langle \nabla w, q \right\rangle + \int |t - \tilde{t}| \, \mathrm{d}x\} \tag{2.8}$$

where p, q are dual variables and their feasible sets are:

$$\begin{aligned} P &= \{p \in R^{2MN}, \|p\|_{\infty} \leq 1\} \\ Q &= \{q \in R^{4MN}, \|q\|_{\infty} \leq 1\} \end{aligned} \tag{2.9}$$

The algorithm for transmission refinement is formally summarized in Algorithm 1.

Algorithm 1 Transmission map refinement by guided TGV

```
**Initialization:** $t^0 = \tilde{t}$, $w^0, \bar{t}^0, \bar{w}^0, p^0, q^0 = 0$, $\sigma_p > 0$, $\sigma_q > 0$, $\tau_t > 0$, $\tau_w > 0$
**for** $k = 0$ **to** $Maxiteration$ **do**
    $p^{k+1} = \mathcal{P}[p^k + \sigma_p \alpha_1 (D^{1/2}(\nabla \bar{t}^k - \bar{w}^k))]$
    $q^{k+1} = \mathcal{P}[q^k + \sigma_q \alpha_0 \nabla \bar{w}^k]$
    $t^{k+1} = thresholding_{\tau}(t^k + \tau_u \alpha_1 \nabla^T D^{1/2} p^{k+1})$
    $w^{k+1} = w^k + \tau_w (\alpha_0 \nabla^T q^{k+1} + \alpha_1 D^{1/2} p^{k+1})$
    $\bar{t}^{k+1} = t^{k+1} + \theta(t^{k+1} - \bar{t}^k)$
    $\bar{w}^{k+1} = w^{k+1} + \theta(w^{k+1} - \bar{w}^k)$
**end for**
```

In the algorithm, $\sigma_p > 0$, $\sigma_q > 0$, $\tau_t > 0$, $\tau_w > 0$ are step sizes and k is the iteration counter. The element-wise projection operator $\mathcal{P}$ is defined:

$$\mathcal{P}[x] = \frac{x}{\max\{1, |x|\}} \tag{2.10}$$

The $thresholding_{\tau}()$ denotes the soft-thresholding operation:

$$thresholding_{\tau}(x) = \max(|x| - \tau, 0)\text{sign}(x) \tag{2.11}$$

The θ is updated in every iteration as suggested by [33]. The divergence and gradient operators in the optimization are approximated using standard finite differences. Please refer to [33] for more details of this optimization method.

Figure 2.2 shows the transmission maps estimated by guided filter, matting followed by bilateral filter and TGV refinement. Compared with guided image filtering or bilateral smoothing, our method is aware of the depth edges while producing a smooth surface within each objects (see the buildings indicated by the yellow circles). In addition, our optimization scheme does not exactly trust the initialization and it can somewhat tolerate the errors (see the house indicated by the blue arrow).

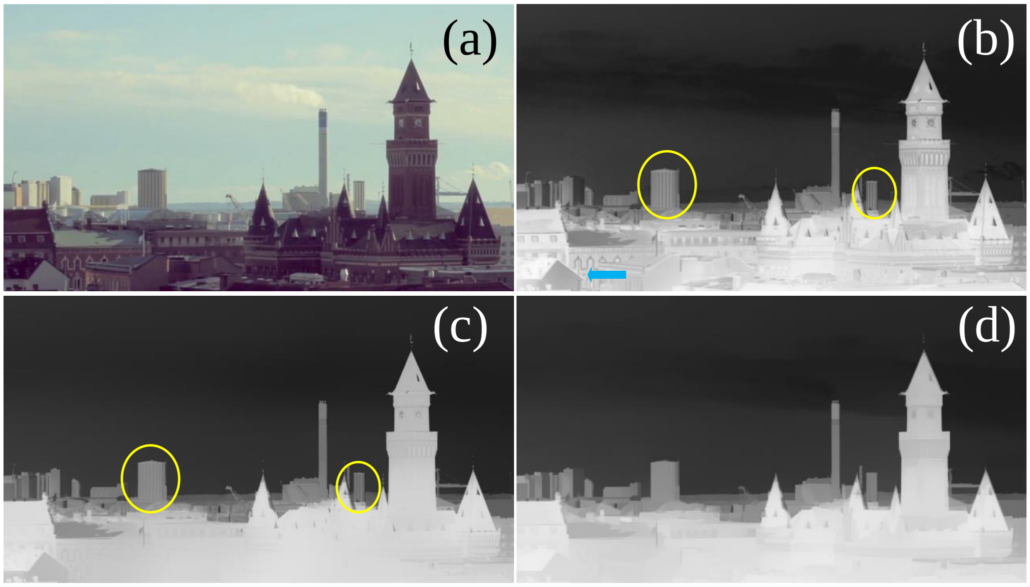


Figure 2.2: Comparisons of transmission refinement methods. (a) Input image. (b) Result of guided image filtering [29]. (c) Result of matting followed by bilateral filtering [17]. (d) Ours.

2.4 Robust Latent Image Recovery by Gradient Residual Minimization

After the transmission map is refined, our next goal is to recovery the scene radiance **J**. Many existing methods obtain it by directly solving the linear haze model (2.5), where the artifacts are treated equally as the true pixels. As a result, the artifacts will be also enhanced after dehazing.

Without any prior information, it is impossible to extract or suppress the artifacts from the input image. We have observed that in practice, the visual artifacts are usually invisible in the input image. After dehazing, they pop up as their gradients are amplified, introduce new image edges that are not consistent with the underlying image content, such as the color bands in Fig. 2.1(b,c). Based on this observation, we propose a novel way to constrain the image edges to be structurally consistent before and after dehazing. This motivates us to minimize the residual of the gradients between the input and output images under the sparse-inducing norm. We call it Gradient Residual Minimization (GRM). Combined with the linear haze model, our optimization problem becomes:

$$\min_{\mathbf{J}}\{\frac{1}{2}\int \|\mathbf{J}t - (\mathbf{I} - \mathbf{A} + \mathbf{A}t)\|_2^2 \, \mathrm{d}x + \eta \int \|\nabla \mathbf{J} - \nabla \mathbf{I}\|_0 \, \mathrm{d}x\} \tag{2.12}$$

where the ℓ_0 norm counts the number of non-zero elements and η is a weighting parameter. It is important to note that the above spares-inducing norm only encourages the non-zero gradients of $\mathbf{J}$ to be at the same positions of the gradients of $\mathbf{I}$. However, their magnitudes do not have to be the same. This good property of the edge-preserving term is very crucial in dehazing, as the contrast of the overall image will be increased after dehazing. With the proposed GRM, new edges (often caused by artifacts) that do not exist in the input image will be penalized but the original strong image edges will be kept.

Due to the existence of the artifacts, it is very possible that the linear haze model does not hold on every corrupted pixel. Unlike previous approaches, we assume there may exist some artifacts or large errors $\mathbf{E}$ in the input image, which violates the linear composition model in Eq. (2.1) locally. Furthermore, we assume $\mathbf{E}$ is sparse. This is reasonable as operations such as compression do not damage image content uniformly: they often cause more errors in high-frequency image content than flat regions. With the above assumptions, to recover the latent image, we solve the following optimization problem:

$$\min_{\mathbf{J},\mathbf{E}}\{\frac{1}{2}\int \|\mathbf{J}t - (\mathbf{I} - \mathbf{E} - \mathbf{A} + \mathbf{A}t)\|_2^2 \, \mathrm{d}x + \lambda \int \|\mathbf{E}\|_0 \, \mathrm{d}x + \eta \int \|\nabla \mathbf{J} - \nabla \mathbf{I}\|_0 \, \mathrm{d}x\} \tag{2.13}$$

where λ is a regularization parameter. Intuitively, the first term says that after subtracting $\mathbf{E}$ from the input image $\mathbf{I}$, the remaining component $\mathbf{I} - \mathbf{E}$, together with the latent image $\mathbf{J}$ and the transmission map $\mathbf{A}$, satisfy the haze model in Eq. (2.1). The second term $\mathbf{E}$ represents large artifacts while the last term encodes our observations on image edges.

However, the ℓ_0 minimization problem is generally difficult to solve. Therefore in practice, we replace it with the closest convex relaxation – ℓ_1 norms

[34, 35]:

$$\min_{\mathbf{J},\mathbf{E}}\{\frac{1}{2}\int \|\mathbf{J}t-(\mathbf{I}-\mathbf{E}-\mathbf{A}+\mathbf{A}t)\|_2^2 \,\mathrm{d}x + \lambda\int \|\mathbf{E}\|_1 \,\mathrm{d}x + \eta\int \|\nabla\mathbf{J}-\nabla\mathbf{I}\|_1 \,\mathrm{d}x\} \tag{2.14}$$

We alternately solve this new problem by minimizing the energy function with respect to **J** and **E**, respectively. Let $\mathbf{Z} = \mathbf{J} - \mathbf{I}$, and the **J** subproblem can be rewritten as:

$$\min_{\mathbf{Z}}\{\frac{1}{2}\int \|(\mathbf{Z}+\mathbf{I})t-(\mathbf{I}-\mathbf{E}-\mathbf{A}+\mathbf{A}t)\|_2^2 \,\mathrm{d}x + \eta\int \|\nabla\mathbf{Z}\|_1 \,\mathrm{d}x\} \tag{2.15}$$

which is a TV minimization problem. We can apply an existing TV solver [36] for this subproblem. After **Z** is solved, **J** can be recovered by $\mathbf{J} = \mathbf{Z} + \mathbf{I}$. For the **E** subproblem:

$$\min_{\mathbf{E}}\{\frac{1}{2}\int \|\mathbf{J}t-(\mathbf{I}-\mathbf{E}-\mathbf{A}+\mathbf{A}t)\|_2^2 \,\mathrm{d}x + \lambda\int \|\mathbf{E}\|_1 \,\mathrm{d}x\} \tag{2.16}$$

It has a closed-form solution by soft-thresholding. The overall algorithm for latent image recovery is summarized in Algorithm 2.

Algorithm 2 Robust image dehazing

```
**Initialization:** $E^0 = 0$, $J^0 = \frac{\mathbf{I}-\mathbf{A}}{t} + \mathbf{A}$
**for** $k = 0$ **to** $Maxiteration$ **do**
    $\mathbf{Z}_b = \mathbf{I} - \mathbf{E}^k - \mathbf{A} + \mathbf{A}t - \mathbf{I}t$
    $\mathbf{Z} = \arg\min_{\mathbf{Z}}\{\frac{1}{2}\int \|\mathbf{Z}t - \mathbf{Z}_b\|_2^2 \mathrm{d}x + \eta\int \|\nabla\mathbf{Z}\|_1 \mathrm{d}x\}$
    $\mathbf{J}^{k+1} = \mathbf{I} + \mathbf{Z}$
    $\mathbf{E}^{k+1} = thresholding_\lambda(\mathbf{I} - \mathbf{J}^{k+1}t - (1-t)\mathbf{A})$
**end for**
```

The convergence of Algorithm 2 is shown in Fig. 2.3. We initialize **J** with the least squares solution without GRM and a zero image **E**. As we can see, the object function in Eq. (2.14) decreased monotonically and our method gradually converged. From the intermediate results, it can be observed that the initial **J** has visible artifacts in the sky region, which is gradually eliminated during the optimization. One may notice that **E** converged to large values on the tower and building edges. These are the aliasing artifacts caused by compression, and our method can successfully separate out these artifacts.

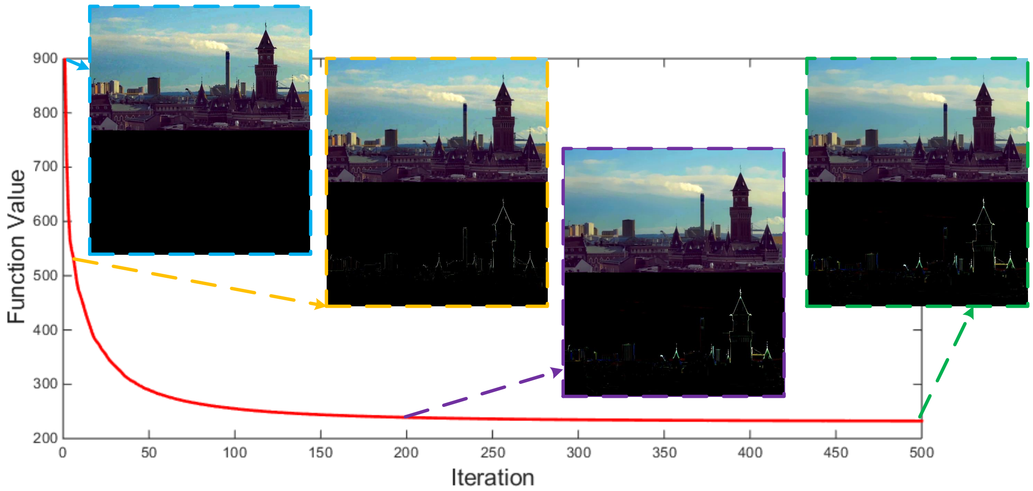


Figure 2.3: The convergence of proposed method. The object function in Eq. (2.14) is monotonically decreasing. The intermediate results of $\mathbf{J}$ and $10\times\mathbf{E}$ at iterations 1, 5, 200 and 500 are shown.

2.5 Experiments

For quality comparisons, all the images should be viewed on-screen instead of in the printed version.

2.5.1 Implementation details

In our implementation, the tensor parameters are set as $\beta = 9$, $\gamma = 0.85$. The regularization parameters are $\alpha_0 = 0.5$, $\alpha_1 = 0.05$, $\lambda = 0.01$ and $\eta = 0.1$. We found our method is not sensitive to these parameters. The same set of parameters are used for all experiments in this chapter. We terminate Algorithm 1 after 300 iterations and Algorithm 2 after 200 iterations.

We use the same method in the He et al. approach to estimate the atmospheric light $\mathbf{A}$. For video inputs, we simply use the $\mathbf{A}$ computed from the first frame for all other frames. We found that fixing $\mathbf{A}$ for all frames is generally sufficient to get temporally coherent results by our model.

Using our MATLAB implementation on a laptop computer with a i7-4800 CPU and 16GB RAM, it takes around 20 seconds to dehaze a 480×270 image. In comparison, 10 minutes per frame is reported in [25] on the same video frames. As many previous works [19], we apply a global gamma correction on images that become too dark after dehazing, just for better displaying.

2.5.2 Evaluation on synthetic data

We first quantitatively evaluate the performance of the proposed transmission estimation method using a synthetic dataset. Similar to previous practices [20], we synthesize hazy images from stereo pairs [37, 38] with known disparity maps. The transmission maps are simulated in the same way as in [20]. Since our method is tailored toward suppressing artifacts, we prepare two test sets: one with high-quality input images, the other with noise and compression corrupted images. To synthesize corruption, we first add 1% of Gaussian noise to the hazy images. These images are then compressed using the JPEG codec in Photoshop, with the compression quality 8 out of 12.

In Tables 2.1 and 2.2 we show the MSE of the haze map and the recovered image by different methods, on the clean and the corrupted datasets, respectively. The results show that our method achieves a more accurate haze map and latent image than previous methods in most cases. One may find that the errors for corrupted inputs sometimes are lower than those of noise-free ones. It is because the dark channel based methods underestimated the transmission on these bright indoor scenes. The transmission may be slightly more precise when noise makes the images more colorful. Comparing the results of the two tables, the improvement by our method is more significant on the second set, which demonstrates its ability to suppress artifacts.

Table 2.1: Quantitative comparisons on the clean synthetic dataset. The table reports the MSE (10^{-3}) of the transmission map (left) and the output image (right).

	Aloe	Barn	Cones	Dolls
He et al. [17]	5.6/17.4	0.9/**7.9**	8.6/13.7	8.0/14.3
Li et al. [10]	4.5/13.2	1.6/13.9	5.8/8.9	4.1 /7.0
Ours	**4.4/10.4**	**0.8**/9.0	**5.6/8.0**	**3.8 /6.8**
	Moebius	Monopoly	Teddy	Rocks
He et al. [17]	7.5/18.6	11.2/30.1	10.3/20.1	4.6/11.4
Li et al. [10]	5.7/**12.7**	8.9/23.8	**4.0/6.6**	3.7/9.5
Ours	**4.4/10.4**	**7.9/20.7**	6.6/10.7	**3.2/8.0**

Table 2.2: Quantitative comparison on the noise and compression corrupted synthetic dataset. The table reports the MSE (10^{-3}) of the transmission map (left) and the output image (right).

	Aloe	Barn	Cones	Dolls
He et al. [17]	5.3/17.0	**1.0/11.2**	8.1/12.9	7.6/13.8
Li et al. [10]	4.4/13.2	1.5/14.2	5.6/8.9	4.0/7.1
Ours	**3.9/9.9**	**1.0**/12.8	**5.2/7.1**	**3.5/6.2**
	Moebius	Monopoly	Teddy	Rocks
He et al. [17]	7.2/17.4	10.7/27.9	10.0/19.3	4.1/10.3
Li et al. [10]	5.7/12.7	8.8/22.6	**4.0/6.7**	3.6/9.3
Ours	**3.9/9.9**	**7.5/18.6**	6.5/10.1	**2.8/6.8**

2.5.3 Real-world images and videos

We compare our method with some recent works [18, 10, 23] on a real video frame in Fig. 2.4. The compression artifacts and image noise become severe after dehazing by the Meng et al. method and the Dehaze feature in Adobe Photoshop. The Galdran et al. result suffers from large color distortion. He et al. have pointed out the similar phenomenon of Tan et al. method [22], which is also based on contrast enhancement. The Li et al. method [10] is designed for blocking artifact suppression. Although their result does not contain such artifacts, the sky region is quite over-smoothed. Our result maintains subtle image features while at the same time successfully avoids boosting these artifacts.

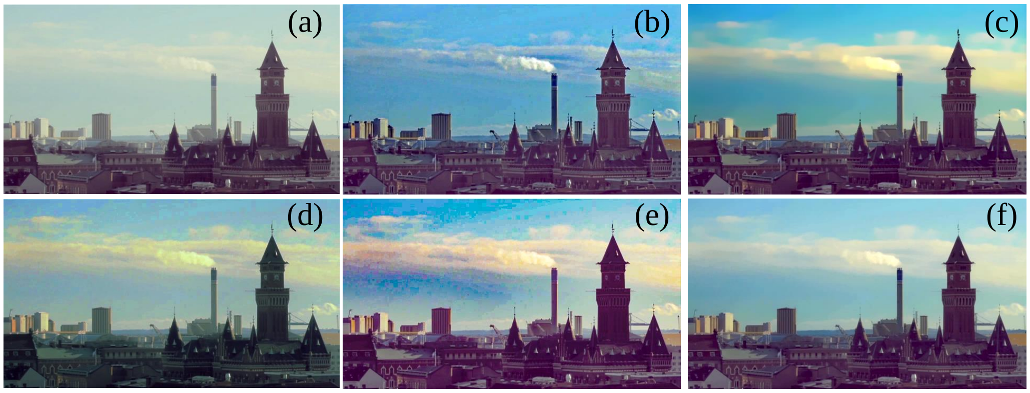


Figure 2.4: Dehazing results of different methods. (a) Input image. (b) Meng et al. result [18]. (c) Li et al. result [10]. (d) Galdran et al. result [23]. (e) Photoshop 2015 dehazing result. (f) Our result.

Our method can especially suppress halo and color aliasing artifacts around

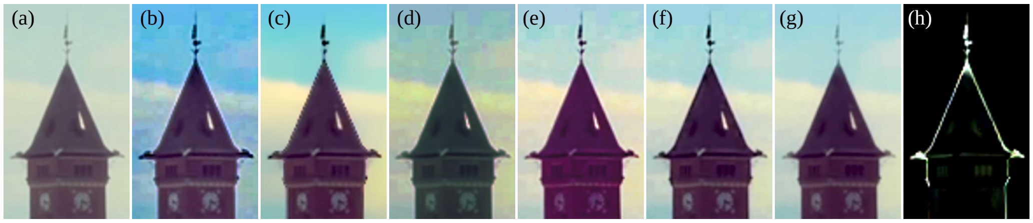


Figure 2.5: Zoomed-in region of Fig. 2.4. (a) Input image. (b) Meng et al. result [18]. (c) Li et al. result [10]. (d) Galdran et al. result [23]. (e) Photoshop 2015 result. (f) Our result without the proposed GRM. (g) Our result. (h) Our $\mathbf{E} \times 10$.

depth edges that are common for previous methods, as shown in the zoomed-in region of the tower in Fig. 2.5. Except the result by our method, all other methods produce severe halo and color aliasing artifacts around the sharp tower boundary. Pay special attention to the flag on the top of the tower: the flag is dilated by all other methods except ours. Figure 2.5(h) visualizes the artifact map $\mathbf{E}$ in Eq. (2.14), it suggests that our image recovery method pays special attention to the boundary pixels to avoid introducing aliasing by dehazing. We also include our result without the proposed GRM in Fig. 2.5 (f). The blocky artifacts and color aliasing around the tower boundary cannot be reduced on this result, which demonstrates the effectiveness of the proposed model.

In Fig. 2.6, we compare our method with two variational methods [25, 23] proposed recently on a video frame. The Galdran et al. method [23] converged in a few iterations on this image, but the result still contains haze. The method in [25] performs simultaneously dehazing and stereo reconstruction, thus it only works when structure-from-motion can be calculated. For general videos contain dynamic scenes or a single image, it cannot be applied. From the results, our method is comparable to that in [25], or even better. For example, our method can remove more haze on the building. This is clearer on Li's depth map, where the shape of the building can hardly be found.

We further compare our method with the deblocking based method [10] on more video sequences in Fig. 2.7. The Li et al. method generates various artifacts in these examples, such as the over-sharpened and over-saturated sea region in the first example, the color distortion in the sky regions of the second, and the halos around the buildings and the color banding in the third

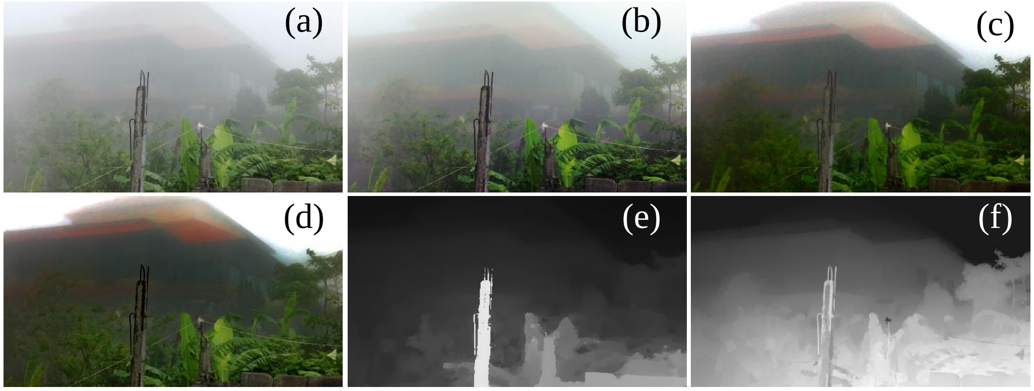


Figure 2.6: Comparison with some recent methods. (a) Input video frame. (b) Galdran et al. result [23]. (c) Li et al. result [25]. (d) Our result. (e) Li's depth [25] (computed using the whole video). (f) Our transmission map.

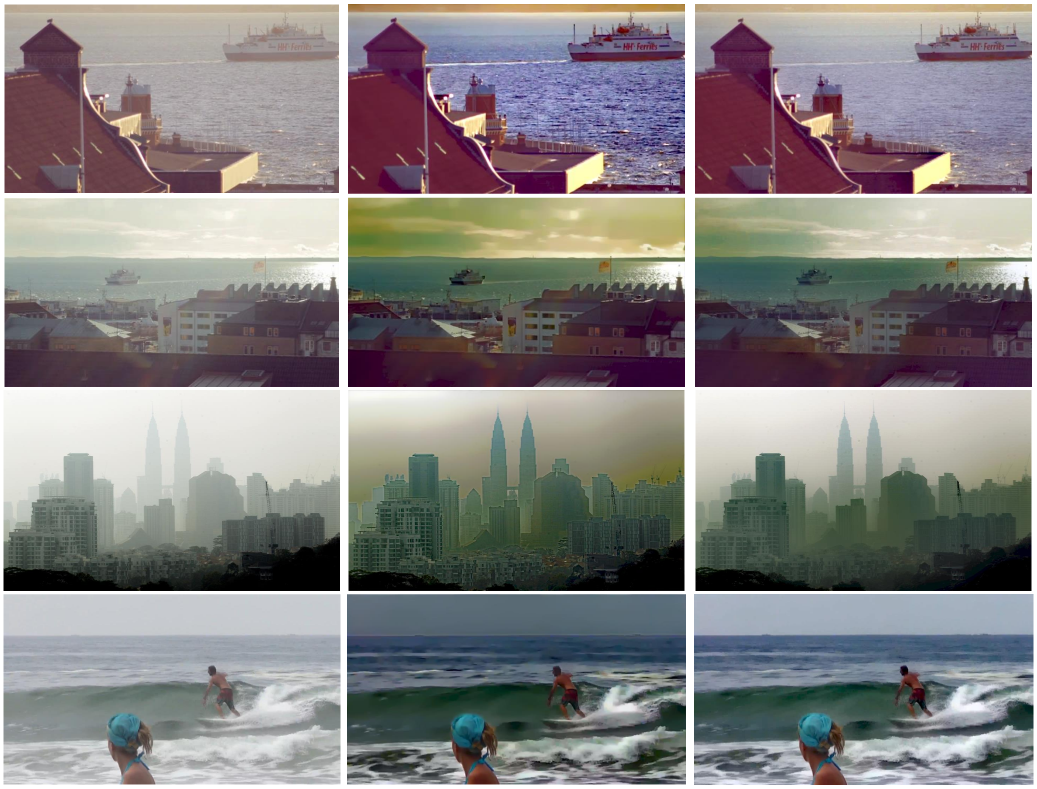


Figure 2.7: Comparison with Li et al. method. First column: input video frame. Second column: Li et al. result [10]. Third column: Our result.

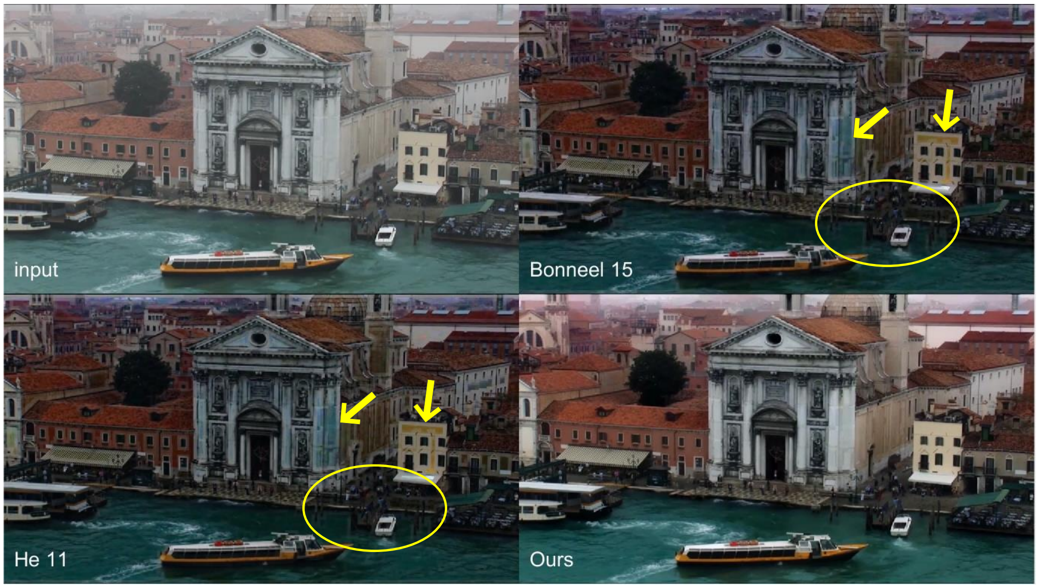


Figure 2.8: A frame of video dehazing results. The full video is in the supplementary material. The halos around the pillars and structured artifacts are indicated by the yellow circle and arrows.

example. In the bottom example, there is a strong halo near the intersection of the sky and sea. Another drawback of the Li et al. method is that fine image details are often lost, such as the sea region in the last example. In contrast, our results contain many fewer visual artifacts and appear to be more natural.

For videos, the flickering artifacts widely exist on the previous frame-by-frame dehazing methods. It is often caused by the artifacts and the change of overall color in the input video. Recently, Bonneel et al. proposed a new method to remove the flickering by enforcing temporal consistency using optical flow [39]. Although their method can successfully remove the temporal artifacts, it does not work for the spatial artifacts on each frame. Figure 2.8 shows one example frame of a video, where their result inherits all the structured artifacts from the existing method. Although we only perform frame-by-frame dehazing, the result shows that our method is able to suppress temporal artifacts as well. This is because the input frames already have good temporal consistency. Such temporal consistency is transferred into our result frame-by-frame by the proposed GRM.

We recruited 34 volunteers through the Adobe mail list for a user study of result quality, which contained researches, interns, managers, photographers etc. For each example, we presented three different results anonymously (always including ours) in random orders, and asked them to pick the best dehazing result, based on realism, dehazing quality, artifacts etc. 52.9% subjects preferred our “bali” result in Fig. 2.6, 47.1% preferred the result in [25] and 0% preferred the result of He et al. [17]. We have mentioned above that [25] requires external structure-from-motion information, while ours does not and can be applied to more general dehazing. For Fig. 2.8, 91.2% preferred our results over those of He et al. [17] and Bonneel et al. [39]. For the remaining of examples in this chapter, our results were the preferred ones also (by 73.5%-91.2% people), where overall 80.0% picked our results over Li et al. [10] (14.7%) and He et al. [17] (5.3%).

2.5.4 Discussion

One may argue there are simpler alternatives to handle artifacts in the dehazing pipeline. One way is to explicitly remove the image artifacts before dehazing, such as the Li et al. method. However, accurately removing all image artifacts itself is a difficult task. If not done perfectly, the quality of the final image will be compromised, as shown in various examples in this chapter. Another alternative is to simply reduce the amount of haze to be removed. However, it will significantly decrease the power of dehazing. Our method is a more principled way to achieve a good balance between dehazing and minimizing visual artifacts.

Despite its effectiveness, our method still has some limitations. Firstly, our method inherits the limitations of the dark channel prior. It may overestimate the amount of haze for white objects that are close to the camera. In addition, for very faraway objects, our method can not significantly increase their contrast, which is due to the ambiguity between the artifacts and true objects covered by very thick haze. It is even difficult for human eyes to distinguish them without image context. Previous methods also have poor performance on such challenging tasks: they either directly amplify all the artifacts or mistakenly remove the distant objects to produce over-smoothed results.

Figure 2.9 shows one such example that contains some faraway buildings surrounded by JPEG artifacts. Both the Fattal et al. and He et al. results

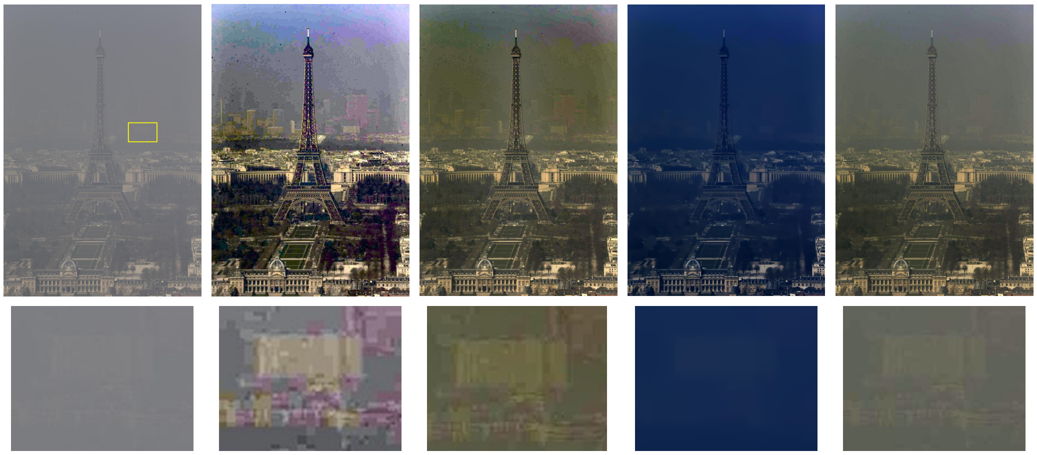


Figure 2.9: Dehazing a low quality JPEG image. From left to right are the input image and the results by: Fattal et al. [19], He et al. [17], Li et al. [10] and ours. The bottom row shows the zoomed-in areas corresponding to the yellow box.

have observable JPEG artifacts after dehazing. On the contrary, in the Li et al. result, the distant buildings are mistakenly removed by their deblocking filter, and become much less visible. Although our method cannot solve the ambiguity mentioned above to greatly enhance the faraway buildings, it can automatically take care of the artifacts and generate a more realistic result.

2.6 Conclusion

We have proposed a new method to suppress visual artifacts in image and video dehazing. By introducing a gradient residual and error layer into the image recovery process, our method is able to remove various artifacts without explicitly modeling each one. A new transmission refinement method is introduced in this work, which contributes to improving the overall accuracy of our results. We have conducted extensive evaluation on both synthetic datasets and real-world examples, and validated the superior performance of our method over the state-of-the-arts for lower-quality inputs. While our method works well on the dehazing task, it can be potentially extended to other image enhancement applications, due to the similar artifacts-amplification nature of them.

CHAPTER 3

IMAGE SYNTHESIS FOR CORRUPTED IMAGES

3.1 Introduction

[1]Semantic inpainting [40] refers to the task of inferring arbitrary large missing regions in images based on image semantics. Since prediction of high-level context is required, this task is significantly more difficult than classical inpainting or image completion which is often more concerned with correcting spurious data corruption or removing entire objects. Numerous applications such as restoration of damaged paintings or image editing [41] benefit from accurate semantic inpainting methods if large regions are missing. However, inpainting becomes increasingly more difficult if large regions are missing or if scenes are complex.

Classical inpainting methods are often based on either local or non-local information to recover the image. Most existing methods are designed for single image inpainting. Hence they are based on the information available in the input image, and exploit image priors to address the ill-posed-ness. For example, total variation (TV) based approaches [42, 43] take into account the smoothness property of natural images, which is useful to fill small missing regions or remove spurious noise. Holes in textured images can be filled by finding a similar texture from the same image [44]. Prior knowledge, such as statistics of patch offsets [45], planarity [46] or low rank (LR) [47] can greatly improve the result as well. PatchMatch (PM) [48] searches for similar patches in the available part of the image and quickly became one of the most successful inpainting methods due to its high quality and efficiency. However, all single image inpainting methods require appropriate information to be contained in the input image, e.g., similar pixels, structures, or patches.

[1] Chapter 3 has been published in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* 2017 [11]. Copyright IEEE.

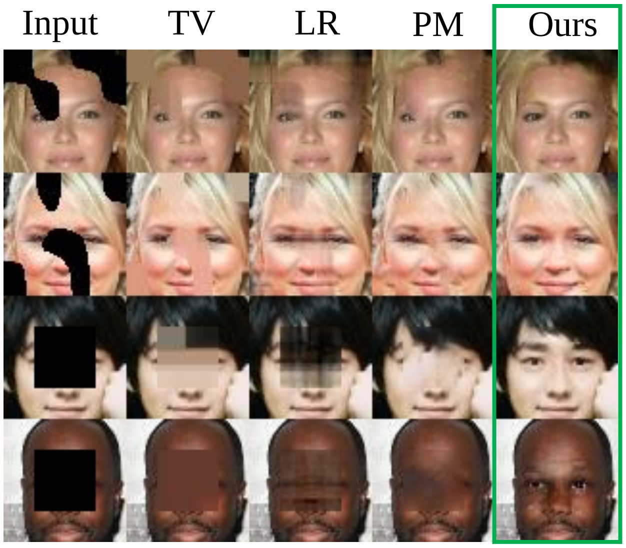


Figure 3.1: Semantic inpainting results by TV, LR, PM and our method. Holes are marked by black color.

This assumption is hard to satisfy, if the missing region is large and possibly of arbitrary shape. Consequently, in this case, these methods are unable to recover the missing information. Figure 3.1 shows some challenging examples with large missing regions, where local methods fail to recover the nose and eyes.

In order to address inpainting in the case of large missing regions, non-local methods try to predict the missing pixels using external data. Hays and Efros [49] proposed to cut and paste a semantically similar patch from a huge database. Internet-based retrieval can be used to replace a target region of a scene [50]. Both methods require exact matching from the database or Internet, and fail easily when the test scene is significantly different from any database image. Unlike previous hand-crafted matching and editing, learning-based methods have shown promising results [51, 52, 53, 54]. After an image dictionary or a neural network is learned, the training set is no longer required for inference. Often, these learning-based methods are designed for small holes or for removing small text in the image.

Instead of filling small holes in the image, we are interested in the more difficult task of semantic inpainting [40]. It aims to predict the detailed content of a large region based on the context of surrounding pixels. A seminal approach for semantic inpainting, and closest to our work is the Context Encoder (CE) by Pathak et al. [40]. Given a mask indicating missing regions, a neural network is trained to encode the context information and predict the unavailable content. However, the CE only takes advantage of the structure of holes during training but not during inference. Hence it results in blurry or unrealistic images especially when missing regions have arbitrary shapes.

In this chapter, we propose a novel method for semantic image inpainting. We consider semantic inpainting as a constrained image generation problem and take advantage of the recent advances in generative modeling. After a deep generative model, i.e., in our case an adversarial network [55, 56], is trained, we search for an encoding of the corrupted image that is "closest" to the image in the latent space. The encoding is then used to reconstruct the image using the generator. We define "closest" by a weighted context loss to condition on the corrupted image, and a prior loss to penalizes unrealistic images. Compared to the CE, one of the major advantages of our method is that it does not require the masks for training and can be applied for arbitrarily structured missing regions during inference. We evaluate our method on three datasets: CelebA [57], SVHN [58] and Stanford Cars [59], with different forms of missing regions. Results demonstrate that on challenging semantic inpainting tasks our method can obtain much more realistic images than the state-of-the-art techniques.

3.2 Related Work

A large body of literature exists for image inpainting, and due to space limitations we are unable to discuss all of it in detail. Seminal work in that direction includes the aforementioned works and references therein. Since our method is based on generative models and deep neural nets, we will review the technically related learning-based work in the following.

Generative Adversarial Networks (GANs) are a framework for training generative parametric models, and have been shown to produce high-quality images [55, 60, 56]. This framework trains two networks, a generator, G, and a discriminator D. G maps a random vector $\mathbf{z}$, sampled from a prior distribution $p_{\mathbf{Z}}$, to the image space while D maps an input image to a likelihood. The purpose of G is to generate realistic images, while D plays an adversarial role, discriminating between the image generated from G, and the real image

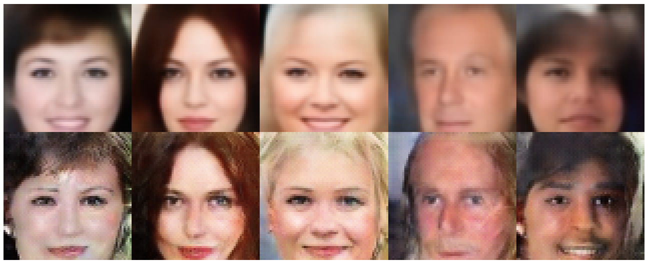


Figure 3.2: Images generated by a VAE and a DCGAN. First row: samples from a VAE. Second row: samples from a DCGAN.

sampled from the data distribution p_{data}.

The G and D networks are trained by optimizing the loss function:

$$\begin{aligned}\min_G \max_D V(G, D) =& \mathbb{E}_{\mathbf{h} \sim p_{data}(\mathbf{h})}[\log(D(\mathbf{h}))] + \\ & \mathbb{E}_{\mathbf{z} \sim p_{\mathbf{Z}}(\mathbf{z})}[\log(1 - D(G(\mathbf{z}))] \end{aligned} \tag{3.1}$$

where $\mathbf{h}$ is the sample from the p_{data} distribution; $\mathbf{z}$ is a random encoding on the latent space.

With some user interaction, GANs have been applied in interactive image editing [61]. However, GANs cannot be directly applied to the inpainting task, because they produce an entirely unrelated image with high probability, unless constrained by the provided corrupted image.

Autoencoders and Variational Autoencoders (VAEs) [62] have become a popular approach to learning of complex distributions in an unsupervised setting. A variety of VAE flavors exist, e.g., extensions to attribute-based image editing tasks [63]. Compared to GANs, VAEs tend to generate overly smooth images, which is not preferred for inpainting tasks. Figure 3.2 shows some examples generated by a VAE and a deep convolutional GAN (DC-GAN) [56]. Note that the DCGAN generates much sharper images. Jointly training VAEs with an adversarial loss prevents the smoothness [64], but may lead to artifacts.

The Context Encoder (CE) [40] can be also viewed as an autoencoder conditioned on the corrupted images. It produces impressive reconstruction results when the structure of holes is fixed during both training and inference, e.g., fixed in the center, but is less effective for arbitrarily structured regions.

3.2.1 Backpropagation to the input

Backpropagation to the input data is employed in our approach to find the encoding which is close to the provided but corrupted image. In earlier work, back propagation to augment data has been used for texture synthesis and style transfer [65, 66, 67]. Google's DeepDream uses back-propagation to create dreamlike images [68]. Additionally, backpropagation has also been used to visualize and understand the learned features in a trained network, by "inverting" the network through updating the gradient at the input layer [69, 70, 71, 72]. Similar to our method, all these backpropagation-based methods require specifically designed loss functions for the particular tasks.

3.3 Semantic Inpainting by Constrained Image Generation

To fill large missing regions in images, our method for image inpainting utilizes the generator G and the discriminator D, both of which are trained with uncorrupted data. After training, the generator G is able to take a point $\mathbf{z}$ drawn from $p_{\mathbf{Z}}$ and generate an image mimicking samples from p_{data}. We hypothesize that if G is efficient in its representation then an image that is not from p_{data} (e.g., corrupted data) should not lie on the learned encoding manifold, $\mathbf{z}$. Therefore, we aim to recover the encoding $\hat{\mathbf{z}}$ "closest" to the corrupted image while being constrained to the manifold, as illustrated in Fig. 3.3; we visualize the latent manifold, using t-SNE [73] on the two-dimensional space, and the intermediate results in the optimization steps of finding $\hat{\mathbf{z}}$. After $\hat{\mathbf{z}}$ is obtained, we can generate the missing content by using the trained generative model G.

More specifically, we formulate the process of finding $\hat{\mathbf{z}}$ as an optimization problem. Let $\mathbf{y}$ be the corrupted image and $\mathbf{M}$ be the binary mask with size equal to the image, to indicate the missing parts. An example of $\mathbf{y}$ and $\mathbf{M}$ is shown in Fig. 3.3 (a).

Using this notation we define the "closest" encoding $\hat{\mathbf{z}}$ via:

$$\hat{\mathbf{z}} = \arg\min_{\mathbf{z}} \{\mathcal{L}_c(\mathbf{z}|\mathbf{y}, \mathbf{M}) + \mathcal{L}_p(\mathbf{z})\} \tag{3.2}$$

where $\mathcal{L}_c$ denotes the context loss, which constrains the generated image

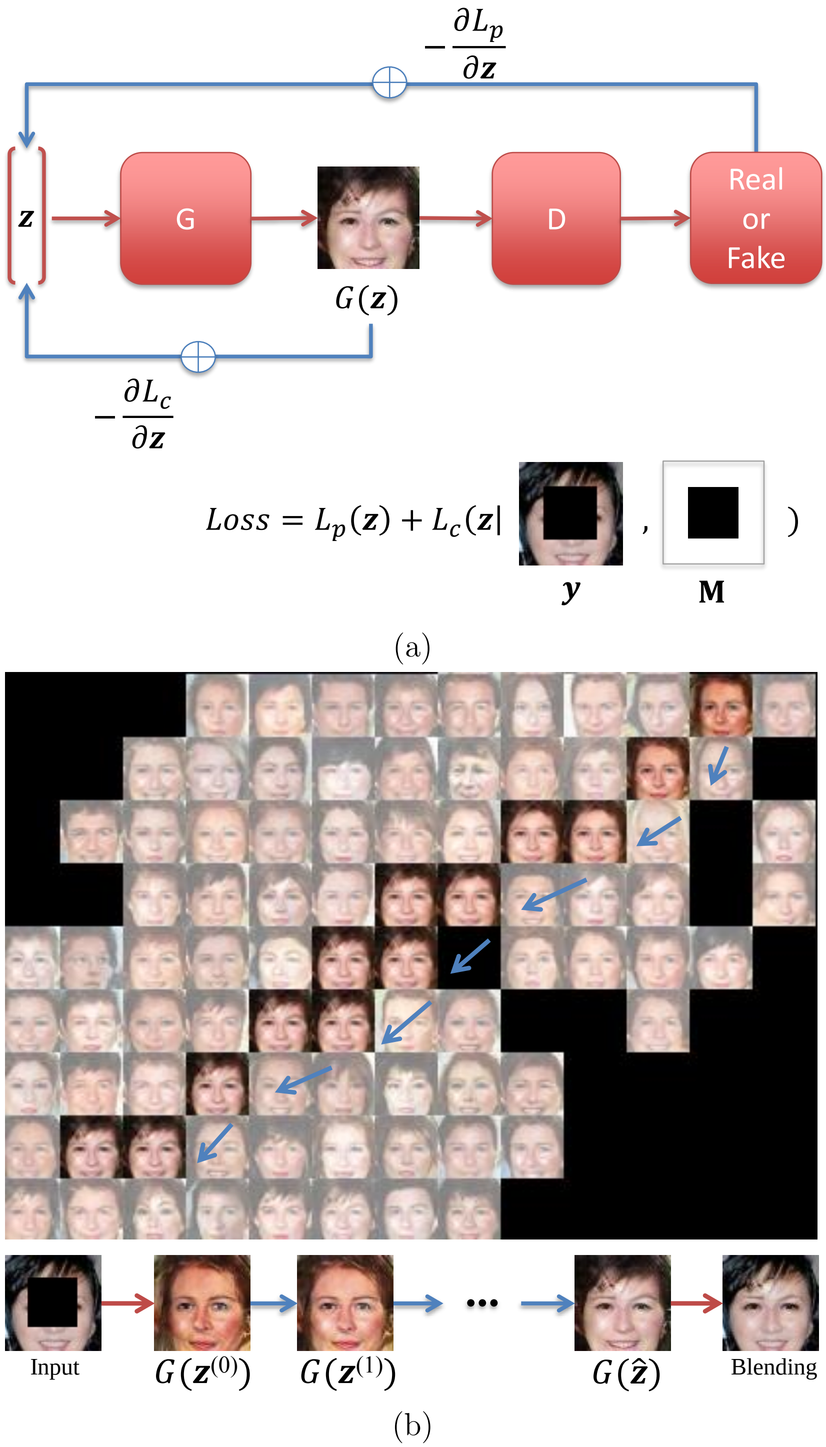


Figure 3.3: The proposed framework for inpainting. (a) Given a GAN model trained on real images, we iteratively update $\mathbf{z}$ to find the closest mapping on the latent image manifold, based on the designed loss functions. (b) Manifold traversing when iteratively updating $\mathbf{z}$ using back propagation. $\mathbf{z}^{(0)}$ is random initialed; $\mathbf{z}^{(k)}$ denotes the result in k-th iteration; and $\hat{\mathbf{z}}$ denotes the final solution.

given the input corrupted image $\mathbf{y}$ and the hole mask $\mathbf{M}$; and $\mathcal{L}_p$ denotes the prior loss, which penalizes unrealistic images. The details of the proposed loss function will be discussed in the following sections.

Besides the proposed method, one may also consider using D to update $\mathbf{y}$ by maximizing $D(\mathbf{y})$, similar to backpropagation in DeepDream [68] or neural style transfer [65]. However, the corrupted data $\mathbf{y}$ is neither drawn from a real image distribution nor the generated image distribution. Therefore, maximizing $D(\mathbf{y})$ may lead to a solution that is far away from the latent image manifold, which may hence lead to results with poor quality.

3.3.1 Importance weighted context loss

To fill large missing regions, our method takes advantage of the remaining available data. We designed the context loss to capture such information. A convenient choice for the context loss is simply the ℓ_2 norm between the generated sample $G(\mathbf{z})$ and the uncorrupted portion of the input image $\mathbf{y}$. However, such a loss treats each pixel equally, which is not desired. Consider the case where the center block is missing: a large portion of the loss will be from pixel locations that are far away from the hole, such as the background behind the face. Therefore, in order to find the correct encoding, we should pay significantly more attention to the missing region that is close to the hole.

To achieve this goal, we propose a context loss with the hypothesis that the importance of an uncorrupted pixel is positively correlated with the number of corrupted pixels surrounding it. A pixel that is very far away from any holes plays a very small role in the inpainting process. We capture this intuition with the importance weighting term, $\mathbf{W}$,

$$\mathbf{W}_i = \begin{cases} \sum\limits_{j \in N(i)} \frac{(1-\mathbf{M}_j)}{|N(i)|} & \text{if } \ \mathbf{M}_i \neq 0 \\ 0 & \text{if } \ \mathbf{M}_i = 0 \end{cases} \tag{3.3}$$

where i is the pixel index, $\mathbf{W}_i$ denotes the importance weight at pixel location i, $N(i)$ refers to the set of neighbors of pixel i in a local window, and $|N(i)|$ denotes the cardinality of $N(i)$. We use a window size of 7 in all experiments.

Empirically, we also found the ℓ_1-norm to perform slightly better than the ℓ_2-norm in our framework. Taking it all together, we define the contextual loss to be a weighted ℓ_1-norm difference between the recovered image and the uncorrupted portion, defined as follows,

$$\mathcal{L}_c(\mathbf{z}|\mathbf{y}, \mathbf{M}) = \|\mathbf{W} \odot (G(\mathbf{z}) - \mathbf{y})\|_1 \tag{3.4}$$

Here, $\odot$ denotes the element-wise multiplication.

3.3.2 Prior loss

The prior loss refers to a class of penalties based on high-level image feature representations instead of pixel-wise differences. In this work, the prior loss encourages the recovered image to be similar to the samples drawn from the training set. Our prior loss is different from the one defined in [74] which uses features from pre-trained neural networks.

Our prior loss penalizes unrealistic images. Recall that in GANs, the discriminator, D, is trained to differentiate generated images from real images. Therefore, we choose the prior loss to be identical to the GAN loss for training the discriminator D, i.e.,

$$\mathcal{L}_p(\mathbf{z}) = \lambda \log(1 - D(G(\mathbf{z}))) \tag{3.5}$$

Here, λ is a parameter to balance between the two losses, and $\mathbf{z}$ is updated to fool D and make the corresponding generated image more realistic. Without $\mathcal{L}_p$, the mapping from $\mathbf{y}$ to $\mathbf{z}$ may converge to a perceptually implausible result. We illustrate this by showing the unstable examples where we optimized with and without $\mathcal{L}_p$ in Fig. 3.4.

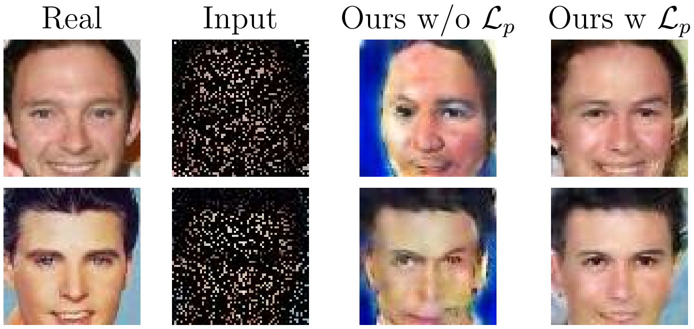


Figure 3.4: Inpainting with and without the prior loss.

3.3.3 Inpainting

With the defined prior and context losses at hand, the corrupted image can be mapped to the closest $\mathbf{z}$ in the latent representation space, which we denote $\hat{\mathbf{z}}$. The $\mathbf{z}$ is randomly initialized and updated using backpropagation on the total loss given in Eq. (3.2). Figure 3.3 (b) shows for one example that $\mathbf{z}$ is approaching the desired solution on the latent image manifold.

After generating $G(\hat{\mathbf{z}})$, the inpainting result can be easily obtained by overlaying the uncorrupted pixels from the input. However, we found that the predicted pixels may not exactly preserve the same intensities of the surrounding pixels, although the content is correct and well aligned. Poisson blending [75] is used to reconstruct our final results. The key idea is to keep the gradients of $G(\hat{\mathbf{z}})$ to preserve image details while shifting the color to match the color in the input image $\mathbf{y}$. Our final solution, $\hat{\mathbf{x}}$, can be obtained by:

$$\begin{aligned} \hat{\mathbf{x}} &= \arg\min_{\mathbf{x}} \|\nabla \mathbf{x} - \nabla G(\hat{\mathbf{z}})\|_2^2 \\ &\text{s.t. } \mathbf{x}_i = \mathbf{y}_i \text{ for } \mathbf{M}_i = 1 \end{aligned} \tag{3.6}$$

where ∇ is the gradient operator. The minimization problem contains a quadratic term, which has a unique solution [75]. Figure 3.5 shows two examples where we can find visible seams without blending.

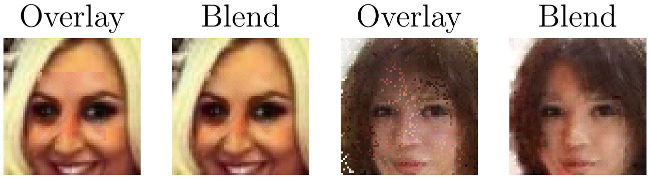


Figure 3.5: Inpainting with and without blending.

3.3.4 Implementation details

In general, our contribution is orthogonal to specific GAN architectures and our method can take advantage of any generative model G. We used the DCGAN model architecture from Radford et al. [56] in the experiments. The generative model, G, takes a random 100-dimensional vector drawn from a uniform distribution between $[-1, 1]$ and generates a $64 \times 64 \times 3$ image. The discriminator model, D, is structured essentially in reverse order. The input layer is an image of dimension $64 \times 64 \times 3$, followed by a series of convolution layers where the image dimension is half, and the number of channels is double the size of the previous layer, and the output layer is a two-class softmax.

For training the DCGAN model, we follow the training procedure in [56] and use Adam [76] for optimization. We choose $\lambda = 0.003$ in all our experiments. We also perform data augmentation of random horizontal flipping on the training images. In the inpainting stage, we need to find $\hat{\mathbf{z}}$ in the latent space using back-propagation. We use Adam for optimization and restrict $\mathbf{z}$ to $[-1, 1]$ in each iteration, which we observe to produce more stable results. We terminate the backpropagation after 1500 iterations. We use the identical setting for all testing datasets and masks.

3.4 Experiments

3.4.1 Datasets and masks

We evaluate our method on three dataset: the CelebFaces Attributes Dataset (CelebA) [57], the Street View House Numbers (SVHN) [58] and the Stanford Cars Dataset [59].

The CelebA contains $202,599$ face images with coarse alignment [57]. We remove approximately 2000 images from the dataset for testing. The images are cropped at the center to 64×64, which contain faces with various viewpoints and expressions.

The SVHN dataset contains a total of 99,289 RGB images of cropped house numbers. The images are resized to 64×64 to fit the DCGAN model architecture. We used the provided training and testing split. The numbers in the images are not aligned and have different backgrounds.

The Stanford Cars dataset contains 16,185 images of 196 classes of cars. Similar as the CelebA dataset, we do not use any attributes or labels for both training and testing. The cars are cropped based on the provided bounding boxes and resized to 64×64. As before, we use the provided training and test set partition.

We test four different shapes of masks: (1) central block masks; (2) random pattern masks [40] in Fig. 3.1, with approximately 25% missing; (3) 80% missing complete random masks; (4) half-missing masks (randomly horizontal or vertical).

3.4.2 Visual comparisons

Comparisons with TV and LR inpainting. We compare our method with local inpainting methods. As we already showed in Fig. 3.1, local methods generally fail for large missing regions. We compare our method with TV inpainting [43] and LR inpainting [77, 47] on images with small random holes. The test images and results are shown in Fig. 3.6. Due to a large number of missing points, TV and LR-based methods cannot recover enough image details, resulting in very blurry and noisy images. PM [48] cannot be applied to this case due to insufficient available patches.

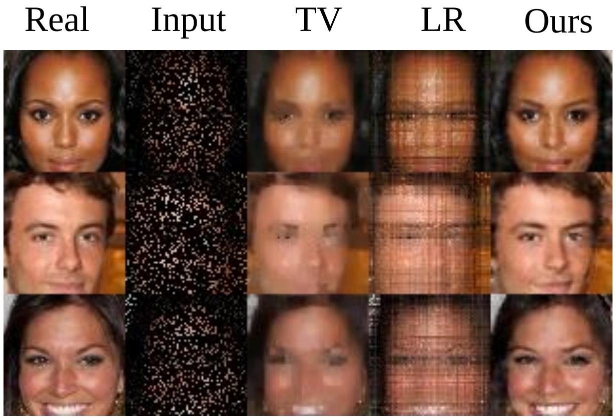


Figure 3.6: Comparisons with local inpainting methods TV and LR inpainting on examples with random 80% missing.

Comparisons with NN inpainting. Next we compare our method with nearest neighbor (NN) filling from the training dataset, which is a key component in retrieval-based methods [49, 50]. Examples are shown in Fig. 3.7, where the misalignment of skin texture, eyebrows, eyes and hair can be clearly observed by using the nearest patches in Euclidean distance. Although people can use different features for retrieval, the inherit misalignment problem cannot be easily solved [40]. Instead, our results are obtained automatically without any registration.

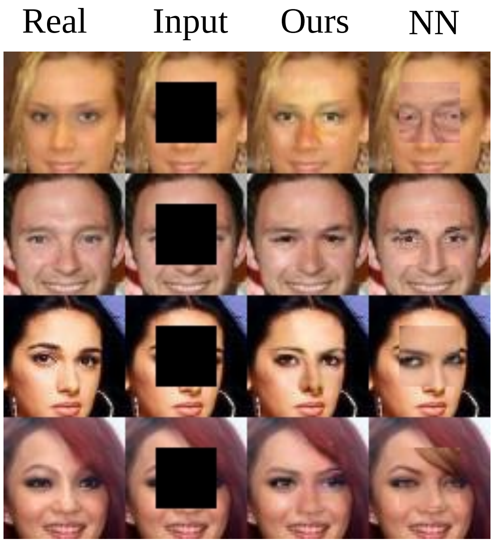


Figure 3.7: Comparisons with nearest patch retrieval.

Comparisons with CE. In the remainder, we compare our result with those obtained from the CE [40], the state-of-the-art method for semantic inpainting. It is important to note that the masks are required to train the CE. For a fair comparison, we use all the test masks in the training phase for the CE. However, there are infinite shapes and missing ratios for the inpainting task. To achieve satisfactory results one may need to re-train the CE. In contrast, our method can be applied to arbitrary masks without re-training the network, which is according to our opinion a huge advantage when considering inpainting applications.

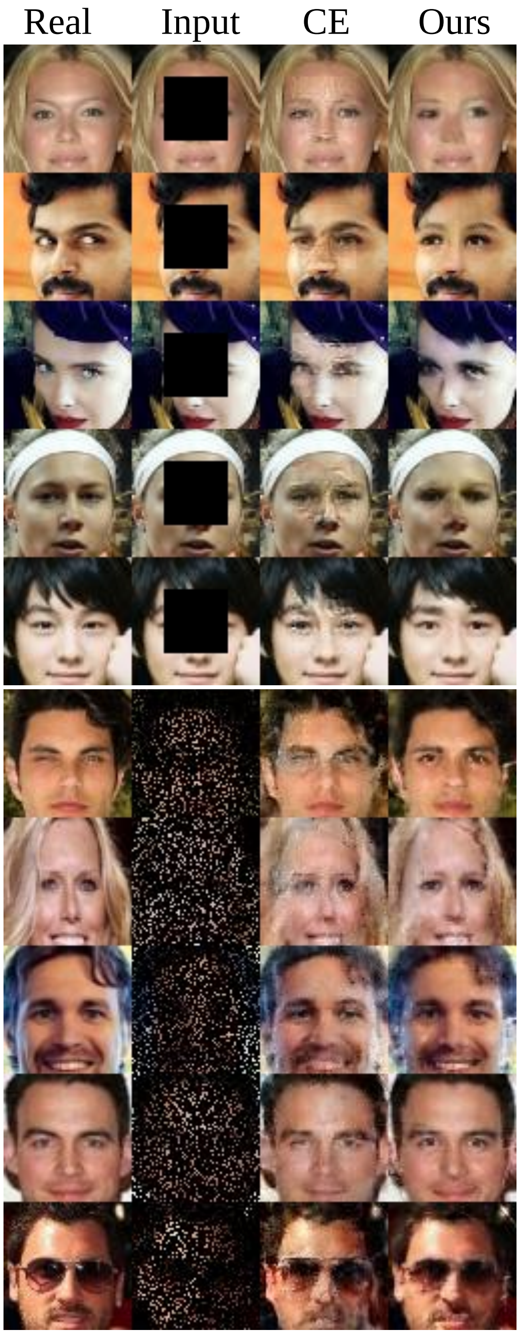


Figure 3.8: Comparisons with CE on the CelebA dataset.

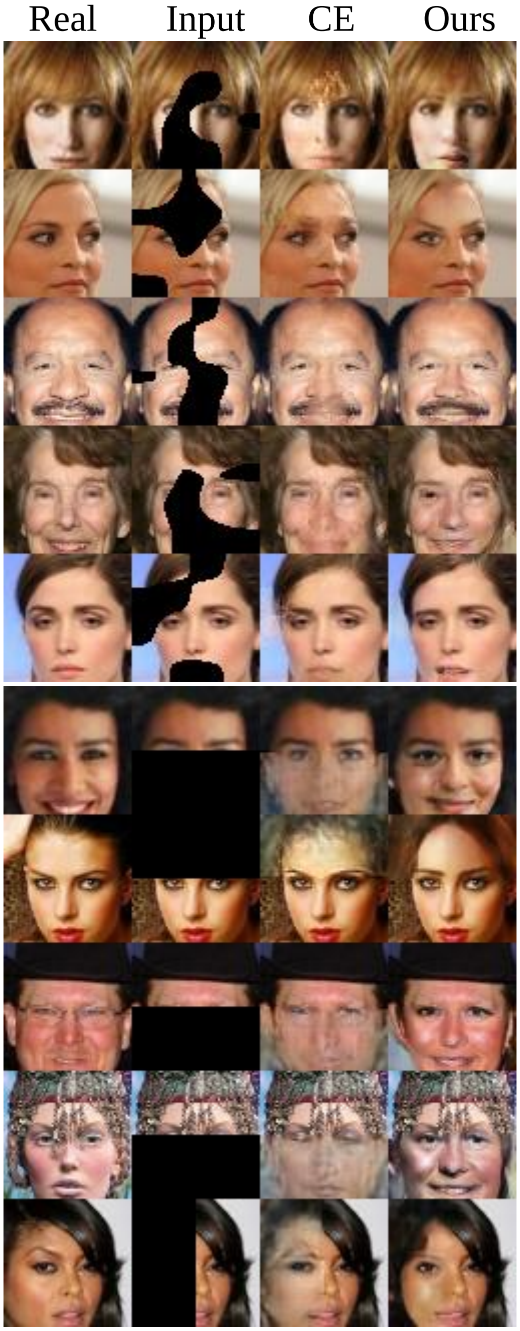


Figure 3.9: Comparisons with CE on the CelebA dataset.

Figures 3.8 and 3.9 show the results on the CelebA dataset with four types of masks. Despite some small artifacts, the CE performs best with central masks. This is due to the fact that the hole is always fixed during both training and testing in this case, and the CE can easily learn to fill the hole from the context. However, random missing data is much more difficult for the CE to learn. In addition, the CE does not use the mask for inference but pre-fills the hole with the mean color. It may mistakenly treat some uncorrupted pixels with similar color as unknown. We could observe that the CE has more artifacts and blurry results when the hole is at random positions. In many cases, our results are as realistic as the real images. Results on SVHN and car datasets are shown in Figs. 3.10 and 3.11, and our method generally produces visually more appealing results than the CE since the images are sharper and contain fewer artifacts.

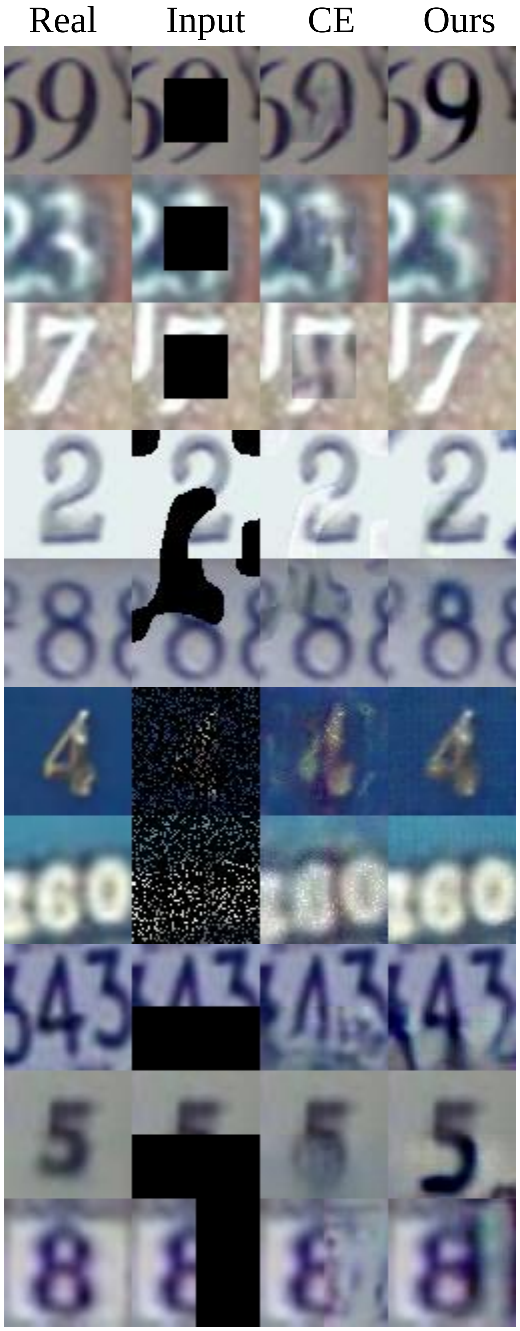


Figure 3.10: Comparisons with CE on the SVHN dataset.

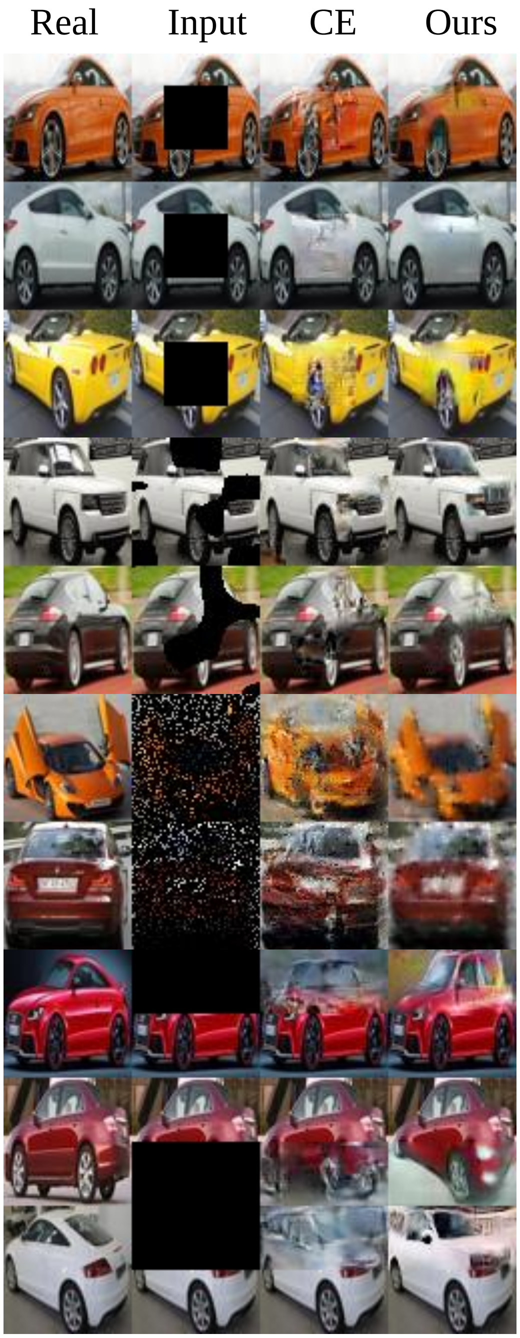


Figure 3.11: Comparisons with CE on the car dataset.

3.4.3 Quantitative comparisons

It is important to note that semantic inpainting is not trying to reconstruct the ground-truth image. The goal is to fill the hole with realistic content. Even the ground-truth image is one of many possibilities. However, readers may be interested in quantitative results, often reported by classical inpaint-

Table 3.1: The PSNR values (dB) on the test sets. Left/right results are by CE [40]/ours.

Masks/Dataset	CelebA	SVHN	Cars
Center	**21.3**/19.4	**22.3**/19.0	**14.1**/13.5
pattern	**19.2**/17.4	**22.3**/19.8	14.0/**14.1**
random	20.6/**22.8**	24.1/**33.0**	16.1/**18.9**
half	**15.5**/13.7	**19.1**/14.6	**12.6**/11.1

ing approaches. Following previous work, we compare the PSNR values of our results and those by the CE. The real images from the dataset are used as ground-truth references. Table 3.1 provides the results on the three datasets. The CE has higher PSNR values in most cases except for the random masks, as they are trained to minimize the mean square error. Similar results are obtained using SSIM [78] instead of PSNR. These results conflict with the aforementioned visual comparisons, where our results generally yield to better perceptual quality.

We investigate this claim by carefully investigating the errors of the results. Figure 3.12 shows the results of one example and the corresponding error images. Judging from the figure, our result looks artifact-free and very realistic, while the result obtained from the CE has visible artifacts in the reconstructed region. However, the PSNR value of CE is 1.73 dB higher than ours. The error image shows that our result has large errors in the hair area, because we generate a hairstyle which is different from the real image. This indicates that quantitative results do not represent well the real performance of different methods when the ground truth is not unique. Similar observations can be found in recent super-resolution works [74, 79], where better visual results corresponds to lower PSNR values.

For random holes, both methods achieve much higher PSNR, even with 80% missing pixels. In this case, our method outperforms the CE. This is because uncorrupted pixels are spread across the entire image, and the flexibility of the reconstruction is strongly restricted; therefore PSNR is more meaningful in this setting which is more similar to the one considered in classical inpainting works.

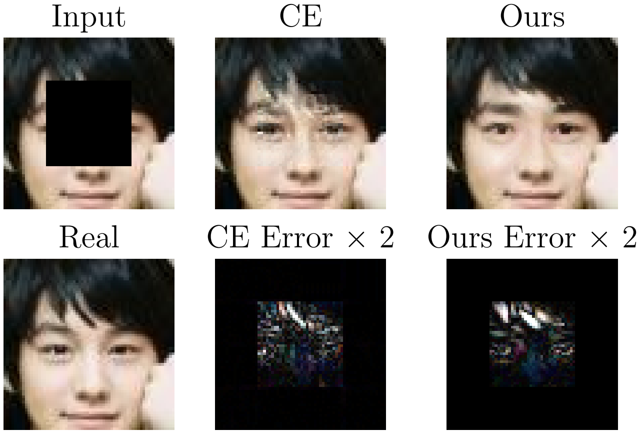


Figure 3.12: The error images for one example. The PSNR for context encoder and ours are 24.71 dB and 22.98 dB, respectively. The errors are amplified for display purpose.

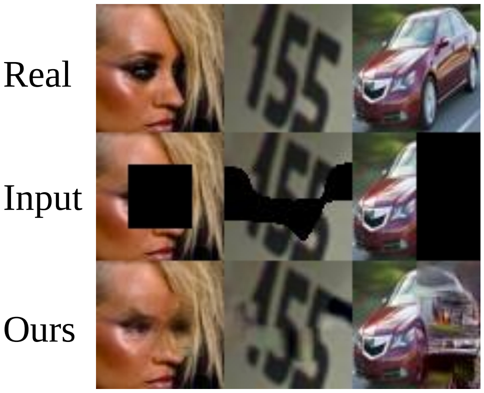


Figure 3.13: Some failure examples.

3.4.4 Discussion

While the results are promising, the limitation of our method is also obvious. Indeed, its prediction performance strongly relies on the generative model and the training procedure. Some failure examples are shown in Fig. 3.13, where our method cannot find the correct $\hat{\mathbf{z}}$ in the latent image manifold. The current GAN model in this chapter works well for relatively simple structures like faces, but is too small to represent complex scenes in the world. Conveniently, stronger generative models, improve our method in a straightforward way.

3.5 Conclusion

In this chapter, we proposed a novel method for semantic inpainting. Compared to existing methods based on local image priors or patches, the proposed method learns the representation of training data, and can therefore predict meaningful content for corrupted images. Compared to CE, our method often obtains images with sharper edges which look much more realistic. Experimental results demonstrated its superior performance on challenging image inpainting examples.

CHAPTER 4

FAST LOW-LIGHT IMAGE PROCESSING

4.1 Introduction

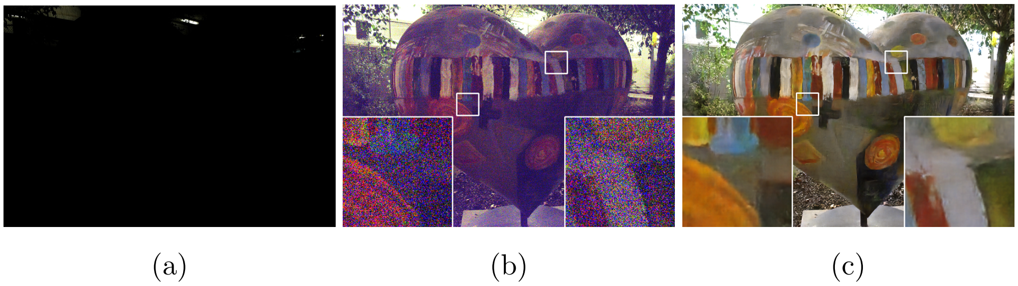


Figure 4.1: (a) An image captured at night by a Fujifilm X-T2 camera with ISO 800, aperture f/7.1, and exposure of 1/30 second. The illuminance at the camera is approximately 1 lux. (b) Processing the raw data by a traditional image processing pipeline does not effectively handle the extreme noise and color bias in the data. (c) Our result obtained from the same raw data.

[1]Noise is present in any imaging system, but it makes imaging particularly challenging in low light. A high ISO is commonly used in nighttime photography to increase the brightness of the image. However, this significantly amplifies noise. One may apply post-processing to improve the quality of low-light images, including scaling or histogram stretching, but these methods do not resolve the low signal-to-noise ratio (SNR) due to low photon counts. There are physical means to increase SNR in low light, including opening the aperture, extending exposure time, and using flash. Unfortunately, each of these has its own characteristic drawbacks. For example, increasing exposure time can introduce blur due to camera shake or object motion.

[1] Chapter 4 has been published in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* 2018 [12]. Copyright IEEE.

The challenge of fast imaging in low light has been studied in the computational photography community, but remains open. Researchers have proposed techniques for denoising [80], deblurring [81], and enhancement of low-light images [82]. These techniques generally assume that images are captured in somewhat dim environments with moderate levels of noise. In contrast, we are interested in extreme low-light imaging with short exposure times (e.g., at video rate). In this regime, the traditional camera processing pipeline breaks down and the image has to be recovered from the raw sensor data.

Figure 4.1 shows an example image captured at night with a 1/30-second exposure. Processing the raw data by a traditional pipeline makes the content visible, but reveals dramatic noise and color bias. As we will show, even state-of-the-art denoising techniques [83] fail to remove such levels of noise and do not address the bias. Burst photography is an alternative approach that uses a burst of images for low-light photography [84, 85]. Although this can significantly reduce noise, using a burst in extreme low-light (sub-lux) conditions is complicated as the robustness of dense correspondence algorithms suffers due to massive noise levels. Furthermore, these methods are not designed for video capture (e.g., due to the use of "lucky imaging" within the burst).

In this chapter, we propose a new image processing pipeline that addresses the challenges of extreme low-light photography via a data-driven approach. Specifically, we train deep neural networks to learn all the image processing operations for raw data, including color transformations, demosaicing, noise reduction, and image enhancement. The pipeline is trained end-to-end to avoid the noise amplification and error accumulation that characterize traditional camera processing pipelines in this regime.

Most existing methods for processing low-light images were evaluated on synthetic data or on real low-light images without ground truth. To the best of our knowledge, there is no public dataset for training and testing techniques for processing fast low-light images with diverse real-world data and ground truth. Therefore, we have collected a new dataset consisting of more than three thousand raw images captured with fast exposure in low-light conditions. Each low-light image has a corresponding long-exposure high-quality reference image. We demonstrate promising results on the new dataset: low-light images are amplified by up to 300 times with successful noise reduction and correct color transformation. We systematically analyze key components of the pipeline and discuss directions for future research.

4.2 Related Work

Computational processing of low-light images has been extensively studied in the literature, with many approaches proposed for noise reduction and contrast enhancement. We provide a short review of these methods.

Image denoising. Image denoising is a traditional topic in low-level vision. Many approaches have been proposed, using techniques such as total variation [86], wavelet-domain processing [87], sparse coding [88, 89], nuclear norm minimization [90], and 3D transform-domain filtering (BM3D) [91]. These methods are often based on specific image priors such as smoothness, sparsity, low rank, or self-similarity. Researchers have also explored the application of deep networks to denoising, including stacked sparse denoising auto-encoders (SSDA) [52, 92], trainable nonlinear reaction diffusion (TNRD) [93], multi-layer perceptrons [94], deep autoencoders [95], and convolutional networks [96, 97]. When trained on certain noise levels, these data-driven methods can compete with state-of-the-art classic techniques such as BM3D and sparse coding. Unfortunately, most existing methods have been evaluated on synthetic data, such as images with added Gaussian or salt&pepper noise. A careful recent evaluation with real data found that BM3D outperforms more recent techniques on real noisy images [83]. Joint denoising and demosaicing has also been studied, including recent work that uses deep networks [98, 99], but these methods have been evaluated on synthetic Bayer patterns and synthetic noise, rather than real images collected in extreme low-light conditions.

In addition to single-image denoising, multiple-image denoising has also been considered and can often achieve better results as more information is collected from the scene [100, 101, 85, 84, 102]. In particular, Liu et al. [85] and Hasinoff et al. [84] propose to denoise low-light images using a burst of images from the same scene. While often effective, these pipelines can be elaborate, involving reference image selection ("lucky imaging") and dense correspondence estimation across images. We focus on a complementary line of investigation and study how far single-image processing of low-light images can be pushed.

Low-light image enhancement. Low-light images suffer from poor visibility and low contrast. A variety of techniques have been applied to improve the contrast of such images. One classic choice is histogram equalization, which balances the histogram of the entire image. Another widely used technique is gamma correction, which increases the brightness of dark regions while compressing bright pixels. More advanced methods perform more global analysis and processing, using for example the inverse dark channel prior [103, 102], the wavelet transform [104], the retinex model [105], and illumination map estimation [106]. However, these methods generally do not explicitly model image noise and typically apply off-the-shelf denoising methods as a post-processing step.

Noisy image datasets. Although there are many studies of image denoising, most existing methods are evaluated on synthetic data, such as clean images with added Gaussian or salt&pepper noise. The RENOIR dataset [107] was proposed to benchmark denoising with real noisy images. However, as reported in the literature [83], image pairs in the RENOIR dataset exhibit spatial misalignment. Bursts of images have been used to reduce noise in low-light conditions [85, 84], but the associated datasets do not contain reliable ground-truth data. The recent Darmstadt Noise Dataset (DND) [83] aims to address the need for real data in the denoising community. Unfortunately, the images in DND were captured at daytime and are not suitable for evaluation of low-light image processing techniques. In addition, the number of images is very limited. To the best of our knowledge, no public dataset exists with raw low-light images and corresponding ground truth. We therefore collect such a dataset to support systematic reproducible research in this area.

4.3 See-in-the-Dark Dataset

We collected a new dataset for training and benchmarking single-image processing of raw low-light images. The See-in-the-Dark (SID) dataset contains 3634 raw images, each with a corresponding reference image. Most images were captured outdoors at night. Since exposure times for the reference images are necessarily long, all the scenes in the dataset are static. The light

Table 4.1: The See-in-the-Dark (SID) dataset contains 3634 raw short-exposure images, each with a reference long-exposure image. The images were collected by two cameras (top and bottom). From left to right: ratio of exposure times between input and reference images, filter array, exposure time of input image, and number of images in each condition.

Sony α7 II	Filter array	Exposure time (s)	# Images
x300	Bayer	1/10	1083
x250	Bayer	1/25	145
x100	Bayer	1/10	721
Fujifilm X-T2	Filter array	Exposure time (s)	# Images
x300	X-Trans	1/30	531
x250	X-Trans	1/25	144
x100	X-Trans	1/10	1010

sources were typically street lights, building lights, and moonlight. We measured illuminance levels for reference using a light meter. The illuminance at the camera is generally between 0.2 lux and 5 lux. Input images were captured with exposure of 1/30 to 1/10 seconds. The corresponding reference (ground truth) images were captured with exposure of 10 or 30 seconds. The dataset is summarized in Table 4.1. A small sample of reference images is shown in Fig. 4.2. Approximately 20% of the images in each condition are randomly selected to form the test set, and another 20% are selected as the validation set.

Images were captured using two mirrorless cameras: Sony α7 II and Fujifilm X-T2. These cameras have very different sensors: the Sony camera has a full-frame Bayer sensor while the Fuji camera has an APS-C X-Trans sensor. This supports evaluation of low-light image processing pipelines on images produced by different filter arrays. The Sony images have resolution 4240×2832, the Fuji images have resolution 6000×4000.

The cameras were mounted on sturdy tripods. We did not use DSLRs to avoid vibration due to mirror flapping. In each scene, camera settings such as aperture, ISO, focus, and focal length were adjusted to maximize the quality of the reference (long-exposure) images. After a long-exposure reference image was taken, a remote smartphone app was used to change the exposure time for the short-exposure image, decreasing the exposure time by a factor of 100 to 300. The camera was not touched between the long-exposure and the short-exposure image. We also collected sequences of short-exposure images to support a comparison with an idealized burst-

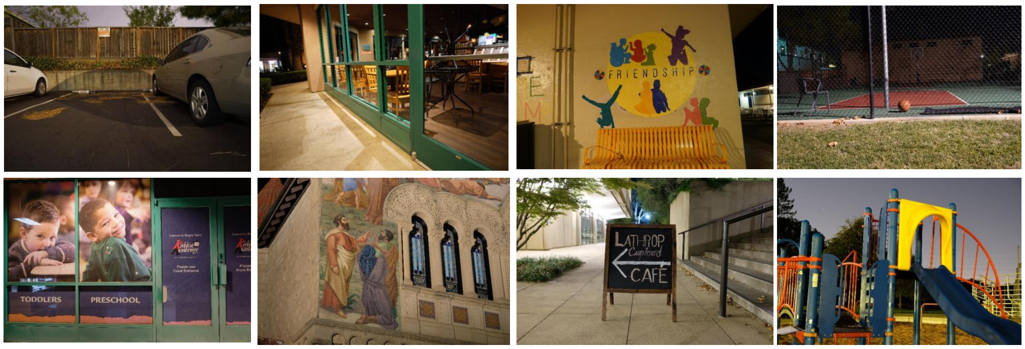


Figure 4.2: Example reference images in the dataset. Top row: images captured by the Sony camera. Bottom row: images captured by the Fuji camera.

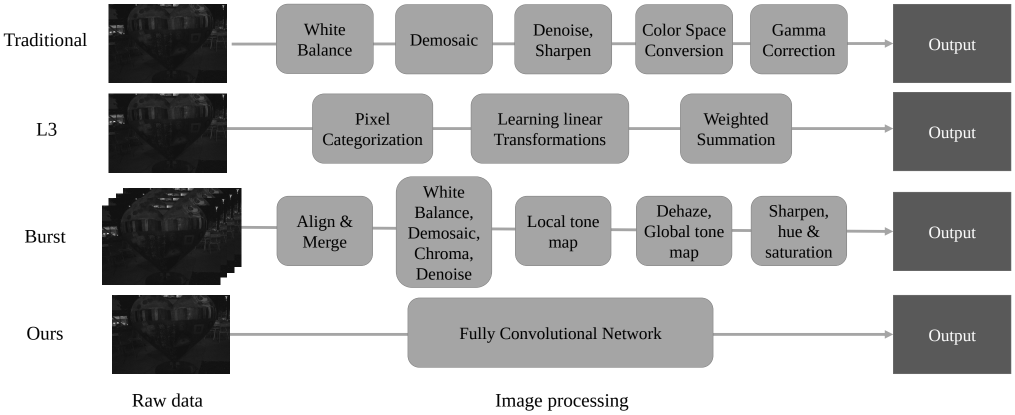


Figure 4.3: The structure of different image processing pipelines. From top to bottom: a traditional image processing pipeline, the L3 pipeline [109], a burst imaging pipeline [84], and our pipeline.

imaging pipeline that benefits from perfect alignment (177 sequences with the Sony, 144 sequences with the Fuji).

We generally avoided very high ISO to minimize blob noise [108]. The long-exposure reference images may still contain some noise, but the perceptual quality is sufficiently good for these images to serve as effective ground truth. We target applications that aim to produce perceptually good images in low-light conditions, rather than exhaustively removing all noise or maximizing image contrast. The entire dataset will be released upon publication.

4.4 Method

4.4.1 Pipeline

After getting the raw data from an imaging sensor, the traditional image processing pipeline applies a sequence of modules such as white balance, demosaicing, denoising, sharpening, color space conversion, gamma correction, and others. These modules often need to be redesigned or tuned for a specific camera. Jiang et al. [109] proposed to use a large collection of local, linear, and learned (L3) filters to approximate the complex nonlinear pipelines found in modern consumer imaging systems. Yet neither the traditional pipeline nor the L3 pipeline successfully deal with fast low-light imaging, as they are not able to handle the extremely low SNR. Hasinoff et al. [84] described a burst imaging pipeline for smartphone cameras. This method can produce very good results by aligning and blending multiple images, but introduces a certain level of complexity, for example due to the need for dense correspondence estimation, and may not easily extend to low-light video capture, for example due to the use of lucky imaging.

We propose to use end-to-end learning for direct single-image processing of fast low-light images. Specifically, we train a fully convolutional network (FCN) [110, 111] to perform the entire image processing pipeline, including both traditional transformations (e.g., white balance and gamma correction) and low-light image denoising and enhancement. Recent work has shown that pure FCNs can effectively represent many image processing algorithms [112, 113]. We are inspired by this work and investigate the application of this approach to extreme low-light imaging. Rather than operating on normal sRGB images produced by traditional camera processing pipelines, we operate on raw sensor data.

We arrange the raw image into multiple channels. For Bayer arrays, we pack the input four channels and correspondingly reduce the spatial resolution by a factor of two in each dimension. For X-Trans arrays, the raw data is arranged in 6×6 blocks. We pack it into nine channels by carefully arranging the RGB values. The raw images are proportional to scene brightness after subtracting the black level. We subtract the black level and scale the data by the appropriate amplification factor (e.g., x100 or x300). The packed and amplified data is fed into the network. The amplification ratio determines the brightness of the output. In this design, the amplification ratio can be set externally to the pipeline. At test time, the input image undergoes blind noise suppression and color transformation. The network directly outputs the high-quality processed image in sRGB space.

4.4.2 Networks and training

Fully convolutional networks are gaining popularity in low-level image processing due to their speed and representational power. We investigate two popular types of architectures for per-pixel regression tasks: a deep full-resolution network [112, 113] and an encoder-decoder structure [114, 115, 116]. Some architectures have incorporated residual connections [117, 80, 97], but we did not find these beneficial in our setting, presumably because our input and output are represented in different color spaces. Another consideration that affected of choice of architectures is memory consumption: we have chosen architectures that can process a full-resolution image (e.g., at 4240×2832 or 6000×4000) in GPU memory. We therefore avoided fully connected layers that require processing small image patches and reassembling them [95].

After preliminary exploration, we have focused on two general structures: a full-resolution network recently used for fast image processing (FastNet) [113] and a U-net [114]. We train the networks from scratch using the mean squared error (MSE) loss and the Adam optimizer [76]. During training, the input to the network is the packed raw data and the ground-truth is the reference image in sRGB space (processed by a traditional pipeline). We first train separate networks for different ratios in Table 4.1 and then experiment with networks trained with multiple amplification ratios. In each iteration, we randomly crop a 512×512 patch for training and apply random flipping and rotation for data augmentation. Initial learning rate is set to 10^{-4} and is decreased to 10^{-5} after 2000 epochs. Training proceeds for 3000 epochs.

4.5 Experiments

In the experiments, we analyze factors that affect the performance of the presented pipeline, and compare to alternative approaches. All images are

Table 4.2: Mean PSNR/SSIM of different models on the Sony test sets.

Ratio	FastNet	U-net	U-net (mixed ratio)
x300	28.34/0.865	29.59/0.890	29.48/0.888
x250	29.22/0.904	28.90/0.912	28.35/0.905
x100	30.80/0.925	30.51/0.935	30.06/0.918

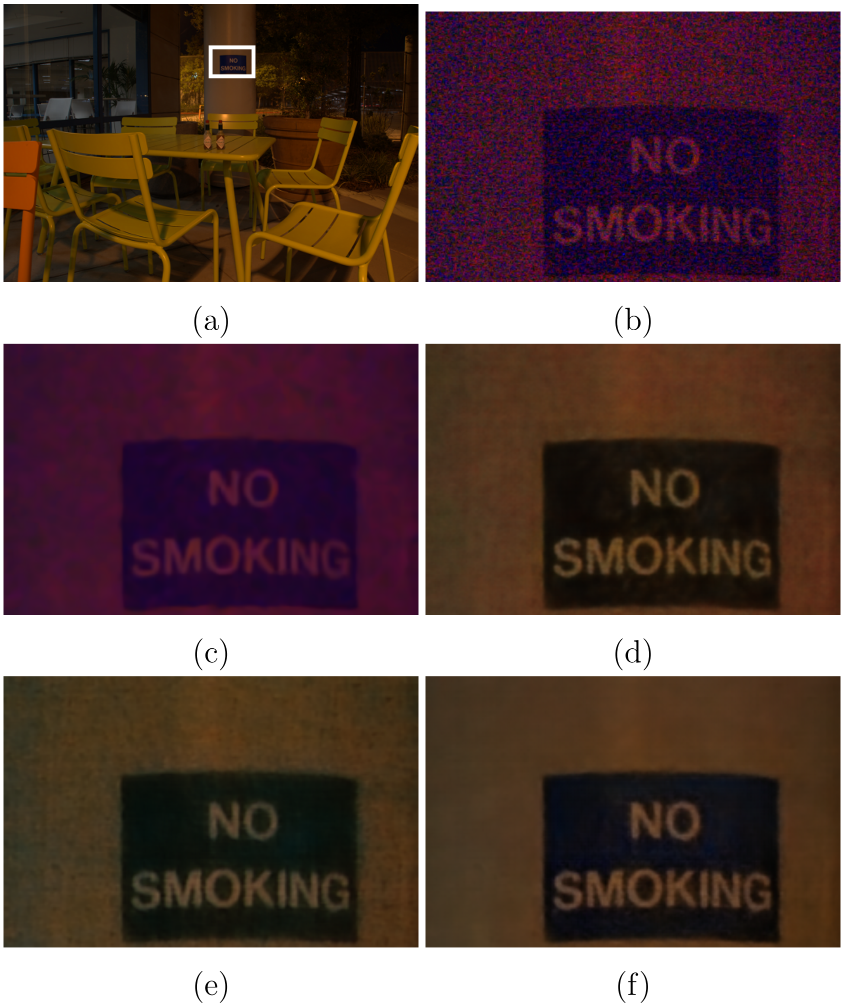
best viewed on the screen, with magnification for detail.

Table 4.3: Mean PSNR/SSIM of different models on the Fuji test sets.

Ratio	FastNet	U-net	U-net (mixed ratio)
x300	28.20/0.909	27.33/0.913	28.17/0.918
x250	29.05/0.907	28.68/0.911	28.84/0.908
x100	29.82/0.933	30.09/0.937	30.01/0.937

We begin by comparing the performance of the two network structures (FastNet and U-net) on the presented dataset. We first train separate models for each set in Table 4.1. The amplification ratio is set to match the ratio of exposure times between input and reference images in the dataset (e.g., x300 for 0.1s input and 30s reference). Results in terms of Peak Signal-to-Noise Ratio (PSNR) and Structural SIMilarity (SSIM) [78] are listed in Tables 4.2 and 4.3. The U-net has lower PSNR in most conditions, but higher SSIM in every condition. In perceptual experiments, we have found the images produced by the U-net to be better in most cases.

Figure 4.4 shows an example from the Sony x300 test set. Figure 4.4(a) shows the reference long-exposure image. As shown in Figure 4.4(b), processing the short-exposure input with the traditional image processing pipeline yields very poor results. (We use `libraw`, extensively optimized for best performance on this data.) As shown in Figure 4.4(c), BM3D, which remains the state-of-the-art denoising method on real data [91, 83], does not successfully process the image produced by the traditional pipeline. This is in part due to poor white balance: as observed in the literature, automatic white balance commonly fails in low-light conditions [84]. The image produced by the FastNet (Figure 4.4(e)) is perceptually noisier than the image produced by the U-Net (Figure 4.4(f)). Likewise, training the U-net on raw data yields better results than training the model on preprocessed sRGB input (Figure 4.4(d)).



(a) (b)

(c) (d)

(e) (f)

Figure 4.4: (a) A reference image from the Sony x300 test set (30s exposure). (b) A patch of the corresponding input image (0.1s exposure), processed by a traditional image processing pipeline. (c) Postprocessing of (b) by BM3D. (d) Output of U-net trained on sRGB images instead of raw data. (e) Output of FastNet. (f) Output of U-net trained on raw data. Zoom in for detail.

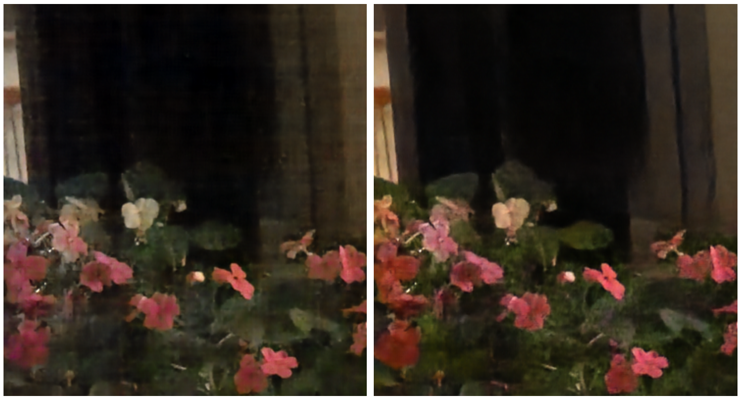


Figure 4.5: A patch from an image in the Fuji x300 test set. Left: output of FastNet with PSNR/SSIM 25.15/0.888. Right: output of U-net with PSNR/SSIM 25.09/0.897. Zoom in for detail.

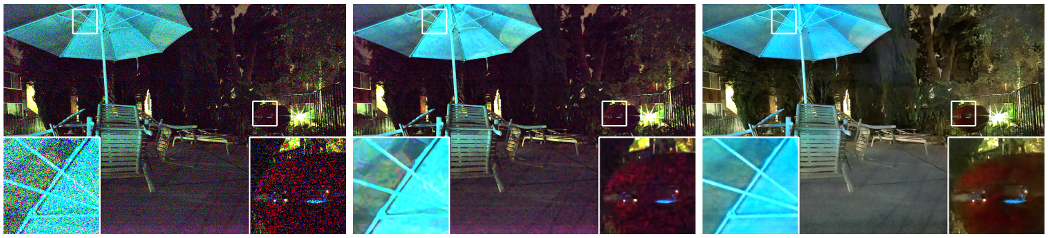


Figure 4.6: An image from the Sony x300 test set. Left: low-light input processed by the traditional image processing pipeline. Middle: same, followed by BM3D denoising. Right: Our result.

A patch from the Fuji x300 set is shown in Fig. 4.5. At high magnification levels, the differences in the output of the FastNet and the U-net are apparent: the U-net produces better results, as reflected in the higher SSIM. In general, we found SSIM to correlate more closely with human judgment on the SID dataset. We will use the U-net as the default architecture in subsequent experiments due to its superior performance.

Note that traditional methods do not successfully handle extreme low-light input even when color bias is not a problem. Figure 4.6 shows one example in which the traditional image processing pipeline yielded correct color balance, which matches the reference image in this respect. However, BM3D still does not successfully handle the huge noise levels in the amplified input, despite extensive parameter search that was conducted to maximize the quality of its output. The result produced by the end-to-end trained network is considerably better.

Most existing denoising methods operate on sRGB images that have already been processed by a traditional image processing pipeline. We have found that training networks directly on raw sensor data is crucial in extreme low-light imaging. Figure 4.4(d) shows the result of the same network architecture (U-net) trained on sRGB input. The result suffers from higher noise levels and artifacts that are not successfully removed by the network. PSNR/SSIM is lower by 1.25/0.028 in comparison to the output of the network that operates on raw data (Fig. 4.4(f)). Our experiments indicate that lossy non-linear steps in the traditional pipeline are not easily reversed, even with end-to-end training of expressive models.

We use the MSE loss by default, but have evaluated many alternative loss functions. The average PSNR/SSIM values on the Sony x300 test set are 29.58/0.892 and 29.46/0.890 for the ℓ_1 and SSIM loss functions, respectively [118]. These are quite close to the performance of the MSE loss, and we have not observed systematic perceptual benefits. Adding a total variation loss does not improve accuracy. Adding a GAN loss [55] significantly reduces accuracy.

Common choices for arranging raw sensor data for processing by a convolutional network are packing the color values into different channels with commensurately lower spatial resolution, or duplicating and masking different colors [99]. We evaluated both options. For Bayer data, packing colors into channels yields a gain of 1.33/0.021 in PSNR/SSIM on the Sony x300 validation set. A typical artifact of the alternative approach – masking different colors in the input – is loss of some hues in the output.

The X-Trans data is very different in structure from the Bayer data and is arranged in 6×6 blocks. One option is to pack it into 36 channels. Instead, we exchange some values between neighboring elements to create a 3×3 pattern, which is packed into nine channels. This arrangement increases the PSNR/SSIM on the Fuji x300 validation set by 0.99/0.025. Figure 4.7 demonstrates the improvement. Our arrangement yields more accurate colors and fine details. We also tried the deep joint demosaicing and denoising network [99], but it was not competitive with the U-net.

The output image is directly upsampled using bilinear interpolation to recover the original spatial resolution. The loss is defined on the upsampled image. We also evaluated the *depth_to_space* layer [119] for increasing resolution and obtained similar performance to simple upsampling.

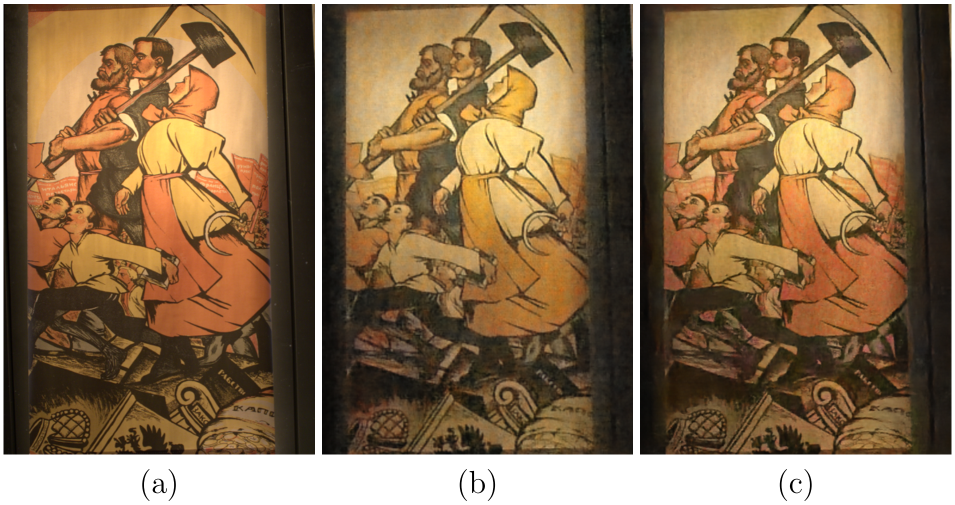


Figure 4.7: Effect of data arrangement on X-Trans data. (a) Reference image. (b) Result of network trained by packing 6 × 6 blocks. (c) Result of network trained by packing rearranged 3 × 3 blocks.

Histogram stretching is a common operation that brightens the image and increases contrast. In initial experiments, we included histogram stretching in the processing pipeline for the reference images. Thus the network had to learn histogram stretching. Despite trying many network architectures and loss functions, we were not successful in training networks to perform this task. With histogram stretching applied to the reference image, the mean PSNR/SSIM drops from 30.09/0.937 to 17.00/0.774 on the Sony x100 test set. Our experiments suggest that convolutional neural networks do not easily learn to model and manipulate global histogram statistics across the entire image, and are prone to overfitting the training data when faced with this task. We thus exclude histogram stretching from the processing pipeline. Figure 4.8 shows a typical result in which attempting to learn histogram stretching yields visible artifacts at test time. The result of training on unstretched reference images has much higher quality but is slightly darker. Histogram stretching is a simple classic operation that can be easily applied post-hoc.

So far, we trained separate networks for each amplification ratio. During training, the network is given amplified raw data as input, with an amplification ratio that matches the reference image. At test time, the raw data can be amplified by any ratio (e.g., x100 or x300) and the network should produce a high-quality image with corresponding brightness. A network trained on

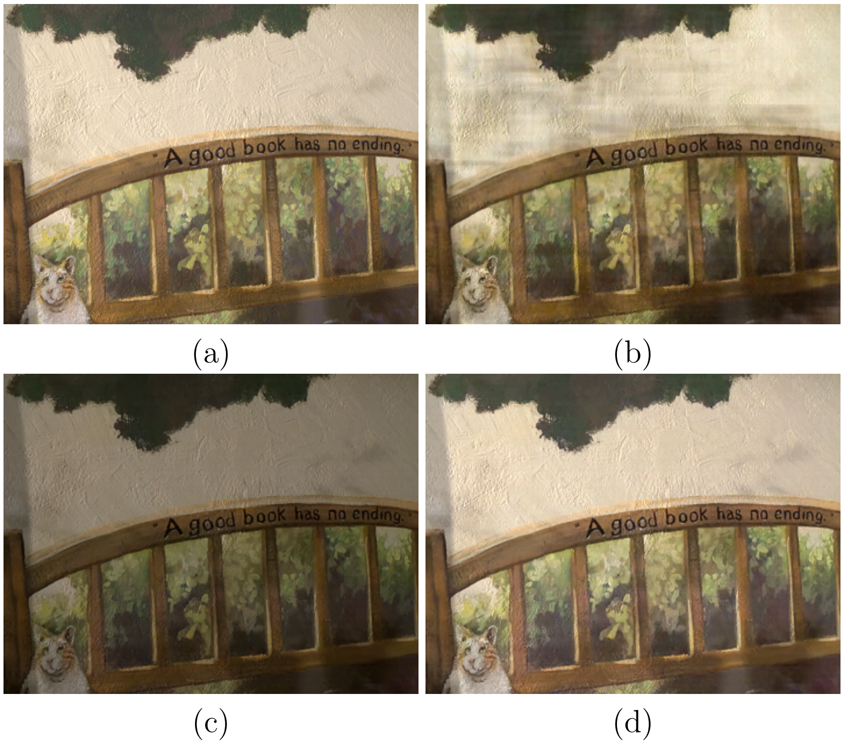
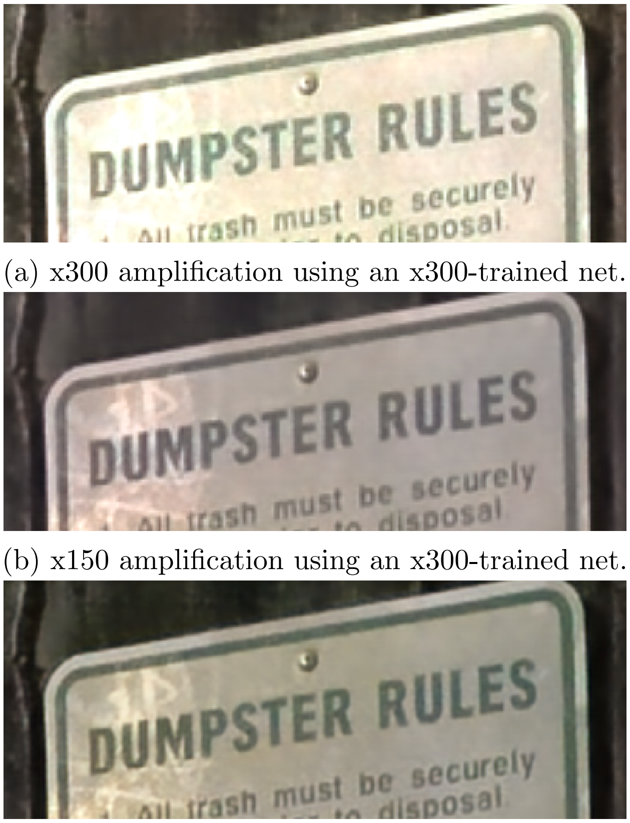


Figure 4.8: Effect of histogram stretching. (a) A reference image in the Sony x100 set, produced with histogram stretching. (b) Output of U-net trained on histogram-stretched images. The result suffers from visible artifacts. (c) Output of the same network trained on images without histogram stretching. The result is darker but much cleaner. (d) The image (c) after histogram stretching.

images with only one amplification level may not generalize well to different brightness levels. For this reason, we also train a single network with all amplification levels in the dataset. This yields a network that handles all of them well: the performance of this single network is reported in Tables 4.2 and 4.3 as “U-net (mixed ratio)”.

We have also tested an alternative design: a network that has multiple heads, one for each representative amplification level. In this condition, we attach multiple copies of the last layer to the network, each of which outputs an image. Each head is trained with images from one of the three amplification levels in the dataset. This improves the results even further: the gain in PSNR/SSIM across the Sony test set is 0.22/0.008 over the single-head mixed-level network. On-demand learning is not required for our problem [120]. Figure 4.9 illustrates the effect of mixed-level training. A network trained on images with only one amplification level exhibits loss of color when the input is amplified by a different ratio at test time. A network trained with diverse amplification levels handles this input well.



(a) x300 amplification using an x300-trained net.

(b) x150 amplification using an x300-trained net.

(c) x150 amplification using a mixed-ratio net.

Figure 4.9: The importance of training with multiple amplification levels. (a) An image from the Sony x300 test set, processed by a network trained on the Sony x300 training set. (b) The output of the same network, when the input image was amplified by a factor of 150; the result suffers from noticeable color shift. (c) A network trained with diverse amplification levels handles the same input well.

We now compare the presented pipeline to BM3D [91] and burst image denoising [85, 84]. (Recall that BM3D remains the state-of-the-art denoising method on real data [83].) Since image sequences in our dataset are already aligned, the burst-imaging pipeline we compare to is idealized: it benefits from perfect alignment, which is not present in practice. (We captured aligned low-light image sequences in order to enable this comparison.) Since alignment is already taken care of, we perform burst denoising by taking the per-pixel median for a sequence of eight images.

Comparison in terms of PSNR/SSIM using the reference long-exposure images would not be fair to BM3D and burst processing, since these baselines have to use input images that undergo a different processing pipeline. For fair comparison, we remove the color bias introduced by poor white balance of low-light images by using the white balance coefficients of the reference image. In addition, we scale the images given to the baselines channel-by-channel to the same mean values as the reference image. These adjustments bring the images given to the baselines closer in appearance to the reference image in terms of color and brightness. Note that this amounts to using privileged information to help the baselines.

To evaluate the relative quality of images produced by our pipeline, BM3D denoising, and burst denoising, we use a perceptual experiment based on blind randomized A/B tests deployed on the Amazon Mechanical Turk platform. Each comparison presents corresponding images produced by two different pipelines to an MTurk worker, who has to determine which image has higher quality. Image pairs are presented in random order, with random left-right order, and no indication of the provenance of different images. A total of 1180 comparisons were performed by 10 MTurk workers. Table 4.4 shows the rates at which workers chose an image produced by the presented pipeline over a corresponding image produced by one of the baselines. We performed the experiment with images from two test sets: Sony x300 (most challenging) and Sony x100 (easier). Our pipeline significantly outperforms the baselines on the challenging x300 set and is on par on the easier x100 set. Recall that the experiment is skewed in favor of the baselines due to the oracle preprocessing of the data provided to the baselines. Note also that burst denoising uses information from eight images with perfect alignment.

Table 4.4: Perceptual experiments were used to compare the presented pipeline with BM3D and burst denoising using eight images with perfect alignment. The experiment is skewed in favor of the baselines, as described in the text. The presented single-image pipeline still significantly outperforms the baselines on the challenging x300 set and is on par on the easier x100 set.

	Sony x300 set	Sony x100 set
Ours > BM3D	92.4%	59.3%
Ours > Burst	85.2%	47.3%

We also collected a set of low-light images with an iPhone 6s smartphone, using an app that supports manual setting of ISO and other parameters. We applied the mixed-ratio network trained on Sony Bayer data to the raw data from the iPhone 6s. A representative result is shown in Figure 4.10. As with data from other cameras, low-light data processed by the traditional pipeline suffers from severe noise and color shift. The result of our network, trained on images from a different camera, has good contrast, low noise, and well-adjusted color.

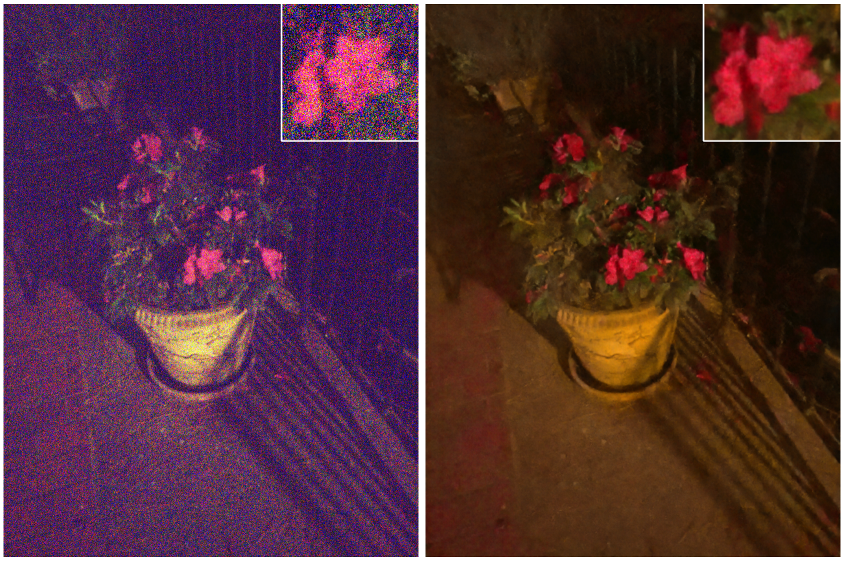


Figure 4.10: Application of a network trained on SID to a low-light raw image taken with an iPhone 6s smartphone. Left: a raw image captured at night with an iPhone 6s with ISO 400, aperture f/2.2, and exposure time 0.05s. This image was processed by the traditional image processing pipeline and scaled to match the brightness of the reference image. Right: the output of our network, with amplification factor x100.

4.6 Conclusion and Discussion

Fast low-light imaging is a formidable challenge due to low photon counts and low SNR. Imaging at video rates at night, in sub-lux conditions, is considered impractical with traditional signal processing techniques. In this chapter, we presented a dataset created to support the development of data-driven approaches that may enable such extreme imaging. Using this dataset, we developed a simple pipeline based on end-to-end training of a fully convolutional network that replaces much of the traditional camera processing pipeline. Experiments demonstrate promising results, with PSNR above 28 and SSIM around 0.9 in 300x amplification of nighttime images (e.g., from 1/30-second exposure to a reference image captured for 10 seconds).

The presented work leaves many opportunities for future research. The presented dataset is limited in that it does not contain humans and dynamic objects. The results of the presented pipeline are limited as well and artifacts are still present, especially in the most challenging x300 regime. Figure 4.11 shows one failure case for extreme imaging, where our result exhibits loss of detail that is apparent upon close examination.

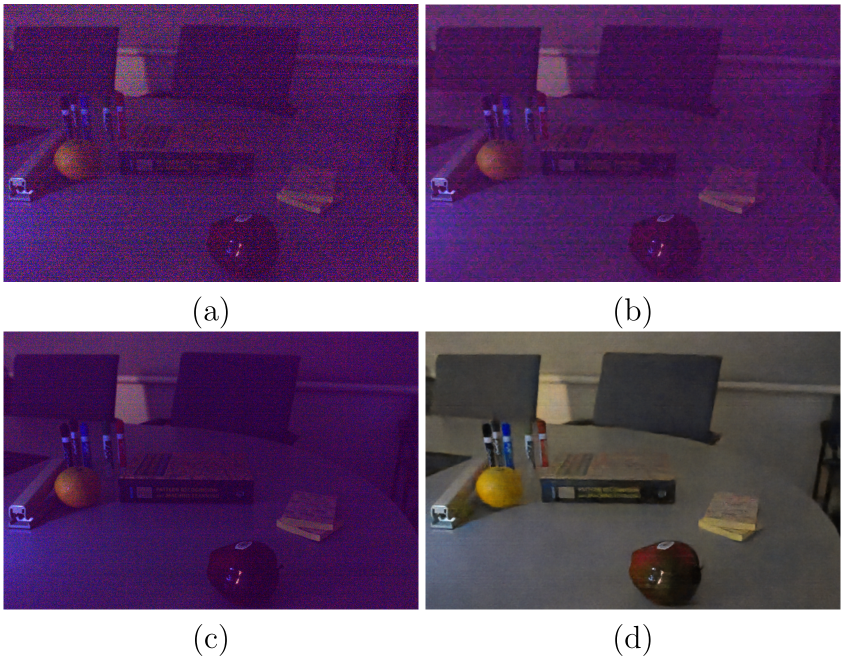

(a) (b) (c) (d)

Figure 4.11: A failure case in extreme low-light conditions (indoor, dark room, 0.2 lux). (a) An input image in the Sony x300 set, processed by the traditional pipeline and scaled to match the reference. (b) BM3D denoising applied to (a). (c) Burst denoising with eight images: the result is still bad due to the severe artifacts in all images in the burst. (d) The result of our network; loss of detail is apparent upon close examination.

Another limitation of the presented pipeline is that the amplification ratio must be chosen externally to the pipeline. Ideally, the ratio that will yield the perceptually optimal output should be inferred from the input. Another opportunity for future work is runtime optimization. The presented pipeline takes 0.38 and 0.66 seconds to process a full-resolution Sony and Fuji image, respectively; this is not fast enough for real-time processing at full resolution, although a low-resolution preview can be produced in real time.

We expect future work to yield significant improvements in image quality, for example by systematically optimizing the network architecture and training procedure. We hope that the presented dataset and experimental findings can stimulate and support such systematic investigation.

CHAPTER 5

FAST LOW-LIGHT VIDEO PROCESSING

5.1 Introduction

We are interested in capturing videos of dynamic scenes in the dark: people dancing in the moonlight, an intimate conversation by candlelight, and a nocturnal animal foraging. Can such scenes ever be captured effectively, in motion, by widely accessible consumer-grade cameras?

Extreme low-light videography is challenging due to low photon counts. Using high ISO can increase brightness but also amplifies noise. Aperture size is limited in consumer-grade cameras and mobile devices. Flash changes the character of the scene and is problematic for videography. And long exposure times (seconds or tens of seconds) are not feasible for videos of dynamic scenes. This leaves us with computational techniques for low-light video processing.

Researchers have developed many techniques to reduce noise for low-light imaging [86, 87, 88, 89, 90, 91, 121, 122, 52, 92, 94, 95, 96, 97]. These techniques generally assume that images are captured in somewhat dim environments with moderate levels of noise. In addition, these methods are often trained and evaluated using synthetic noise models, which do not reflect the severe quantization, bias, and clipping that arise in extreme low-light conditions.

Recent work proposed end-to-end learning for low-light image processing [12, 123]. The idea is to train a deep network on a dataset of short-exposure raw and long-exposure reference images, such that the network learns the entire image processing pipeline to maximize low-light imaging performance. However, these datasets contain images of static scenes and do not address video, and the trained networks exhibit temporal instability that is not easily remedied with post-hoc temporal consistency enhancement. Another approach to low-light photography that has recent achieved significant progress is burst processing [84, 85, 124, 125]. However, these methods are generally not designed for video capture (e.g., due to the use of "lucky imaging") and require dense correspondence estimation across the input frames, which can fail due to massive noise in the conditions we consider.

In this chapter, we tackle end-to-end processing of extreme low-light video, from raw sensor data to sRGB output. This presents challenges beyond those associated with individual low-light images. For example, long-exposure videos of dynamic scenes cannot be obtained, since videos must be acquired at video rate. Thus "ground-truth" long-exposure video of dark dynamic scenes is not available. It is thus not a priori clear how to train models that produce temporally consistent output in this regime.

To study this problem, we collect a new low-light video dataset. We captured 202 static raw videos for training and evaluation, each of which has corresponding long-exposure ground truth. We also capture real-world low-light videos with camera shake and scene motion. For these videos, long-exposure ground truth is not available, and they are used for perceptual experiments. Using the collected data, we develop a new learning-based method for extreme low-light video processing. The method involves training a deep network with a specially designed loss that encourages temporal stability. We show that the network can be trained on static videos but generalizes to dynamic scenes. Experimental results demonstrate that our method outperforms state-of-the-art approaches, as measured by reference-based distortion metrics as well as reference-free perceptual studies.

5.2 Related Work

Single-image denoising. Image denoising has been extensively studied [86, 87, 88, 89, 90, 91, 121]. Most approaches are based on specific image priors such as smoothness, sparsity, low rank, or self-similarity. Learning-based methods further advanced performance in recent years [52, 92, 93, 94, 95, 96, 97]. Lehtinen et al. [126] showed that a denoising network can be trained without clean ground truth if the noise is unbiased; this assumption does not hold in our case. Some networks can denoise and demosaic images jointly [98, 99] or even replace much of the image processing pipeline [12, 123]. However, as demonstrated in our experiments, these single-image processing methods can exhibit significant temporal artifacts when applied to video.

Multiple-image denoising. When video or burst images are available, noise can be reduced using spatial and temporal correlations. Liu et al. [85] and Hasinoff et al. [84] propose to merge a burst of images by robust and efficient aligment methods. Godard et al. [125] propose to use recurrent networks for multi-frame denoising, where the burst sequence needs to be pre-warped to the reference frame. Mildenhall et al. [124] propose to align and denoise bursts via learned per-pixel kernels.

In these works, burst denoising involves reference image selection and outputs a single frame. In contrast, video denoising is more challenging since every frame needs to be processed for the output video, which should be temporally consistent. State-of-the-art video denoising methods include VBM4D [127] and non-local Bayes [128], which rely on grouping similar patches and jointly filtering them to form the result. When noise is small or moderate, these methods can achieve excellent results. However, in extreme low-light conditions, the patch correlation criteria used by these methods can break down. Furthermore, these methods do not address the biases present in the data we consider.

Low-light image and video enhancement. Methods have been developed that can enhance brightness and contrast of images and video acquired in moderately dim environments [103, 102, 104, 105, 106, 129]. However, these methods generally assume that image details are preserved in the sRGB camera output. In contrast, in the extreme low-light settings we consider, the sRGB camera output is unusable and we must work with raw data, with much lower SNR and associated processing challenges that are not addressed by these models.

Noisy image datasets. Image and video denoising datasets have traditionally been created using synthetic noise models, such as Gaussian and Poisson noise, applied to clean images and videos; see Plötz and Roth for review [83]. More recently, datasets were created with real noisy images produced by imaging sensors. These include the RENOIR dataset [107], the Darmstadt Noise Dataset (DND) [83], the Smartphone Image Denoising Dataset (SIDD) [130], and the See in the Dark (SID) dataset [12]. Burst image datasets [85, 84, 125] have also been used for low-light image denoising; however, the bursts are short (less than 10 frames) and scene motion is

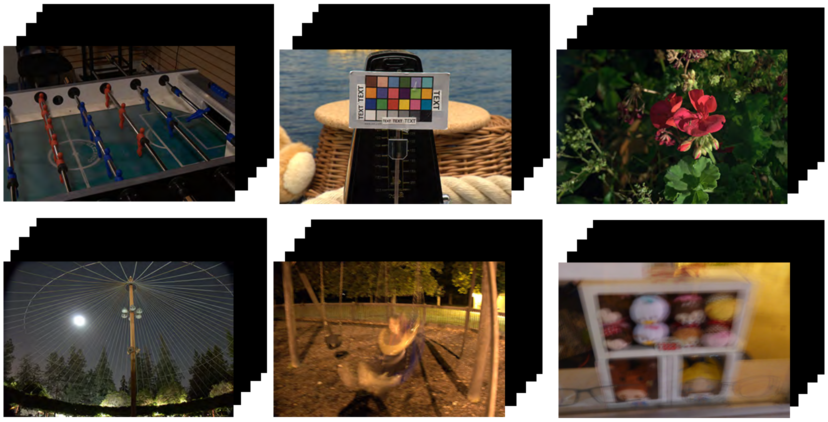


Figure 5.1: Example videos from the dataset. For each video, the first image is the long exposure reference image. The later frames are the short exposed image, which are almost dark in extreme low-light conditions. Note that the last two videos in the second row are from the test set containing motions. The reference long exposure images are blurry due to subject and camera motion, which cannot be served as the ground truth for quantitative evaluations.

small. We collect a new dataset of extreme low-light raw videos, with up to 110 frames each. To the best of our knowledge, this is the first dataset with real-world low-light raw video sequences.

5.3 Dark Raw Video Dataset

We collected a new dataset for low-light raw video processing. We refer to the dataset as Dark Raw Video (DRV). We used a Sony RX100 VI camera, which can capture raw image sequences at approximately 16 to 18 frames per second in continuous shooting mode, and the buffer can keep around 110 frames in total. This is equivalent to 5.5-second videos at 20 fps. The resolution for the Bayer image is 3672 × 5496. The dataset contains both indoor and outdoor scenes.

Following the SID dataset [12], we capture low-light raw data and corresponding long-exposure images. However, this scheme only works for static scenes. Therefore, we collect two sets of videos: one contains static videos with corresponding long-exposure ground truth, and the other contains dynamic videos without ground truth. Examples are shown in Fig. 5.1. Most scenes in the dataset are in the 0.5 to 5 lux range.

To collect a static video, we used a tripod and controlled the camera remotely via a mobile app. Since the scenes in this subset are static, the low-light sequences and the ground truth image (a single frame) are perfectly aligned. We have 202 static videos for training and quantitative evaluation. We randomly divide them into approximately 64% for training, 12% for validation, and 24% for testing. Sometimes there are multiple videos of the same physical scene in different lighting conditions (e.g., light sources with different color temperatures, illuminance, and positions). Each such cluster is placed together in the training, testing, or validation set; images from within a cluster are not distributed across the sets.

We capture a separate set of dynamic video sequences. The motion is due to scene motion, camera motion, or both. These videos do not have ground truth long-exposure references. They are used for perceptual experiments.

The exposure differences between the raw low-light input and the long-exposure ground truth in the static set are between factors of 120 and 300. The low-light raw videos are amplified with these ratios before being fed to the network for processing.

We analyze the noise distribution in the DRV dataset and compare it with a synthetic noise model used in recent work [124]. In the synthetic model, a noisy pixel is assumed to be distributed according to

$$x_p \sim \mathcal{N}(y_p, \sigma_r^2 + \sigma_s y_p) \tag{5.1}$$

Here x_p is the noisy observation, y_p is the true pixel value, and σ_r and σ_s are parameters for read and shot noise. To simulate synthetic noise for comparison, we use the same sampling strategies for σ_r and σ_s as [124].

The low-light data is processed by first subtracting the black level and then amplifying the data by the exposure difference of the low-light input and the long-exposure reference. After this processing, the overall intensity of the processed input matches that of the ground truth. And thus of the synthetic noise model, which is applied to the ground truth for comparison. The comparison is shown in Fig. 5.2. This figure shows the distribution of the real data, compared to the distribution of the synthetic noise model. The distributions are estimated via Parzen density estimation.

Perfectly clean data would correspond to a delta function at 1. As can be seen in the figure, the synthetic noise model is symmetric about 1. On

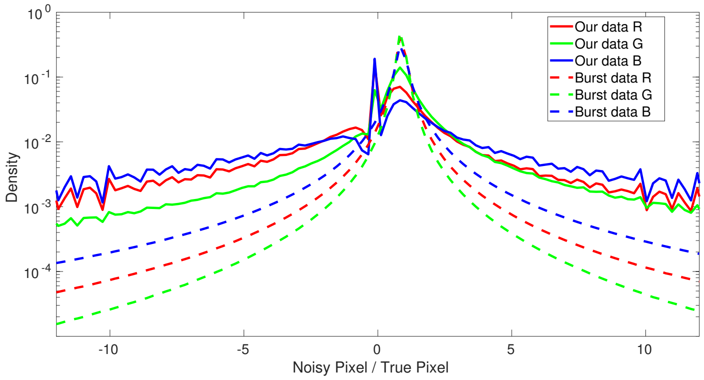


Figure 5.2: Comparison between real low-light noise in the DRV dataset and a synthetic noise model (used in [124]). The X-axis is the ratio between a noisy raw pixel and the corresponding ground-truth value. The Y-axis is the normalized density of these ratios across all pixels in the dataset, computed via kernel density estimation. Note that the Y-axis is in log-scale. Noise in the real data is stronger by an order of magnitude, and exhibits severe non-monotonic bias.

the other hand, the real data is severely biased, in part due to clipping and quantization. For example, there is a peak at zero because many sensor readings are too weak and are quantized to zero even in the 14-bit raw input data. Furthermore, the noise in the DRV data is an order of magnitude stronger than predicted by the synthetic model. (Note that the density is plotted on a logarithmic scale and observe the data at very high noise ratios, such as -10 and 10.) Overall, the average signal-to-noise ratio (SNR) for the synthetic model is 18.59 dB, while in the real data it is -3.24 dB. That is, the SNR in our data, as measured in dB, is negative.

5.4 Method

Our dataset contains two subsets of videos: static videos with ground truth and dynamic videos without ground truth. We think the following criteria are desired for a low-light video processing system:

(a). **Start from raw**. In our extreme low-light dataset, the raw sensor readings are extremely small. In 8-bit JPEG camera output, most of the signal is destroyed and most pixel values are quantized to zero. We should take the 14-bit raw frames as input.

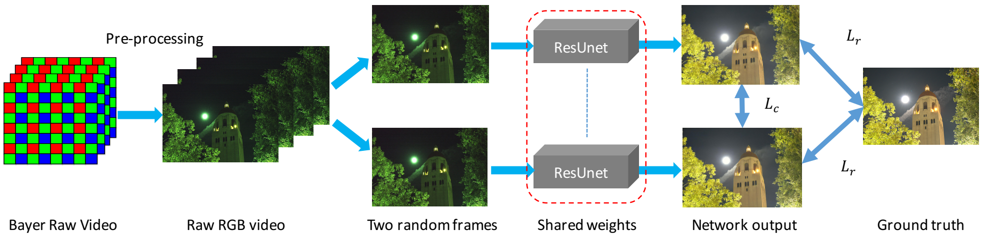


Figure 5.3: The entire training phase of our method on static videos with ground truth.

(b). **End-to-end**. We should train the network end-to-end, from the raw data to sRGB output [12]. Multi-step processing without end-to-end optimization may result in error accumulation if the different processing steps cannot adapt to each other.

(c). **Spatial and temporal denoising**. Both spatial and temporal correlations should be utilized to reduce noise.

(d). **Generalization**. While ground truth is only available for static sequences in DRV, the trained model must generalize to dynamic videos.

(e). **Temporal consistency**. The output video should be temporally stable, without salient flickering artifacts.

In accordance with these requirements, we designed a new learning-based pipeline that uses a deep network to process extreme low-light videos. The training is schematically summarized in Fig. 5.3. First, the raw Bayer video frames are preprocessed. The preprocessing includes Bayer to raw RGB conversion, black level subtraction, and pixel amplification. The Bayer data is split into separate RGB channels to form the raw RGB where the green channel is obtained by averaging of every two green pixels in two-by-two blocks. The pixel values are amplified by the exposure difference with the ground truth. In addition, we apply VBM4D [127] to reduce the noise using spatial and temporal correlations. Note that VBM4D does not need training data and can be applied on both static and dynamic videos.

The preprocessed raw RGB frames are fed to a deep network that is trained to perform all subsequent processing needed to obtain the results demonstrated in the ground truth images. The network takes a single frame as input. For training, two frames from a static sequence in DRV are sampled at random and are fed to the network in siamese mode. Let $\hat{Y}^1$ and $\hat{Y}^2$ denote these two frames and let the ground truth for this sequence be denoted by Y^*. The loss for this training pair is defined as follows:

$$\mathcal{L}(\hat{Y}^1, \hat{Y}^2, Y^*) = \mathcal{L}_r + \mathcal{L}_c \tag{5.2}$$

where $\mathcal{L}_r$ is referred to as the the recovery loss and $\mathcal{L}_c$ is called the self-consistency loss. They are defined as follows:

$$\mathcal{L}_r = \sum_l \frac{1}{N^l} \sum_{k=1,2} \|\Phi^l(Y^*) - \Phi^l(\hat{Y}^k)\|_1 \tag{5.3}$$

$$\mathcal{L}_c = \sum_l \frac{\lambda}{N^l} \|\Phi^l(\hat{Y}^1) - \Phi^l(\hat{Y}^2)\|_1 \tag{5.4}$$

Here Φ^l denotes the VGG [131] features at the l-th layer and N^l is the number of such features. λ is a regularization parameter and was empirically set to 0.05 for all results. The recovery loss $\mathcal{L}_r$ encourages the output to be close to the ground truth. However, this alone is not sufficient for temporal consistency. As shown in Fig. 5.4, two outputs may have the same ℓ_1 distance to the ground truth in feature space, but may be far from each other. This corresponds to temporal instability (flickering). To alleviate temporal instability, we use the self-consistency loss, which encourages the two outputs to be close to each other. Refer again to Fig. 5.4 for the geometric intuition.

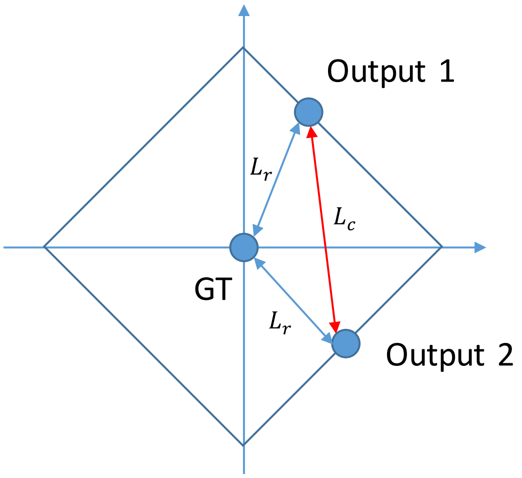


Figure 5.4: The distance between outputs and ground truth (GT) in the feature space.

The network produces output in sRGB space. We use a ResUnet structure akin to [132] by adding 16 residual blocks [133] to a Unet [114, 12].

Our method easily satisfies the first and second criteria discussed earlier.

Noise is reduced using spatial and temporal correlations in preprocessing by VBM4D. The trained network can then adapt to the characteristics of the preprocessed input and optimize for fidelity given this input. The self-consistency loss, used during training, encourages the network to produce temporally stable output. (As we shall see in the experiments, this temporal stability characteristic carries over into dynamic videos at test time.) Since the network operates on a single frame at test time, it generalizes to dynamic videos.

5.4.1 Implementation details

We implement our method using Tensorflow [134]. We found that training on complete images rather than patches is very important for accuracy, as some of the pipeline operations may need global statistics (e.g., white balance). In the first stage of training, both input and ground truth were downsized by a factor of two. This allows full images at this resolution to fit into GPU memory for training. We clip large noise outside the [0,1] range (after amplification). We train our model on an Nvidia Titan Xp GPU with 12 GB of memory. We use the Adam optimizer [76]. The initial learning rate is 2×10^{-4} and is reduced to 10^{-4} and 10^{-5} after 32500 and 97500 iterations, respectively. We train the network for 130000 iterations on GPU. After that, we fine-tune the model on CPU for 3000 iterations. We use the input, "conv1_2", "conv2_2", "conv3_2", and "conv4_2" layers of the VGG network as features in the loss.

5.4.2 Discussion of alternative options

Dense correspondence. If the frames can be perfectly aligned, noise can be significantly reduced by averaging or smoothing temporally. Some existing methods rely on pre-warping [85, 84, 125] or learning to align [124]. Almost all optical flow methods assume that little to no noise is present in the input frames. However, this assumption breaks down in our setting. Even after preprocessing with VBM4D, there is still substantial noise and artifacts. Figure 5.5 shows the optical flow estimated on our data (dynamic DRV sequence) using the state-of-the-art PWC-Net [135]. The flow has significant

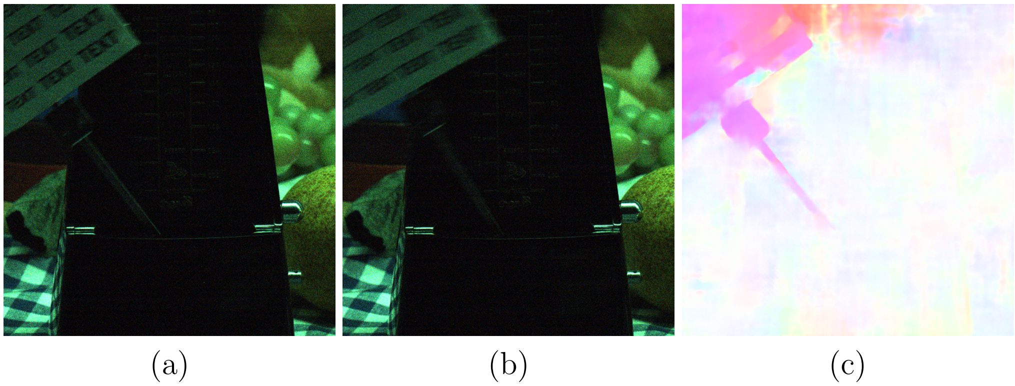

(a) (b) (c)

Figure 5.5: Optical flow on preprocessed raw RGB images. (a) and (b) are two consecutive frames from a dynamic DRV sequence. In this sequence, the metronome, with a text sign, is ticking, while the rest of the scene is static. (c) Optical flow between the two frames, estimated by PWC-Net [135]. The flow contains significant errors.

errors. This suggests that prewarping via dense correspondence estimation is problematic in our setting. (We have further verified this experimentally.)

Denoising followed by color conversion. Another option is to learn joint alignment and denoising [124], followed by subsequent processing (e.g., to map from raw RGB to 8-bit sRGB). As we will see in controlled experiments, such decoupled processing is suboptimal because the later processing stages do not optimally adapt to the characteristics of the input provided to them. In practice, the later processing stages significantly amplify errors from earlier stages.

Temporal consistency. Another possibility is to enhance temporal consistency in post-processing. Existing methods use the input videos to guide such enhancement [39, 132]. The underlying assumption is that the input video is temporally consistent. This is not true in our case. Due to the extremely low SNR, the input video is temporally unstable. Applying state-of-the-art temporal consistency techniques to our data (e.g., SID [12] for per-frame processing, followed by learned blind temporal consistency [132]) therefore yields severe visible artifacts, as shown in Fig. 5.6.

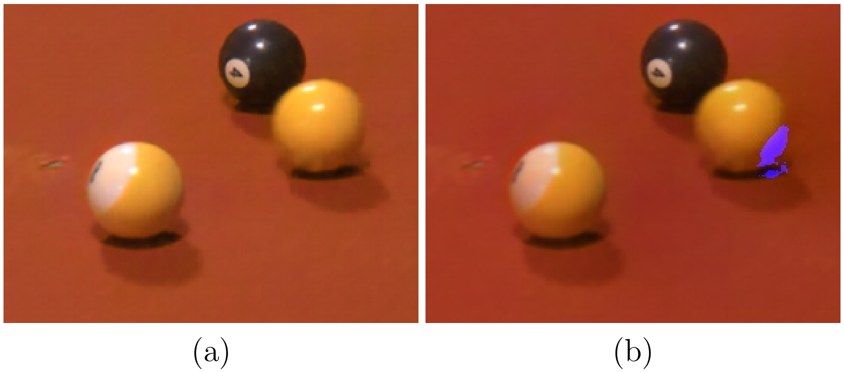


Figure 5.6: (a) Result by SID [12]. (b) Applying learned blind temporal consistency [132] as post-processing results in visible artifacts. Note the blue patch on the ball.

5.5 Experiments

5.5.1 Experimental setup

Following SID [12], we use Rawpy (a Python wrapper for LibRaw) as the reference traditional non-learning-based pipeline. The long exposure raw images are processed by Rawpy to form the sRGB ground truth. We use the metadata of the ground truth raw data for all Rawpy processing. This benefits the traditional pipeline, as the white balance estimation is worse in low-light conditions. For our method and SID [12], we use the preprocessed low-light raw frames as input and learn to convert the colors without the need for metadata. As one of our baselines, we train the kernel prediction network (KPN) [124] with default settings using the author-provided code, on amplified raw RGB images and the corresponding long-exposure raw images; the denoised results are processed by Rawpy to produce the sRGB output. Both VBM4D [127] and KPN use eight frames for temporal denoising.

5.5.2 Image quality evaluation

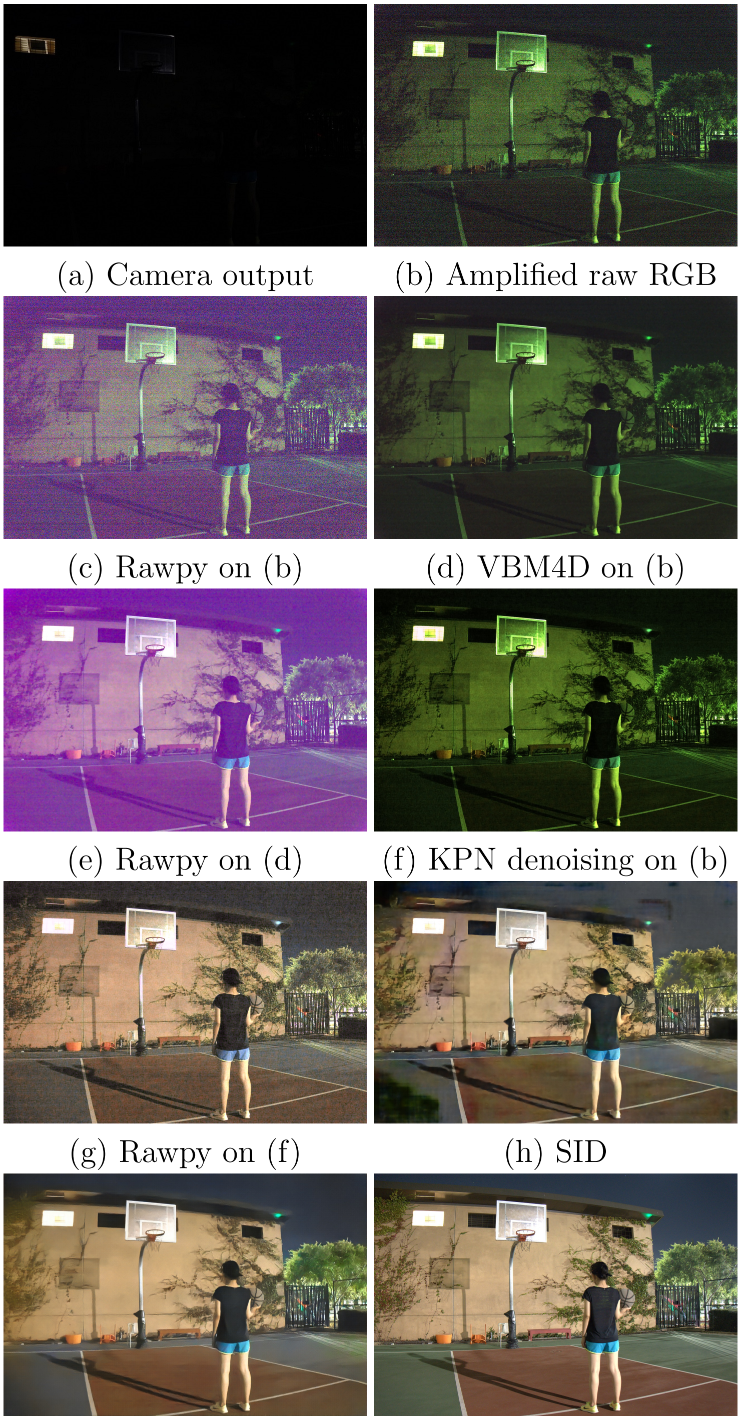
We evaluate different methods on the static test videos. The 5th frame of the output video is compared with the ground truth using Peak Signal to Noise Ratio (PSNR) and the Structural Similarity Index (SSIM) [78]. The average results over the entire test set are listed in Table 5.1. Our method significantly outperforms the baselines. The ablations confirm the benefits of VBM4D preprocessing and the self-consistency loss.

Table 5.1: Quantitative evaluation of image quality on the static video test set.

	PSNR (dB)	SSIM
Input+Rawpy	12.94	0.165
VBM4D [127]+Rawpy	14.77	0.315
KPN [124]+Rawpy	18.81	0.540
SID [12] w/o VBM4D	27.32	0.790
SID [12]	27.69	0.803
Ours	**28.37**	**0.821**
Ours w/o $\mathcal{L}_c$	27.90	0.801
Ours w/o VBM4D [127] & $\mathcal{L}_c$	27.30	0.786

An example is shown in Fig. 5.7. The camera output is almost completely black when the exposure time is fixed to 1/30 seconds in this dark environment. Linearly amplifying the raw RGB image, as shown in Fig. 5.7(b), reveals the content and shows the noise and bias in the data. Applying the traditional image processing pipeline (including while balance), as shown in Fig. 5.7(c), boosts red and blue channels due to bias in the data. VBM4D [127] can remove the noise to an extent on the raw RGB space or the output sRGB space, but cannot correct the color shift. As shown in Fig. 5.7(f), KPN learns to remove the noise in the raw color space. However, as shown in Fig. 5.7(g), when the traditional pipeline converts from the raw color space to sRGB and increases image contrast, it significantly amplifies the residual errors. This suggests that denoising followed by the traditional pipeline is sub-optimal: the whole pipeline has not been optimized end-to-end, and the denoising stage leaves some residual noise that is boosted by later processing. Our method is trained end-to-end to avoid such error accumulation. In comparison with our results (Fig. 5.7(i)), SID [12] has strong artifacts in the sky and on the wall (Fig. 5.7(h)).

Since SID is the strongest baseline, we further use it for comparison in Fig. 5.8. While SID has strong denoising ability, it sometimes exhibits discoloration artifacts. For example, the letters in “Voice” and “Daily Post” in Fig. 5.8(a) (top) shift to a yellow tint. In Fig. 5.8(a) (bottom), the green box and orange book lose their color in the center. Such artifacts happen less frequently on the SID dataset [12], captured by a more expensive camera with longer exposure (up to 1/10 seconds), but occur more prominently in



(a) Camera output (b) Amplified raw RGB

(c) Rawpy on (b) (d) VBM4D on (b)

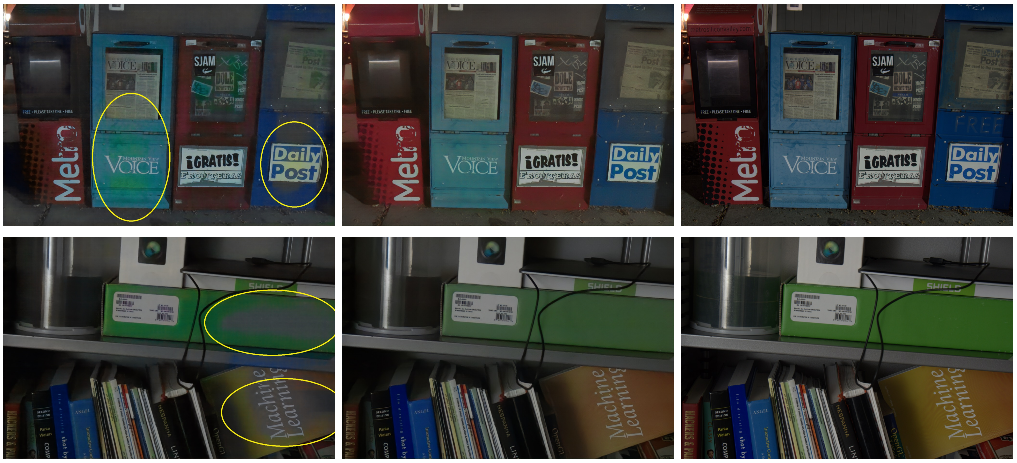
(e) Rawpy on (d) (f) KPN denoising on (b)

(g) Rawpy on (f) (h) SID

(i) Ours (j) Ground truth

Figure 5.7: An example from a nighttime sequence captured with a Sony RX100 VI camera with aperture f/4, ISO 2000, and 1/30 second exposure. This is a static DRV sequence, so ground truth is available for reference. Zoom in for details.

the DRV dataset. Although both SID and our method use exactly the same input, our results have consistently higher quality.



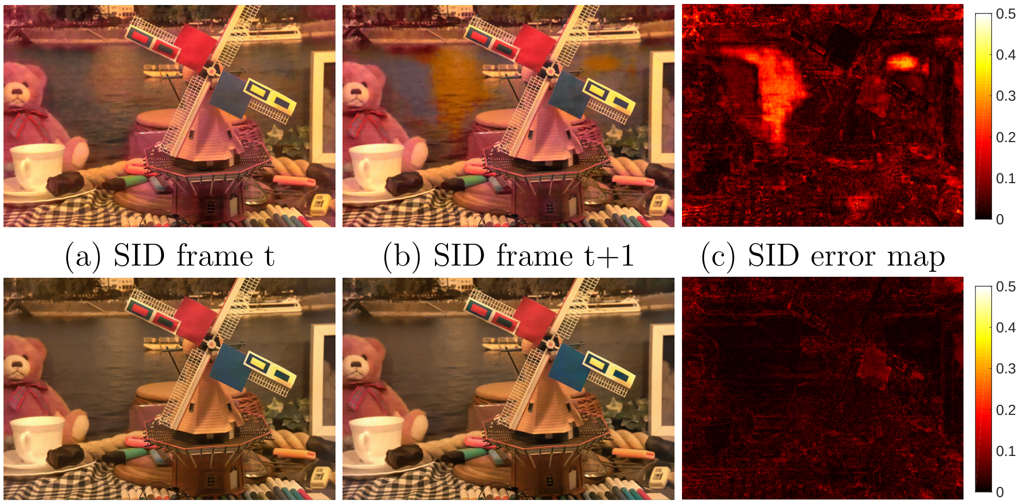
(a) SID (b) Ours (c) Ground truth

Figure 5.8: Image quality comparison with SID [12] on two examples. Note the discoloration artifacts in the demarcated regions.

5.5.3 Video quality evaluation

We further evaluate the video quality of SID and our method. Adopting the methodology of [132], we measure temporal error on every pair of consecutive frames using PSNR, SSIM, and mean absolute error (MAE) on the static test videos. We use the terms temporal PSNR (TPSNR), temporal SSIM (TSSIM), and temporal MAE (TMAE) to distinguish the temporal variants from single-image metrics. Since the frames in the static DRV sequences are perfectly aligned, there is no need to apply warping. The results are shown in Table 5.2, where the images are scaled to [0, 1] for TMAE calculation. It demonstrates that our method has much lower temporal error than SID. We found that larger λ leads to smaller temporal errors, but at the cost of somewhat lower spatial accuracy. At the inference stage, SID and our method take 0.24 and 0.30 seconds on average to process a frame on a Titan Xp GPU, respectively.

Although the input frames are static, they contains strong flickering artifacts due to random noise. We further visualize the temporal errors in Fig. 5.9, which shows two consecutive frames from a static DRV sequence,



(a) SID frame t (b) SID frame t+1 (c) SID error map

(d) Our frame t (e) Our frame t+1 (f) Our error map

Figure 5.9: The visual results of two consecutive frames on a static video. The error maps show per-pixel error, measured by Euclidean distance in [0,1] sRGB space. Brighter means larger errors.

Table 5.2: Temporal errors on the static video test set for SID and our method.

	TPSNR	TSSIM	TMAE
	(dB)		($\times 10^2$)
SID [12] w/o VBM4D	33.72	0.939	1.56
SID [12]	37.05	0.961	1.05
Ours	**38.60**	**0.974**	**0.87**
Ours w/o $\mathcal{L}_c$	38.19	0.970	0.92
Ours w/o VBM4D & $\mathcal{L}_c$	34.61	0.950	1.39

processed by SID and by our method. The SID results exhibit temporal instability in the form of discoloration. Our method is much more stable.

5.5.4 Perceptual evaluation

For dynamic videos, we do not have reference long-exposure ground truth. Therefore, we conduct a perceptual experiment that compares the results of SID and our approach. In blind randomized A/B tests, we display corresponding video pairs and ask workers to indicate which of the two videos has better quality. Order is randomized both within and across pairs. There are 34 workers who participated in the experiment, ranking results on 10 dy-

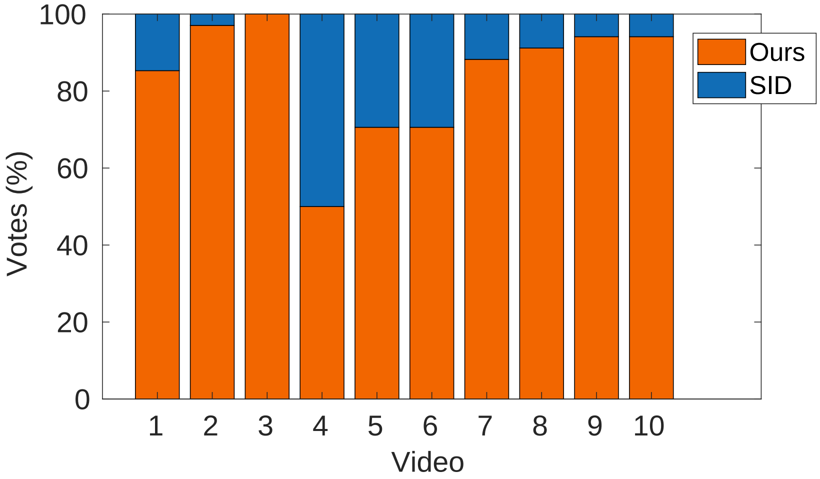


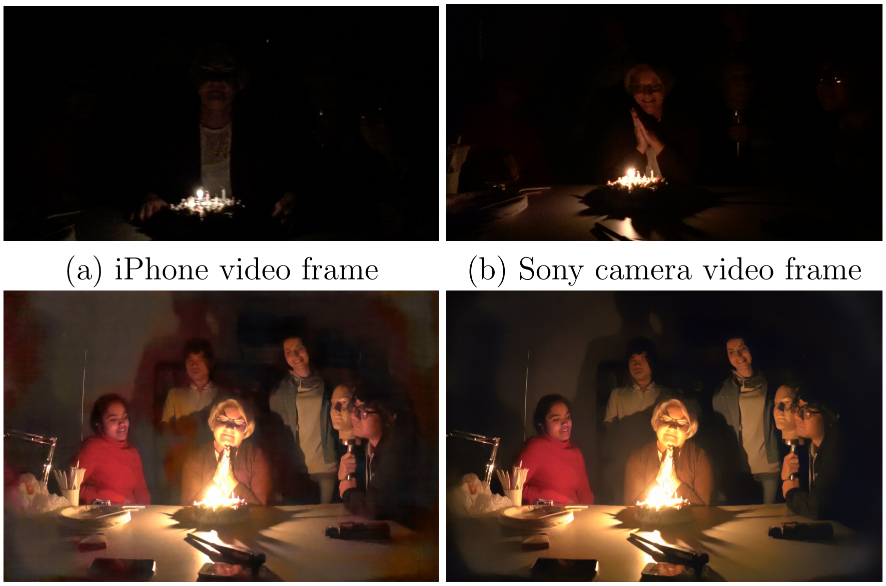
Figure 5.10: Perceptual experiment. Results of blind randomized A/B tests on 10 dynamic videos. The figure shows preferred percentage for each video.

namic video sequences. Figure 5.10 shows the results. Overall, the workers rate videos produced by our method as superior in quality in 84.12% of the comparisons. The result is statistically significant with $p < 10^{-3}$.

5.5.5 Extreme imaging

Finally, we demonstrate our method qualitatively in Fig. 5.11. Videos from an iPhone X and the Sony RX100 VI camera video mode are used for reference. In this mock birthday party video, illumination was provided by a single candle. This is a sub-lux setting. The iPhone video was captured using the auto mode. For the Sony video, we fixed the exposure time to 1/30 seconds while keeping the maximum aperture and ISO. The raw image sequences for SID and our method were captured with ISO 2000 in continuous shooting mode.

Light intensity is inversely proportional to the square of the distance from the source. We thus see in Fig. 5.11(a,b) that in the iPhone and Sony sequences only the birthday lady can be (dimly) made out in the image. Both SID and our method reveal the entire scene. However, the SID result suffers from both spatial and temporal artifacts, while our result is cleaner and more stable. This is video #10 in the perceptual experiment (Fig. 5.10), for which 94.1% of the comparisons are in favor of our result.



(a) iPhone video frame

(b) Sony camera video frame

(c) SID result

(d) Our result

Figure 5.11: Video of a dynamic scene lit with a single candle. The illuminance is 0.73 lux at the birthday lady's ear.

5.6 Conclusion

We presented a new dataset and a new method for extreme low-light video processing. The dataset contains both static and dynamic videos. Quantitative and qualitative results demonstrate that the presented method achieves superior performance over a range of baselines, particularly in the more extreme low-light scenarios. While the improvement is significant, certain failure modes remain. For example, our method (as well as the baselines) completely failed on moonlight videos (approximately 0.01-0.03 lux). Furthermore, we did not consider HDR tone mapping. For example, the area around the candles in Fig. 5.11 is over-exposed. There is scope for much exciting future work.

CHAPTER 6

CONCLUSION

In this work, we studied three image restoration problems: image dehazing, image inpainting and fast low-light imaging.

In the first problem image dehazing, we proposed a new method that minimized the gradient residual between the input and output pairs. The optimized function value leads to the desired result with significant artifact supression. This method is inspired by that the artifacts generated by previous methods often are not visible in the input hazy images.

In the second problem, we proposed to use deep generative models for semantic image inpainting. It is assumed that if a deep generative model can generate high-quality images from a paticular distribution, we can find the missing part by optimizing the latent code.

In the third problem, the images captured in low-light conditions with short exposures often have extreme low SNR. The traditional image processing pipeline has difficulties to process such data, particularly, in white balance and noise suppression steps. We proposed to recover low-light images using end-to-end supervised learning. A new dataset was collected using two cameras with a Bayer filter and an x-Trans filter, respectively. We demonstrate promissing results compared with traditional image processing pipeline and the state-of-the-art image denoising method BM3D.

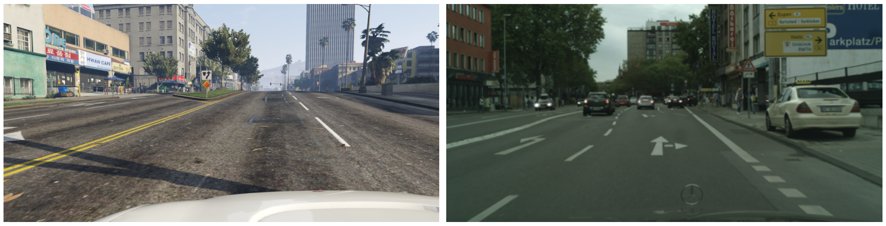
The last problem of extreme low-light video processing is much more challenging. A new dataset was collected to fulfill this purpose. We have to exploit spatial and temporal correlations without ground-truth data for the dynamic videos. Also, the output should be temporally smooth. We demonstrate an effective processing pipeline by using deep learning and VBM4D preprocessing. The results show the superior performance of our method compared with the state-of-the-art. It also indicates lots of future opportunities.

There are some differences among these methods. In image dehazing, we optimize an object function to obtain the desired results. Our method is general to be applied to a wide range of data without using any training data. However, this process is too slow to be used in real-time applications. In the second problem, we need pre-trained deep generative models. Although we still need training data with the same distribution, the training is unsupervised without using any label. During the test phase, we optimize the latent code in order to find a solution that close to the original image. We use supervised training to solve the fast low-light imaging and video processing problem. The training is time comsuming and requires a large dataset. The benefit is that the test speed is very fast.

CHAPTER 7

FUTURE WORK

Image synthesis from computer graphics (CG) images is an interesting future research direction. For example, we aim to transfer the style of the Grand Theft Auto (GTA) dataset [136] to the Cityscapes dataset [137]. Figure 7.1 shows the example images of two datasets. The goal is to synthesize realistic images without artifacts. However, the task is very challenging because there is no correspondence between the two datasets. We have to use unsupervised learning.

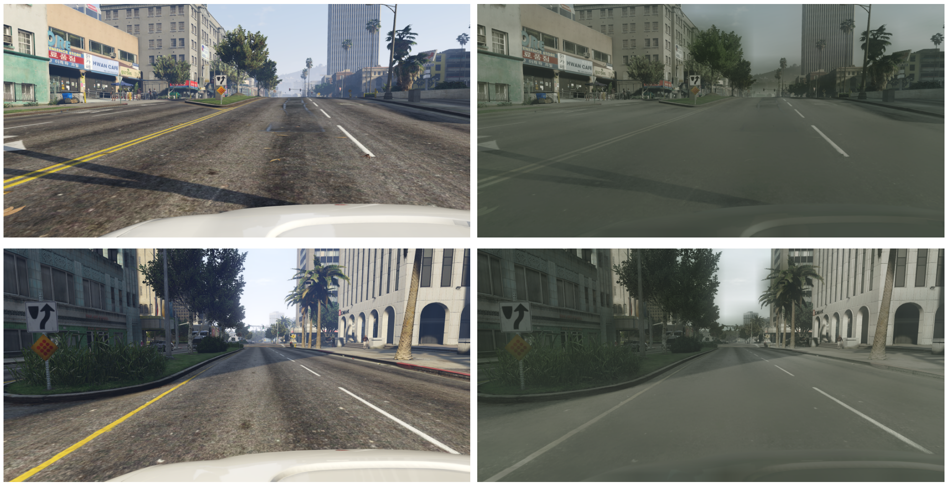


(a) A GTA example (b) A Cityscapes example

Figure 7.1: Example images from the GTA dataset [136] and the Cityscapes dataset [137].

Many existing methods are based on the GAN architecture with an additional cycle consistency loss [138, 139, 140, 141]. A common problem of these methods is artifacts. We plan to use the structure preserving networks (e.g. [142]) to constrain the structure of the image, probably with additional semantic or instance labels to make the problem more tractable.

While CyCADA [138] can transfer the style between two domains, the results suffer from severe artifacts. Although the results contain distortion artifacts, they are coarsely aligned with the input images. We train a structure preserving network [142] using GTA images as input and CyCADA results as the reference. Some preliminary results are shown in Fig. 7.2. We can observe the structure of the input is preserved in the results and the style is successfully transferred. The goal of future work is to train a structure preserving network without the use of CyCADA.



(a) The original GTA images (b) The transferred images

Figure 7.2: GTA to real image synthesis results.

REFERENCES

[1] M. Bertero, T. A. Poggio, and V. Torre, "Ill-posed problems in early vision," in *Proceedings of the IEEE*, vol. 76, no. 8, 1988, pp. 869–889.

[2] J. Wright, Y. Ma, J. Mairal, G. Sapiro, T. S. Huang, and S. Yan, "Sparse representation for computer vision and pattern recognition," in *Proceedings of the IEEE*, vol. 98, no. 6, 2010, pp. 1031–1044.

[3] J. Yang, J. Wright, T. S. Huang, and Y. Ma, "Image super-resolution via sparse representation," *IEEE Transactions on Image Processing*, vol. 19, no. 11, pp. 2861–2873, 2010.

[4] B. Liu, J. Huang, C. Kulikowski, and L. Yang, "Robust visual tracking using local sparse appearance model and k-selection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 12, pp. 2968–2981, 2013.

[5] C. Chen and J. Huang, "Compressive sensing MRI with wavelet tree sparsity," in *Advances in Neural Information Processing Systems*, 2012, pp. 1115–1123.

[6] C. Chen, Y. Li, W. Liu, and J. Huang, "SIRF: Simultaneous satellite image registration and fusion in a unified framework," *IEEE Transactions on Image Processing*, vol. 24, no. 11, pp. 4213–4224, 2015.

[7] Y. Li, C. Chen, F. Yang, and J. Huang, "Hierarchical sparse representation for robust image registration," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 40, no. 9, pp. 2151–2164, 2018.

[8] J. Huang, C. Chen, and L. Axel, "Fast multi-contrast MRI reconstruction," *Magnetic Resonance Imaging*, vol. 32, no. 10, pp. 1344–1352, 2014.

[9] C. Chen, M. N. Do, and J. Wang, "Robust image and video dehazing with visual artifact suppression via gradient residual minimization," in *Proceedings of the European Conference on Computer Vision*, 2016, pp. 576–591.

[10] Y. Li, F. Guo, R. T. Tan, and M. S. Brown, "A contrast enhancement framework with JPEG artifacts suppression," in *Proceedings of the European Conference on Computer Vision*, 2014, pp. 174–188.

[11] R. A. Yeh, C. Chen, T. Y. Lim, A. G. Schwing, M. Hasegawa-Johnson, and M. N. Do, "Semantic image inpainting with deep generative models," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 5485–5493.

[12] C. Chen, Q. Chen, J. Xu, and V. Koltun, "Learning to see in the dark," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 3291–3300.

[13] Y. Y. Schechner, S. G. Narasimhan, and S. K. Nayar, "Instant dehazing of images using polarization," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, vol. 1, 2001, p. 325.

[14] S. Shwartz, E. Namer, and Y. Y. Schechner, "Blind haze separation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, vol. 2, 2006, pp. 1984–1991.

[15] J. Kopf, B. Neubert, B. Chen, M. Cohen, D. Cohen-Or, O. Deussen, M. Uyttendaele, and D. Lischinski, "Deep photo: Model-based photograph enhancement and viewing," *ACM Transactions on Graphics*, vol. 27, no. 5, p. 116, 2008.

[16] S. G. Narasimhan and S. K. Nayar, "Contrast restoration of weather degraded images," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 25, no. 6, pp. 713–724, 2003.

[17] K. He, J. Sun, and X. Tang, "Single image haze removal using dark channel prior," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, no. 12, pp. 2341–2353, 2011.

[18] G. Meng, Y. Wang, J. Duan, S. Xiang, and C. Pan, "Efficient image dehazing with boundary constraint and contextual regularization," in *Proceedings of the IEEE International Conference on Computer Vision*, 2013, pp. 617–624.

[19] R. Fattal, "Dehazing using color-lines," *ACM Transactions on Graphics*, vol. 34, no. 1, p. 13, 2014.

[20] K. Tang, J. Yang, and J. Wang, "Investigating haze-relevant features in a learning framework for image dehazing," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 2995–3002.

[21] R. Fattal, "Single image dehazing," *ACM Transactions on Graphics*, vol. 27, no. 3, p. 72, 2008.

[22] R. T. Tan, "Visibility in bad weather from a single image," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2008, pp. 1–8.

[23] A. Galdran, J. Vazquez-Corral, D. Pardo, and M. Bertalmio, "Enhanced variational image dehazing," *SIAM Journal on Imaging Sciences*, vol. 8, no. 3, pp. 1519–1546, 2015.

[24] J.-P. Tarel, N. Hautière, L. Caraffa, A. Cord, H. Halmaoui, and D. Gruyer, "Vision enhancement in homogeneous and heterogeneous fog," *IEEE Intelligent Transportation Systems Magazine*, vol. 4, no. 2, pp. 6–20, 2012.

[25] Z. Li, P. Tan, R. T. Tan, D. Zou, S. Zhiying Zhou, and L.-F. Cheong, "Simultaneous video defogging and stereo reconstruction," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 4988–4997.

[26] K. Bredies, K. Kunisch, and T. Pock, "Total generalized variation," *SIAM Journal on Imaging Sciences*, vol. 3, no. 3, pp. 492–526, 2010.

[27] H. Koschmieder, *Theorie der horizontalen Sichtweite.* Beitrge zur Physik der freien Atmosphre, 1924.

[28] A. Levin, D. Lischinski, and Y. Weiss, "A closed-form solution to natural image matting," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 30, no. 2, pp. 228–242, 2008.

[29] K. He, J. Sun, and X. Tang, "Guided image filtering," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 6, pp. 1397–1409, 2013.

[30] M. Werlberger, W. Trobin, T. Pock, A. Wedel, D. Cremers, and H. Bischof, "Anisotropic Huber-L1 optical flow," in *Proceedings of the British Machine Vision Conference*, vol. 1, no. 2, 2009, p. 3.

[31] R. Ranftl, S. Gehrig, T. Pock, and H. Bischof, "Pushing the limits of stereo using variational stereo estimation," in *IEEE Intelligent Vehicles Symposium*, 2012, pp. 401–407.

[32] D. Ferstl, C. Reinbacher, R. Ranftl, M. Rüther, and H. Bischof, "Image guided depth upsampling using anisotropic total generalized variation," in *Proceedings of the IEEE International Conference on Computer Vision*, 2013, pp. 993–1000.

[33] A. Chambolle and T. Pock, "A first-order primal-dual algorithm for convex problems with applications to imaging," *Journal of Mathematical Imaging and Vision*, vol. 40, no. 1, pp. 120–145, 2011.

[34] C. Chen, Y. Li, W. Liu, and J. Huang, "Image fusion with local spectral consistency and dynamic gradient sparsity," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 2760–2765.

[35] Y. Li, C. Chen, F. Yang, and J. Huang, "Deep sparse representation for robust image registration," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 4894–4901.

[36] A. Beck and M. Teboulle, "Fast gradient-based algorithms for constrained total variation image denoising and deblurring problems," *IEEE Transactions on Image Processing*, vol. 18, no. 11, pp. 2419–2434, 2009.

[37] D. Scharstein and R. Szeliski, "A taxonomy and evaluation of dense two-frame stereo correspondence algorithms," *International Journal of Computer Vision*, vol. 47, no. 1-3, pp. 7–42, 2002.

[38] S. Lu, X. Ren, and F. Liu, "Depth enhancement via low-rank matrix completion," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 3390–3397.

[39] N. Bonneel, J. Tompkin, K. Sunkavalli, D. Sun, S. Paris, and H. Pfister, "Blind video temporal consistency," *ACM Transactions on Graphics*, vol. 34, no. 6, p. 196, 2015.

[40] D. Pathak, P. Krahenbuhl, J. Donahue, T. Darrell, and A. A. Efros, "Context encoders: Feature learning by inpainting," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 2536–2544.

[41] M. Bertalmio, G. Sapiro, V. Caselles, and C. Ballester, "Image inpainting," in *Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques*, 2000.

[42] J. Shen and T. F. Chan, "Mathematical models for local nontexture inpaintings," *SIAM Journal on Applied Mathematics*, vol. 62, no. 3, pp. 1019–1043, 2002.

[43] M. V. Afonso, J. M. Bioucas-Dias, and M. A. Figueiredo, "An augmented lagrangian approach to the constrained optimization formulation of imaging inverse problems," *IEEE Transactions on Image Processing*, vol. 20, no. 3, pp. 681–695, 2011.

[44] A. A. Efros and T. K. Leung, "Texture synthesis by non-parametric sampling," in *Proceedings of the IEEE International Conference on Computer Vision*. IEEE, 1999, p. 1033.

[45] K. He and J. Sun, "Statistics of patch offsets for image completion," in *Proceedings of the European Conference on Computer Vision*, 2012, pp. 16–29.

[46] J.-B. Huang, S. B. Kang, N. Ahuja, and J. Kopf, "Image completion using planar structure guidance," *ACM Transactions on Graphics*, vol. 33, no. 4, p. 129, 2014.

[47] Y. Hu, D. Zhang, J. Ye, X. Li, and X. He, "Fast and accurate matrix completion via truncated nuclear norm regularization," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 9, pp. 2117–2130, 2013.

[48] C. Barnes, E. Shechtman, A. Finkelstein, and D. B. Goldman, "Patchmatch: A randomized correspondence algorithm for structural image editing," *ACM Transactions on Graphics*, vol. 28, no. 3, p. 24, 2009.

[49] J. Hays and A. A. Efros, "Scene completion using millions of photographs," in *ACM Transactions on Graphics*, vol. 26, no. 3. ACM, 2007, p. 4.

[50] O. Whyte, J. Sivic, and A. Zisserman, "Get out of my picture! internet-based inpainting." in *Proceedings of the British Machine Vision Conference*, vol. 2, no. 4, 2009, p. 5.

[51] J. Mairal, M. Elad, and G. Sapiro, "Sparse representation for color image restoration," *IEEE Transactions on Image Processing*, vol. 17, no. 1, pp. 53–69, 2008.

[52] J. Xie, L. Xu, and E. Chen, "Image denoising and inpainting with deep neural networks," in *Advances in Neural Information Processing Systems*, 2012, pp. 341–349.

[53] J. S. Ren, L. Xu, Q. Yan, and W. Sun, "Shepard convolutional neural networks," in *Advances in Neural Information Processing Systems*, 2015, pp. 901–909.

[54] S. Liu, J. Pan, and M.-H. Yang, "Learning recursive filters for low-level vision via a hybrid neural network," in *Proceedings of the European Conference on Computer Vision*. Springer, 2016, pp. 560–576.

[55] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Advances in Neural Information Processing Systems*, 2014, pp. 2672–2680.

[56] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," *arXiv preprint arXiv:1511.06434*, 2015.

[57] Z. Liu, P. Luo, X. Wang, and X. Tang, "Deep learning face attributes in the wild," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 3730–3738.

[58] Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, and A. Y. Ng, "Reading digits in natural images with unsupervised feature learning," in *Advances in Neural Information Processing Systems Workshops*, vol. 2011, no. 2, 2011, p. 5.

[59] J. Krause, M. Stark, J. Deng, and L. Fei-Fei, "3D object representations for fine-grained categorization," in *Proceedings of the IEEE International Conference on Computer Vision Workshops*, 2013, pp. 554–561.

[60] E. L. Denton, S. Chintala, R. Fergus et al., "Deep generative image models using a Laplacian pyramid of adversarial networks," in *Advances in Neural Information Processing Systems*, 2015, pp. 1486–1494.

[61] J.-Y. Zhu, P. Krähenbühl, E. Shechtman, and A. A. Efros, "Generative visual manipulation on the natural image manifold," in *Proceedings of the European Conference on Computer Vision.* Springer, 2016, pp. 597–613.

[62] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," *arXiv preprint arXiv:1312.6114*, 2013.

[63] X. Yan, J. Yang, K. Sohn, and H. Lee, "Attribute2Image: Conditional image generation from visual attributes," *arXiv preprint arXiv:1512.00570*, 2015.

[64] A. B. L. Larsen, S. K. Sønderby, H. Larochelle, and O. Winther, "Autoencoding beyond pixels using a learned similarity metric," *arXiv preprint arXiv:1512.09300*, 2015.

[65] L. A. Gatys, A. S. Ecker, and M. Bethge, "Image style transfer using convolutional neural networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition.* IEEE, 2016, pp. 2414–2423.

[66] L. Gatys, A. S. Ecker, and M. Bethge, "Texture synthesis using convolutional neural networks," in *Advances in Neural Information Processing Systems*, 2015, pp. 262–270.

[67] C. Li and M. Wand, "Combining Markov random fields and convolutional neural networks for image synthesis," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 2479–2486.

[68] A. Mordvintsev, C. Olah, and M. Tyka, "Inceptionism: Going deeper into neural networks," *Google Research Blog. Retrieved June*, vol. 20, 2015.

[69] A. Mahendran and A. Vedaldi, "Understanding deep image representations by inverting them," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 5188–5196.

[70] A. Dosovitskiy and T. Brox, "Inverting visual representations with convolutional networks," *arXiv preprint arXiv:1506.02753*, 2015.

[71] K. Simonyan, A. Vedaldi, and A. Zisserman, "Deep inside convolutional networks: Visualising image classification models and saliency maps," *arXiv preprint arXiv:1312.6034*, 2013.

[72] A. Linden and J. Kindermann, "Inversion of multilayer nets," in *Proceedings of the International Joint Conference on Neural Networks*, vol. 2, 1989, pp. 425–430.

[73] L. v. d. Maaten and G. Hinton, "Visualizing data using t-SNE," *Journal of Machine Learning Research*, vol. 9, no. Nov, pp. 2579–2605, 2008.

[74] J. Johnson, A. Alahi, and L. Fei-Fei, "Perceptual losses for real-time style transfer and super-resolution," in *Proceedings of the European Conference on Computer Vision*. Springer, 2016, pp. 694–711.

[75] P. Pérez, M. Gangnet, and A. Blake, "Poisson image editing," *ACM Transactions on Graphics*, vol. 22, no. 3, pp. 313–318, 2003.

[76] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[77] C. Lu, J. Tang, S. Yan, and Z. Lin, "Generalized nonconvex nonsmooth low-rank minimization," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 4130–4137.

[78] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: From error visibility to structural similarity," *IEEE Transactions on Image Processing*, vol. 13, no. 4, pp. 600–612, 2004.

[79] C. Ledig, L. Theis, F. Huszár, J. Caballero, A. Aitken, A. Tejani, J. Totz, Z. Wang, and W. Shi, "Photo-realistic single image super-resolution using a generative adversarial network," *arXiv preprint arXiv:1609.04802*, 2016.

[80] T. Remez, O. Litany, R. Giryes, and A. M. Bronstein, “Deep convolutional denoising of low-light images,” *arXiv preprint arXiv:1701.01687*, 2017.

[81] Z. Hu, S. Cho, J. Wang, and M.-H. Yang, “Deblurring low-light images with light streaks,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 3382–3389.

[82] X. Zhang, P. Shen, L. Luo, L. Zhang, and J. Song, “Enhancement and noise reduction of very low light level images,” in *Proceedings of the International Conference on Pattern Recognition*, 2012, pp. 2034–2037.

[83] T. Plötz and S. Roth, “Benchmarking denoising algorithms with real photographs,” *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017.

[84] S. W. Hasinoff, D. Sharlet, R. Geiss, A. Adams, J. T. Barron, F. Kainz, J. Chen, and M. Levoy, “Burst photography for high dynamic range and low-light imaging on mobile cameras,” *ACM Transactions on Graphics*, vol. 35, no. 6, p. 192, 2016.

[85] Z. Liu, L. Yuan, X. Tang, M. Uyttendaele, and J. Sun, “Fast burst images denoising,” *ACM Transactions on Graphics*, vol. 33, no. 6, p. 232, 2014.

[86] L. I. Rudin, S. Osher, and E. Fatemi, “Nonlinear total variation based noise removal algorithms,” *Physica D: Nonlinear Phenomena*, vol. 60, no. 1-4, pp. 259–268, 1992.

[87] J. Portilla, V. Strela, M. J. Wainwright, and E. P. Simoncelli, “Image denoising using scale mixtures of Gaussians in the wavelet domain,” *IEEE Transactions on Image Processing*, vol. 12, no. 11, pp. 1338–1351, 2003.

[88] M. Elad and M. Aharon, “Image denoising via sparse and redundant representations over learned dictionaries,” *IEEE Transactions on Image Processing*, vol. 15, no. 12, pp. 3736–3745, 2006.

[89] J. Mairal, F. Bach, J. Ponce, G. Sapiro, and A. Zisserman, “Non-local sparse models for image restoration,” in *Proceedings of the IEEE International Conference on Computer Vision*, 2009, pp. 2272–2279.

[90] S. Gu, L. Zhang, W. Zuo, and X. Feng, “Weighted nuclear norm minimization with application to image denoising,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 2862–2869.

[91] K. Dabov, A. Foi, V. Katkovnik, and K. Egiazarian, "Image denoising by sparse 3-d transform-domain collaborative filtering," *IEEE Transactions on Image Processing*, vol. 16, no. 8, pp. 2080–2095, 2007.

[92] F. Agostinelli, M. R. Anderson, and H. Lee, "Adaptive multi-column deep neural networks with application to robust image denoising," in *Advances in Neural Information Processing Systems*, 2013, pp. 1493–1501.

[93] Y. Chen and T. Pock, "Trainable nonlinear reaction diffusion: A flexible framework for fast and effective image restoration," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 6, pp. 1256–1272, 2017.

[94] H. C. Burger, C. J. Schuler, and S. Harmeling, "Image denoising: Can plain neural networks compete with BM3D?" in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2012, pp. 2392–2399.

[95] K. G. Lore, A. Akintayo, and S. Sarkar, "LLNet: A deep autoencoder approach to natural low-light image enhancement," *Pattern Recognition*, vol. 61, pp. 650–662, 2017.

[96] V. Jain and S. Seung, "Natural image denoising with convolutional networks," in *Advances in Neural Information Processing Systems*, 2008, pp. 769–776.

[97] K. Zhang, W. Zuo, Y. Chen, D. Meng, and L. Zhang, "Beyond a Gaussian denoiser: Residual learning of deep cnn for image denoising," *IEEE Transactions on Image Processing*, vol. 26, no. 7, pp. 3142–3155, 2017.

[98] K. Hirakawa and T. W. Parks, "Joint demosaicing and denoising," *IEEE Transactions on Image Processing*, vol. 15, no. 8, pp. 2146–2157, 2006.

[99] M. Gharbi, G. Chaurasia, S. Paris, and F. Durand, "Deep joint demosaicking and denoising," *ACM Transactions on Graphics*, vol. 35, no. 6, p. 191, 2016.

[100] C. Liu and W. T. Freeman, "A high-quality video denoising algorithm based on reliable motion estimation," in *Proceedings of the European Conference on Computer Vision*, 2010, pp. 706–719.

[101] N. Joshi and M. F. Cohen, "Seeing Mt. Rainier: Lucky imaging for multi-image denoising, sharpening, and haze removal," in *Proceedings of the IEEE International Conference on Computational Photography*, 2010, pp. 1–8.

[102] H. Malm, M. Oskarsson, E. Warrant, P. Clarberg, J. Hasselgren, and C. Lejdfors, "Adaptive enhancement and noise reduction in very low light-level video," in *Proceedings of the IEEE International Conference on Computer Vision*, 2007, pp. 1–8.

[103] X. Dong, G. Wang, Y. Pang, W. Li, J. Wen, W. Meng, and Y. Lu, "Fast efficient algorithm for enhancement of low lighting video," in *Proceedings of the IEEE International Conference on Multimedia and Expo*. IEEE Computer Society, 2011, pp. 1–6.

[104] A. Łoza, D. R. Bull, P. R. Hill, and A. M. Achim, "Automatic contrast enhancement of low-light images based on local statistics of wavelet coefficients," *Digital Signal Processing*, vol. 23, no. 6, pp. 1856–1866, 2013.

[105] S. Park, S. Yu, B. Moon, S. Ko, and J. Paik, "Low-light image enhancement using variational optimization-based retinex model," *IEEE Transactions on Consumer Electronics*, vol. 63, no. 2, pp. 178–184, 2017.

[106] X. Guo, Y. Li, and H. Ling, "Lime: Low-light image enhancement via illumination map estimation," *IEEE Transactions on Image Processing*, vol. 26, no. 2, pp. 982–993, 2017.

[107] J. Anaya and A. Barbu, "Renoir-a dataset for real low-light image noise reduction," *arXiv preprint arXiv:1409.8230*, 2014.

[108] G. Wang, C. Lopez-Molina, and B. De Baets, "Blob reconstruction using unilateral second order Gaussian kernels with application to high-ISO long-exposure image denoising," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 4817–4825.

[109] H. Jiang, Q. Tian, J. Farrell, and B. A. Wandell, "Learning the image processing pipeline," *IEEE Transactions on Image Processing*, vol. 26, no. 10, pp. 5032–5042, 2017.

[110] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel, "Backpropagation applied to handwritten zip code recognition," *Neural Computation*, vol. 1, no. 4, 1989.

[111] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 3431–3440.

[112] L. Xu, J. Ren, Q. Yan, R. Liao, and J. Jia, "Deep edge-aware filters," in *Proceedings of the International Conference on Machine Learning*, 2015, pp. 1669–1678.

[113] Q. Chen, J. Xu, and V. Koltun, "Fast image processing with fully-convolutional networks," in *Proceedings of the IEEE International Conference on Computer Vision*, vol. 9, 2017, pp. 2516–2525.

[114] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *International Conference on Medical Image Computing and Computer-assisted Intervention.* Springer, 2015, pp. 234–241.

[115] X. Tao, H. Gao, R. Liao, J. Wang, and J. Jia, "Detail-revealing deep video super-resolution," *arXiv preprint arXiv:1704.02738*, 2017.

[116] S. Su, M. Delbracio, J. Wang, G. Sapiro, W. Heidrich, and O. Wang, "Deep video deblurring for hand-held cameras," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 1279–1288.

[117] J. Kim, J. Kwon Lee, and K. Mu Lee, "Accurate image super-resolution using very deep convolutional networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 1646–1654.

[118] H. Zhao, O. Gallo, I. Frosio, and J. Kautz, "Loss functions for image restoration with neural networks," *IEEE Transactions on Computational Imaging*, vol. 3, no. 1, pp. 47–57, 2017.

[119] W. Shi, J. Caballero, F. Huszár, J. Totz, A. P. Aitken, R. Bishop, D. Rueckert, and Z. Wang, "Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 1874–1883.

[120] R. Gao and K. Grauman, "On-demand learning for deep image restoration," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 1086–1095.

[121] F. Heide, M. Steinberger, Y.-T. Tsai, M. Rouf, D. Pajak, D. Reddy, O. Gallo, J. Liu, W. Heidrich, K. Egiazarian et al., "FlexISP: A flexible camera image processing framework," *ACM Transactions on Graphics*, vol. 33, no. 6, p. 231, 2014.

[122] B. Wen, Y. Li, L. Pfister, and Y. Bresler, "Joint adaptive sparsity and low-rankness on the fly: An online tensor reconstruction scheme for video denoising," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017.

[123] E. Schwartz, R. Giryes, and A. M. Bronstein, “DeepISP: Learning end-to-end image processing pipeline,” *IEEE Transactions on Image Processing*, vol. 28, no. 2, pp. 912–923, 2019.

[124] B. Mildenhall, J. T. Barron, J. Chen, D. Sharlet, R. Ng, and R. Carroll, “Burst denoising with kernel prediction networks,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 2502–2510.

[125] C. Godard, K. Matzen, and M. Uyttendaele, “Deep burst denoising,” in *Proceedings of the European Conference on Computer Vision*, 2018, pp. 538–554.

[126] J. Lehtinen, J. Munkberg, J. Hasselgren, S. Laine, T. Karras, M. Aittala, and T. Aila, “Noise2noise: Learning image restoration without clean data,” in *Proceedings of the International Conference on Machine Learning*, 2018, pp. 2965–2974.

[127] M. Maggioni, G. Boracchi, A. Foi, and K. Egiazarian, “Video denoising, deblocking, and enhancement through separable 4-d nonlocal spatiotemporal transforms,” *IEEE Transactions on Image Processing*, vol. 21, no. 9, pp. 3952–3966, 2012.

[128] M. Lebrun, A. Buades, and J.-M. Morel, “A nonlocal Bayesian image denoising algorithm,” *SIAM Journal on Imaging Sciences*, vol. 6, no. 3, pp. 1665–1688, 2013.

[129] F. Lv, F. Lu, J. Wu, and C. Lim, “MBLLEN: Low-light image/video enhancement using cnns,” in *Proceedings of the British Machine Vision Conference*, 2018, p. 220.

[130] A. Abdelhamed, S. Lin, and M. S. Brown, “A high-quality denoising dataset for smartphone cameras,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 1692–1700.

[131] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” in *Proceedings of the International Conference for Learning Representations*, 2015.

[132] W.-S. Lai, J.-B. Huang, O. Wang, E. Shechtman, E. Yumer, and M.-H. Yang, “Learning blind video temporal consistency,” in *Proceedings of the European Conference on Computer Vision*, 2018, pp. 179–195.

[133] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 770–778.

[134] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard et al., "Tensorflow: A system for large-scale machine learning." in *Proceedings of the12th USENIX Symposium on Operating Systems Design and Implementation*, 2016, pp. 265–283.

[135] D. Sun, X. Yang, M.-Y. Liu, and J. Kautz, "PWC-NET: CNNs for optical flow using pyramid, warping, and cost volume," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 8934–8943.

[136] S. R. Richter, V. Vineet, S. Roth, and V. Koltun, "Playing for data: Ground truth from computer games," in *Proceedings of the European Conference on Computer Vision*, vol. 9906, 2016, pp. 102–118.

[137] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele, "The cityscapes dataset for semantic urban scene understanding," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 3213–3223.

[138] J. Hoffman, E. Tzeng, T. Park, J.-Y. Zhu, P. Isola, K. Saenko, A. A. Efros, and T. Darrell, "CyCADA: Cycle-consistent adversarial domain adaptation," in *Proceedings of the International Conference on Machine Learning*, 2018, pp. 1989–1998.

[139] P. Li, X. Liang, D. Jia, and E. P. Xing, "Semantic-aware grad-GAN for virtual-to-real urban scene adaption," *arXiv preprint arXiv:1801.01726*, 2018.

[140] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, "Unpaired image-to-image translation using cycle-consistent adversarial networks," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 2223–2232.

[141] M.-Y. Liu, T. Breuel, and J. Kautz, "Unsupervised image-to-image translation networks," in *Advances in Neural Information Processing Systems*, 2017, pp. 700–708.

[142] M. Gharbi, J. Chen, J. T. Barron, S. W. Hasinoff, and F. Durand, "Deep bilateral learning for real-time image enhancement," *ACM Transactions on Graphics*, vol. 36, no. 4, p. 118, 2017.