FLORIDA STATE UNIVERSITY

COLLEGE OF ARTS AND SCIENCES

FIRST STEPS TOWARDS IMAGE DENOISING UNDER LOW-LIGHT CONDITIONS

By

JOSUE ANAYA

A Dissertation submitted to the
Department of Statistics
in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy

Josue Anaya defended this dissertation on November 18, 2016.
The members of the supervisory committee were:

Adrian Barbu

Professor Directing Dissertation

Anke Meyer-Baese

University Representative

Antonio Linero

Committee Member

Jinfeng Zhang

Committee Member

The Graduate School has verified and approved the above-named committee members, and certifies that the dissertation has been approved in accordance with university requirements.

To my friends and family for always believing in me

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABSTRACT

The application of noise reduction or perfoming denoising on an image is a very important topic in the field of computer vision and computational photography. Many popular state of the art denoising algorithms are trained and evaluated using images with artificial noise. These trained algorithms and their evaluations on synthetic data may lead to incorrect conclusions about their performances. In this paper we will first introduce a benchmark dataset of uncompressed color images corrupted by natural noise due to low-light conditions, together with spatially and intensity-aligned low noise images of the same scenes. The dataset contains over 100 scenes and more than 500 images, including both RAW formatted images and 8 bit BMP pixel and intensity aligned images. We will also introduce a method for estimating the true noise level in each of our images, since even the low noise images contain a small amount of noise. Through this noise estimation method we develop a convolutional neural network model for automatic noise estimation in single noisy images. Finally, we improve upon a state-of-the-art denoising algorithm Block Matching through 3D filtering (BM3D) by learning a specialized denoising parameter using another developed convolutional neural network.

CHAPTER 1

INTRODUCTION AND MOTIVATION

In the field of computer vision and computational photography, noise reduction is the application in which granular discrepancies found in images are removed. The task of performing noise reduction is synonymous with improvement in image quality. Many consumer cameras and mobile phones deal with the issues of low-light noise due to small sensor size and insufficient exposure. The issue of noise for a particular digital camera is so important that it is used as a valuable metric of the camera sensor and for determining how well the camera performs[20]. Another example of images that deal with noise due to limited acquisition time are Magnetic Resonance Images (MRI). Other important types of image modalities such as Xray and CT (Computed Tomography) also suffer from noise artifacts due to insufficient exposure because of low radiation dose limits. While the image acquisition process is different in all of these examples, the reason for the noise is in most part the same. This is why the problem of low-light image noise reduction is studied and has led to a variety of different noise reduction methods [5, 31, 10, 4, 11, 26, 6, 34, 8].

In general most of the performance evaluations for these various noise reduction methods are done on small images contaminated with artificial noise (Gaussian, Poisson, salt and pepper, etc), which is artificially added to a clean image to obtain a noisy version. Measuring the performance of a noise reduction algorithm on small images corrupted by artificial noise might not give an accurate enough picture of the denoising performance of the algorithm on real digital camera or mobile images in low-light conditions. The nature of the noise in low-light camera images is more complex than just i.i.d., for example its variance depends on the image intensity [13] and has small-range correlations [24], so it would be desirable to obtain images naturally corrupted by low-light noise and their noise-free counterparts. For this purpose, we bring the following contributions:

- A dataset of color images naturally corrupted by low-light noise, taken with two digital cameras (Canon PowerShot S90, Canon EOS Rebel T3i) and a mobile phone camera (Xioami Mi3).
- A process for the collection of noisy and low-noise pixel-aligned images of the same scene.

- A method for aligning the intensity values of all the images of the same scene.
- A technique for computing the noise level and Peak Signal-to-Noise Ratio (PSNR) [35] of the images in our dataset.
- An evaluation of the denoising performance of four algorithms on our dataset, with some surprising results.

Different cameras produce different kinds of noise due to their sensor size, sensor type, and other factors on the imaging pipeline. A learning-based denoising method (e.g. [4, 6] or [8]) could be trained for a specific type of camera on noisy-clean image pairs for that specific camera, but it is not clear how well it would generalize to images from another camera. Trying to construct a dataset of various image pairs from different cameras may help determine which denoising method generalizes well over many different cameras, however it does not evaluate the full potential of a method on any one specific camera at various noise levels. We therefore selected to obtain an equal number of images from three cameras with different sensor sizes: one with a small sensor (Xiaomi Mi3), one with a slightly larger sensor (Canon S90), and one with a mid-size sensor (Canon T3i) to obtain many images with different noise levels for each camera.

CHAPTER 2

RENOIR - A DATASET FOR REAL LOW-LIGHT IMAGE DENOISING

Many various denoising methods [6, 4] train models from the noisy-clean image pairs that are supposed to generalize well to future noisy images. For this reason and for evaluation in general it is very important to maintain a careful construction of these noisy-clean image pairs and to have many examples for a representative dataset. The difficulty in constructing such pairs is why artificial noise is used in practice.

2.1 Related Works

Although a variety of color image databases exist, such as [30], the only database for benchmarking image denoising is [12] (discussed in [11]), which evaluates various denoising methods on color images corrupted by artificial Gaussian noise. The problem with artificial Gaussian noise is that it is a very simple noise model that is not present in real world images where many times the noise level changes with the image intensity.

Furthermore, it has been shown that the distribution of low-light camera noise is not Gaussian, but follows a more complex Poisson-Gaussian mixture distribution with intensity dependent variance [13, 25]. Work by both [13] and [24] use a few real low-light noise and clean image pairs to study the noise and intensity relationship. However, we are not aware of any public database or collection of images that have been corrupted by real low-light noise like the ones presented in this dissertation, and which took all the necessary steps for carefully acquiring the images, intensity aligning them and diagnosing the quality of the obtained pairs.

A dataset with real low-light noisy images and their clean counterparts would bring many benefits to just using images artificially corrupted by noise from a Poisson-Gaussian mixture model:

- It would contribute to the further study of the noise structure in digital cameras, such as how much it differs from the Poisson-Gaussian mixture model, spatial correlation structure, how noise parameters relate to camera acquisition parameters, etc.

- It would present a more realistic range of levels of noise, similar to what happens "in the wild". In contrast, Poisson-Gaussian noise model is usually studied with fixed (and known) noise parameters, such as in [28].
- It would allow for an end-to-end evaluation of denoising algorithms. Evaluating using artificial noise models, even if they are accurate, might not entirely reflect the reality of noise in digital cameras.

2.2 Acquisition of Natural Image Pairs

The dataset[1] acquired in this dissertation consists low-light uncompressed natural images of multiple scenes. About four images per scene were acquired, where two images contain noise and the other two images contain very little noise. The presence of noise in the images is mainly due to the number of photons that are received by the camera's sensor and the amplification process, as shown in [17].

All the images in our dataset are of static scenes and are acquired under low-light conditions using the following "sandwich" procedure:

- A low-noise image is obtained with low light sensitivity (ISO 100) and long exposure time. This will be the *reference* image.
- One or two noisy images are then obtained with increased light sensitivity and reduced exposure time.
- Finally, another low noise image is taken with the same parameters as the reference image. This will be the *clean* image.

The two low-noise (reference and clean) images are acquired at the beginning and at the end of the sequence, while the one or two noisy images are shot in between. This is done to evaluate the quality of the whole acquisition process for that particular scene. This process is very similar to the process discussed in [24] which used pair images that were taken with flash. The problem with taking the images with flash is that the flash can change the scene illumination. Moreover, in [24] no brightness alignment has been performed on their images. Any motion or lighting change during acquisition could make the two low-noise images be sufficiently different, as measured by the PSNR. In fact, we discarded the scenes with PSNR of the clean images less than 34.

The actual acquisition parameters for each camera are presented in Table 2.1.

Table 2.1: ISO (and Exposure time) per camera

	Reference/Clean Images		Noisy Images	
Camera	ISO	Time(s)	ISO	Time(s)
Mi3	100	auto	1600 or 3200	auto
S90	100	3.2	640 or 1000	auto
T3i	100	auto	3200 or 6400	auto

Using the Canon Developer Tool Kit for the Canon S90 and the EOS Utility for the Canon Rebel T3i we were able to program the automatic collection of the four images while trying to preserve the static scene in the images by not moving or refocusing the camera. The sandwich approach that we used to obtain our images also helped insure that the only visual difference between the images of a scene was simply due to noise. All of the images were collected and saved in RAW format (CR2 for Canon). The Mi2Raw Camera app was used to capture the RAW images for the Xiaomi Mi3 (in DNG format).

An example of one of the images in the dataset can be seen in Figure 2.1. In the end we collected 40 scenes for the S90, 40 for the T3i, and another 40 for the Mi3. The adaptive homogeneity directed demosaicing algorithm [15] was used for interpolating the pixel data values when constructing the BMP images from the RAW images. The RAW images for all three cameras are included in the dataset for researchers that are interested in experimenting with the demosaicing process, such as [29, 16].

The image denoising database in [12] contains 300 noisy images at 5 noise levels (σ = 5, 10, 15, 25, and 35) for a total of 1500 images. The dimensionality of these images is 481 by 321 . The dimensionality of the images for just the S90 images is 3684 by 2760 while the images from the other cameras are even larger, as shown in Table 2.2. Although our image database contains far fewer noisy images, our images contain about 60 times more pixels and therefore more patch variability for studying noise models from just one of the three cameras.

Many various denoising methods [4, 6, 8] train models from noisy-clean image pairs that are supposed to generalize well to future noisy images. For this reason and for evaluation in general it is very important to maintain a careful construction of these noisy-clean image pairs and to have many examples for a representative dataset. The difficulty in constructing such pairs is why artificial noise is used in practice.

[1] Available at `http://stat.fsu.edu/~abarbu/Renoir.html`

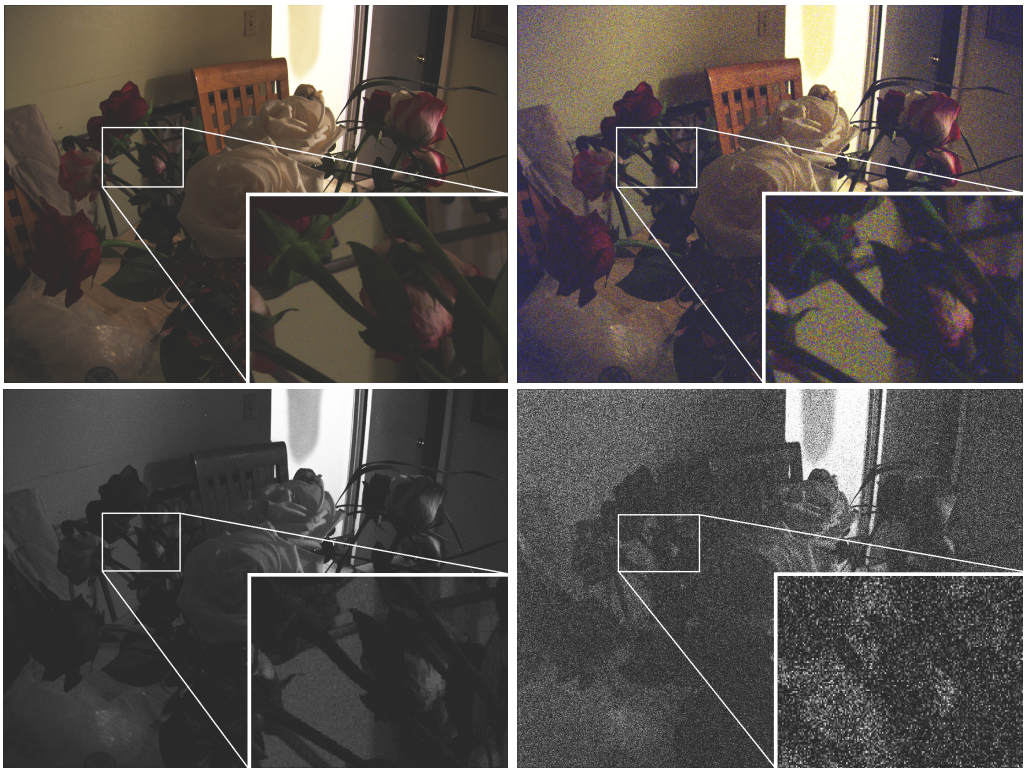

Figure 2.1: An example of a clean and noisy image pair as well as their corresponding blue channel. The noise present is a result of the low-light environment. The images were taken using a Canon PowerShot S90.

2.2.1 Mobile Camera Difficulties and Complexities

In trying to collect images for our dataset we decided on also collecting mobile phone camera images. In doing so we ran into a variety of difficulties, some of which can be seen in Figure 2.2.

The first difficulty that arose was collecting the RAW images of these phone cameras. Only a few mobile phones collect true RAW images (data dump directly from the sensor). For example, the Iphone can have an application installed that will allow it to take RAW images, however these

Table 2.2: Description of the dataset and size

				Noisy Images				Clean Images			
Camera	RAW Image Size	Sensor Size(mm)	# of Scenes	σ Avg.	PSNR Min.	PSNR Avg.	PSNR Max.	σ Avg.	PSNR Min.	PSNR Avg.	PSNR Max.
S90	3684×2760	7.4×5.6	51	18.249	17.429	26.187	33.388	3.067	35.032	38.698	43.425
T3i	5202×3465	22.3×14.9	40	11.714	18.938	27.442	35.256	2.568	34.983	40.426	48.128
Mi3	4208×3120	4.69×3.52	40	19.227	12.751	23.492	36.678	3.712	33.495	37.093	45.252

RAW images are not in fact truly RAW because the sensor data has gone through some unknown post processing. Therefore, only some of the most recent phones that have been allowed by the device manufacturer can truly collect RAW images.

The second difficulty that we found when trying to collect mobile phone images came in the control over certain image acquisition parameters such as the exposure time and ISO values. These mobile phone cameras already have a very small sensor, so when we tried to use an LG Google Nexus 5 with the FV-5 camera application to capture RAW images, the limits of control over settings like the exposure time and ISO, and its tiny sensor size led to many scenes failing our 'sandwich' procedure selection benchmark (did not have sufficient amount of light needed for the PSNR to be around 35 for the reference and clean image.) We also noted for a phone like the LG Google Nexus 5 a non-linearity relationship issue in the brightness alignment procedure (this could also be due to an insufficient amount of light.)

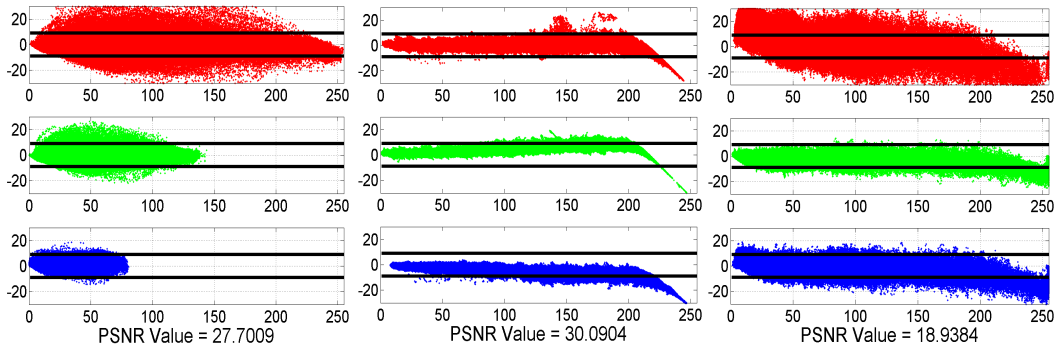

Figure 2.2: Examples of alignment issues observed on scatter plots of the intensity difference between the aligned reference image and noisy image vs reference image intensity. The horizontal line in the plots are 95% noise bounds for images with at least PSNR =35. Left: an example of movement during the 'sandwich' procedure. Middle: an example of light saturation during the 'sandwich' procedure. Right: an example of an image with too much noise and in need of some nonlinear transformation for proper brightness alignment.

The final difficulty we experienced came in the form of tools to help maintain a static scene. With the other cameras we used a tripod and were able to program the automatic acquisition of the scene. With the mobile phone camera we had to use a small phone tripod and a bluetooth mouse to preserve the static scene when taking the images manually.

Settling on the Xiaomi Mi3 phone we collected 6 images per scene. The first two images were both low-noise images and these images were averaged and set as the reference image in the

alignment process. Similarly, the last two images were also both low-noise images and the last two images as well were averaged and used in the overall PSNR computation of the 'sandwich' procedure. If any movement or saturation was detected the images were cropped appropriately post alignment. In the end many of the scenes for the Mi3 were cropped, but all are static with PSNR around 35 or more.

2.3 Main Assumptions and Notations

In this section we present the main assumptions that form the basis of the acquisition procedure, intensity alignment, and noise level estimation.

The following notations will be used in this dissertation:

- R, I^r – the reference image
- C, I^c – the clean image
- N, I^n – the noisy image
- X – one of the clean or noisy images
- GT, I^{GT} – the unknown ground truth image
- $\epsilon, \epsilon_r, \epsilon_c$ – random variables for noise in the low-noise images
- ϵ_n – random variable for noise the noisy images
- $\sigma^2(X) = \text{var}(X)$ the variance of a random variable X

We assume that the two low-noise images I^r (reference) and I^c (clean) as well as the noisy image(s) I^n (acquired with the "sandwich" procedure from Section 2.2) are all noisy versions of a common (unknown) ground truth image I^{GT}, corrupted by zero-mean independent noise. Thus:

$$\begin{aligned} I^n(x,y) &= I^{GT}(x,y) + \epsilon_n(x,y) \\ I^r(x,y) &= I^{GT}(x,y) + \epsilon_r(x,y) \\ I^c(x,y) &= I^{GT}(x,y) + \epsilon_c(x,y) \end{aligned} \tag{2.1}$$

where $\epsilon_n(x), \epsilon_r(x)$, and $\epsilon_c(x)$ are zero-mean and independent.

We also assume that the reference and clean images have the same noise distribution since the two images are of the same static scene with the same ISO level and exposure time. Note

that the reference and clean images have low amounts of noise because many photons have been accumulated on the sensor during the long exposure time.

In summary our assumptions are:

1. The images are formed as described in eq. (2.1) with

$$E[\epsilon_n(x)] = E[\epsilon_r(x)] = E[\epsilon_c(x)] = 0.$$

2. For any x, the random variables $\epsilon_n(x), \epsilon_r(x), \epsilon_c(x)$ are independent.

3. For any x, $\epsilon_r(x)$ and $\epsilon_c(x)$ are identically distributed.

It is shown in [24] that the noise in the digital camera images has short range correlations. We don't need to make any assumptions about the spatial correlations inside one image, just between the three images at the same location.

We will see in experiments that our estimation method based on these assumptions works very well in estimating the noise level in images.

2.4 Intensity Alignment

The dataset construction went beyond just the acquisition of the images. For the purposes of properly aligning the pixel intensities of the image pairs we developed a new form of brightness adjustment that mapped our RAW images to an 8-bit uncompressed format.

The reference image was first mapped from 16-bit to 8-bit as follows. We computed the cumulative distribution of the 16-bit pixel intensities of the RAW reference image and constructed a linear scaling of the RAW reference image that sets the 99th percentile value to the intensity value 230 in the 8-bit image. Thus 1% of the pixels are mapped to intensities above 230, and even fewer will be saturated to value 255. We chose the value 230 so that most of the noisy images will not have much saturation after alignment with the reference image.

Each of the other images of the same scene is at the same time reduced to 8-bit and aligned to the 8-bit reference image by finding a linear mapping specified by parameter α such that if I is the 16-bit image, the 8-bit aligned image is obtained from αI after its values larger than 255 or less than 0 are truncated. For better accuracy, instead of working with the two images I and R, we use blurred versions $\tilde{I}$ and $\tilde{R}$ obtained by convolution with a Gaussian kernel of $\sigma = 5$ to

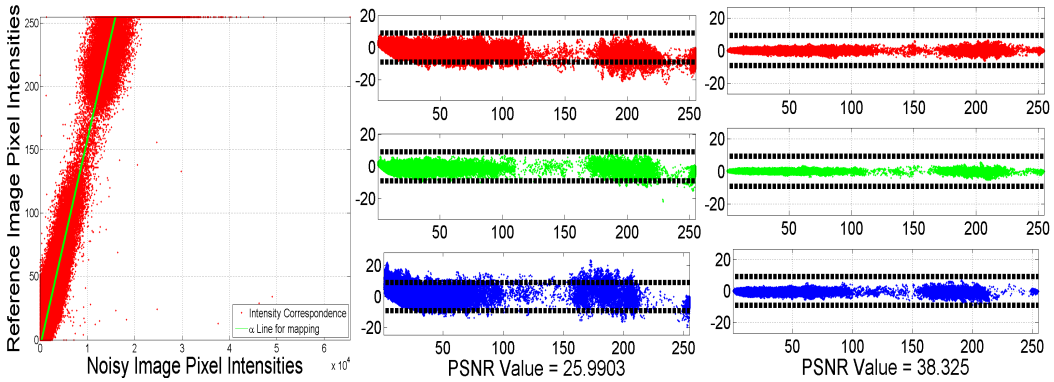

Figure 2.3: Scatter plots of the pixel intensity correspondence between a reference image and its noisy counterpart. Left: the correspondence between the red channel of the 8-bit reference image and the red channel for the 16-bit noisy image. The line shows the estimated linear mapping to align the noisy image to the reference image. Middle: the difference between corresponding pixel intensities of the reference 8-bit and aligned noisy 8-bit image vs reference image intensities for all three color channels. Right: the difference between corresponding pixel intensities between the reference 8-bit and the aligned clean 8-bit image vs reference image intensities.

estimate the intensity alignment parameter α. This way the level of noise is reduced. To avoid biases obtained near intensity discontinuities, the alignment parameter is computed based on the low gradient pixels $M = \{i, |\nabla \tilde{R}(i)| < 1\}$.

The parameter α is found to minimize

$$E(\alpha) = \sum_{i \in M} (\tilde{R}(i) - \max[\min(\alpha \tilde{I}(i), 255), 0])^2$$

This is done by coordinate optimization using the Golden section search in one dimension [32] optimizing on α until convergence. The parameter α obtained for the mapping is robust to outliers.

In Figure 2.3, left is shown an example of the correspondence between the pixels of the 8-bit reference image R and the RAW noisy image I of the same scene, with the alignment line parameterized by α superimposed. The alignment of the obtained 8-bit image I with the 8-bit reference can be diagnosed by plotting the intensity difference $\tilde{I} - \tilde{R}$ (blurred versions) vs the reference intensity $\tilde{R}$. This is shown in Figure 2.3, middle for the noisy image and Figure 2.3, right for the clean image. We obtained plots like Figure 2.3 for all the images in the dataset as a way of diagnosing any misalignment or nonlinear correspondence between the reference image and the corresponding noisy or clean images. The dark dashed horizontal line in the middle and right plots

are 95% noise bounds for clean images with at least PSNR = 35. Figure 2.4 shows an example of the green channel for a particular image in the dataset before and after alignment is performed.

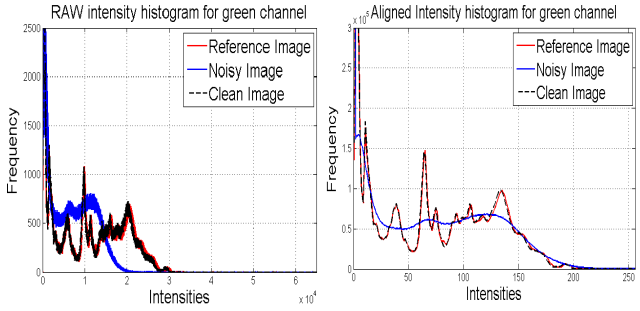

Figure 2.4: An example of the pixel intensity histogram for the Clean and Noisy Green Channels before and after our brightness alignment.

2.5 Dataset Information

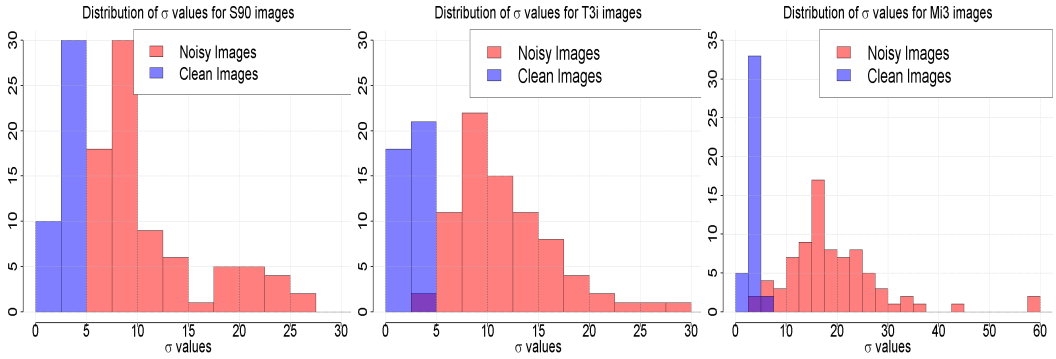

Figure 2.5: Frequency distributions of various noise levels for the noisy and clean images obtained from the S90, T3i, and Mi3 cameras respectively.

Aside from estimating the noise level in every image, we also quantified the image fidelity across the image batches using various metrics such as PSNR [35], SSIM [36], and VSNR [7]. In particular we modified the PSNR measurement by incorporating our estimate of the noise from (2.3) as opposed to using the standard noise estimate from the difference image between a clean and noisy image pair. Although there exist specialized metrics for low-light conditions such as [2] we decided to use measures that are the most prevalent and common in practice.

Table 2.2 lists some specific characteristics about the various cameras and their images in the dataset. Note that the σ in Table 2.2 comes from the estimates from equations (2.2) and (2.3).

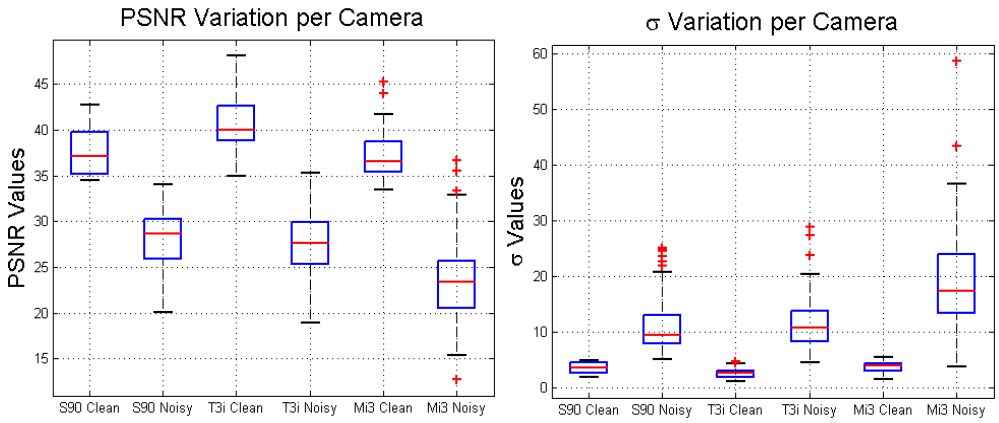

Figure 2.6: Variation of PSNR and σ values for noisy and clean images for each camera.

Figure 2.5 shows the distribution of noise levels for the noisy and clean images for each camera. Figure 2.6 shows box-plots of the variation in PSNR and noise levels for each camera.

One interesting observation that we draw from our dataset is that the low noise images still have a noise level σ of about 3 and as high as 5, which is invisible to the eye. It tells us something about the local nature of the manifold of natural image patches, that the manifold is "thick" in the sense that perturbing a patch with (probably even Gaussian) noise of $\sigma \leq 5$ obtains another natural image patch. Such information could be useful for people that study natural image statistics or learn generative models from natural images (e.g. autoencoders).

2.6 Experiments

As stated previously the amount of noise present in the dataset is due to the sensor and the amplification process. The fact that not all of the images were taken in the same environment under the same camera settings means that we have a wide variety of noise in our images. The fact that we are not dealing with artificial noise also means that we do not know beforehand what will be the noise variance σ^2. Thankfully our "sandwich" procedure for image acquisition, as influenced by [14, 24], allows us to estimate the noise level for any one of our images. The noise level can be estimated locally for an image patch or globally for the entire image.

We will use the fact that if two random variables A, B are independent, then $\text{var}(A - B) = \text{var}(A) + \text{var}(B)$, or in other words $\sigma^2(A-B) = \sigma^2(A) + \sigma^2(B)$ where $\text{var}(A), \sigma(A)$ are the variance

and standard deviation of A respectively. Then from equation (2.1) we get

$$\sigma^2(I^r(x) - I^c(x)) = \text{var}(\epsilon_r(x) - \epsilon_c(x)) = \text{var}(\epsilon_r(x)) + \text{var}(\epsilon_c(x)) = 2\sigma^2(\epsilon(x))$$

from the independence of $\epsilon_r(x)$ and $\epsilon_c(x)$ and the fact that $\epsilon_r(x)$ and $\epsilon_c(x)$ are identically distributed (so we can represent them as $\epsilon(x)$). We obtain the estimation of the noise level in the clean and reference images:

$$\sigma^2(I^r(x) - I^{GT}(x)) = \sigma^2(I^c(x) - I^{GT}(x)) = \sigma^2(\epsilon(x)) = \frac{1}{2}\sigma^2(I^r(x) - I^c(x)) \tag{2.2}$$

For the noisy images we use

$$\sigma^2(I^n(x) - I^r(x)) = \text{var}(\epsilon_n(x) - \epsilon_r(x)) = \sigma^2(\epsilon_n(x)) + \sigma^2(\epsilon_r(x))$$

to obtain the estimation of the noise level as

$$\sigma^2(\epsilon_n(x)) = \sigma^2(I^n(x) - I^{GT}(x)) = \sigma^2(I^n(x) - I^r(x)) - \frac{1}{2}\sigma^2(I^r(x) - I^c(x)) \tag{2.3}$$

If we want to use the best estimate of the GT, which is $I^a(x) = (I^r(x) + I^c(x))/2$, then we have an alternative formula for the noise level in the noisy images

$$\sigma^2(\epsilon_n(x)) = \text{var}(I^n(x) - I^{GT}(x)) = \text{var}(I^n(x) - I^a(x)) - \frac{1}{4}\text{var}(I^r(x) - I^c(x)) \tag{2.4}$$

We can use equations (2.2) and (2.3) to estimate the true noise level for any image in our dataset. Again, these noise levels can be computed globally for the whole image or locally on a patch basis.

2.6.1 Evaluation of the Intensity Alignment and Noise Estimation Using Artificial Noise

To evaluate our noise estimation method we constructed scenes with added artificial noise, as illustrated in Figure 2.7. For this we chose ten random 16-bit RAW reference images from the three digital cameras and used them as ground truth images I^{GT} for constructing artificial sequences from them. We then used our alignment method as described in 2.4 to construct an 8-bit version of I^{GT}. We then generated I^r,I^n, and I^c by adding artificial Gaussian noise to the 16-bit I^{GT}. For 16-bit I^r and I^c we added $\sigma = \frac{3}{\gamma}$ amount of noise where γ is the multiplications factor to map the 16-bit I^{GT} to an 8-bit I^{GT}. A 16-bit I^n was generated using $\sigma = \frac{10}{\gamma}$. This way the standard deviation

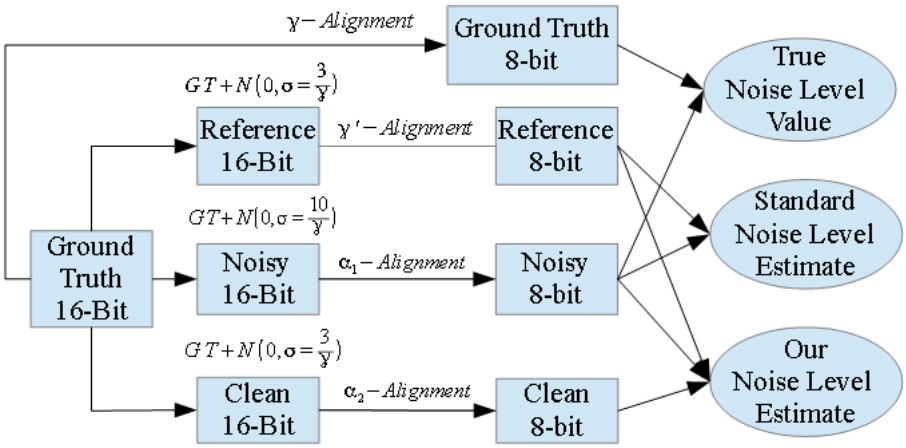

Figure 2.7: The process for constructing the proper reference, clean, noisy, and ground truth images necessary for the noise estimation evaluation. The values of γ', α_1, and α_2 represent the usual alignment of those respective images from 16-bit to 8-bit as defined in section 2.4.

of the difference from the 8-bit I^{GT} to the 8-bit I^r (or I^c) will be 3 and to the 8-bit I^n it will be 10. We then performed our standard alignment on I^r,I^n, and I^c to map them over to 8-bit images obtaining parameters $\gamma', \alpha_i, \alpha_2$ as illustrated in Figure 2.7.

Observe that multiplying the alignment parameters $\gamma', \alpha_i, \alpha_2$ by the same factor produces another alignment of the same quality, thus the alignment is only identifiable up to a constant. For this reason, the alignment is evaluated indirectly, through the quality of the noise level estimation.

The true values of the noise levels for I^r,I^n, and I^c can be computed as the standard deviation of the difference image between each of them and I^{GT} (all in 8-bit versions). Our noise level estimation method for I^c and I^n described in Section 2.6 is compared with the true noise level to obtain the relative error (defined as estimation error divided by the true noise level value). The same type of relative error is also computed for the standard estimate of the noise as $\sigma(I^n - I^r)$ and $\sigma(I^c - I^r)$.

Figure 2.8 shows the relative error (defined as error divided by the true value) of estimating the noise level for both I^n and I^c . When it came to estimating the noise level I^n our method of estimation kept the relative error to below 0.5%, while the standard method of estimating the noise level had a relative error around 5%.

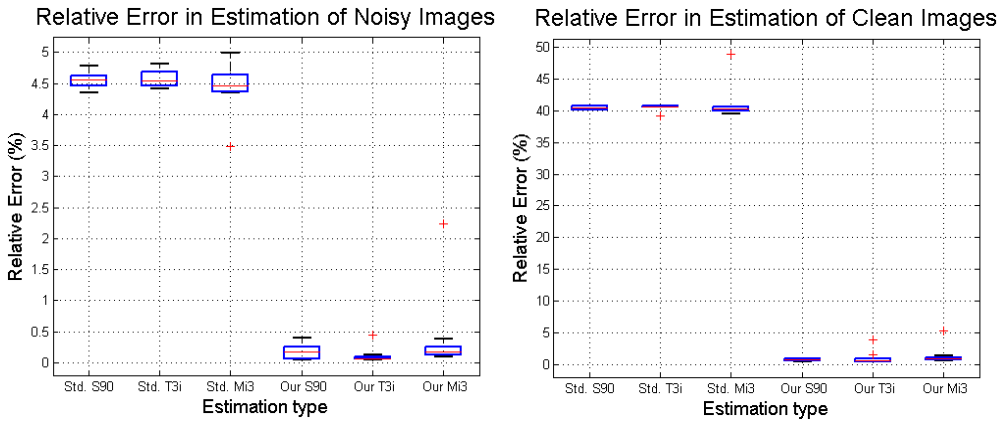

Figure 2.8: The relative error in estimating noisy and clean images.

For all but four of the 120 images evaluated, the relative estimation error for our method was below 1%. This gives us confidence that our intensity alignment method together with the proposed noise level estimation method provide an accurate estimation of the true noise level in images, at least on data with artificial noise.

2.6.2 Evaluation of the Noise Estimation Using Real Noise

To further investigate how well the assumptions we made in Section 2.3 about the noise hold, we acquired a special scene with the S90 camera. The scene was of a constant intensity surface in low-light settings. Using our intensity alignment methodology, instead of mapping our clean image from the 99th quantile to intensity 230; we mapped the median to intensity 128. Using this mapping we then aligned the other two noisy and the clean image using the Golden section method, as described in Section 2.4. Figure 2.9 shows the alignments of the calibration dataset as well as a histogram of pixel difference between the reference image and the other images in the calibration dataset.

Because we know that the I^{GT} was constant since the scene contained a constant intensity surface, we can immediately obtain a true value for σ^2 for each image by directly computing the intensity variance in the image. However, to account for smoothly changing illumination, we constructed a GT version for each image by Gaussian blurring it with a large spatial kernel ($\sigma = 20$) and then calculated the noise level as the variance of the difference image between the original image and its smoothed version. We then looked to see if the standard estimate of using the difference

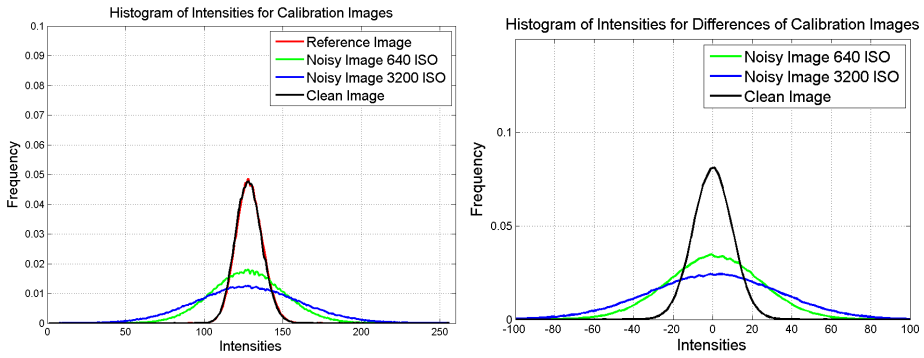

Figure 2.9: Analysis of the calibration images. Left: the intensity histograms of the green channels of the calibration images. Right: the distribution of the intensity difference between the reference image and the various other images in the calibration dataset.

image between the reference image and the other calibration images provided similar results to those we obtained using our methodology from equations (2.2) and (2.3). Analysis of the estimated noise levels for the three image channels and the overall estimate are summarized with boxplots in Figure 2.10.

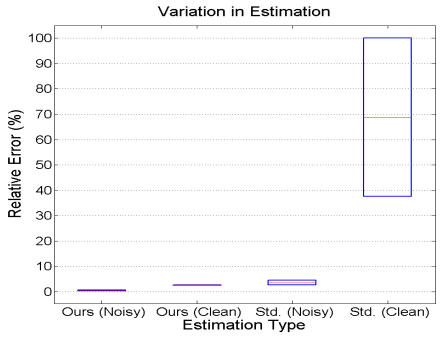

Figure 2.10: Comparison between our method of estimating σ and the standard method based on the difference image. Both methods were tasked with estimating the σ for the red, green, blue channels, and the overall image for the calibration scene.

As Figure 2.10 shows our estimated σ values are less biased and have smaller variance than the standard estimation of σ from the difference images. The average relative error for our method of estimation is 1.58% and for the standard method of estimation is 36.22%. The results that we obtained for this evaluation are in line with the results we obtained for noise estimation for images with artificial noise. Thus our investigation gives us enough confidence in our estimation going

forward. Consequently, the noise estimation described in Section 2.6 will be used as our noise estimation method for all of the images in our dataset and for estimating the PSNR of the denoised images.

2.6.3 Evaluation of the Poisson-Gaussian Noise Model

As stated previously the Poisson-Gaussian noise model [13, 28] depends on the image intensity. Using our notation introduced in Section 2.3, the observed image intensity $I^n(x)$ is represented as

$$I^n(x) = \alpha p(x) + n(x) \tag{2.5}$$

where $p(x)$ is an independent Poisson random variable with expected value $y(x) = I^{GT}(x)/\alpha$ and $n(x)$ is an i.i.d. Gaussian $n(x) \sim N(0, \tau^2)$. This way we obtain the noise model

$$\epsilon_n(x) = I^n(x) - I^{GT}(x) = \alpha p(x) + n(x) - I^{GT}(x) \tag{2.6}$$

which is independent and has zero mean. Therefore the Poisson-Gaussian noise model obeys the assumptions made in Section 2.3.

Under the Poisson-Gaussian noise model the noise level (standard deviation) has an exact relationship with the noise-free image through $\sigma(\epsilon_n(x)) = \sqrt{\alpha I^{GT}(x) + \tau^2}$.

In [13] is presented a maximum likelihood estimate of the Poisson-Gaussian noise parameters (α, b), (where $b = \tau^2$) from a single image. The authors also observe that b could also be negative due to the pedestal level, a constant offset from zero of the digital imaging sensor.

Using local noise estimation through equations (2.2), (2.3), and (2.4), we can calculate the variance v_i of the noise in an image for each patch as well as the intensity level I_i^{GT} of that patch and find the parameters (α, b) so that $v_i = \alpha I_i^{GT} + b$ by the maximum likelihood method from [13]. We can then see how well the Poisson-Gaussian noise model fits our data and we can we can compare our model parameters with the single-image parameter estimates from [13].

A special scene was acquired for this purpose using a uniform background with a smoothly changing intensity and our "sandwich" procedure. We converted the images to gray-scale to be able to compute the model from [13] and divided the image into 400×400 blocks and the blocks into intensity level sets of a smoothed image, following the method described in [13]. Based on the different ways to estimate the noise variance σ in each level set we considered the following variants:

- In the Foi model, the noise level σ is estimated as the standard deviation of the wavelet detail coefficients z^{wdet}, as described in [13].
- In the three image model, the noise level σ is estimated using three images (reference, clean and noisy) and equations (2.2) and (2.4).
- In the two image model, the noise level σ is estimated as the standard deviation of the difference between the reference and the noisy image.
- In the blurred reference model, the noise level σ is estimated as the standard deviation of the difference between the noisy image and the blurred reference image. The blurred reference image was obtained by blurring the reference image with a large Gaussian kernel to better approximate the smooth ground truth image.

For the Foi method we used as smooth image for obtaining the level sets the blurred z^{wapp} wavelet approximation image. For the three image model we obtained the level sets from the average $(I^r + I^c)/2$ between the reference and clean images. For the other two methods we used the blurred reference as the smooth image for the levels sets.

From the obtained $(\hat{y}_i, \hat{\sigma}_i)$ intensity-noise level pairs we fitted Poisson-Gauss model parameters (α, b) by maximum likelihood as described in [13]. The obtained curves are shown in Figure 2.11.

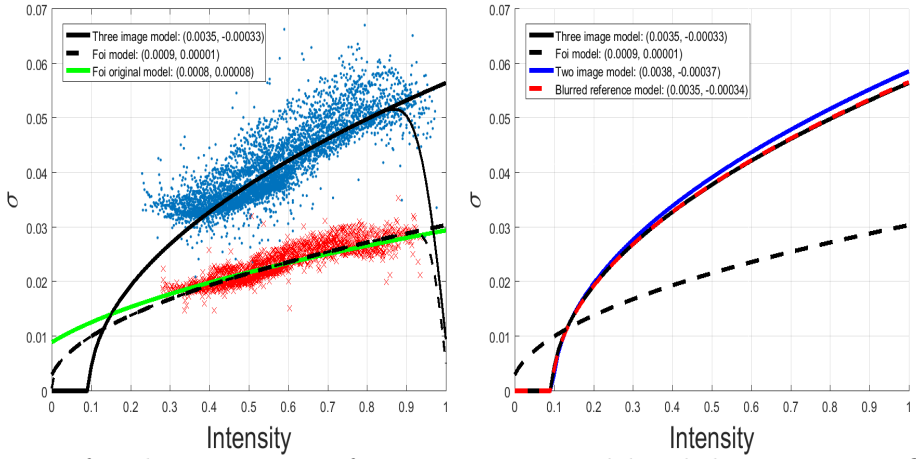

Figure 2.11: Left: The noise curve of our pair image model and the Foi Poisson-Gaussian Mixture model. Right: The noise curves for various noise estimation models using image pairs and the Foi Poisson-Gaussian Mixture model.

In Figure 2.11, left are also shown the data points $(\hat{y}_i, \hat{\sigma}_i)$ from which the three image model and the Foi model were obtained, as well as the Foi original model parameters that was from the noisy image using .

It can be immediately noted that the Foi estimation method is consistently below the other three curves, which are quite close to each other. At high intensities (around 0.9 on the normalized scale), Foi's estimate and the corresponding data points are underestimating the noise level by about 40%. This is possibly due to the local correlations in the image noise, which interfere with the noise estimation based on wavelets and a simple convolution.

Foi's Poisson-Gaussian noise model is using just the noisy image to infer the noise level in the image. With our data acuisition methodology we have both clean and noisy images and are able to infer more accurately the noise level in the image. This is why in Figure 2.11, right we can see all the three estimation methods based on our data are very close to each other and the Foi estimation is quite far. Also, note that this evaluation was on a special scene of a uniform background of continuously changing intensity and no edges. Foi's estimation method would have more difficulty in estimating the noise curve in images with a lot of edges or a lot of textures. At the same time, the three image approach only used the aligned images and no blurring, so it should work as well when edges are present.

CHAPTER 3

SINGLE IMAGE NOISE ESTIMATION USING A CONVOLUTIONAL NEURAL NETWORK

Until recently, noise level estimation was studied for only the artificial Gaussian noise case as seen in both [1] and [21]. It was not until [22] and further extended in [23] that a noise level function could be estimated from a single noisy image. This model presented in [23] used only local statistics from patches of an image to estimate the noise level function. In this chapter we will be training a convolutional neural network (CNN) on the RENOIR dataset and then we will show the performance our noise level estimation model has on low-light noisy images.

3.1 CNN Architecture

In computer vision CNNs have shown a lot of incredible results for highly challenging data such as [19] and [9]. For our application of noise estimation we used Cuda Convnet[1] to develop our CNN. We first preprocessed our data by dividing the noisy images RGB channels by their respective standard deviations. This was the only form of preprocessing done to the data and this was only done for later evaluations. To prevent any possible over fitting we trained the model to predict the noise level for independent RGB channels. The first convolutional layer filters the 32 x 32 x 1 input image patches with 16 kernels of size 5 x 5 x 1 with 1 pixel stride and the output neurons are fitted with rectified linear units (ReLU). Two fully connected layers a 64 and 2048 then take the input of the normalized and pooled output of the first convolutional layer. The overall architecture of our CNN model can be observed in Figure 3.1.

3.1.1 Training Details

We trained our CNNs using stochastic gradient descent with momentum of 0.9 and weight decay of 0.0001. Only the convolutional layer had shared biases enabled and padding was set to 2 to minimize border effects. Our initial learning rate was set to 0.1 and after 15 epochs the learning

[1] 1 http://code.google.com/p/cuda-convnet/

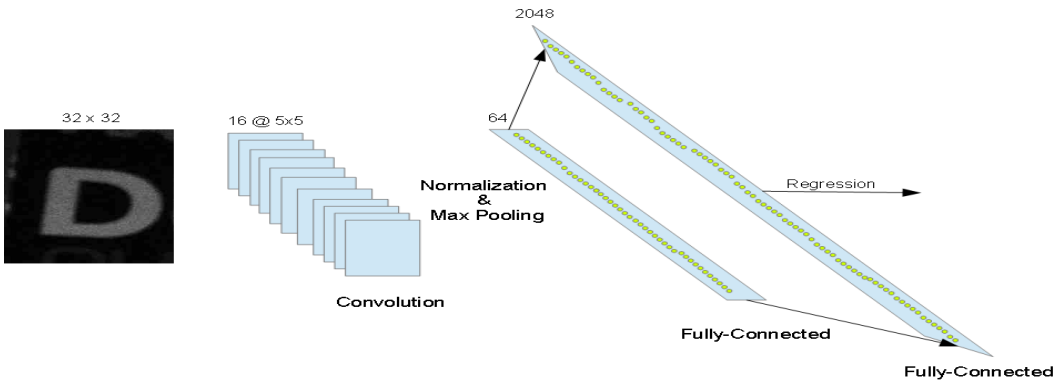

Figure 3.1: An example of our CNNs architecture. It contains a single convolution layer and two fully connected layers. A sum of squares cost layer is used for regression predictions of the noise level for the input patch.

rate was reduced to 0.001. The models trained for a total of 25 epochs using over 3 million image patches per camera. Using a 12 GB NVidia Tesla K40c GPU the training per camera took about 5 minutes.

3.2 Performance on the RENOIR Data

The first experiment we conducted was to see how well our CNN model would predict the noise level for each camera in the RENOIR dataset. To examine how well the CNNs trained we performed 4-fold cross validation by leaving out ten different image batches for each fold.

As stated earlier we preprocessed the data by dividing the images into color channels and the color channels into patches of size 32×32. For each color channel we normalize the per-patch noise estimates by dividing to the standard deviation of the entire color channel. This way the noise level can be recovered by multiplying the prediction with the standard deviation of the color channel.

To evaluate the model fitting we use the standard goodness of fit measure

$$R^2 = 1 - \frac{SS_{res}}{SS_{tot}} = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \overline{y})^2} \tag{3.1}$$

where

- y_i is the true noise in the image patch divided by the standard deviation of its respective noise channel

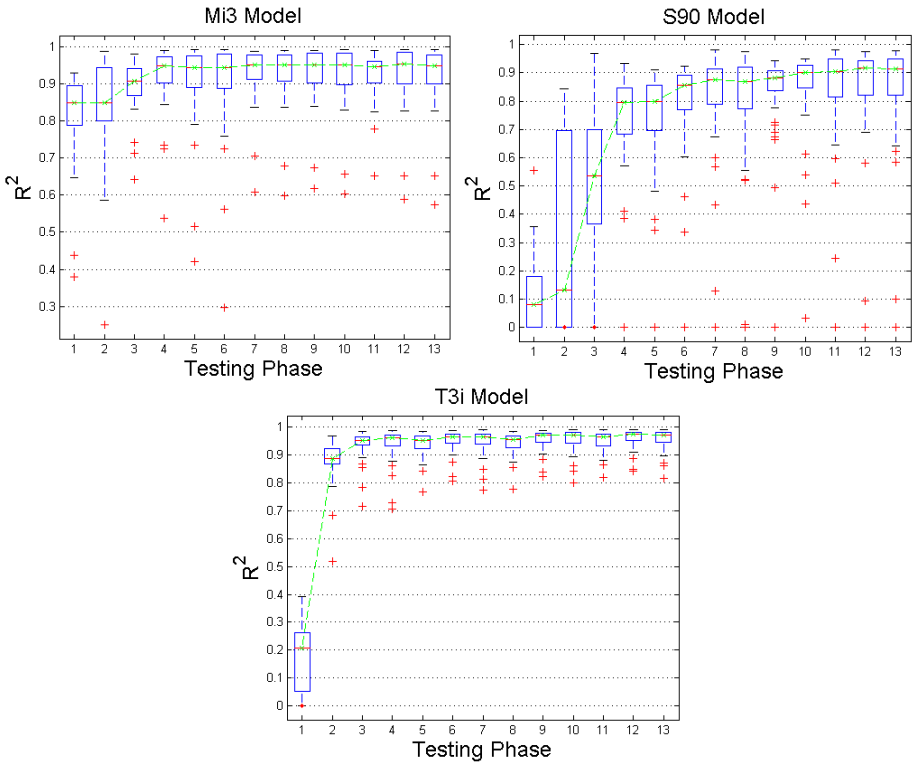

Figure 3.2: The evolution of the R^2 goodness of fit measure for all the image batches. The testing phase occurred after every 2 complete epochs of training. The green line shows the median R^2 at each testing time.

- $\hat{y}_i$ – the predicted noise to standard deviation ratio for a particular image patch
- $\overline{y}$ – the mean of y_i.

The cross-validated results on the test set can be seen in Figure 3.2. From Figure 3.2 one could see that after a few epochs of training the CNN models were able to perform very good noise estimation on the testing data. For all 3 cameras the models were able to reach an average R^2 greater than 90%. The S90 model however had some difficulty with a few image batches. Figure 3.3 shows the learned filters of each of the models in the first convolutional layer.

3.2.1 Cross Camera Performance

The next study that we performed was to observe if the models were capable of cross camera independence. Using the models we developed in section 3.2 we tested their noise estimation performance on images from cameras that were different than the camera they had been trained on. The results of this cross camera experiment can be seen in Figure 3.4.

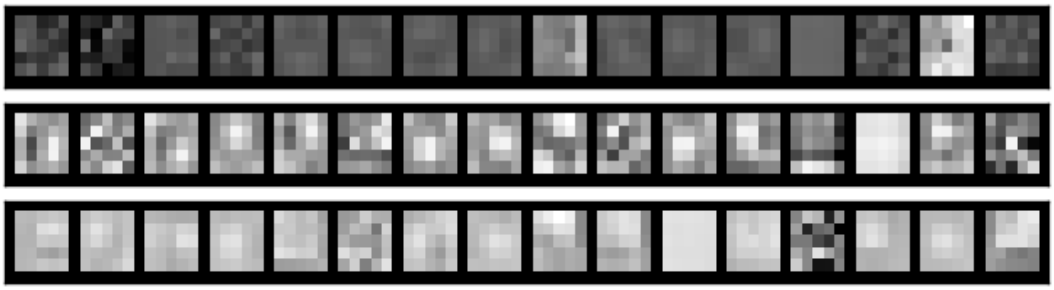

Figure 3.3: The 16 convolutional kernals learned in the first convolutional layer. The RENOIR dataset contained images for a Xiaomi Mi3 phone camera, a Canon S90 digital camera, and a Canon Eos Rebel T3i. Top: Mi3 filters. Middle: S90 filters. Bottom: T3i filters.

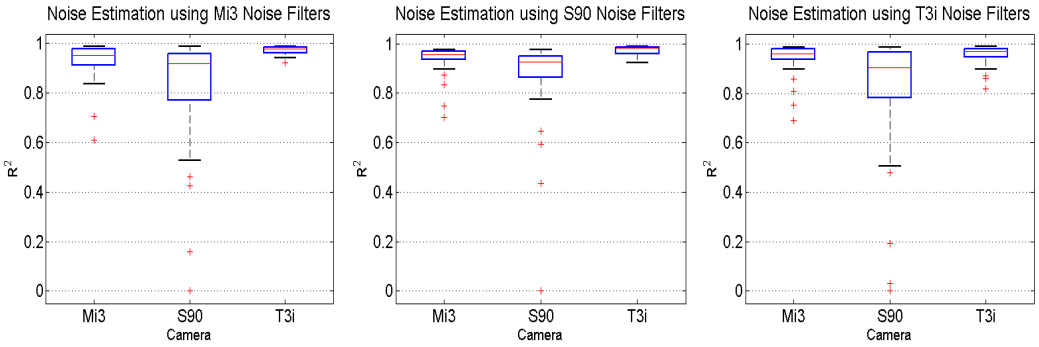

Figure 3.4: A comparison of the R^2 goodness of fit measure for the cross camera performances.

It can be seen that the models performed well in noise estimation regardless of the camera images that were used in training. Also, recall the Xiaomi Mi3 images are from a mobile phone camera while the S90 and T3i images are from digital cameras. The median R^2 being above 90% for all the cross camera testing means that the models are capable of being camera and device independent. The S90 model in 3.4 shows the best fitting across all three cameras.

3.2.2 Comparison with Other Methods

We first begin by comparing the results of our model with the results of other possible models or methods on the RENOIR Mi3 phone images. For this comparison we looked at a simple linear model, a single-layer feed forward neural network, a mode of local variance model that does well in estimating additive and multiplicative Gaussian noise [1], a signal dependent noise model [23], our proposed CNN model, and a deeper CNN version of our own model with an additional convolutional layer prior to the fully connected layers. Each model tried to estimate the noise in various $32 \times 32 \times 1$

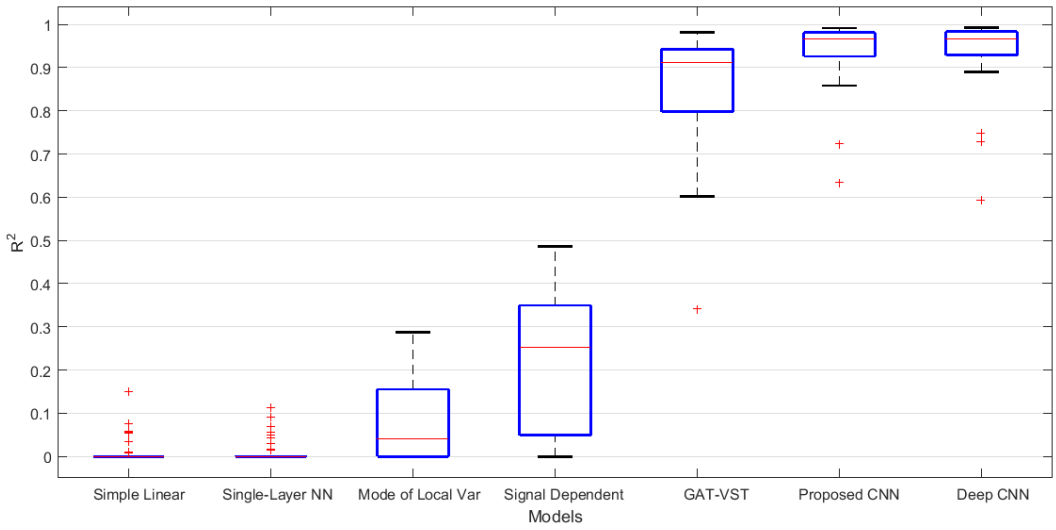

Figure 3.5: A comparison of the R^2 goodness of fit measure for various noise estimation models and our proposed CNN model.

image patches for a total of about 3,060,000 patches from the Mi3 camera. The results of these comparisons can be seen in Figure 3.5. We also tried to incorporate the variance stabilization noise estimation (VST) procedure seen in [3]for Poisson noisy images, however for our small image patches the stabilization was imprecise on Poisson-Gaussian data and we observed very large noise estimates. For this reason we did not include Poisson-VST results in this particular experiment.

The simple linear regression and the single feed forward neural network models made terrible estimations shown by a median R^2 measure of 0. The mode of local variance model [1] performed slightly better on real low-light noise, however as stated previously this model was meant for Gaussian noise estimation and this is reflected by its median R^2 measure of 0.05. The signal dependent noise model [23] was meant for real low-light noise estimation. To get the signal dependent noise model to estimate a $32 \times 32 \times 1$ noise patch, we had it estimate the noise level functions of various cropped patches (1000×1000) for a given image. We did this because we observed difficulties when trying estimate the noise level function for the original 3000×3000 image resolution. We then found the mean intensity value of our patch of interest and then used the appropriate local noise level function for that patch. The median R^2 measure for the signal dependent model was around 0.25.

The Poisson-Gaussian VST method (Generalized Anscombe Transformation) introduced in [27] and further improved upon in [28] for single image noise estimation was also examined. The median R^2 for the Poisson-Gaussian VST method was measured as 0.91. Finally, both our proposed CNN and the deeper version (with an extra convolutional layer) performed slightly better and with less variation. The training time for this deeper CNN model on the 12 GB Nvidia Tesla K40c was about 1 hour.

From this experiment we conclude that the CNN is important for obtaining a good estimation of the noise level and a simple linear regression or even a simple 1-layer NN are not good enough for this purpose. Furthermore with the proposed CNN we obtain noise level estimation results much more accurately to other methods proposed in the literature such as the mode of local variance model [1] or the signal dependent noise model [23] for small image patches.

Table 3.1: Whole image noise estimation experiments. Average difference between the noise level estimated by different methods and the true noise level.

	Avg Mi3	Std. Mi3	Avg S90	Std. S90	Avg T3i	Std. T3i
CNN	1.51	1.09	1.46	1.64	1.14	1.11
Signal Dependent	8.33	18.90	4.40	5.65	5.80	9.05
GAT-VST	6.24	6.22	3.05	2.25	12.22	5.89
Poisson-VST	6.96	6.15	6.86	3.60	9.15	6.20

We also compared the various single image methods on their quality of whole image noise estimation. In particular we examined the signal dependent, CNN, and the VST methods on all the noisy images of the RENOIR dataset. We first estimated the noise in a particular image using one of the methods and then looked at the absolute difference between the estimated noise value and its true noise value given by the RENOIR dataset. These results can be seen in Table 3.1. On average our proposed CNN estimates the image noise more accurately than the other single image models. Note that while the Iterative Poisson-VST is meant for Poisson noisy images it still gives reasonable noise estimation results.

CHAPTER 4

IMAGE DENOISING

We used the RENOIR dataset to evaluate four popular image denoising algorithms: the Active Random Field (ARF)[4], Block Matching and 3D Filtering (BM3D) [10], Bilevel optimization (opt-MRF) [8], and Multi Layer Perceptron (MLP) [6]. These algorithms were selected because they are efficient enough to handle our large images and have code available online. Each of these methods depends on a noise level parameter σ. The methods were evaluated for a number of values of the noise level σ and the best results were reported for each method. We tested the ARF filters [1] that were trained using Gaussian noise (in particular the trained filters for σ = 10, 15, 20, 25 and 50) and using four iterations. A special version of the BM3D algorithm meant for color image denoising [2] was used on the noisy images. For BM3D we evaluated the algorithm's performance at σ = 5, 10, 15, 20, 25, and 50. For opt-MRF we used the Gaussian trained filters (σ = 15 and 25) and a maximum limit of 30 iterations for the optimization [3]. Finally, we used MLPs trained on Gaussian filters [4] to denoise our images. In particular we used filters for σ =10, 25, 35, 50, and 75. For the ARF, opt-MRF, and MLP algorithms the image channels were denoised in the YUV color space for better performance. As BM3D directly handled color images, no transformation was necessary.

In Table 4.1 are shown the denoising results of the various methods on the three cameras. We computed the PSNR, SSIM, and VSNR values between the denoised and the best GT estimate which is the average of the two clean images.

Note the high values given by the SSIM prior to denoising and the lower values for two cameras after denoising. It is not clear how to interpret the SSIM results since the other two measures (PSNR and VSNR) are consistent with each other and with the fact that denoising was performed, while SSIM is not.

[1] from `http://www.stat.fsu.edu/~abarbu/ARF/demo.zip`
[2] from `http://www.cs.tut.fi/~foi/GCF-BM3D/BM3D.zip`
[3] from `http://pan.baidu.com/share/link?shareid=1707746373&uk=2974149445`
[4] from `http://people.tuebingen.mpg.de/burger/neural_denoising/`

Table 4.1: Denoising results

Camera	Before Denoising	ARF	BM3D	opt-MRF	MLP
PSNR					
Mi3	23.492	30.918	32.347	31.641	31.230
S90	26.187	33.797	36.752	34.983	34.073
T3i	27.442	36.550	39.966	38.646	37.584
Average	25.707	33.755	36.355	35.090	34.296
SSIM					
Mi3	0.989	0.972	0.982	0.964	0.929
S90	0.988	0.959	0.979	0.958	0.920
T3i	0.991	0.993	0.994	0.993	0.933
Average	0.989	0.981	0.985	0.972	0.927
VSNR					
Mi3	17.746	22.387	24.820	22.521	24.132
S90	23.789	26.769	28.635	27.357	27.255
T3i	22.318	28.567	30.481	29.803	29.429
Average	21.284	25.908	27.979	26.560	26.605

The best results obtained for the ARF, opt-MRF, and MLP methods occurred with a $\sigma = 25$ filter while the BM3D provided its best results with a $\sigma = 50$ filter. The results from Table 4.1 show that the BM3D outperformed the other methods on all the cameras using all similarity measures. In particular when comparing the performance of BM3D with MLP and opt-MRF for real noisy images these results do not lead to the same conclusions as in [6] and [8] where these methods slightly outperformed BM3D.

We tested four algorithms ARF, BM3D, opt-MRF and MLP on the RENOIR dataset. We were then able to calculate and measure the noise levels in the denoised images using a variety of different methods such as PSNR, VSNR, and SSIM. Note that these methods were trained or tuned on images corrupted by artificial Gaussian noise. Some of these methodologies (ARF opt-MRF, and MLP) and many other recent state-of-the-art denoising methods such as: CSF [33], LSSC [26], and RTF [18] learn the noise structure from the noisy-clean image pairs. These methods could in fact perform even better for denoising low light images if trained on our dataset. With so many different denoising methods having been developed or currently in development, our dataset allows for proper analysis of these tools, and for the quantitative evaluation of noise models for digital and mobile phone cameras.

Our dataset poses one more training and testing challenge compared to using images corrupted by artificial noise. The images in our dataset have a large range of noise levels in them, while usually denoising methods are trained and evaluated for one noise level only. Data with different noise levels poses many challenges in training and testing, but at the same time it helps denoising algorithms advance to the level where they could be used in practice for automatically denoising digital camera images, without any user interaction.

4.1 Estimating the Denoising Parameter for BM3D

Results from Table 4.1 showed an incredible performance of the BM3D denoising algorithm [10] over the other algorithms tested. We noticed however that while the other denoising algorithms for the most part used Gaussian trained filters at $\sigma = 20$ or $\sigma = 25$ levels, the BM3D algorithm best performed using a $\sigma = 50$ noise level even though the average estimated noise level of the RENOIR data was around $\sigma = 20$. In [23] they believed it was possible that the true overall noise level estimate for an image might not be the best parameter for denoising when it comes to BM3D.

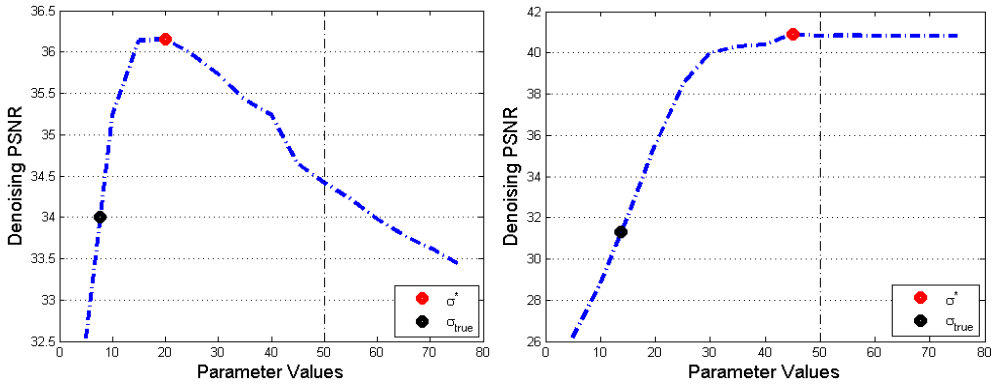

Figure 4.1: Examples of the BM3D denoising performance using various values of the BM3D parameter. The black point is the true noise level of the image and the red point corresponds to $\sigma = 50$. The best BM3D denoising performance occurs at the parameter value that provides the largest PSNR.

Figure 4.1 shows the BM3D performance (measured as the PSNR of the denoised image) for various values of the BM3D parameter σ for two S90 images. Also shown as a black dot is the

performance at the true noise level of the image and we can observe that the best denoising does not take place at the true noise level, but for much larger values of σ.

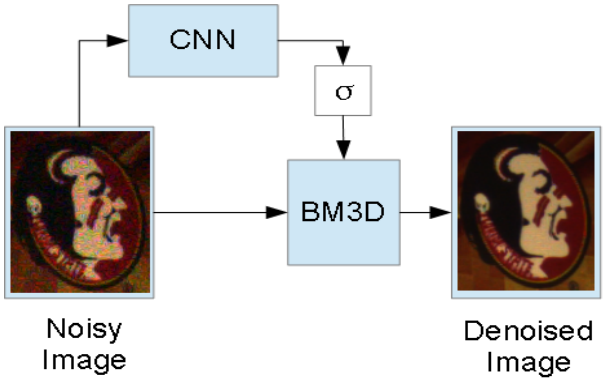

Figure 4.2: Example of the BM3D denoising scheme with the estimated σ.

4.1.1 Formulation as Loss Function Optimization

From Figure 4.1 we also observe that the PSNR depends smoothly on the BM3D parameter σ. Based on this observation, we can improve BM3D by predicting the best possible BM3D parameter σ using a CNN. Figure 4.2 shows our proposed BM3D denoising scheme.

More exactly, given an image patch I the CNN is a function $\sigma_{\mathbf{w}}(I)$ with the vector of weights to be learned $\mathbf{w}$. It is trained to optimize a loss function over the training examples $(I_i, y_i), i = 1, ..., n$:

$$L(\mathbf{w}) = \sum_{i=1}^{n} \ell(\sigma_{\mathbf{w}}(I_i), y_i) \tag{4.1}$$

where the form of y_i and the loss function $\ell(u, y)$ depend on what we are trying to optimize.

In this work we will investigate three loss functions defined in terms of the smooth PSNR functions $p_i(\sigma)$ obtained for each training example I_i as piecewise linear approximations over a grid of values σ (as showed in Figure 4.1). These loss functions are characterized as:

1. Predicting $\sigma_i^* = \text{argmax}_\sigma\, p_i(\sigma)$ directly using the L_2 loss $\ell(u, y) = \|u - y\|^2$. In this case $y_i = \sigma_i^*$ is the parameter that maximized the BM3D denoising PSNR for training example I_i (showed as a red dot in Figure 4.1).

2. Maximizing the PSNR loss

$$L(\mathbf{w}) = \sum_{i=1}^{n} p_i(\sigma_{\mathbf{w}}(I_i)), \tag{4.2}$$

thus $\ell(u, p) = p(u)$ and $y_i = p_i$, given as a vector of PSNR values on a grid and approximated as a piecewise linear function.

3. Minimizing the variance

$$L(\mathbf{w}) = \sum_{i=1}^{n} \mathrm{var}_i(\sigma_{\mathbf{w}}(I_i)), \tag{4.3}$$

where $p_i(\sigma) = 10 \log_{10}(255^2/\mathrm{var}_i(\sigma))$, thus $\mathrm{var}_i(\sigma)$ is the variance of the difference between denoised image I_i by BM3D with parameter σ and its clean counterpart. In this case $y_i = \mathrm{var}_i$ is a vector of values on a grid and is approximated as a piecewise linear function.

Observe that the loss functions (4.2) and (4.3) are novel loss functions that to the best of our knowledge have not been used in CNN training before. The derivative of a piecewise linear function is a piecewise constant function and this fact is used in back-propagation when training the CNN.

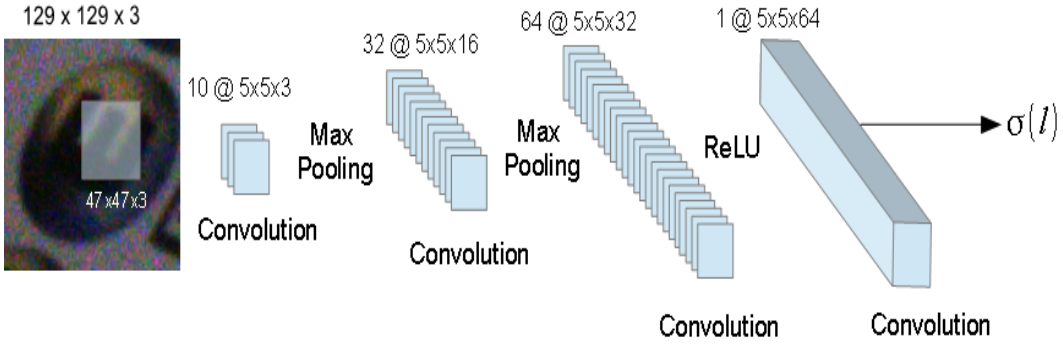

Figure 4.3: An example of our CNN meant to predict the BM3D σ parameter. It contains three convolutional layers and two max pooling layers in between the convolutional layers. The final convolutional layer fits the output neurons with ReLU.

4.1.2 CNN Architecture

For our application in estimating the BM3D tuning parameter σ we modified Matconvnet[5] by adding the three loss functions from Section 4.1.1.

The training examples are image patches of size $129 \times 129 \times 3$ and their corresponding y_i (where the variance and PSNR functions were approximated as piecewise linear with 15 endpoints). The CNN uses a randomly cropped patch of size $47 \times 47 \times 3$.

[5]From http://www.vlfeat.org/matconvnet/

The first layer of the CNN is a convolutional layer with 16 kernels of size $5 \times 5 \times 3$ followed by max-pooling. The second convolutional layer has 32 kernels of size $5 \times 5 \times 16$ followed by max-pooling. Then follows a third convolutional layer with 64 kernels of size $5 \times 5 \times 32$ followed by a ReLU (rectified linear unit) function. Then follows a single kernel of size $5 \times 5 \times 64$ which is the output neuron returning $\sigma(I)$. This architecture is shown in Figure 4.3.

4.2 CNN-BM3D Denoising Experiments

As stated earlier in section 4.1 we approximated the BM3D noise parameter using a CNN with our defined loss functions. Table 4.2 show the image denoising results of our CNN with CBM3D in PSNR for one noisy image from each scene in the RENOIR image dataset. On average a high noise value estimate like $\sigma = 50$ gives incredible denoising results, however we show that our CNN model was capable of helping make CBM3D perform even better.

Table 4.2: Image denoising PSNR results

Denoising Methods	Mi3	S90	T3i	Average
Before denoising	23.49	27.80	27.44	26.24
Use $\sigma = 50$	32.35	36.75	39.97	36.36
Regress σ directly	32.46	36.80	39.81	36.36
Optimize Var	32.87	36.46	39.81	36.38
Optimize PSNR	32.91	36.97	39.85	36.58
Signal dependent w/ CBM3D	24.97	31.85	33.83	30.22
GAT-VST w/ CBM3D	26.88	27.40	28.70	27.66
Iterative Poisson-VST RGB w/ BM3D	32.28	29.74	38.53	33.52

Figure 4.4 shows an example of the various denoising methods on a noisy image from the S90 data.

Figure 4.4: An example of the various denoising methods and their appropriate PSNR. Top left: Noisy image (28.48). Top Middle: $\sigma = 50$ (36.75). Top right: Regress σ (36.94). Middle left: Optimize Var (36.96). Middle middle: Optimize PSNR (36.69). Middle right: GAT-VST (27.47). Bottom left: Poisson-VST (28.82). Bottom right: Signal Dependent (34.56).

CHAPTER 5

CONCLUSIONS

We began this dissertation by first introducing the RENOIR image dataset. A dataset of over 400 uncompressed real low-light noisy and clean images using two digital cameras and a mobile phone camera. We collected the images using a procedure we called the "sandwich procedure" which allowed us to formulate a true noise level value for each image in the dataset. Using this information we developed a convolutional neural network model that estimated the noise level of single images for each camera. On average our CNN model for noise level estimation performed with a median $R^2 > 90\%$ for each camera. We then used the RENOIR dataset to evaluate and compare various image denoising algorithms trained on artificial Gaussian Noise. We noticed that on real low-light noisy images the BM3D algorithm performed better than all the other denoising algorithms contrary to what we were lead to believe from results on artificial Gaussian noisy images. We noted that the BM3D algorithm does not work best when its tuning parameter is the true noise value for real low-light noisy images and so we developed another convolutional neural network to best identify the tuning parameter value that lead us to the best image denoising we studied. We were able to do this by using two novel loss functions we believe haven't yet been used before in any other CNNs.

REFERENCES

[1] Santiago Aja-Fernández, Gonzalo Vegas-Sánchez-Ferrero, Marcos Martín-Fernández, and Carlos Alberola-López. Automatic noise estimation in images using local statistics. additive and multiplicative cases. *Image and Vision Computing*, 27(6):756–770, 2009.

[2] F. Alter, Y. Matsushita, and X. Tang. An intensity similarity measure in low-light conditions. In *ECCV*, pages 267–80, 2006.

[3] Lucio Azzari and Alessandro Foi. Variance stabilization for noisy+estimate combination in iterative poisson denoising, 2016.

[4] A. Barbu. Training an active random field for real-time image denoising. *Image Processing, IEEE Transactions on*, 18(11):2451–2462, 2009.

[5] Antoni Buades, Bartomeu Coll, and J-M Morel. A non-local algorithm for image denoising. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 2, pages 60–65. IEEE, 2005.

[6] H. C. Burger, C. J. Schuler, and S. Hamerling. Image denoising: Can plain neural networks compete with bm3d? In *CVPR*, pages 2392 – 2399, 2012.

[7] D. M. Chandler and S. S. Hemami. Vsnr: A wavelet-based visual signal-to-noise ratio for natural images. 16:2284 –2298, 2007.

[8] Y. Chen, T. Pock, R. Ranftl, and H. Bischof. Revisiting loss-specific training of filter-based mrfs for image restoration. In *German Conference Pattern Recognition*, pages 271–281, 2013.

[9] Dan Ciregan, Ueli Meier, and Jürgen Schmidhuber. Multi-column deep neural networks for image classification. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 3642–3649. IEEE, 2012.

[10] K. Dabov, A. Foi, V. Katkovnik, and K. Egiazarian. Image denoising by sparse 3-d transform-domain collaborative filtering. *Image Processing, IEEE Transactions on*, 16(8):2080–2095, 2007.

[11] F. Estrada, D. Fleet, and A. Jepson. Stochastic image denoising, 2009.

[12] F. Estrada, D. Fleet, and A. Jepson. Image denoising benchmark. `http://www.cs.utoronto.ca/~strider/Denoise/Benchmark/`, 2010. [Online; accessed 15-April-2014].

[13] A. Foi, M. Trimeche, V. Katkovnik, and K. Egiazarian. Practical poissonian-gaussian noise modeling and fitting for single-image raw-data. *Image Processing, IEEE Transactions on*, 17(10):1737–1754, 2008.

[14] G. E. Healey and R. Kondepudy. Radiometric ccd camera calibration and noise estimation. 16:267–276, 1994.

[15] K. Hirakawa and T. W. Parks. Adaptive homogeneity-directed demosaicing algorithm. 14:360–369, 2005.

[16] K. Hirakawa and T. W. Parks. Joint demosaicing and denoising. 15:2146–2157, 2006.

[17] Y. Ishii, T. Saito, and T. Komatsu. Denoising via nonlinear image decomposition for a digital color camera. In *ICIP*, volume 1, pages I–309. IEEE, 2007.

[18] J. Jancsary, S. Nowozin, and C. Rother. Loss-specific training of nonparametric image restoration models: A new state of the art. In *European Conf. Computer Vision*, pages 112–125, 2012.

[19] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[20] DxO Labs. Dxomark sensor scores. `http://www.dxomark.com/About/Sensor-scores/Use-Case-Scores/`, 2009. [Online; accessed 4-April-2014].

[21] Xinhao Liu, Masayuki Tanaka, and Masatoshi Okutomi. Noise level estimation using weak textured patches of a single noisy image. In *2012 19th IEEE International Conference on Image Processing*, pages 665–668. IEEE, 2012.

[22] Xinhao Liu, Masayuki Tanaka, and Masatoshi Okutomi. Estimation of signal dependent noise parameters from a single image. In *2013 IEEE International Conference on Image Processing*, pages 79–82. IEEE, 2013.

[23] Xinhao Liu, Masayuki Tanaka, and Masatoshi Okutomi. Practical signal-dependent noise parameter estimation from a single noisy image. *IEEE Transactions on Image Processing*, 23(10):4361–4371, 2014.

[24] C. Lui, R. Szeliski, S. B. Kang, C. L. Zitnick, and W. T. Freeman. Automatic estimation and removal of noise from a single image. 30:299–314, 2008.

[25] F. Luisier, B. Thierry, and M. Unser. Image denoising in mixed poisson-gaussian noise. *Image Processing, IEEE Transactions on*, 20:696–708, 2011.

[26] Julien Mairal, Francis Bach, Jean Ponce, Guillermo Sapiro, and Andrew Zisserman. Non-local sparse models for image restoration. In *2009 IEEE 12th International Conference on Computer Vision*, pages 2272–2279, 2009.

[27] Markku Mäkitalo and Alessandro Foi. Poisson-gaussian denoising using the exact unbiased inverse of the generalized anscombe transformation. In *2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1081–1084. IEEE, 2012.

[28] Markku Mäkitalo and Alessandro Foi. Noise parameter mismatch in variance stabilization, with an application to poisson–gaussian noise estimation. *IEEE Transactions on Image Processing*, 23(12):5348–5359, 2014.

[29] S. H. Park, H. S. Kim, S. Lansel, Parmar M., and B. A. Wandell. A case for denoising before demosaicking color filter array data. In *Asilomar Conference on Signals, Systems and Computers*, pages 860–864.

[30] N. Ponomarnko, V. L. Lukin, K. O. Egiazarian, J. Astola, M. Carli, and F. Battisti. Color image database for evaluation of image quality metrics. In *International Workshop on Multimedia Signal Processing*. IEEE, 2008.

[31] Javier Portilla, Vasily Strela, Martin J Wainwright, and Eero P Simoncelli. Image denoising using scale mixtures of gaussians in the wavelet domain. *IEEE Transactions on Image processing*, 12(11):1338–1351, 2003.

[32] William H Press. *Numerical recipes 3rd edition: The art of scientific computing*. Cambridge University Press, 2007.

[33] U. Schmidt and S. Roth. Shrinkage fields for effective image restoration. In *IEEE Int'l Conf. Computer Vision and Pattern Recognition*, 2014.

[34] Uwe Schmidt, Qi Gao, and Stefan Roth. A generative perspective on mrfs in low-level vision. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 1751–1758, 2010.

[35] P. C. Teo and D. J Heeger. Perceptual image distortion. In *International Symposium on Electronic Imaging: Science and Technology*, pages 127–141, 1994.

[36] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli. Image quality assessment:from error visibility to structural similarity. 13:600–612, 2004.

BIOGRAPHICAL SKETCH

Josue Anaya was born in 1989 in Hialeah, Florida. He graduated from Barbara Goleman Senior High School in 2007. He then attended Florida State University and began working with Dr. Adrian Barbu in the Spring of 2010. Josue graduated in the following Spring of 2011 with a Bachelor of Science degree in Statistics. In the Fall of 2011 Josue enrolled into the Florida State University graduate school to work under the supervision of Dr. Adrian Barbu. During his graduate studies Josue worked on course development from Fall 2011 to Spring 2012 and then taught introductory level courses in Statistics from Fall 2012 to Spring 2016. He recieved his Master of Science in Statistics in Spring 2015 and is expecting to graduate with his Doctorate in the Fall of 2016. His research interests include image processing, machine learning, and distribution theory.