Learning-Based Single Image Super-Resolution

Algorithm Analysis and Data Synthesis for Real-World Performance

by

Samim Zahoor Taray

Master Thesis

Faculty of Mathematics and Computer Science
Department of Computer Science
Saarland University

Supervisor
Prof. Dr.-Ing. Philipp Slusallek

Advisor
Dr. habil. Ivo Ihrke

Reviewers
Prof. Dr.-Ing. Philipp Slusallek
Dr. habil. Ivo Ihrke

September 29, 2020

UNIVERSITÄT
DES
SAARLANDES

Declaration of Authorship

Eidesstattliche Erklärung

Ich erkläre hiermit an Eides Statt, dass ich die vorliegende Arbeit selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel verwendet habe.

Statement in Lieu of an Oath

I hereby confirm that I have written this thesis on my own and that I have not used any other media or materials than the ones referred to in this thesis.

Einverständniserklärung

Ich bin damit einverstanden, dass meine (bestandene) Arbeit in beiden Versionen in die Bibliothek der Informatik aufgenommen und damit veröffentlicht wird.

Declaration of Consent

I agree to make both versions of my thesis (with a passing grade) accessible to the public by having them added to the library of the Computer Science Department.

Datum/Date: _____

Unterschrift/Signature: _____

SAARLAND UNIVERSITY
Department of Computer Science

Abstract

Learning-Based Single Image Super-Resolution:
Algorithm Analysis and Data Synthesis for Real-World Performance

by Samim Zahoor Taray

Most recent learning-based methods for super-resolution assume that images are blurred with a spatially invariant blur kernel. This assumption rarely holds for real world images as blur in real world images can vary spatially. As an example, the lens blur which affects every image has been shown to vary spatially even for high end lenses used by professional photographers and the effects can be more severe for low end lenses. In this thesis, a super-resolution method is proposed which can deal with arbitrary spatially varying blurs. The method is based on the Plug and Play optimization framework and allows using deep convolutional neural networks as priors via the variable splitting technique. Experiments on real and synthetic data are performed to show that the method can effectively remove spatially varying blurs from images and doing so results in better results in terms of quality.

An additional common assumption of state-of-the-art algorithms is that noise present in the images is additive white Gaussian noise. For real world images this characterization is not accurate because noise in real world images contains both signal-dependent and signal-independent components. Furthermore, noise in color images can be spatially and chromatically correlated. Therefore, these existing super-resolution methods perform poorly on real world images. In the second part of the thesis, a method to synthesize data with realistic blur and noise characteristics is proposed. The data is used to train deep convolutional neural networks for image super-resolution. Experiments on synthetic data are performed to show that a single network can effectively deal with images corrupted with a range of noise levels. Tests on real world images of a variety of scenes taken from different cameras show that the method generalizes well and produces sharp edges and detailed textures while at the same time effectively removing noise.

Acknowledgements

This thesis is the result of joint work between Saarland University and K|Lens GmbH. I would like to take this opportunity to express my sincere gratitude to all the colleagues I worked with during the thesis.

Firstly, I would like to thank Matthias Schmidt, CEO, K|Lens GmbH. and Dr. Sunil Jaiswal, Computer Vision Researcher, K|Lens GmbH. for proposing to conduct the thesis in collaboration with Saarland University. The thesis was jointly supervised by Dr. habil. Ivo Ihrke, Dr. Sunil Jaiswal and Prof. Dr.-Ing. Philipp Slusallek. I am very grateful to my advisor Ivo for guiding me throughout the thesis and agreeing to be the second reviewer. He gave enormous support, valuable comments and working with him was a wonderful learning experience. I want to thank him for always having a positive outlook towards my work and it has motivated me to put forward my best effort. I am incredibly thankful to Sunil for supervising my work at K|Lens. He has been the person that I can always go to and has regularly helped with insightful discussions on the topic. I would like to express special gratitude to Prof. Slusallek for agreeing to be the supervisor from the Saarland University and also the first reviewer of the thesis. I would also like to thank Dr.-Ing. Klaus Illgner-Fehns, CTO K|Lens GmbH for all the insightful discussions on the topic. They have helped me in gaining and pushed me towards seeking deeper understanding of all things technical.

I am very thankful to Noshaba Cheema for all the time and effort she has spent in coordinating matters related to the thesis between K-Lens and Saarland University and for agreeing to proof-read the thesis. I am deeply grateful to my colleagues at K|Lens GmbH for their companionship and kind support. I want to especially thank Kireeti Bodduna for his insights about the problems I discussed with him.

Last but not the least, I want to thank my parents, my family and all my friends for supporting and encouraging me throughout the thesis.

Content

Dedicated to my parents Nighat and Zahoor.

Chapter 1

Introduction

The performance of an imaging system is characterized by its resolution which determines the quality of images that the system can capture. The resolution of all imaging systems is limited because of various constraints on the components that make up the imaging system. The constraints come in the form of unavoidable theoretical constraints such as the phenomenon of diffraction, i.e. the bending of light waves which occurs when light passes through a finite opening or aperture. The constraints also come in the form of practical limits on the actual construction of parts that make up the optical system. For example, the lens setup which is used to focus light can suffer from aberrations, the sensor elements that record the intensity of light can only be packed up to a certain density and the process of recording invariably introduces noise in the measurement. Together, these constraints limit the resolution of the optical system which results in loss of fine details of objects in their recorded images. The goal of image super-resolution (SR) is to overcome the limits of resolution imposed by the imaging systems and generate higher resolution images. It is almost always desirable to have higher resolution images with maximum possible details of the imaged objects. Therefore, SR finds applications in diverse areas such as medical imaging [1, 2], astronomy [3, 4, 5] and surveillance [6, 7, 8] to name a few.

In this thesis we focus on the problem of Single Image Super-Resolution (SISR) in which the goal is to generate a higher resolution image from a single low resolution image. The problem has been extensively studied and several algorithms have been proposed to deal with this problem. Most of these algorithms have to rely on a database consisting of paired low resolution and high resolution images [9, 10, 11, 12, 13, 14, 15, 16, 17]. The process of obtaining real world image pairs of low and high resolution images is tedious and cumbersome. Therefore, these methods have to rely on synthetically generated data. The methods work quite well to generate high resolution images from synthetic low resolution images. However, they usually fail to generalize to real world

low resolution images. This is because these methods ignore or incorrectly characterize the blur and noise which invariably degrades real world images. For example, blur is assumed to be spatially invariant and noise is either completely ignored or characterized as independent white Gaussian noise. Thus, the methods are bound to fail on real world images because blur in real world images can vary spatially [18] and the noise in real world images is neither independent nor fully characterized by a zero mean Gaussian. Recently an effort has been made to overcome these problems and devise methods that generalize well to real world images [19, 20, 21, 22, 23, 24]. For example Zhang et al. [17] propose a method which can deal with arbitrary blurs. However, their method assumes noise to be additive white Gaussian noise and blurs to be spatially invariant so it still fails to generalize well to real world images. Fritsche et al. [19] attempt to synthesize realistic low resolution images by training convolutional neural networks for image to image translations from the domain of clean images to the domain of real world low resolution images. They achieve good results for real world images. However, we found that their method also tends to produce images which look heavily processed and can be perceived as artificial. The method also relies completely on data and lacks any kind of modelling framework for blurring or noise characteristics. Thus it is difficult to rationalize the method and adapt it to various different settings.

1.1 Contributions

The goal of this thesis is to address the problems of the existing methods as outlined in the previous paragraph and develop a method for SISR that works well for real world images. Towards this goal, we make the following contributions:

- In Chapter 3 of this thesis, we propose an algorithm for SISR that can deal with images degraded with arbitrary spatially varying blurs and additive white Gaussian noise. The algorithm is based on the Plug and Play (PnP) optimization technique and allows using deep neural networks as priors for joint denoising and deblurring [25, 17]. We perform experiments to show that the algorithm can effectively remove varying blurs and doing so improves the quality of results. We also show that our method is stable for a range of noise levels.
- In Chapter 4 we propose an approach to further improve the quality of results on real world images. This approach is based on synthesizing data with realistic noise and blur characteristics. The data is then used to train convolutional neural networks to perform the task of deblurring, denoising and super-resolution all together. We perform experiments to show that a single network can deal with

a range of noise levels and different blur kernels. We also test our method on real world images of a variety of scenes taken from different cameras. Qualitative assessment of these results shows that our method generalizes well to real world images and can effectively remove the noise in real world images while at the same time enhancing the resolution by generating images containing rich textures and sharp details.

1.2 Structure

In Chapter 2 we recap important concepts and background material that is used throughout the thesis. We also present a brief survey of existing methods for single and multiple image super-resolution. In Chapter 3 we present our method based on the Plug and Play optimization framework for arbitrary spatially varying blurs. We also present the results from our experiments which show that our method is effective for super-resolving images degraded with spatially varying blurs and different levels of noise. In Chapter 4 we present the details of our data synthesis approach for the task of real-world super-resolution. We also present the details of network architectures and loss functions for training. We end this chapter by presenting results from our experiments which show that our approach generalizes well to real world images of a variety of scenes captured using different devices. Lastly, we conclude the thesis by summarizing all our findings in Chapter 5 and providing an outlook on future research directions for further improvements.

Chapter 2

Background and Related Work

2.1 Resolution, Image Formation and Inverse Filtering

The resolution of an optical system can be defined in-terms of the capability of the system to distinguish objects placed close to each other. Consider a point source of light. Ideally the point source should appear as a point of infinitesimally small dimensions in the image. However, in real systems, the point source appears smeared and spread out over a region in the image. This is because of the phenomenon of diffraction, lens aberrations and noise. For example diffraction through a finite circular aperture produces the Airy disk pattern for a point source of light. The pattern is shown in Figure 2.1 (a). If we now consider two point sources of light, then as the point sources get closer to each other, their patterns in the image start to overlap and they become indistinguishable from each other as depicted in Figure 2.1 (b) and (c). The minimum distance at which two point sources can be distinguished from each other in the image determines the resolving power or resolution of the optical system.

To determine the minimum distance at which the point sources can be distinguished so that the performance of optical systems can be quantified, several criteria have been devised. One of them is the Rayleigh criterion which specifies that the *diffraction patterns are considered just resolved when the central maximum of the first pattern coincides with the first minimum of the other pattern* [26]. For the case of the Airy disc diffraction pattern produced by diffraction through a circular aperture in the absence of aberrations and noise, this distance is given by:

$$R = 1.22\frac{\lambda}{2\mathrm{NA}}, \tag{2.1}$$

where NA is the numerical apperture of the system. It is approximated by the ratio of the radius of the circular aperture and the distance between the aperture and the focal

(a) (b) (c)

FIGURE 2.1: The airy disc pattern produced from a single point source is shown in (a). The overlapping patterns from two sources are shown in (b). As the objects get closer, the overlapping patterns start to merge and become difficult to distinguish as shown in (c). Images by: OpenStax College. Located at: http://cnx.org/contents/031da8d3-b525-429c-80cf-6c8ed997733a/College_Physics. License: CC BY: Attribution

plane and λ is the wavelength of light [27]. R is known as Rayleigh limit and $1/R$ gives the resolving power.

The resolution of an optical system can also be defined in-terms of *bandwidth* of the system using Fourier analysis. The optical system is viewed as a linear system fully characterized by its impulse response or *point spread function (PSF)* which is the image of a point source of light placed at infinity [28]. Let $g(x, y)$ represent a 2D continuous scene where x and y denote Cartesian coordinates. Assuming PSF to be spatially invariant, the image of the scene formed in the image plane denoted by $f(x, y)$ is given by:

$$f(x, y) = g(x, y) * h(x, y) \tag{2.2}$$

where $*$ denotes convolution. The 2D continuous scene $g(x, y)$ is blurred by the PSF $h(x, y)$. The blurring is modeled by the convolution operation which is given by

$$g(x, y) * h(x, y) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f(x - x^{'}, y - y^{'}) h(x^{'}, y^{'}) dx^{'} dy^{'}. \tag{2.3}$$

The convolution theorem states that the Fourier transform of the convolution of two signals in the spatial domain equals point-wise multiplication of their Fourier transforms. According to this theorem, Equation 2.2 can be written as

$$\begin{aligned} \mathcal{F}\{g(x, y) * h(x, y)\} &= \mathcal{F}\{g(x, y)\}\mathcal{F}\{h(x, y)\} \\ \mathcal{F}\{f(x, y)\} &= \mathcal{F}\{g(x, y)\}\mathcal{F}\{h(x, y)\} \end{aligned} \tag{2.4}$$

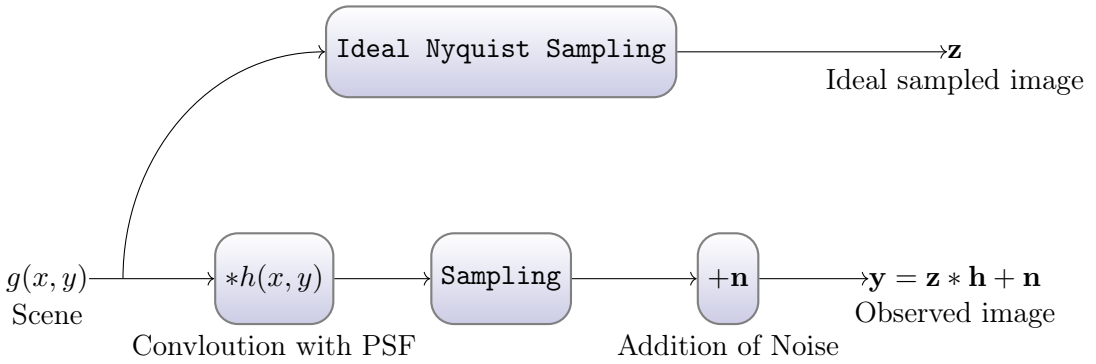


FIGURE 2.2: Block diagram depicting image formation

where $\mathcal{F}\{.\}$ denotes taking the two dimensional continuous Fourier transform and for the function say $f(x, y)$, it is given by $F(u, v)$ and is defined as

$$F(u,v) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(x,y)e^{-j2\pi(ux+vy)}dx\,dy, \tag{2.5}$$

where u and v denote spatial frequencies. Similarly, we also get the Fourier transforms of $G(u, v)$ and $H(u, v)$ of the functions $g(x, y)$ and $h(x, y)$ respectively. Thus, Equation 2.4 can also be written as

$$F(u,v) = G(u,v)\,H(u,v). \tag{2.6}$$

In case of diffraction limited systems (possibly with aberrations) $H(u, v)$ vanishes outside some bounded set $\mathcal{B}$ and the frequencies within the set constitute the pass-band of the optical system [29]. Thus, all images produced by the system are also band-limited with the same band because of multiplication in Equation 2.4. The result is that high frequency components lying outside the band which correspond to the sharpest details of the objects being imaged are smoothed out and the object appears blurred in the image. A band-limited approximation of the the image $g(x, y)$ can be recovered using inverse filtering, that can be derived from Equations 2.4 and 2.6 as

$$\begin{aligned} g(x,y) &= \mathcal{F}^{-1}\{G(u,v)\}, \\ g(x,y) &= \mathcal{F}^{-1}\left\{\frac{F(u,v)}{H(u,v)}\right\}, \end{aligned} \tag{2.7}$$

where $\mathcal{F}^{-1}\{.\}$ denotes the inverse Fourier transform and for the function say $G(u, v)$, it is given by

$$\mathcal{F}^{-1}\{G(u,v)\} = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} G(u,v)e^{j2\pi(ux+vy)}du\,dv. \tag{2.8}$$

However, before the inverse filtering operation can be performed, the image has to be recorded. This is achieved using an array of detectors called the photosites which measure scene irradiance. The photosites are arranged in a two dimensional grid and constitute the sensor. The spacing between the photosites of the sensor, also known as

the pixel pitch, determines the sampling rate. With an ideal sampling rate dictated by the Nyquist criterion, the original band-limited scene can be perfectly recovered from its samples. Let the vector representing this ideal sampling of the scene be denoted by $\mathbf{z}$. The process of recording invariably introduces noise in the samples. To account for noise, the image formation model of Equation 2.2 has to be extended as:

$$\mathbf{y} = \mathbf{z} * \mathbf{h} + \mathbf{n}, \tag{2.9}$$

where $\mathbf{y}$ denotes the recorded image which is obtained from the ideal image of the scene $\mathbf{z}$ blurred with the discrete version of the system PSF $\mathbf{h}$ and contaminated with noise $\mathbf{n}$. A block diagram depicting this process is shown in Figure 2.2. The discrete version of the convolution theorem states for circular boundary conditions, the discrete Fourier transform (DFT) of the convolution of two signals is equal to a point-wise multiplication of the DFT of two signals. Using the discrete convolution theorem, Equation 2.9 can be written as:

$$\mathcal{F}\{\mathbf{y}\} = \mathcal{F}\{\mathbf{z}\} \odot \mathcal{F}\{\mathbf{h}\} + \mathcal{F}\{\mathbf{n}\}, \tag{2.10}$$

where $\odot$ denotes element-wise multiplication of the arrays and $\mathcal{F}\{.\}$ now denotes taking the two-dimensional DFT. For the 2D array, say $\mathbf{y}$, of dimensions $M_1 \times M_2$ it is given by $\mathbf{Y}$ and is defined as

$$\mathbf{Y}(u,v) = \frac{1}{M_1 M_2} \sum_{r=0}^{M_1-1} \sum_{s=0}^{M_2-1} \mathbf{y}(r,s) e^{-j2\pi\left(\frac{us}{M_1}+\frac{vr}{M_2}\right)} \tag{2.11}$$

where u and v now denote discrete spatial frequencies. Similarly, we can also calculate the DFTs for $\mathbf{z}$, $\mathbf{h}$, and $\mathbf{n}$ and let these be denoted as $\mathbf{Z}$, $\mathbf{H}$ and $\mathbf{N}$ respectively. Using these in Equation 2.10 we get

$$\mathbf{Y} = \mathbf{Z} \odot \mathbf{H} + \mathbf{N}. \tag{2.12}$$

Following the same steps as the continuous case, a band-limited approximation of the samples of the scene denoted by $\hat{\mathbf{z}}$ can be recovered using inverse filtering as

$$\hat{\mathbf{z}} = \mathcal{F}^{-1}\left\{\mathbf{Y} \odot \frac{1}{\mathbf{H}}\right\} \tag{2.13}$$

where $\mathcal{F}^{-1}$ denotes the discrete inverse Fourier transform (IDFT) and for the array $\mathbf{Y}$, it is defined as

$$\mathcal{F}^{-1}\{\mathbf{Y}\} = \sum_{u=0}^{M_1-1} \sum_{v=0}^{M_2-1} \mathbf{Y}(u,v) e^{-j2\pi\left(\frac{us}{M_1}+\frac{vr}{M_2}\right)}. \tag{2.14}$$

Using Equation 2.12 in 2.13, we get

$$\hat{\mathbf{z}} = \mathcal{F}^{-1}\left\{\mathbf{Z} + \mathbf{N} \odot \frac{1}{\mathbf{H}}\right\}. \tag{2.15}$$

From this equation, we can see that in practice inverse filtering can result in an enhancement of noise. This is because the amplitude of noise is assumed to be constant over the entire band whereas $\mathbf{H}$ tends to 0 towards the boundary of the band. The situation is depicted in Figure 2.3. This is a well known problem in inverse filtering. A commonly used approach to overcome this problem is the well known Wiener filter. For the current discussion, this means that it is only possible to recover a bandlimited approximation of the image $\hat{\mathbf{z}}$ on a band $\mathcal{B}_{\text{eff}}$ which is smaller than the band of optics $\mathcal{B}$. Some works refer to $\mathcal{B}_{\text{eff}}$ as the *effective band*. The effective band $\mathcal{B}_{\text{eff}}$ determines the true resolution capabilities of the optical system. Berterto and De Mol [29] define super-resolution as *the process of restoring frequency components lying outside the effective band*. On the other hand, restoration within the effective band is referred to as deblurring.

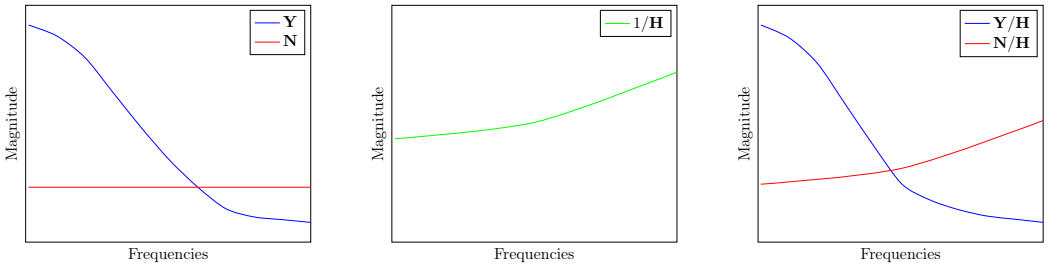


FIGURE 2.3: Enhancement of noise in inverse filtering

2.2 Related Work

Super-resolution has been an active area of research for many years and has been extensively studied. The rest of this chapter is dedicated to a brief review of existing approaches that aim to solve the problem. Existing methods can be classified into one of two types:

- Multiple Image Super Resolution (MISR) in which the input to the algorithm is a set of low resolution images of a scene and the goal is to synthesize a high resolution image of the scene covered by the low resolution images.
- Single Image Super Resolution (SISR) in which the input to the algorithm is a single low resolution image and the goal is to generate the high resolution image corresponding to the input image.

Main Category	**Sub Category**	**Methods**
Multiple Image Super-Resolution		[30, 31, 32, 33, 34, 35, 36, 37]
Single Image Super-Resolution	Internal Example Based	[38, 39, 40]
	External Example Based	Dictionary based: [9, 10, 11, 12, 13, 41]. DNN based: [14, 42, 43, 44, 45, 15]

TABLE 2.1: Classification of existing SR approaches.

2.2.1 Multiple Image Super Resolution

Although works on super-resolution were published as far back as 1974 [46], initial works on MISR [30, 31] were published starting 1984. These works were developed to super-resolve images captured by satellites of the surface of earth where the low resolution images are assumed to be shifted and under-sampled images of the same continuous scene i.e the surface of earth and the restoration algorithm is developed in the frequency domain. A drawback of the frequency domain techniques used by these methods is that they cannot handle more complicated motions between the LR images and are limited to only translations between the LR images.

Irani and Peleg[32] propose the *backprojection* method for MISR which is inspired by reconstruction methods used for Tomography. The algorithm requires accurate knowledge of motion between the LR frames and the motion model can be arbitrary. They start from an initial guess of the HR image and "backproject" it onto each of the LR images. They then calculate errors between the projected LR images and the captured LR images and iteritively refine the estimated HR image by minimizing this error. Farsiu et al. [33] observe that only minimizing the error between the projected images and recorded images can result in unstable optimization. They use the so called *Bilateral Total Variation (BTV)* norm of the HR estimate as the regularization term to make the optimization procedure stable. In [34], Li et al. show that BTV regularization can result in enhancement of noise and propose a locally adaptive bilateral total variation to suppress noise during reconstruction. The authors in [35] propose to estimate motions between the LR frames using optic flow which allows them to remove any assumption on the motion between LR frames. They can handle complicated motions using accurate flow estimation between the LR frames. Cheeseman et al. [47] formulate the problem in the MAP framework, in which the HR image is estimated by maximizing the *posterior distribution*. An important aspect of such approaches is choosing a suitable *prior* which is used to enforce beliefs about "what the HR image should look like" and additionally

stabilize the optimization procedure. The choice of prior is critical and determines the quality of reconstruction to a large extent. The prior term used by Cheeseman et al. enforces that pixels in a local neighbourhood have similar values or in other words pixel values should have spatial smoothness. This is desirable for noisy pixel values. However, this also results in edges being smoothed over which is undesirable. Subsequently better suited priors based on gradients of images were utilized to prevent smoothing over edges [36, 48]. Instead of relying on an independent registration step, Hardie et al. [37] incorporate the estimation of registration parameters into their MAP framework. This allows them to perform registration and SR reconstruction jointly using a combined optimization scheme.

2.2.2 Single Image Super Resolution

The goal of SISR is to generate an HR image from one LR image. Most methods aim to to hallucinate details and textures that fit nicely with the input LR image and produce a realistic looking higher resolution image to achieve this goal. SISR is the only choice if a single input image is available and we wish to super-resolve it. However, even if multiple images are available, SISR can still be preferable in certain scenarios over MISR. This is because most MISR approaches generally assume some amount of aliasing is present in the LR images and then try to reconstruct a single HR image from all LR images that is free from aliasing artifacts. If there is little to no aliasing in the LR images, then SISR approaches could be more suitable as suggested in [49]. MISR algorithms also rely on accurate registration of the LR images. In case of complicated motions this becomes a difficult problem in itself and therefore SISR can be more suitable.

Most of the methods for SISR have to rely on a database consisting of paired low resolution and high resolution images [9, 10, 11, 12, 13, 38, 14, 15, 16, 17] and thus Example-Based Super-Resolution is sometimes also used interchangeably with SISR. Existing methods for SISR can further be divided into two categories namely Internal Example-based methods and External Example-based methods.

Internal Example-based Methods

These methods rely on the assumption that a natural image has repetitive content which can be exploited to produce a higher resolution version of the image. Based on this assumption, Glasner et al. [38] divide the image into small 5x5 patches and show that patches tend to recur in the image. They exploit this recurrence by essentially replacing each patch within the LR image with a higher resolution corresponding patch calculated

using just the input image. Huang et al. [50] expand the number of exemplar patches by accounting for additional geometric transformations in the matching process and they show that having larger number of exemplar patches improves results in better quality HR images. Shocher et al. [40] propose Zero Shot Super-Resolution wherein they train a small CNN using patches taken only from the test image. They then use this network to super-resolve the test image.

External Example-based

The main characteristic of these methods is that they rely on an external database of paired LR/HR images. The pioneering work of Freeman et al. [9] started research into external example based SISR. Their method is based on generating a dictionary of LR patches and their corresponding HR patches. An external dataset of paired LR/HR images is used to create the dictionary. To super-resolve an image, they first divide it into small overlapping patches. For each patch, they look it up in the dictionary and match it with its nearest neighbor. The LR patch is then replaced with the HR patch from the dictionary. Chang at al. [10] extended this method by looking up k nearest neighbors of each patch and generating the corresponding HR patch by taking a weighted average of all k corresponding HR patches. Zeyde et al. [11] learn sparse representations of LR and HR patches and use these representations as the dictionary. This leads to a more efficient algorithm without degrading quality of the HR images. A faster algorithm is proposed by Tomofte et al. [12, 13] which they call *Anchored Neighborhood Regression*. Their idea is to divide the dictionary into neighborhoods of similar dictionary atoms. For each atom in the dictionary they pre-calculate and store projection matrices which contain a sparse representation of a dictionary atom using other atoms within its neighborhood. At test time, the patch from the test image is matched to its nearest atom. The HR patch is obtained using the pre-calculated projection matrix of the nearest atom and the HR patches of the neighborhood to which the matched atom belongs. The dictionary based methods were superseded by Convolutional Neural Network based methods which have been very successful and continue to dominate standard SISR benchmarks. We provide a brief overview of these methods for SISR next.

Fully Convolutional Neural Networks for SISR

A Convolutional Neural Network (CNN) is a type of neural network that employs the discrete convolution operation as its building block. CNNs consist of at least one or more convolutional layers and in each convolutional layer, the input is transformed by performing the convolution operation with certain kernels. These kernels constitute the

weights or the parameters of the layer and are updated during training. CNNs are ideal for data that have a grid like topology e.g. images and time series data because of properties like sparse connectivity, parameter sharing and translation invariance [51]. Normally, the size of the kernels is much smaller than the size of the inputs which gives rise to sparse connectivity because the output of a layer is only affected by a portion of the inputs and not the entire input. Sparse connectivity is desirable for processing inputs with a very large size which is the case for images. Parameter sharing occurs as each filter is applied at every position in the input because of convolution operation. This property is especially useful for extracting features from the images efficiently. Suppose we want to extract edges from the image. The same edge can occur at any position and response of one filter can be used to detect the edge at any position in the image. Translation invariance refers to the property by which, if the input is shifted by a certain amount, then the response of the network is also shifted by the same amount. This property is also desirable for processing images as objects can appear anywhere in the image and under most circumstances, we do not want the objects to be treated differently depending on where they occur in the image. Suppose we are trying to classify if something is a cat or not in an image, then it should not matter where the cat is in the image as long as it is there. It should still be classified as a cat. Sparse connectivity, parameter sharing and translation invariance, have together made CNNs ideal tools for various image processing and computer vision tasks. Thus, CNNs have been adopted for solving various high-level computer vision tasks like image classification [52], object detection [53], semantic segmentation [54] as well as various low-level computer vision tasks such as denoising [55], optical flow [56] and super-resolution [14] etc.

Dong et al. [14] were the first to employ a Fully Convolutional Neural Network i.e. a network which does not consist any fully connected layers for SISR. Their network called SRCNN consists of 3 convolutional layers:

- The first layer is composed of 64 convolution kernels each of size 9x9 followed by ReLU [57] non-linearity where $\text{ReLU}(x) = x$ if $x > 0$ and $\text{ReLU}(x) = 0$ otherwise.
- The second layer is composed of 32 convolution kernels each of size 64x5x5 also followed by ReLU non linearity.
- The last layer consists of 1 convolution kernel of size 32x5x5. This layer performs the reconstruction of final HR image from the extracted features.

Their method relies on a dataset D consisting of pairs of LR and HR images denoted by $\mathbf{y}$ and $\mathbf{x}$ respectively i.e $D = \{(\mathbf{y_i}, \mathbf{x_i})\}_{i=1}^{m}$. Here m denotes the number of images

in the dataset. The dataset is generated as follows. They start with some good quality images taken from existing datasets like the Imagenet [52] dataset. They then extract sub-images of size 33x33 which are considered as the HR images $\mathbf{x}$. They then apply blurring using a Gaussian blur kernel and sub-sample the images by picking every s^{th} pixel along height and width where s becomes the scale factor for upsampling. This gives the LR image $\mathbf{y}$ corresponding to $\mathbf{x}$. Their network takes as input pre-upsampled images i.e $\mathbf{y}$ is first upsampled by the desired scaling factor s using bicubic upsampling and then processed by the network.

This dataset is used to "train" the CNN denoted by $\mathcal{G}_\theta$, where θ denote the parameters of the network. Training is accomplished by minimizing a loss function $\mathcal{L}$ which characterizes "how far away" the image predicted by the network is from the corresponding ground truth HR image i.e

$$\theta^* = \arg\min_\theta \frac{1}{m} \sum_{i=1}^{m} \mathcal{L}(\mathcal{G}_\theta(\mathbf{y_i}), \mathbf{x_i}), \tag{2.16}$$

where θ^* denotes the trained parameters of the CNN. They use the L_2 Loss which is computed as the $\mathcal{L}_2$ norm of the difference between the predicted and the ground truth image i.e $\mathcal{L}_2(\mathcal{G}_\theta(\mathbf{y_i}), \mathbf{x_i}) = \|\mathcal{G}_\theta(\mathbf{y_i}) - \mathbf{x_i}\|_2^2$.

They rely on stochastic gradient descent for the minimization of Equation 2.16 with a constant step size i.e.,

$$\theta^{t+1} = \theta^t - \alpha \frac{1}{m} \sum_{i=1}^{m} \nabla_\theta \|\mathcal{G}_\theta(\mathbf{y_i}) - \mathbf{x_i}\|_2^2, \tag{2.17}$$

where α denotes the constant step size or the learning rate, t denotes the current step index and m now denotes the number of images in a single batch instead of the entire dataset. The gradients of the network are calculated using back propogation.

Kim et al. [42] show that significant performance gains can be achieved by using deeper networks, i.e. networks with more convolutional layers. This is because deeper architectures allow aggregation of features over a larger receptive field in the input image which provides more information resulting in better reconstruction. They propose VDSR (Very Deep SR), a 20 layer CNN where each layer (except the first and the last layer) consists of 64 filters of size 3x3x64. To overcome the difficulty associated with training deeper networks they employ two strategies. First, they only predict the difference between the upsampled LR and HR image by making the output of the network $\hat{\mathbf{x}} =$

$\mathcal{G}_\theta(\mathbf{y}) + \mathbf{y}$. The loss function to be minimized thus becomes $\|\mathcal{G}_\theta(\mathbf{y}) - (\mathbf{x} - \mathbf{y})\|_2^2 = \|\mathcal{G}_\theta(\mathbf{y}) - \mathbf{r}\|_2^2$, where $\mathbf{r} = \mathbf{x} - \mathbf{y}$ is the residual or the difference between the upsampled LR image and the HR image which the network has to learn. The second strategy they employ is using large learning rates along with gradient clipping to avoid the exploding and vanishing gradient problems. The input to both VDSR and SRCNN is a pre-upscaled LR image using bicubic interpolation. The network can then be thought of as filling in fine details and textures that make up the HR image. Working with pre-upscaled images requires more computation which makes the approach in-efficient. To overcome this, Dong et al. [43] propose to operate directly on the LR image without pre-upscaling. They use a transpose convolution layer at the end of the network to perform upscaling to the desired scale. Instead of relying on transpose convolution, Shi et al. [44] propose the Efficient Sub-Pixel Convolution Layer(ESPCN) to perform upscaling. The ESPCN layer simply re-arranges values along the channel dimension of the extracted feature map into spatial dimensions to achieve upsampling and has also been widely adopted.

A requirement of the super-resolved images is that they should appear photo-realistic. That is to say super-resolved images should ideally be indistinguishable from very high quality images produced by real cameras. Recent works on super-resolution have shown that the $\mathcal{L}_1$ and $\mathcal{L}_2$ loss functions which are commonly used to train super-resolution networks are not the best choices when it comes to producing photo realistic images [45, 15]. The reason is that these loss functions have limited correlation with human perception of similarity. This means that networks trained to minimize the $\mathcal{L}_1$ or $\mathcal{L}_2$ loss between the ground truth and generated images cannot produce images which humans perceive to be close to high resolution images or high quality images. Thus, several recent works [45, 15, 16] have aimed to devise loss functions which correlate better with human perception for training neural networks to produce better super-resolved images.

Johnson et al. [45] propose a loss function computed by taking the Euclidean distance of high-level feature representations of the ground truth HR image and the predicted HR image. To get the feature representations they use a pretrained VGG-16 network trained for the task of classification on the Imagenet dataset. Their experiments show that using this perceptual loss results in visually pleasing SR results with better details and sharper edges. Ledig et al. [15] propose SRGAN which uses a combination of three loss functions to achieve photo-realistic super-resolution. Their loss function is given by the following equation:

$$\mathcal{L}_{\text{per}} = \mathcal{L}_{\text{mse}} + \mu_1 \mathcal{L}_{\text{adv}} + \mu_2 \mathcal{L}_{\text{fea}}, \tag{2.18}$$

where $\mathcal{L}_{\text{mse}}$ denotes the content loss, $\mathcal{L}_{\text{adv}}$ denotes the adversarial loss which guides the network towards synthesizing photo-realistic images and $\mathcal{L}_{\text{fea}}$ denotes the feature loss

which ensures that the generated images are perceptually similar to the ground truth high resolution images. μ_1 and μ_2 are constants that are used to control the relative strengths of each term. The $\mathcal{L}_{\text{mse}}$ term is computed in the usual way by taking the mean squared error between the ground truth and the generated image:

$$\mathcal{L}_{\text{mse}} = \frac{1}{m}\sum_{i=1}^{m}\frac{\|\mathcal{G}(\mathbf{y}^{(i)}) - \mathbf{x}^{(i)}\|_2^2}{M}, \tag{2.19}$$

where m denotes the batch size, $\mathbf{y}^{(i)}$ is the i^{th} low resolution image in the batch and $\mathbf{x}^{(i)}$ is the corresponding high resolution ground truth image and M denotes the number of pixels in the high resolution image. The $\mathcal{L}_{\text{mse}}$ term ensures that the content of the generated image on the pixel level is close to the ground truth image on the pixel level.

To guide the network towards generating photo-realistic images, they adapt the GAN loss [58] for the $\mathcal{L}_{\text{adv}}$ term. The GAN loss can be formulated as a two player minimax game [59]. The two players involved in the game are the generator $\mathcal{G}$ and the discriminator $\mathcal{D}$. The game proceeds such that the generator $\mathcal{G}$ tries to synthesize fake data points from an input such that the discriminator $\mathcal{D}$ cannot distinguish fake data points from real ones. On the other hand the discriminator $\mathcal{D}$ outputs the probability of a data point being real. It tries to classify datapoints from the dataset as real by outputting 1 and the generated datapoints as fake by outputting 0 for them. The GAN loss is formulated as:

$$\mathcal{L}_{\text{gan}} = \min_{\mathcal{G}}\max_{\mathcal{D}}\mathbb{E}_{\mathbf{x}\sim p_{\mathbf{x}}}\left[\log(\mathcal{D}(\mathbf{x}))\right] + \mathbb{E}_{\mathbf{y}\sim p_{\mathbf{y}}}\left[\log(1-\mathcal{D}(\mathcal{G}(\mathbf{y})))\right], \tag{2.20}$$

where $p_{\mathbf{x}}$ denotes the probability distribution of the real data and $p_{\mathbf{y}}$ denotes the probability distribution of inputs to the generator. The solution of the above problem is the optimum generator $\mathcal{G}^*$ that can generate samples from the distribution of real images $p_{\mathbf{x}}$ and the optimum discriminator $\mathcal{D}^*$ that outputs $\frac{1}{2}$ everywhere. If $\mathcal{G}$ and $\mathcal{D}$ are neural networks which have to be trained with stochastic gradient descent methods, Goodfellow et al.[58] propose a slightly different formulation of the GAN loss to prevent saturation of gradients in the generator. The loss function for the generator network is formulated as

$$\mathcal{L}_{\mathcal{G}} = -\mathbb{E}_{\mathbf{y}\sim p_{\mathbf{y}}}\left[\log(\mathcal{D}(\mathcal{G}(\mathbf{y})))\right] \tag{2.21}$$

i.e the generator tries to maximize the log probability of the discriminator to incorrectly label the generated data as real. The loss function for the discriminator remains the same and is given by:

$$\mathcal{L}_{\mathcal{D}} = -\mathbb{E}_{\mathbf{x}\sim p_{\mathbf{x}}}\left[\log(\mathcal{D}(\mathbf{x}))\right] - \mathbb{E}_{\mathbf{y}\sim p_{\mathbf{y}}}\left[\log(1-\mathcal{D}(\mathcal{G}(\mathbf{y})))\right], \tag{2.22}$$

i.e the discriminator tries to maximize the log probability of real data points and to minimize the log probability of fake data points which the generator produces. In practice, the networks are trained using stochastic gradient descent based methods where gradients are only calculated over a batch of images instead of the entire dataset. Thus the expectation in the loss function is replaced with means over all images in the batch. The adversarial loss for the generator is given by

$$\mathcal{L}_{\text{adv}} = \frac{1}{m}\sum_{i=1}^{m} \log(\mathcal{D}(\mathcal{G}(\mathbf{y}^{(i)}))). \tag{2.23}$$

Lastly, the feature loss term $\mathcal{L}_{\text{feat}}$ is calculated as the mean squared error between high level features of the generated and the ground truth image. The high level features are extracted by feeding both the generated and the ground truth images into a VGG19 [60] network trained for the task of image recognition. It is given by:

$$\mathcal{L}_{\text{per}} = \frac{1}{m}\sum_{i=1}^{m} \frac{\|\text{VGG}(\mathcal{G}(\mathbf{y}^{(i)})) - \text{VGG}(\mathbf{x}^{(i)})\|_2^2}{M'}, \tag{2.24}$$

where VGG denotes the feature extraction part of the VGG19 network and $M^{'}$ denotes the number of elements in the feature vector $\text{VGG}(\mathbf{x})$. Ledig et al.[15] show that networks trained with their proposed perceptual loss function can produce photo-realistic super-resolved images.

Wang et al. [16] comprehensively study the factors that affect the performance of SRGAN. To calculate the feature loss, they choose to take the features of the VGG network at the fourth convolution layer *before* the activation layer because the feature map after activation becomes sparse and they show that using sparse features to calculate the feature loss is not the most optimal choice for super-resolution. They also propose to use the so called Relativistic GAN loss instead of the original GAN loss used in SRGAN. The Relativistic GAN loss is based on assessing the relative realness of a batch of generated images with respect to the corresponding ground truth images. They achieved first place on the perceptual PIRM2018 perceptual super-resolution challenge [20] with their method ESRGAN. SRGAN and ESRGAN tend to produce images with textures where no textures are actually needed. This results in images containing perceptually unpleasing artifacts. To overcome this problem, Fritsche et al. propose the frequency separation method as a part of their method in [19]. To prevent artificial textures being synthesized in the low frequency regions of the image they apply the GAN loss only to high pass filtered images. To replicate the low frequency content of the low resolution images accurately, they also propose to apply the content loss only on the low pass

filtered images. This frequency separation produces better quality images and prevents the network from introducing texture artifacts where they are not needed.

2.3 Summary

In this chapter, we first discussed the problems associated with the recovery of high frequency components of an image which are lost during the various steps involved in recording the image. A consideration of these problems is essential for real-world SR because the goal in SR is to generate high frequency details that fit nicely with the image. We also presented a brief survey of the various methods that have been proposed to tackle the problem of SR. Since SR has been an active area of research for many years, a large body of works exists on it. Early works have focussed on MISR, where the goal is to generate a single higher resolution image of the scene from multiple lower resolution images. Most methods for MISR assume that some amount of aliasing is present in the LR image and aim to remove the aliasing by fusing the information from multiple LR images. The methods for SISR generally rely on an external database of HR and LR images and aim to utilize the information in this database to generate HR images. Deep Convolutional Neural Networks trained from large datasets give state-of-the-art performance on standard SISR benchmarks.

Chapter 3

Spatially Varying Blurs in SISR

Real world images are degraded by noise and blurring. The goal in SISR is to generate high resolution images from such noisy and blurred images. The generated images should possess sharp details, contain realistic textures and be free from noise. Therefore, dealing with blurs and noise in real world images is critical for good super-resolution performance. In this chapter, we focus on the problem of removing blur from the images in the context of super-resolution. Objects can appear blurry in their images because of several reasons such as relative motion of the objects and the camera (motion blur), turbulence of air in the atmosphere (atmospheric blur), wrong focus settings (defocus blur) and lens blur which arises because of the optics of the camera system i.e the lens and other components of the camera. We focus only on lens blur in this chapter because it affects the quality of every image.

Lens blur can be characterized by the blur kernel or the point spread function (PSF) of the optical system. There exist several methods to estimate the PSF [61, 18, 62], so for now we assume that the PSF is known or it can be estimated. The PSF has been shown to have non-negligible spread and vary spatially even for top of the shelf commercial lenses [18] and for low end lenses, these effects can be more severe. However, existing methods for SISR are designed to invert the effects of blurring of a single PSF. For example, most existing CNN based methods [14, 42, 15, 63] train their networks on synthetic data which consists of paired low resolution and high resolution images. The low resolution images are synthesized by blurring with one specific PSF and the network is trained to invert the effects of this specific PSF. Recently, some works have been proposed that can deal with arbitrary blur kernels [17, 64, 65]. However, these methods still assume that the PSF is invariant over the domain of the image. In this chapter, we propose an SISR method that can deal with arbitrary spatially varying blurs. We derive the method by first formulating the problem of SISR in the Maximum A-Posterirori (MAP) framework. We tackle the resulting regularization problem using the Alternating

Direction Method of Multipliers (ADMM) method. On a high level, we can say that this allows us to decompose the solution to the problem of SISR in two iterative steps. The first step consist of removing the spatially varying blur and we provide an efficient scheme for this. The second step consists of joint denoising and super-resolution and we use pre-trained CNNs for this step.

The rest of the chapter is organized as follows:

- In Section 3.1, we formulate the problem of SISR in the MAP framework and derive the solution using the ADMM method.
- In Section 3.2, we describe our method for removing spatially varying blurs from the images and also explain the implementation of the operators that allow us to implement deblurring efficiently.
- In Section 3.3, we present our experiments with real and simulated data which show that our method can effectively remove spatially varying blur and noise present in the images which results in sharper super-resolved images.
- Lastly we conclude the chapter with an analysis of the performance and remaining limitations of the approach. This motivates our work in Chapter 4 which is aimed at overcoming the remaining limitations to achieve good performance on a wide variety of real world images.

3.1 Deep Super-Resolver Priors for SISR

The observed image denoted by vector $\mathbf{y} \in R^{MN}$ is related to the desired HR image $\mathbf{x} \in R^{s^2MN}$ via blurring, sampling and addition of noise. Here M and N are the height and width of the LR image and s is the scaling factor. The effect of spatially varying blurs can be modelled by a matrix operator which operates on $\mathbf{x}$. Let this matrix operator be denoted by H. The sampling of the blurred image can also be modeled with a matrix operator and let this be denoted by D. Finally, the sampled image is degraded by additive noise denoted by vector $\mathbf{n} \in R^{MN}$. This results in the forward model

$$\mathbf{y} = DH\mathbf{x} + \mathbf{n}, \tag{3.1}$$

In-order to arrive at the formulation that allows deep neural networks to be used as priors for super resolution, the order of D and H has to be reversed as is proposed by

Zhang et al. [17]. This results in the forward model given by

$$\mathbf{y} = HD\mathbf{x} + \mathbf{n}. \tag{3.2}$$

Equation 3.2 conveys that the unknown HR image $\mathbf{x}$ is first downsampled by the operator D and then blurred by the operator H. Although, this exchange of order in H and D results in the image formation model which does not truly reflect the physics of image formation, we still continue with it because it has useful mathematical properties as we will see in the sequel. The dimensions of matrix which implements the downsampling operator D are $D \in R^{MN \times s^2 MN}$. This matrix is too large to handle even for moderate dimensions of the image, however, we note that the operator can be implemented by element-wise operations on the operand. So the matrix is never explicitly stored. The order of matrix H is given by $H \in R^{MN \times MN}$. As noted earlier, H represents the matrix of spatially varying blurs and under some assumptions, operations of type $H\mathbf{z}$ and $H^T\mathbf{z}$ can also be implemented efficiently. We cover the details of our implementation in a later part of this chapter in Section 3.2.1.

The MAP estimate denoted by $\hat{\mathbf{x}}$ of the desired HR image $\mathbf{x}$ is given by

$$\begin{aligned}
\hat{\mathbf{x}} &= \arg\max_{\mathbf{x}} p(\mathbf{x}|\mathbf{y}) \\
&= \arg\max_{\mathbf{x}} \frac{p(\mathbf{y}|\mathbf{x})p(\mathbf{x})}{p(\mathbf{y})} \\
&= \arg\max_{\mathbf{x}} p(\mathbf{y}|\mathbf{x})p(\mathbf{x}) \\
&= \arg\max_{\mathbf{x}} \log(p(\mathbf{y}|\mathbf{x})) + \log(p(\mathbf{x})) \\
&= \arg\min_{\mathbf{x}} -\log(p(\mathbf{y}|\mathbf{x})) - \log(p(\mathbf{x}))
\end{aligned} \tag{3.3}$$

where $p(\mathbf{y}|\mathbf{x})$ is the likelihood term and $p(\mathbf{x})$ is the prior term. Assuming $\mathbf{n}$ to be additive white Guassian noise in Equation 3.2 with noise level σ, the likelihood $p(\mathbf{y}|\mathbf{x}) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{\frac{\|\mathbf{y}-HD\mathbf{x}\|^2}{2\sigma^2}}$. Substituting it in Equation 3.3 we can write

$$\hat{\mathbf{x}} = \arg\min_{\mathbf{x}} \frac{1}{2\sigma^2}\|\mathbf{y} - HD\mathbf{x}\|^2 + \lambda\Phi(\mathbf{x}), \tag{3.4}$$

where the first term is the data-fidelity term and $\Phi(\mathbf{x}) = -\log(p(\mathbf{x}))$ is the regularization term. The scalar λ is introduced to control the trade-off between the two terms. We have assumed that the noise level σ is the same at each pixel for ease of notation. A more realistic assumption would be to assume that the noise level at each pixel is different and we note that the results derived still apply with minor modifications. The unconstrained optimization problem in Equation 3.4 can be equivalently formulated as a constrained

optimization problem given by

$$\hat{\mathbf{x}} = \arg\min_{\mathbf{x}} \frac{1}{2\sigma^2}\|\mathbf{y} - H\mathbf{z}\|_2^2 + \lambda\Phi(\mathbf{x}) \quad \text{subject to } D\mathbf{x} - \mathbf{z} = 0. \tag{3.5}$$

Doing so decouples the data-fidelity term and the regularization term when using the ADMM method [66] to solve Equation 3.5. The augmented Lagrangian for the above problem is given by:

$$L_\rho(\mathbf{x}, \mathbf{z}, \mathbf{u}) = \frac{1}{2\sigma^2}\|\mathbf{y} - H\mathbf{z}\|_2^2 + \lambda\Phi(\mathbf{x}) + \mathbf{u}^T(D\mathbf{x} - \mathbf{z}) + \frac{\rho}{2}\|D\mathbf{x} - \mathbf{z}\|_2^2. \tag{3.6}$$

In Equation 3.6, $\mathbf{u}$ is the dual variable or the Lagrange multiplier associated with the equality constraint and ρ is a penalty parameter. The ADMM method consists of performing the following iterations to solve the problem:

$$\begin{aligned}
\mathbf{x}^{k+1} &= \arg\min_{\mathbf{x}} L_\rho\left(\mathbf{x}, \mathbf{z}^k, \mathbf{u}^k\right), \\
\mathbf{z}^{k+1} &= \arg\min_{\mathbf{z}} L_\rho\left(\mathbf{x}^{k+1}, \mathbf{z}, \mathbf{u}^k\right), \\
\mathbf{u}^{k+1} &= \mathbf{u}^k + \left(D\mathbf{x}^{k+1} - \mathbf{z}^{k+1}\right).
\end{aligned}$$

By re-defining the dual variable $\mathbf{u}$ to be the scaled variable $\frac{\mathbf{u}}{\rho}$, we can write the Lagrangian in Equation 3.6 as:

$$L_\rho(\mathbf{x}, \mathbf{z}, \mathbf{u}) = \frac{1}{2\sigma^2}\|\mathbf{y} - H\mathbf{z}\|_2^2 + \lambda\Phi(\mathbf{x}) + \frac{\rho}{2}\|D\mathbf{x} - \mathbf{z} + \mathbf{u}\|_2^2 - \frac{\rho}{2}\|\mathbf{u}\|_2^2. \tag{3.7}$$

The ADMM iterations can finally be written as:

$$\mathbf{x}^{k+1} = \arg\min_{\mathbf{x}} \lambda\Phi(\mathbf{x}) + \frac{\rho}{2}\|D\mathbf{x} - \mathbf{z}^k + \mathbf{u}^k\|_2^2, \tag{3.8}$$

$$\mathbf{z}^{k+1} = \arg\min_{\mathbf{z}} \frac{1}{2\sigma^2}\|\mathbf{y} - H\mathbf{z}\|_2^2 + \frac{\rho}{2}\|D\mathbf{x}^{k+1} - \mathbf{z} + \mathbf{u}^k\|_2^2, \tag{3.9}$$

$$\mathbf{u}^{k+1} = \mathbf{u}^k + \left(D\mathbf{x}^{k+1} - \mathbf{z}^{k+1}\right). \tag{3.10}$$

$$\tag{3.11}$$

The second update step given in Equation 3.9 can be interpreted as a deblurring step which results in the intermediate deblurred image $\mathbf{z}$. We discuss this minimization problem in detail in Section 3.2 of this chapter. The third update step given in Equation 3.10

is simply the update of the dual variable. Equation 3.8 can equivalently be written as

$$\mathbf{x}^{k+1} = \arg\min_{\mathbf{x}} \frac{1}{2\sqrt{1/\rho^2}} \|\bar{\mathbf{z}}^k - D\mathbf{x}\|_2^2 + \lambda\Phi(\mathbf{x}), \tag{3.12}$$

where for simplification of expressions we have introduced $\bar{\mathbf{z}}^k = \mathbf{z}^k - \mathbf{u}^k$. It is now easy to see that this minimization problem is a simplified super-resolution and denoising problem that arises from the forward model given by:

$$\bar{\mathbf{z}}^k = D\mathbf{x} + \mathbf{n}_2,$$

where $\bar{\mathbf{z}}^k$ can be interpreted as the observed image obtained by downsampling of the unknown image $\mathbf{x}$ by the downsampling operator D and addition of noise $\mathbf{n}_2$ where $\mathbf{n}_2$ is additive white Gaussian noise with noise level $\sqrt{1/\rho}$. The problem is simpler because the downsamping operator D is the same for all possibly differently blurred images. We follow the procedure of Zhang et al. [17] to obtain $\mathbf{x}^{k+1}$ which is using a pre-trained CNN. We use the same architecture as [17] i.e SRResnet+ and train it for variable levels of additive white gaussian noise in the range $[0, 50]$ for joint de-noising and super-resolution. We denote the network by $\mathcal{SR}_D$ to note that the network is trained for a fixed downsampling operator D. To adapt a single network for various noise levels, a noise level map of the same size as the input image is created and concatenated with the input image. The image and the noise level map concatenated together become the input to the network which outputs a super-resolved and denoised image $\mathbf{x}^{k+1}$ i.e

$$\mathbf{x}^{k+1} = \mathcal{SR}_D\left(\mathbf{z}^k, \rho\right) \tag{3.13}$$

Zhang et al. coin the term *super-resolver prior* for $\mathcal{SR}_D$ in their work because it absorbs the prior term of Equation 3.4 and 3.8.

3.2 Spatially Varying De-blurring

In this section we describe our approach to tackle the first minimization problem of Equation 3.9 which can be re-written in the following way:

$$\mathbf{z}^{k+1} = \arg\min_{\mathbf{z}} \underbrace{\|\mathbf{y} - H\mathbf{z}\|_2^2 + \sigma^2\rho\|\tilde{\mathbf{x}}^{k+1} - \mathbf{z}\|_2^2}_{G(\mathbf{z})}. \tag{3.14}$$

For ease of notation, we have used $\tilde{\mathbf{x}}^{k+1} = D\mathbf{x}^{k+1} + \mathbf{u}^k$. Recall that $\mathbf{y}$ is the input LR image that we wish to super-resolve and σ is the noise level of each pixel in $\mathbf{y}$, ρ is the penalty parameter associated with the equality constraints of the constrained

minimization problem of Equation 3.6. It is easy to see that Equation 3.14 is a deblurring problem where H represents the blurring matrix. Previous works consider the blur to be spatially invariant. However, we consider the more general problem where the blur is not spatially invariant. Therefore, H now represents the spatially varying blur matrix. The objective $G(\mathbf{z})$ is smooth and convex. To minimize the objective, we rely on the Accelerated Gradient [67] method with constant step size and momentum parameters [68] given by

$$\mathbf{w}_{t+1} = \mathbf{z}_t + \beta(\mathbf{z}_t - \mathbf{z}_{t-1}) \tag{3.15}$$

$$\mathbf{z}_{t+1} = \mathbf{w}_{t+1} - \alpha\nabla_{\mathbf{z}} G(\mathbf{w}_{t+1}), \tag{3.16}$$

where $\nabla_{\mathbf{z}}$ denotes the gradient operator with respect to $\mathbf{z}$ and $\mathbf{w}$ is just an intermediate vector introduced to shorten the notation. We set $\alpha = 0.2$ and $\beta = 0.1$ for all our experiments. The details of the solver are given in Algorithm 1. Note that the solver only relies on matrix vector multiplications (MVMs) of the form $H\mathbf{x}$ and $H^T\mathbf{z}$ and we explain how such MVMs can be implemented efficiently using the *Filter Flow Framework* of Hirsch et al. [62] in Section 3.2.1. For convenience, let us denote the solver by $\mathcal{AG}$ which allows us to write:

$$\mathbf{z}^{k+1} = \mathcal{AG}(H, H^T, \tilde{\mathbf{x}}^{k+1}, \mathbf{y}, \sigma, \rho). \tag{3.17}$$

The solver $\mathcal{AG}$ takes as inputs the operators H and H^T along with the vectors $\mathbf{y}$, $\tilde{\mathbf{x}}^k$ and the constants ρ and σ, and outputs the deblurred image $\mathbf{z}^{k+1}$. Finally, the overall scheme to solve the original SISR problem of Equation 3.4 presented is in Algorithm 2.

Algorithm 1 $\mathcal{AG}$ solver

Input Operators H and H^T. Constant quantities $\mathbf{y}$, $\tilde{\mathbf{x}}^{k+1}$, ρ and σ. Parameters α and β and total iterations N.

1: Initialize $\mathbf{z}_{-1} = \mathbf{y}$ and $\mathbf{z}_0 = \mathbf{0}$
2: **while** $t < N_{\mathcal{AG}}$ **do**
3: Update $\mathbf{w}_{t+1} = \mathbf{z}_t + \beta(\mathbf{z}_t - \mathbf{z}_{t-1})$
4: Update $\mathbf{z}_{t+1} = \mathbf{w}_{t+1} - 2\alpha\left(H^T H\mathbf{w}_{t+1} + \rho\sigma^2\mathbf{w}_{t+1} - H^T y - \rho\sigma^2\tilde{\mathbf{x}}^{k+1}\right)$

Output $\mathbf{z}_{N_{\mathcal{AG}}}$

Algorithm 2 PnP Optimization with DNN prior for SISR

Input LR image $\mathbf{y}$, noise level σ, operators H and H^T total iterations T

1: Initialize $\mathbf{z}^0 = \mathbf{0}$ and $\mathbf{u}^0 = \mathbf{0}$
2: **while** $k < N_{\mathcal{SR}}$ **do**
3: Update $\mathbf{x}^{k+1} = \mathcal{SR}_D\left(\mathbf{z}^k, \rho\right)$ [Joint SR and denoising]
4: Update $\mathbf{z}^{k+1} = \mathcal{AG}(H, H^T, \mathbf{x}^{k+1}, \mathbf{y}, \sigma, \rho)$[Deblurring Step]
5: Update $\mathbf{u}^{k+1} = \mathbf{u}^k + \left(D\mathbf{x}^{k+1} - \mathbf{z}^{k+1}\right)$ [Dual variable update]

Output $\mathbf{x}^{N_{\mathcal{SR}}}$

3.2.1 Efficient Implementation of Operators H and H^T

Recall that H represents the matrix that models spatially varying blurs. Naively handling H i.e actually storing the matrix and performing regular matrix vector operations is computationally infeasible even for moderate dimensions of the image. To model spatially varying blurs, Hirsch et al. [62] assume that the image can be divided into *overlapping patches* and within each patch the blur is invariant. Based on this assumption, they propose the *Efficient Filter Flow Framework* (EFF). We rely on their method to efficiently evaluate matrix vector multiplications of the form $H\mathbf{z}$ and $H^T\mathbf{z}$ which is all that is needed for our problem. For completeness, we describe the approach next.

We assume that the input image denoted by $\mathbf{z}$ can be divided into p overlapping patches and the blur kernel within each patch is invariant. Let the blur kernel corresponding to patch r be denoted by $\mathbf{h}^r$. Since blur within a patch is invariant, it can be modeled using the convolution operation. The overall image which is degraded with the spatially varying blur (denoted by $\mathbf{y}$) is given by:

$$\mathbf{y} = \sum_{r=1}^{p} \mathbf{h}^{(r)} * (\mathbf{w}^{(r)} \odot \mathbf{z}), \tag{3.18}$$

where $\mathbf{w}^{(r)}$ is the vector that performs the task of extracting the r^{th} patch from the input image $\mathbf{z}$ and damping the borders of the patch by elementwise multiplications (denoted by $\odot$) with $\mathbf{z}$. The right hand side of Equation 3.18 can further be expanded as:

$$\mathbf{y} = \sum_{r=1}^{p} \mathbf{h}^{(r)} * \text{Diag}(\mathbf{w}^{(r)})\mathbf{z}, \tag{3.19}$$

Where $\text{Diag}(\mathbf{w}^{(r)})$ denotes a diagonal matrix constructed out of the entries of $\mathbf{w}^{(r)}$. It follows from the direct application of the convolution theorem that:

$$\mathbf{y} = \sum_{r=1}^{p} C_r^T \, \mathcal{F}^{-1} \left[\text{Diag}(\mathcal{F}(\mathbf{h}^{(r)})) \, \mathcal{F}(C_r \, \text{Diag}(\mathbf{w}^{(r)})) \right] \mathbf{z}, \tag{3.20}$$

where $\mathcal{F}$ denotes Discrete Fourier Transform, $\mathcal{F}^{-1}$ denotes Inverse Discrete Fourier Transform and C_r denotes the cropping matrix corresponding to r^{th} patch. Equation 3.20 can simply be read from left to right to understand the operations involved i.e the input image is first divided into p overlapping patches using $\text{Diag}(\mathbf{w}^{(r)})$. The patches are cropped using the cropping matrix C_r for r^{th} patch. The convolution operation representing blurring of the patch is implemented by multiplication in the Fourier domain and the results are transformed back to the spatial domain by taking the IDFT. Lastly, each patch is aligned and stacked back to its correct position by the transpose of the cropping matrix matrix C_r^T and overlapping regions are added up. A simplified visual example of the steps involved is depicted in Figure 3.1.

Furthermore, if we let $H^{(r)}$ denote the Toeplitz matrix associated with the blur kernel $\mathbf{h}^{(r)}$ which is to say that rows of $H^{(r)}$ consist of shifted copies of the kernel $\mathbf{h}^{(r)}$, then Equation 3.18 can equivalently be written as

$$\begin{aligned} \mathbf{y} &= \sum_{r=1}^{p} \mathbf{h}^{(r)} * (\mathbf{w}^{(r)} \odot \mathbf{z}), \\ \mathbf{y} &= \sum_{r=1}^{p} H^{(r)} \, \text{Diag}(\mathbf{w}^{(r)}) \mathbf{z}, \\ \mathbf{y} &= H\mathbf{z}, \end{aligned} \tag{3.21}$$

where the matrix H represents the spatially varying blurring operator. Comparing the right hand side of Equations 3.20 and 3.21, we see that

$$\boxed{H = \sum_{r=1}^{p} C_r^T \, \mathcal{F}^{-1} \left[\text{Diag}(\mathcal{F}(\mathbf{h}^{(r)})) \, \mathcal{F}(C_r \, \text{Diag}(\mathbf{w}^{(r)})) \right]} \tag{3.22}$$

The transpose operator needed can now be derived by taking the transpose on both sides of Equation 3.22 which yields

$$\boxed{H^T = \sum_{r=1}^{p} \text{Diag}(\mathbf{w})^{(r)} C_r^T \, \mathcal{F}^{-1} \left[\overline{\text{Diag}(\mathcal{F}(\mathbf{h}^{(r)}))} \, \mathcal{F}(C_r) \right].} \tag{3.23}$$

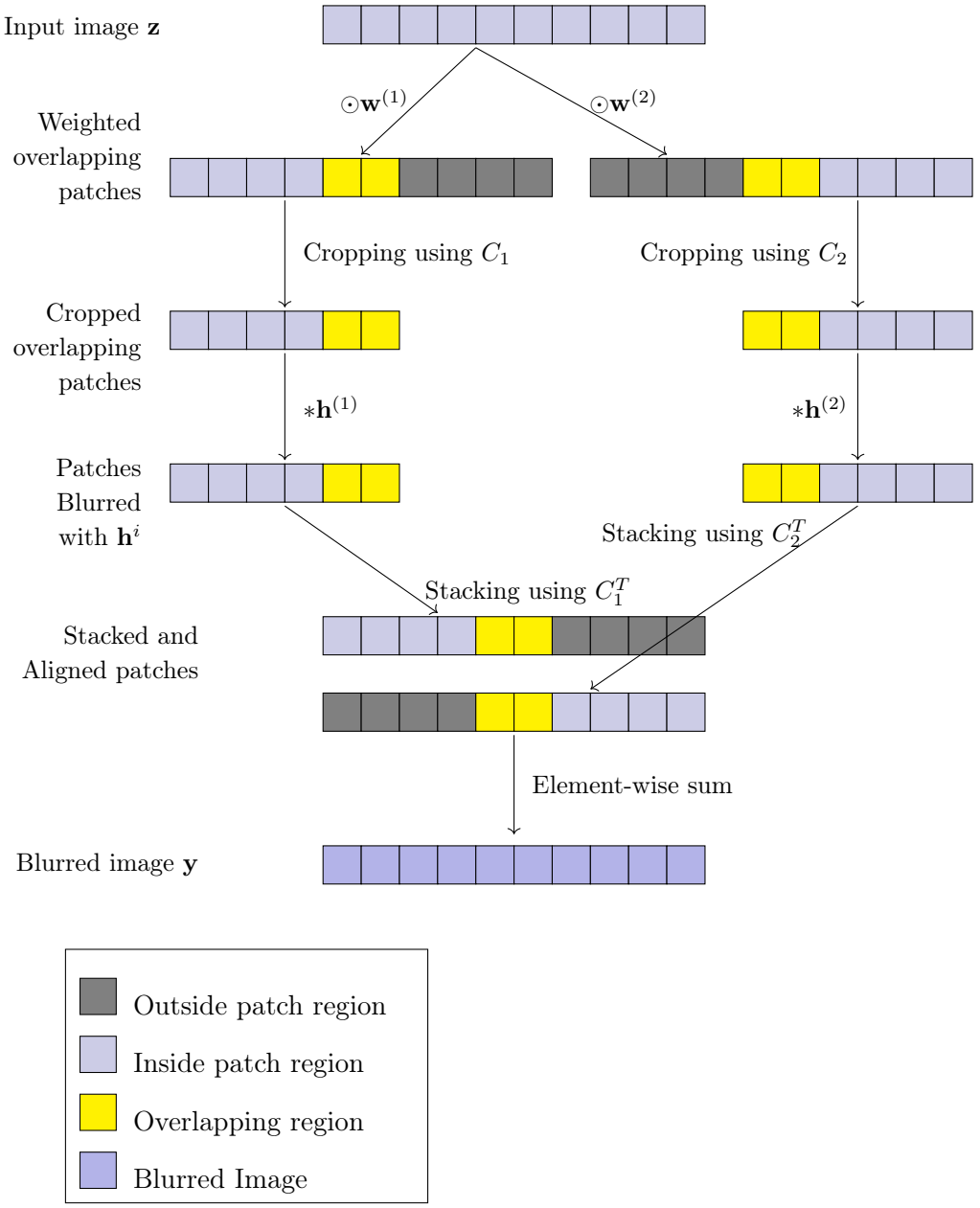


FIGURE 3.1: Example showing steps involved in the implementation of operator H using the EFF Framework.

As before, the transpose operator can be efficiently implemented by reading Equation 3.23 from right to left. The difference here is that the weighting is applied as the last operation and the blur kernel appears as the complex conjugate $\overline{\text{Diag}(\mathcal{F}(\mathbf{h}^{(r)}))}$ (see Appendix 5.2 for detailed derivation). Lastly we note that, although we have shown the cropping operation as a matrix (C_r and inverse cropping operation as C_r^T), no actual matrices are involed in the implementation it as the operations involved can be efficiently implemented by only knowing the dimensions of the image, the number of overlapping patches in which the image is divided and the extent of overlap between each patch.

3.3 Experiments and Results

In this section, we present the results of our experiments on simulated as well as real world data. The benefit of experimenting with synthetic data is that it is easier to perform experiments under a wide range of scenarios and parameter settings. For example we can easily control the noise level and study the effect on performance. It also allows us to study the effect of the parameters ρ and T on the proposed algorithm. We describe our procedure to set these parameters next.

3.3.1 Parameter Settings

We fix the number of iterations of Algorithm 2 by setting $N_{\mathcal{SR}} = 10$. Recall that $\mathcal{AG}$ is also an iterative solver and we fix the number of itreations of $\mathcal{AG}$ by setting $N_{\mathcal{AG}} = 30$ in Algorithm 1. This means after every 30 iterations of the deblurring step, we perform one iteration of the joint denoising and super-resolution step. We got good and sharp results and the algorithm remained stable for a wide range of noise levels with this setting. The most important parameter which is crucial to get a good performance out of the algorithm is ρ. As noted earlier, ρ controls the extent of denoising in the joint denoising and SR network $\mathcal{SR}$ with the extent of denoising proportional to $\sqrt{1/\rho}$. ρ also controls the extent to which the equality constraint is enforced in the deblurring step of Algorithm 2. We found that a simple scheme to update ρ between the iterations works well for a wide range of noise levels. We initialize $\sqrt{1/\rho} = 50$ which is a relatively high value. After each iteration, we gradually decrease the value on an exponential scale with the final value being $1/\rho = \sigma/2$ where σ is the noise level of the image. The updates of the parameter are depicted in Figure 3.2

3.3.2 Super Resolution on Simulated Data

We generate a set of 9 anisotropic and rotated Gaussian kernels as shown in Figure 3.3. The kernels are arranged in a 3x3 grid. The image is also divided into a 3x3 grid of overlapping patches and the blur kernel falling under each patch affect that particular patch only. Together, the set of kernels define the spatially varying blurring operator H. Following the image formation model in Equation 3.2, we start with a high quality image assumed to be the HR image. We use 25 images from the validation set of the DIV2K dataset for this experiment which were not used for training. The HR image is first downsampled by the fixed downsampling operator D. We fix D to be Matlab's

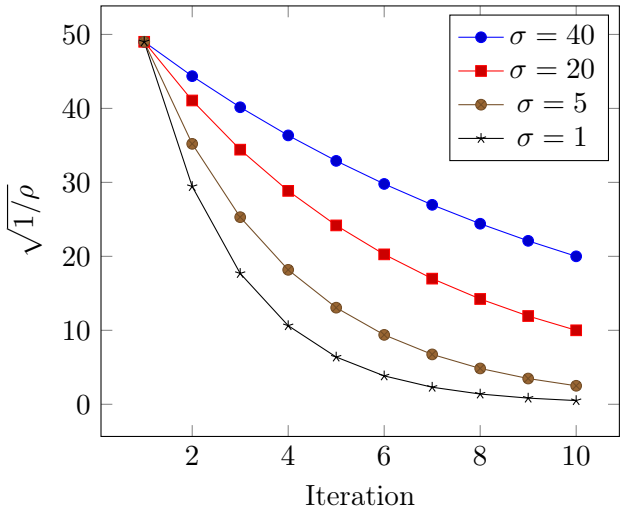


FIGURE 3.2: Parameter updates for each iteration of the ADMM at different noise levels.

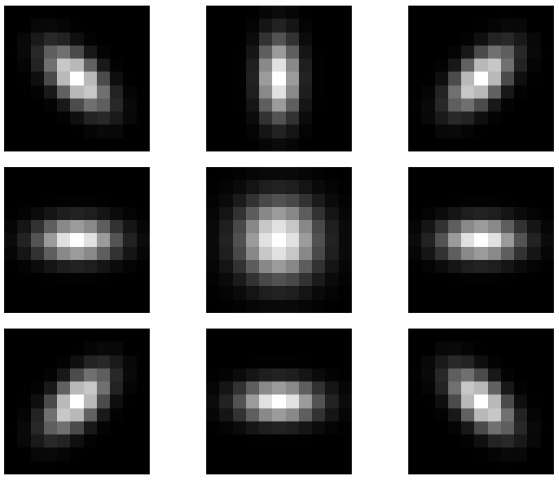


FIGURE 3.3: Spatially varying blur kernels

default `imresize` function and call this image the clean downsampled image. Next, the downsampled image is blurred using the operator H and we call this the blurred image. Finally, additive white Gaussian noise is added to the blurred image which gives the LR image. We then super-resolve the LR image using the algorithm proposed in the previous section.

The PSNR and SSIM values for the super-resolved images are shown in Table 3.1. For comparison we also upscale the LR image using Bicubic upscaling. We also super-resolve the image using only the $\mathcal{SR}$ network without the deblurring step and show the PSNR and SSIM values under the column No-Deblur in Table 3.1. This serves as the baseline approach. Our approach produces consistently better results over all scale factors and noise levels. The gain in the performance is more significant at lower noise

Degradations		Average PSNR/SSIM		
Scale Factor	Noise Level	Bicubic	No-Deblur	**Ours**
x2	0	25.6840/0.7572	24.5201, 0.7106	28.2599/0.8156
	5	25.4024/0.6987	24.5204/0.7102	27.6231/0.7860
	20	22.7840/0.3747	24.4745/0.7060	26.1024/0.7439
	40	19.3182/0.1859	24.1195/0.7014	24.3021/0.7025
x3	0	24.0393/0.7121	23.0131/0.6797	26.2921/0.7574
	5	23.8036/0.6532	23.0131/0.6792	25.8432/0.7397
	20	21.4550/0.3453	22.9877/0.6754	24.5121/0.7047
	40	18.1578/0.1698	22.7479/0.6710	22.9314/0.6718
x4	0	22.9269/0.6858	21.9514/0.6611	25.02656/0.7217
	5	22.7701/0.6481	21.9518/0.6605	24.6741/0.7107
	20	21.0295/0.4050	21.9441/0.6570	23.3811/0.6799
	40	18.2183/0.2233	21.7896/0.6533	21.9272/0.6534

TABLE 3.1: Average PSNR and SSIM values for results on the DIV2K validation set at different noise levels and scale factors obtained using Bicubic upsampling,

levels than that at higher noise levels. For example for scale factor x2, the gain in performance in terms of PSNR/SSIM is around 4dB/0.1 and 3dB/0.07 at noise levels 0 and 5 respectively. The gain in performance decreases to 2dB/0.03 and 0.2dB/0.001 at noise levels 20 and 40 respectively as compared with the baseline No-deblur approach. Figure 3.4 shows the results visually for one of the images in the test set. Here we can see that our approach produces images with crisp and fine details and low blurring at low noise levels. At higher noise levels higher denoising strength is needed to remove the noise from the images. This results in images that have more smooth regions and less texture and fine details. This explains the disparity in performance gain at different noise levels.

3.3.3 Convergence and Running Time

The algorithm converges fairly rapidly in practice for a wide range of noise levels using the universal parameter settings. We show this empirically for one of the images in the test set by plotting the value of the objective function at each iteration in Figure 3.5. We can see that the algorithm converges rapidly and remains stable for a range of noise levels.

We implement both parts of our algorithm i.e., the joint denoising and super-resolution network $\mathcal{SR}_D$ and the accelerated gradient solver $\mathcal{AG}$ in PyTorch. Although PyTorch is designed as a library to rapidly prototype and train neural networks, it also provides most of the operations we need like array manipulation and Fast Fourier Transform computation with default GPU support. Thus, it is well suited for our algorithm. The

FIGURE 3.4: The first image (top left) is image number 865 in the DIV2K validation set which we consider as the ground truth HR image. It is degraded with spatially varying blurs and additive white Gaussian noise at different noise levels which are shown in the right column. The corresponding super-resolved images are shown in the left column.

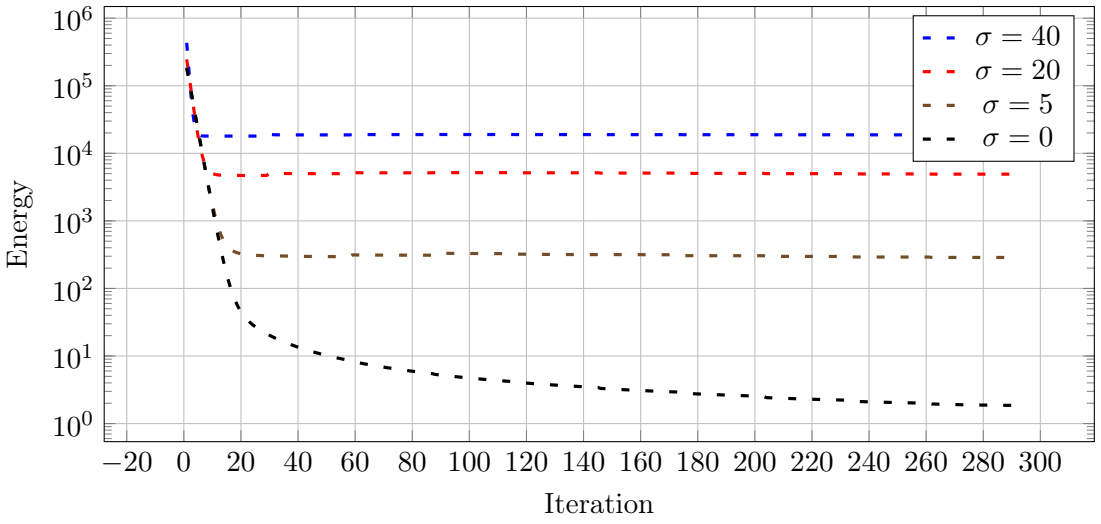


FIGURE 3.5: Convergence of our algorithm on image 865 of the DIV2K validation set at different noise levels.

algorithm takes around 50 seconds on a workstation equipped with a Nvidia 1080Ti, 11GB VRAM, Intel Xeon E5-1650@3.60GHz CPU on images of size 1024x768.

3.3.4 Real-world Super Resolution

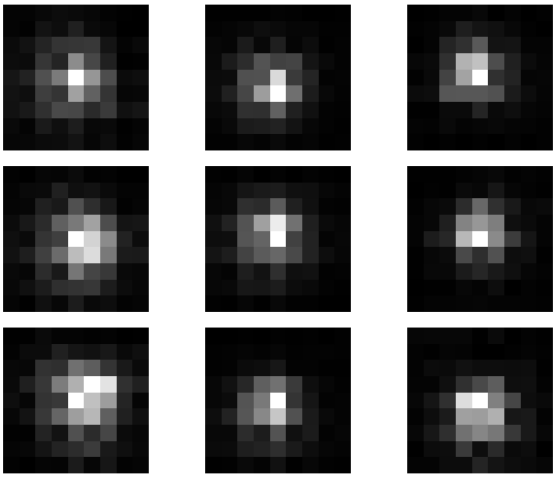


FIGURE 3.6: Experimentally measured spatially varying PSF of an optical system

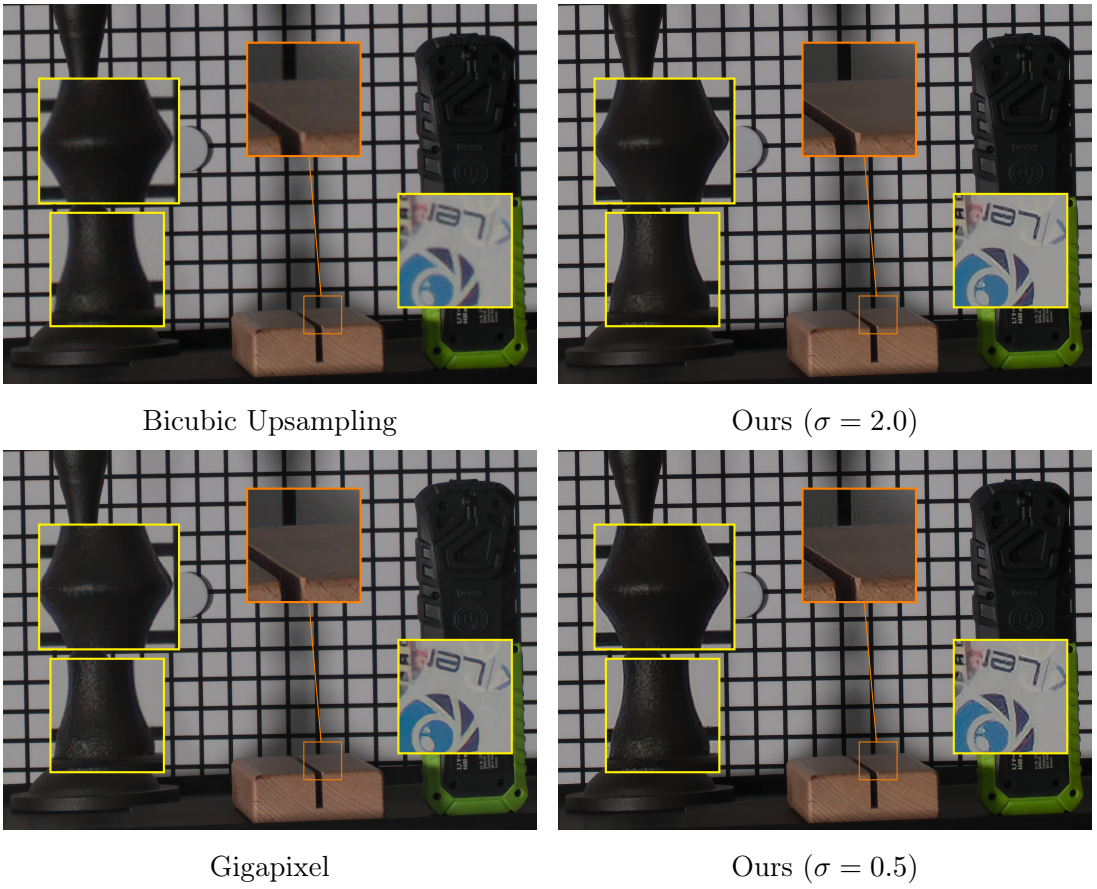
The camera setup for our real world experiments consists of a Nikon D810 camera. The lens we use is a multi-view lens called the K-Lens. K-Lens [69] allows imaging of 9 different perspectives of a scene using a single shot. The different perspectives can be used for a host of post-processing tasks and applications like depth estimation, post capture focus etc. Since the sensor now captures 9 different perspectives (sub-views), the number of pixels in each sub-view is $1/3^{\text{rd}}$ of the full sensor. Our goal is to enhance the resolution of each sub-view with our proposed algorithm. We first focus on the center

view for this experiment. We start with measuring the point spread functions of the camera-lens setup. To do this, we image a point source of light (white LED light around a covering with a single $30\mu m$ hole) in a dark room using the camera setup. To obtain estimates at different spatial postions, we image the point source on a regular 3x3 grid. We set the exposure time to 1/6 seconds and ISO to 100. To extract the PSF, we crop a window of 9x9 pixels around the brightest point in each image. This window becomes the PSF at that position and the measured PSFs from our experiment are shown in Figure 3.6. Using the same camera setup as above we also image some scenes for testing.

Qualitative results from this experiment are shown in Figures 3.7 and 3.8. The images labeled Gigapixel were produced using a commercial software from Topaz Labs called Gigapixel. We show these images from Gigapixel for comparison. Since no ground truth data is available for real scenes, we have to rely on visually assessing the results. We show results from setting $\sigma = 2.0$ and $\sigma = 0.5$ for our approach. It can be seen that the super-resolved images produced by our approach look sharper and more crisp. This is especially evident around the checkerboard patterns. Furthermore, noise present in the original images is also reduced especially for the setting with $\sigma = 2.0$. The results from Gigapixel are less sharp than our results. However, the textures produced by Gigapixel seem to be richer and more detailed. This demonstrates the effectiveness of our approach on real world images with known noise levels and known arbitrary spatially varying point spread functions.

3.4 Summary

Using the ADMM algorithm, we devise an approach for SISR for images degraded with arbitrary spatially varying blur kernels and additive white Gaussian noise. On a high level, the algorithm consists of performing two steps iteratively to arrive at the final super-resolved image. The first step consists of removing the spatially varying blurs. We provide an efficient method to achieve this goal. The second step consists of joint denoising and super-resolution. Our formulation allows using a single deep neural network trained for a fixed downsampling operator and different noise levels to be used for this task. Using experiments on synthetic data, we show that our approach out-performs the baseline approaches by significant marigins for different noise levels and scale factors. We perform experiments on real world images and demonstrate that the super-resolved images from our approach are visually pleasing and contain sharp and crisp details.



Bicubic Upsampling | Ours ($\sigma = 2.0$)

Gigapixel | Ours ($\sigma = 0.5$)

FIGURE 3.7: Qualitative results on a real world scene. Best viewed on a high resolution display by zooming in.

The noise level parameter which controls the strength of denoising should be high enough to prevent enhancement of noise artifacts in the super-resolved images. However, if the parameter is too high, the textures can be smoothed over. Furthermore, the amount and characteristics of noise varies within real world images. We address the problem of oversmoothing of textures and better noise characterization in the next chapter and extend our approach in this chapter to have excellent performance on a wide variety of real world images.

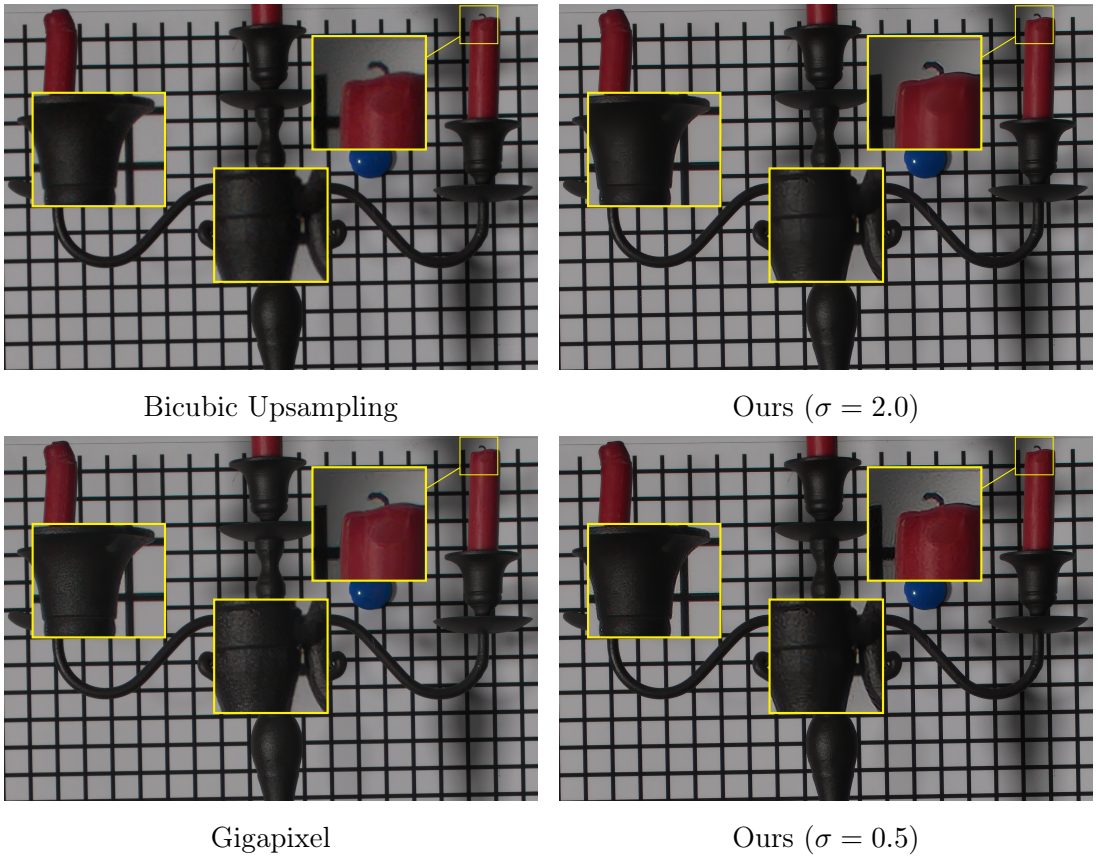


FIGURE 3.8: Qualitative results on a real world scene. Best viewed on a high resolution display by zooming in.

Chapter 4

Learning Denoising, Deblurring and SR all Together

In the previous chapter, we tackled the problem of blur in the context of super-resolution. We showed that removing blurs results in better super-resolved images. Our super-resolution algorithm works for arbitrary spatially varying blurs. We also showed that accounting for noise is essential to get good super-resolved images. If the noise is left unchecked, then it can be enhanced and appear as undesired artifacts in the super-resolved image. We assumed the additive noise in the image to be independent for each pixel. Furthermore, we also assumed that noise at each pixel follows a Gaussian distribution with zero mean and a certain standard deviation. The independent additive white Gaussian noise assumption is nice because of ease of formulation and analysis and suffices for some applications. However, this characterization of noise turns out to be too simplistic when it comes to real world images [70]. Low level computer vision tasks like denoising and super-resolution can therefore benefit from incorporating more realistic noise models.

In this chapter, we incorporate realistic noise models for synthesizing low resolution noisy images from high resolution images. We use the images to train CNN models for the task of super-resolution. We perform experiments on real world images to show that our models yield good performance on a wide variety of real scenes. The rest of the chapter is organized as follows:

- In Section 4.1, we provide a review of realistic noise models that have been successful for the task of denoising real-world images.
- In Section 4.2, we explain our method to synthesize low resolution and high resolution images which we use to train CNN models for super-resolution.

- In Section 4.4 we provide details of our models, training and experiments on real world images.

4.1 Realistic Noise Models

The noise which degrades a color image that we finally see originates and is transformed at the various stages of processing that are performed to arrive at the desired color image. The process of recording an image of a scene starts at the photosites of a sensor which measure scene irradiance. The photosites are arranged in a two dimensional grid that constitutes the whole sensor. Each photosite counts the number of photons incident on it. Photon counting is a classical Poisson process [70] and the uncertainty of the process gives rise to *photon noise* in the images. The number of photons counted by each photosite can therefore, be modeled by a Poisson distribution. The probability mass function of the Poisson distribution is given by

$$P(x = N) = \frac{\lambda^N e^{-\lambda}}{N!}, \tag{4.1}$$

where N is the count of photons at the photosite and λ is a parameter of the distribution that gives the expectation of the distribution. It is equal to the actual number of photons incident on the photosite and is therefore proportional to the scene irradiance. The amount of photon noise is given by the variance of the Poisson distribution [70]. Poisson distributions have the property that their variance is equal to their expectation. Therefore, the amount of photon noise is also proportional to the scene irradiance. Photon noise constitutes the signal dependent part of noise in real world images. In modern digital camera sensors which are predominantly manufactured using the CMOS fabrication process, photon noise is the performance limiting noise component [71]. In practice, the photon noise component is modeled using a heteroskedastic Gaussian [72] such that:

$$N \sim \mathcal{N}(\lambda, \lambda) \tag{4.2}$$

The photon counts are stored as charge at each photosite which accumulates during the time period for which the sensor is exposed. Eventually, charge is converted into voltage, it is amplified, read out of the sensor, digitized and stored on the camera storage. The data at this point constitutes the *RAW* sensor image. The processes associated with amplification, reading and digitizing also introduce noise in the data. Together, this noise is usually termed *read noise*. Foi et al. [72] assume that read noise is signal independent and model it using a zero mean Gaussian distribution. They propose the

Poisson-Gaussian model for noise in the formation of RAW images given by

$$\mathbf{r} = \mathbf{x} + \mathbf{n}, \tag{4.3}$$

where $\mathbf{r}$ is the noisy RAW image, $\mathbf{x}$ is clean image and $\mathbf{n}$ is additive noise. The noise $\mathbf{n}$ in their model is assumed to follow a heteroskedastic Gaussian distribution i.e

$$\mathbf{n} \sim \mathcal{N}(0, \sigma^2(\mathbf{r})). \tag{4.4}$$

The variance of noise $\sigma^2(\mathbf{r})$ depends on the irradiance of the scene. It is given by

$$\sigma^2(\mathbf{r}) = a\mathbf{r} + b, \tag{4.5}$$

where a and b are the parameters that determine the strength of the signal-dependent photon noise and signal-independent read noise respectively. The values of a and b depend on factors like the quantum efficiency of the sensor which determines how efficiently the sensor converts incident photons into charge, analog gain which is used to amplify the voltages and is determined by the ISO setting on the camera, the pedestal or the base charge that is always present in the sensor etc. For a more detailed discussion on parameters a and b, we point the interested reader to [72, 73]. The noise model of Foi et al. has been used to synthesize noisy RAW images from clean images for training denoising CNNs [73] and achieves good denoising performance on real world noisy images [74].

The RAW sensor image is further transformed by the camera image signal processor (ISP) using several steps to arrive at a display ready sRGB image. For example, the RAW sensor image is gamma corrected and demosaiced. Demosaicing converts the single channel RAW data into three channel RGB data. The demosaicing step makes the noise spatially and chromatically correlated [75]. Several other processing steps like tone mapping, white balancing, color correction and compression may optionally be also applied to finally arrive at the display ready sRGB image. The net effect of all these steps is that the noise distribution present in the RAW images is heavily transformed by the image processing pipeline. Therefore, the noise model of Foi et al. [72] which has been successful for real world RAW denoising [74] cannot be directly applied to sRGB images to synthesize paired noisy and clean images. To overcome this problem recent works have taken the approach of inverting clean sRGB images to RAW images, injecting noise according to the mixed Poisson Gaussian model of Equation 4.3 and converting the raw images back to sRGB images.

Guo et al. [76] convert sRGB images to RAW images by inverting each step of the camera ISP. They model the RAW to sRGB conversion using the equation

$$\mathbf{y} = \text{JPEG}(\text{DM}(f(\mathbf{r}))), \tag{4.6}$$

where $\mathbf{r}$ is the raw sensor image which is first processed by the camera response function f that includes gamma compression. DM denotes the demosaicing operation which is performed next. Finally the image may optionally be compressed using the JPEG compression algorithm. They perform the inverse of each operation on the sRGB image to arrive at the RAW image:

$$\mathbf{r} = f^{-1}\text{DM}^{-1}(\mathbf{y}), \tag{4.7}$$

where DM^{-1} denotes the inverse of the demosaicing operation i.e the mosaic operation which converts the sRGB image into a one channel bayer image. A simple method to implement it is by alternatively selecting values from the B,G,G,R channels to arrive at the BGGR bayer pattern. This bayer image is then operated upon by the inverse of the camera response function f^{-1}. The camera response function varies from one camera to another and sometimes it is not even known because the manufacturers of the camera may not make it available publicly. To overcome this problem, they use a set of 201 camera response functions provided by [77]. At this point, they end up with a clean RAW image. They now inject noise using the Poisson-Gaussian model with different levels of Poisson and Gaussian noise into the RAW image. Next, they apply the camera response function f that they picked from the set and apply demosaicing to get the noisy sRGB image following Equation 4.6. So they start of with a clean sRGB image and by inverting the camera ISP, injection noise and applying the same camera ISP functions, they end up with paired noisy and clean sRGB images with realistic noise. They use the paired clean and noisy images to train a denoising CNN. Their method achieved state of the art performance on the DND [74] benchmark for blind sRGB denoising at that time.

Zamir et al. [78] build on the techniques of Guo et al. however, they take a different approach to the problem of converting sRGB images to RAW space. They propose CycleISP, a device agnostic framework which does not require any knowledge of camera parameters or camera response functions on which which the previous approaches like [76, 73] rely on. The framework consists of two CNNs. The first CNN which they call the RGB2RAW network is trained to convert sRGB images to RAW space. This network is trained using RAW and sRGB images from the MIT-Adobe 5k [79] dataset which contains RAW images from 4 DSLR cameras and corresponding processed sRGB images. The second CNN is trained to map RAW images back to sRGB images as shown in the

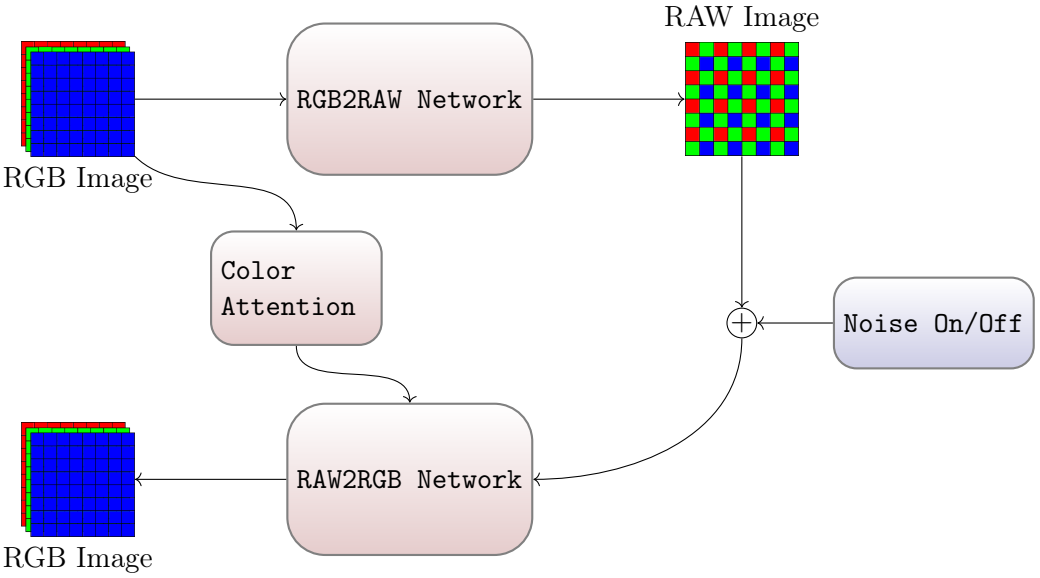


FIGURE 4.1: The RGB2RAW network takes as input an RGB image and produces a RAW image. Noise can optionally be injected into the RAW image according to the Poisson-Gaussian noise model. The noisy image is mapped backed to sRGB space using the RAW2RGB network.

bottom part of Figure 4.1. A RAW image can be mapped to an arbitrary number of sRGB images by changing any of the processing steps like gamma compression, white balance, tone mapping etc. To overcome this problem and to guide their RAW2RGB network towards learning to generate the desired sRGB image, they additionally propose a color correction module. This is also a CNN that takes as input the low pass filtered sRGB image and maps it into a higher dimensional feature space. The mapped features from the color attention network are then fused with the features of the RAW2RGB network. They mention that this constitutes passing only color information which guides the network towards the desired sRGB image. Their framework is depicted in Figure 4.1. Once both the networks are trained, they can generate arbitrary number of paired clean and noisy sRGB images. To do this, they start from a clean sRGB image and feed it to the RGB2RAW network to get the clean RAW image. They now switch on the noise injection module (Figure 4.1). This module injects noise according to the Poisson-Gaussian model with different levels of Poisson and Gaussian components. The noisy RAW image is fed into the RAW2RGB to get the noisy sRGB image with realistic noise characteristics. This allows them to generate an arbitrary number of paired clean and noisy images and simplifies the problem of large scale data generation to train CNNs significantly. Although, they learn the color to RAW and the reverse mapping only using paired RAW and color images taken from a few specific cameras, their method still generalizes well for all kinds of color images for generating paired noisy and clean data. This can be explained using the following reasoning: To generate a noisy color image given a clean color image with for the purpose of training CNNs, we need to know how to

map the color image to *a RAW image*. It does not matter if the RAW image is different from the one that actually produced the color image. We also need to know how to map this RAW image back to the color image. Then by injecting noise into the RAW image and mapping it back to the color image, we can generate paired clean and noisy color images with realistic noise characteristics from any color image. At the time of writing, their method was the state of the art on the DND real world denoising benchmark [74].

To summarize, simple noise models like additive white Gaussian noise do not suffice for real-world images because the noise in real-world color images is complicated. Therefore, the CNN models trained on synthetic data generated using simple Gaussian noise models do not work well on real-world images. Collecting paired noisy and clean images is a tedious process and not suited for large scale learning either. The noise characteristics of RAW images on the other hand are well understood and accurately modeled by the Poisson-Gaussian model. To overcome the problem of generating realistic noisy and clean color images recent methods like [73, 76, 78] have proposed to first convert color images into RAW images, inject noise according to the Poisson Gaussian model into the RAW images and then convert the RAW images back to color images. Methods based on this idea give state of the art performance on real-world denoising benchmarks [74] and also generalize well to a wide range of real world images.

Accurate characterization of noise in images is important to avoid enhancing noise and loosing details in the task of super-resolution. Since the idea of converting color images to RAW space, injecting noise according to well understood characteristics of noise in the RAW image and then converting the noisy images back to color images has proven to be useful for real world denoising problem, it can also be utilized to synthesize data for the super-resolution problem. In the next section, we explain our method which incorporates this idea to synthesize realistic data for super-resolution.

4.2 Realistic Data Synthesis For Super-Resolution

A significant amount of research effort has been put into designing neural network architectures for super resolution [14, 42, 44, 15, 63]. These networks perform well on synthetic benchmarks however, on real world images their performance benefits diminish significantly [21]. The reason is that neural networks require large amounts of training data consisting of paired low resolution and high resolution images. Acquiring such data for real world images is a time consuming and tedious process. Therefore, traditionally, these methods have relied on synthetic data. The general approach is to use images

from existing datasets like DIV2K [80] and Flickr2K [81] and synthesize low resolution images by using simple bicubic downsampling. This procedure tends to remove noise and blurring which makes the synthesized low resolution images different from the low resolution images that are seen in the real world. The mismatch between the real and synthesized low resolution images causes neural networks trained on such images to fail.

Our goal is to synthesize low resolution images such that the mismatch between synthesized low resolution images and real world low resolution images is minimized. To achieve this goal, we aim to match the blurring and noise characteristics found in real world images. We assume that a dataset containing good quality images of a wide variety of scenes is given (several datasets are available freely [81, 80]). Since the images in these datasets can also contain noise, we first have to cleanup these images which gives the clean HR image. For each high resolution image, we perform a sequence of steps to arrive at the desired low resolution image. The steps consists of a blurring operation with certain blur kernels, subsampling and addition of realistic noise. We explaine each of these steps in complete detail next with the help of Figure 4.2 which depicts the overall procedure.

The first step is a simple cleanup step to obtain the HR images from a dataset containing high quality images. To achieve this we simply blur the image with a 3x3 gaussian blur kernel followed by initial downsampling. This gives us the clean HR image and let this image be denoted by $\mathbf{x}$. The HR image is blurred to obtain a blurred image denoted by $\mathbf{x}_{\text{blurred}}$ by convolving $\mathbf{x}$ with a filter $\mathbf{h}$ i.e

$$\mathbf{x}_{\text{blurred}} = \mathbf{x} * \mathbf{h}. \tag{4.8}$$

The filter is obtained by randomly selecting one filter from the filter bank. The filter bank is a set of filters consisting of measured PSFs as described in the previous chapter. Additionally, we augment the filter bank with 9x9, 7x7 and 5x5 Gaussian filters. This gives the clean but blurred image $\mathbf{x}_{\text{blurred}}$ as shown in Figure 4.2. $\mathbf{x}_{\text{blurred}}$ is then downsampled with the desired scale factor s giving the downsampled image denoted by $\mathbf{y}_{\text{s}}$ where s determines the factor with which we ultimately wish to super-resolve the images. The next step consists of injecting realistic noise into $\mathbf{y}_{\text{s}}$ to finally arrive at the desired low resolution image. To achieve this we first convert $\mathbf{y}_{\text{clean}}$ from sRGB to RAW space. The `RGB to RAW` block depicted in Figure 4.2 performs the conversion which gives the downsampled raw image $\mathbf{r}_s$. Next, we generate noise according to the Poisson-Gaussian noise model of Foi et el.[72] given in Equation 4.4. The noise vector $\mathbf{n}(\mathbf{r}_s)$ is generated

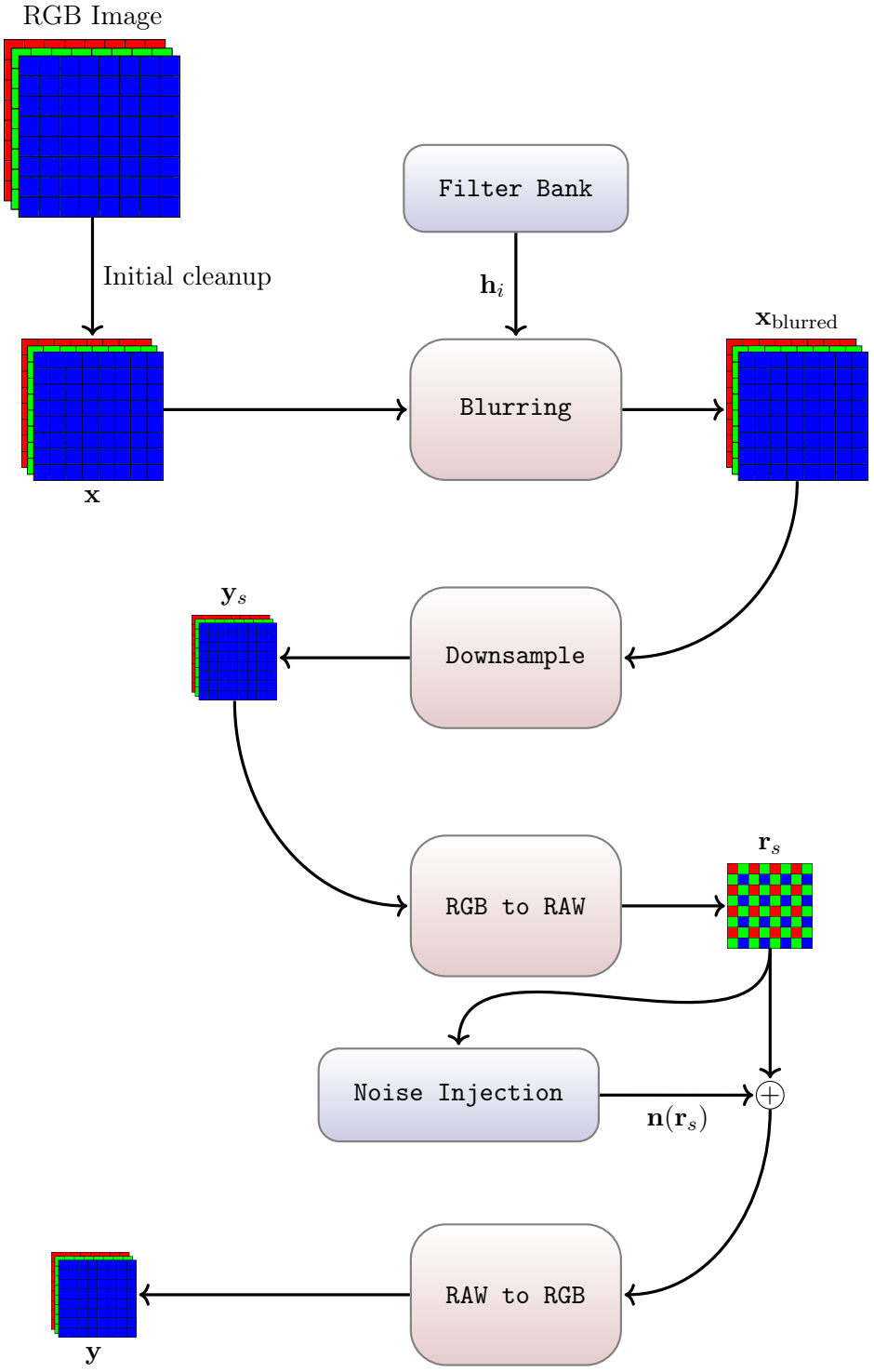


FIGURE 4.2: Procedure to synthesize realistic data for super resolution.

by sampling from a heteroskedastic Gaussian distribution i.e

$$\mathbf{n}(\mathbf{r}_s) \sim \mathcal{N}(0, \sigma^2(\mathbf{r}_s)), \tag{4.9}$$

where $\sigma^2(\mathbf{r}_s)$ denotes the variance of the Gaussian distribution which is a function of the RAW image and is given by

$$\sigma^2(\mathbf{r}_s) = a\mathbf{r}_s + b. \tag{4.10}$$

The parameter a determines the amount of the Poisson noise component and the parameter b determines the amount of the Gaussian noise component. The values of a and b should be determined by the amount of noise present in the target images that we want to super-resolve. However as shown by Guo et al. [76], deep convolutional neural networks can effectively denoise images containing noise with different noise levels if the network is also trained for different noise levels. Therefore, we also generate images containing different levels of Poisson and Gaussian noise by sampling a and b from a sensible range of values (see Section 4.4 for details about the range). The last step consists of converting the noisy RAW image back into sRGB space which finally gives the noisy low resolution sRGB image $\mathbf{y}$ corresponding to the high resolution image $\mathbf{x}$. Figure 4.3 shows an example of a high resolution image and the corresponding synthesized low resolution image with realistic blur and noise characteristics with this method. For the conversion steps i.e to convert the downsampled sRGB image to RAW space and then back to the sRGB space, we used the CycleISP [78] of Zamir et al. depicted in Figure 4.1. We note that our method should work with any other method for sRGB to RAW conversion.

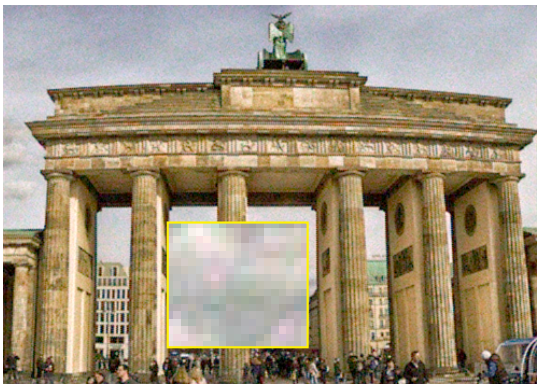
4.3 Network Architecture

We avoid complicated architectural engineering because a significant amount of research has been done in designing efficient architectures for super resolution. For all our experiments, we use the Residual in Residual Dense Network (RRDBNet) proposed in [16] because it is among the state of the art. The network design follows the established conventions. The first part of the network consists of an initial convolution layer to transform the image into the feature space. This is followed by several basic blocks where most of the computation takes place. The resulting features are upsampled using a convolution transpose layer. The upsampled features are compressed to 3 channels via the final convolution layer giving the super resolved image. The architecture of the network is depicted in Figure 4.4.

The basic block is the Residual in Residual Dense Block (RRDB). It is composed of three Residual Dense Blocks (RDB) with skip connections in between. The skip connections are achieved by adding the input feature maps to the output feature maps of each block and therefore having a path which *skips* the block as depicted in Figure 4.5. Skip connections ensure that a block has to learn only the *residual* mapping from the

(a) Image 512 from DIV2K dataset.

(b) Clean image $\mathbf{x}$



(c) Noisy image $\mathbf{y}$ ($a = 0.0016$ and $b = 0.00016$)

(d) Noisy image $\mathbf{y}$ ($a = 0.0122$ and $b = 0.00068$)

FIGURE 4.3: A high quality image taken from the DIV2K data set is shown in (a). The cleaned up high resolution image is shown in (b). Low resolution i.e blurred and noisy images with different levels of noise are shown in (c) and (d).

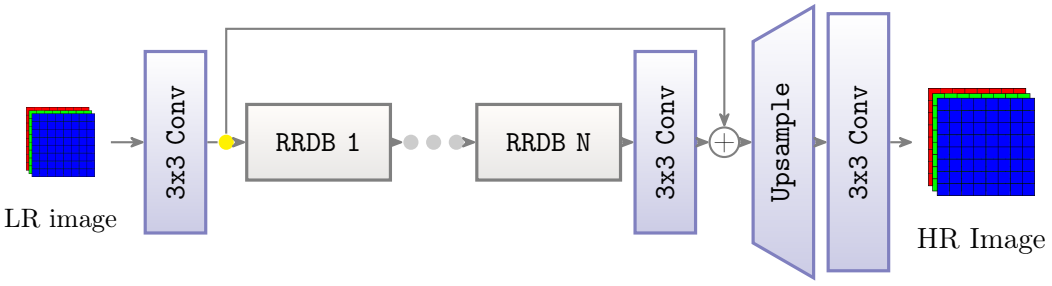


FIGURE 4.4: Architecture of RRDBNet. The network consists of an initial convolution layer followed by a series of Residual in Residual Dense blocks to extract features. Finally the features are upsampled and compressed to give the super-resolved image.

input and thus enable training very deep networks i.e. networks with several convolution layers. Scaling the values of the feature maps by a constant between 0 and 1 before applying skip connection to the input of the block stabilizes training because with a large number of layers and corresponding skip connections, the values in the feature map can become very large.

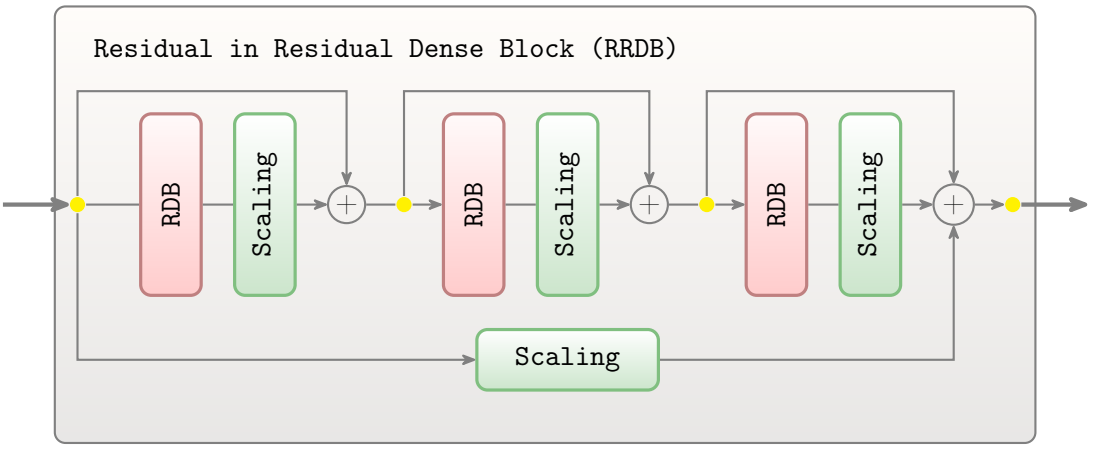


FIGURE 4.5: Basic block of RRDBNet. It consists of of three Residual Dense Blocks with skip connections. Scaling simply multiplies the values in the feature map by a constant (0.2 for our experiments) before addition.

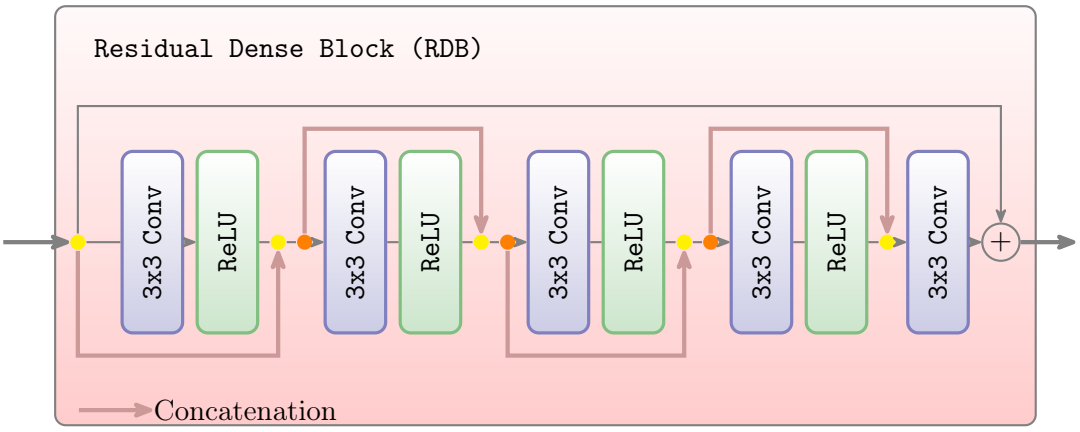


FIGURE 4.6: The RDB consists of four convolution layers. Dense connections are achieved by concatenating outputs of all previous layers. Finally a skip connection to the input of the block is applied.

The Residual Dense Blocks (RDBs) which make up the basic block of the network are composed of 4 convolution layers each followed by a ReLU[57] non-linearity given by

$$\text{ReLU}(x) = \begin{cases} x & \text{if} \quad x > 0, \\ 0 & \text{otherwise} \end{cases}$$

The output of each convolution layer is concatenated with the output of all previous layers within the block including the input which becomes the input to the next layer. This makes the layers in the block *densely connected*. The architecture of a single RDB is depicted in Figure 4.6. The concatenated outputs of all the convolution layers within the dense block are finally compressed using a final convolution layer. This is followed by a skip connection to the input of the block for *residual* learning.

4.4 Experiments and Results

4.4.1 Dataset Generation

For all our experiments we use our method described in Section 4.2 to synthesize the training data. We source high quality sRGB images by combining the DIV2K [80] and Flickr2K [81] datasets. The resulting DF2K [19] dataset contains 3450 training images and 100 validation images. We fix the inital cleanup downsampling factor to 2. For converting sRGB images to RAW space we use the RAW2RGB network of CycleISP [78] as described in Figure 4.1. We add noise according to the Poisson-Gaussian noise model of Equation 4.10 into the raw images by sampling a and b uniformly from a range of values $[a_{\min}, a_{\max}]$ and $[b_{\min}, b_{\max}]$ respectively. Recall that a controls the amount of Poisson noise component and b controls the amount of Gaussian noise components. Finally we convert the low resolution noisy RAW imgae back to sRGB space using the RGB2RAW network of CycleISP. The entire procedure is demonstrated in Figure 4.2.

4.4.2 Training

The next step is to train a network for super-resolution using the generated dataset. We use the architecture of RRDBNet [16] with 23 RRDBs described in Section 4.3 as the main SR network in all our experiments. The images are cropped to a size of 128x128 during training and data augmentation consisting of random horizontal and vertical flips is also applied for better generalization. The network is optimized by minimizing the $\mathcal{L}_1$ between the network output and the ground truth HR image. We implement the network in PyTorch and use the built in Adam optimizer with $\beta_1 = 0.9$ and $\beta_2 = 0.99$ and an initial learning rate of 2×10^{-4} for optimization. We set the batch size to 24 and train the network for 500 epochs. Training the network of one Nvidia Quadro 6000 RTX takes around 10 hours.

4.4.3 Effect of Noise

To study and understand the effect of noise on the quality of super-resolution we perform an experiment in which we train two networks. Let us call these networks `model1` and `model2`. The network architecture and training parameters are the same as described in the previous section for both models. The models differ in that the synthesized data used to train the two networks contains different amount of noise. We train `model1` on data for which the noise parameter a is uniformly from the range $[0.0001, 0.032]$. Similarly, the noise parameter b is uniformly sampled from the range $[10^{-6}, 0.025]$. We

select this range because it covers a wide range of ISO levels on different cameras [74]. We also found that empirically these noise ranges gave good performance on real world images. Similarly, we train `model2` with a sampled uniformly from the range $[0.0001, 0.01]$ and b sampled from range $[10^{-6}, 0.012]$ i.e the upper limit of noise levels that `model2` is trained on is half of the upper limits for `model1`.

To evaluate the performance of the models and compare them, we synthsize low resolution and high resolution images for a range of noise levels. We source the images from a subset of the validation set of DIV2K dataset which was not used in training the two models. For each image, we synthesize 50 low resolution images by gradually increasing the amount of noise injected into the images. Specifically we first fix a low level of the Gaussian component which models the read noise by setting $b = 0.0001$. We gradually increase the amount of the Poisson noise component by increasing a from 0.0001 to 0.1 in 50 steps. Then we use `model1` and `model2` to super-resolve each image. This allows us to study the effect of noise on the performance. Since we rely on synthesized low resolution images, we also possess ground truth high resolution images and thus we can quantify the performance of each model using metrics like PSNR, SSIM which measure similarity between the processed and the ground truth image. We also use the LPIPS [82] metric which is calculated by comparing the features of the processed image with the features of the ground truth image. The features are obtained using pre-trained CNNs like Alexnet [83] for visual recognition tasks. LPIPS has been shown to correlate well with human perception of similarity [82]. As an example, the ranking based on mean opinion scores obtained from human evaluation for the performance of various super-resolution methods in the NTIRE2020 super-resolution challenge [23] very closely matches the rankings obtained by LPIPS scores.

The results of our experiment are shown in Figure 4.7. The figure shows the performance of `model1` and `model2` in terms of LPIPS, SSIM and PSNR at various levels of the Poisson noise components with the Gaussian component fixed at a low level in the first column. Similarly, in the second column, it shows the performance of of `model1` and `model2` at various levels of Gaussian noise component with Poisson component fixed at a low level. The first conclusion we can draw from Figure 4.7 is that the performance of both the models deteriorates with increase in the level of noise (be it Poisson or Gaussian). This is expected because it is harder to super-resolve images with higher amounts of noise. We can also observe that `model1` outperforms `model2` for higher levels of the Poisson noise component. This is evident from the first column of the figure where `model1` consistently outperforms `model2` for values of a beyond 0.01 (first row left part) by achieving a lower LPIPS score and higher SSIM and PSNR. Similarly, `model1` also

outperforms `model2` for higher levels of the Gaussian nosie component as well. However, for lower levels of noise, the performance of both the models is close to each other. This is evident from all the graphs because at lower levels of noise, the LPIPS curves of both of the models almost overlap. The PSNR and SSIM curves are also very close for both the models. This leads us to the conclusion that our network trained for a range of higher noise levels can effectively super-resolve low resolution images with higher noise levels without sacrificing performance at lower noise levels.

Figure 4.8 shows the results of our experiment on one image from our test set. The noisy image was synthesized using $a = 0.029$ and $b = 0.0001$ and is shown in Figure 4.8(b). It is evident by looking at the images that `model1` produces much better looking results than `model2`. Another interesting observation is how `model2` deals with the noise in the image as the noise level is much larger than the maximum noise level that `model2` is trained for. It seems that noise is enhanced in the form of blotchy artifacts that make the resulting image shown in Figure 4.8 (d) perceptually less pleasing.

Figure 4.9 shows the results on a real world image. It is interesting to see that the super-resolved image obtained using `model2` contains blotchy artifacts as well which can be seen in the zoomed in regions of Figure 4.9 (a). On the other hand the super-resolved image obtained using `model1` does not contain these blotchy artifacts as can be seen from the zoomed in regions of Figure 4.9 (b). This suggests that the blotchy artifacts also appear in real world images where the noise level is is higher than what the network can deal with.

4.4.4 A Single Network for Varying Blurs

To verify that a single network can remove spatially varying blurs from images we perform a simple experiment in which we train two networks. Let us call them `model3` and `model4`. Both the networks are trained for 500 epochs with images containing the same level of noise ($a \in [0, 0.024]$ and $b \in [0, 0.012]$). The two models differ in that images used for training the two models are blurred differently. Specifically, for `model3` we use a blur kernel picked randomly from 9 different blur kernels (see Figure 3.3), whereas for `model4` we use one specific blur kernel (center kernel in Figure 3.3). We use validation data which consists of blurred, downsampled and noisy images. The images are blurred with spatially varying blurs using the same 9 blur kernels used for `model3` (see Section 3.2.1 for details on how we do this). Next we follow the same procedure as done in previous experiments to downsample and inject realistic noise into the validation images. We use this validation set to evaluate the performance of the two models after every 100 epochs of

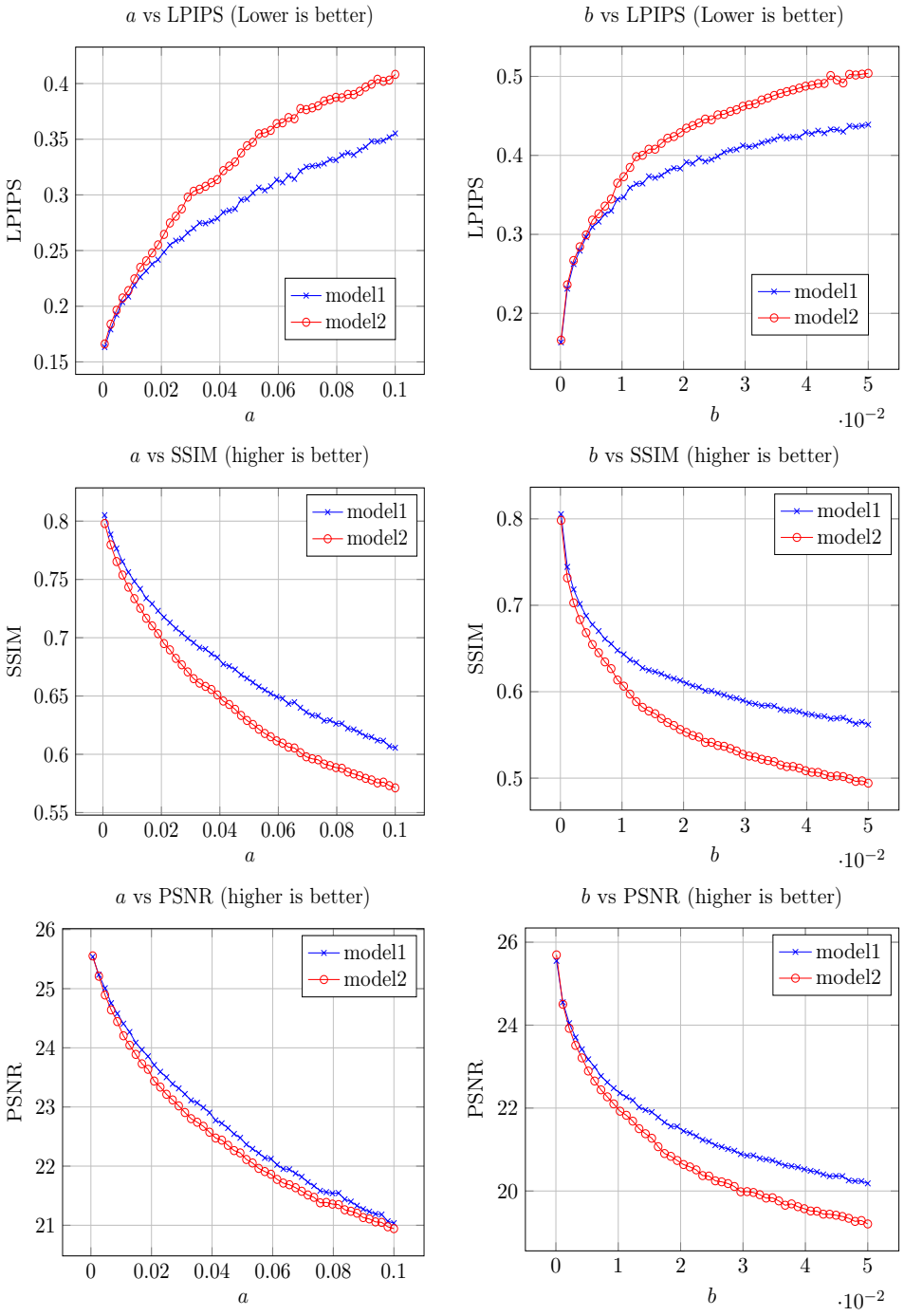


FIGURE 4.7: Performance of two models trained for different levels of noise in terms of LPIPS, PSNR and SSIM. `model1` which is trained for higher levels of noise can effectively super-resolve images containing higher levels of noise without sacrificing performnce at lower levels of noise.

(a) Ground Truth image.

(b) LR image ($a = 0.029, b = 0.0001$).

(c) `model1` result (LPIPS=0.167, SSIM=0.825).

(d) `model2` result (LPIPS=0.237, SSIM=0.772)

FIGURE 4.8: The image in (a) is a ground truth image taken from the validation set and (b) is the low resolution noisy image with noise parameters $a = 0.029$ and $b = 0.0001$. The image in (c) is the super-resolved image produced with `model1` trained for higher levels of noise and (d) is the super-resolved image produced with `model2` which is trained for lower levels of noise. The results from model2 are less accurate and contain blotchy artifacts. These artifacts are not present in (c) which suggest that they arise because `model2` is not trained for the level of noise present in the input image.

training on the basis of LPIPS score, SSIM and PSNR. The results of our experiment are shown in Figure 4.10. `model3` scores better on all metrics than `model4` at all the epochs which suggests that indeed a single network can deal with images containing spatially varying blurs if the network is also trained to remove these blurs. This can prove to be useful in the blind real-world super-resolution setting where we do not have information about the blur kernels that may have degraded the images. Another interesting thing to note is that the LPIPS score of `model4` increses after epoch 200 (see Figure 4.10 Epoch vs LPIPS graph) which means that the performance of `model4` on the validation set gets worse as the training progresses. This suggests that `model4` overfits to its training data which is synthesized only with a single blur kernel and cannot deal with the validation data synthesized with spatially varying blurs.

(a) `model1`

(b) `model2`

FIGURE 4.9: The super-resolved real image produced from `model2` trained at lower levels of noise is shown in (b). The zoomed in regions show presence of blotchy artifacts similar to those observed in Figure 4.8 for synthetic images. The super-resolved image in (b) is produced from the `model1` trained for higher levels of noise. The zoomed in regions have less of these blotchy artifacts which suggests that indeed these artifacts result from noise that the network cannot handle.

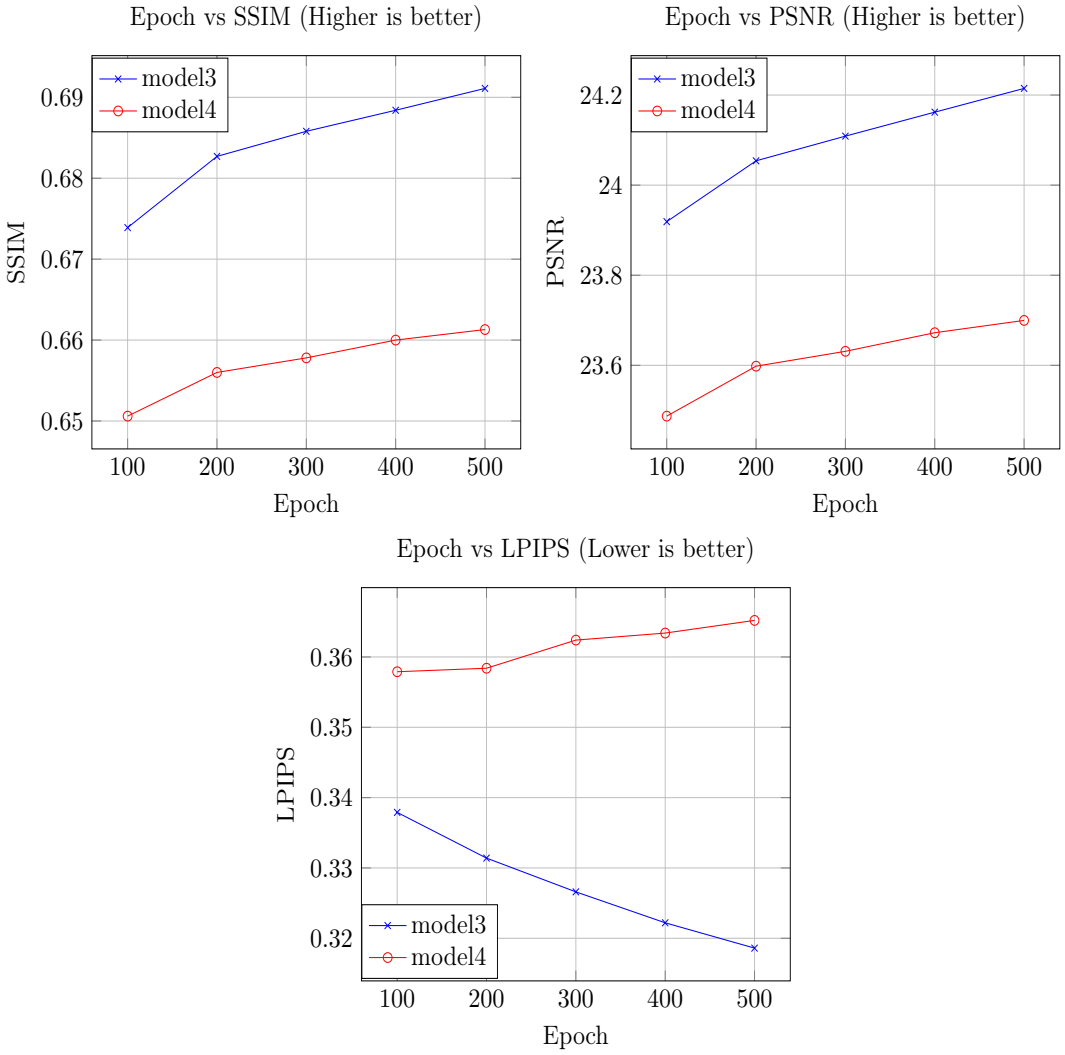


FIGURE 4.10: Average LPIPS, SSIM and PSNR values of `model3` and `model4` on a validation set containing varying blurs. `model3` which is trained for varying blurs produces consistently better scores than `model4` which is trained for only one blur kenel.

4.4.5 Comparison with PnP Method

For comparison between the Plug and Play (PnP) optimization method for spatially varying SR developed in Chapter 3 with the single network joint denoising, deblurring and super-resolution method proposed in this chapter, we rely on synthetic data because the two approaches differ in two key ways. First, the PnP method is based on the image formation model in which the images are first downsampled and then blurred (see Section 3.2 for more details) whereas the single network method developed in this chapter is based on the image formation model in which the images are first blurred and then downsampled. Second, the PnP method is also based on the assumption that noise at each pixel is zero mean Gaussian with known standard deviation, whereas the noise in the single network approach is not assumed to be Gaussian. Furthermore, the network

is trained on a range of noise levels and the noise level is not assumed to be known during testing. By relying on synthetic test data, we are able to perform an exhaustive comparison between the two methods for all possible scnearios that arise out of these differences. This allows us to compare

- The performance of the two methods when the assumptions of the methods are fulfilled to judge which approach is better overall.
- The robustness of the methods when the test data does not follow the assumptions.

To assess the robustness of the methods, we have to be able to synthesize test images containing the two types of noise with similar noise levels for a fair comparison. Matching the level of noise for both types of noise, i.e. independent Gaussian noise and realistic noise is a little tricky because realistic noise is signal dependent. However, we found a simple method to synthesize test images of both types of noise with approximately the same noise level. We first fix the level of realistic noise in all images by setting $a = 0.01$ and $b = 0.001$ and generating noisy images with spatially varying blurs as explained for the previous experiments. Next, we calculate the standard deviation (denoted by σ_{real}) of the difference between the blurred and noisy image and the blurred but clean image. σ_{real} gives the noise level in the images degraded with realistic signal dependent noise. To generate images degraded with independent white Gaussian noise with the same noise level, we sample noise from a Gaussian distribution with zero mean and standard deviation σ_{real} and add it to the blurred image. This ensures that noise levels of both types of noise are the comparable.

For testing, we use `model3` to represent the single network approach. Recall that `model3` is trained for varying blur and realistic noise (see Section 4.4.4 for more details). For the PnP approach, we use the default parameter setting as explained in Section 3.2 of Chapter 3. Table 4.1 shows the results for the image formation model in which the images are first blurred and downsampled. As expected, `model3` outperforms the PnP approach for realistic noise on all metrics. On Gaussian noise the PnP approach performs better in-terms of SSIM and PSNR. However, on the LPIPS metric, `model3` outperforms the PnP approach. This suggests that `model3` may be able to produce super-resolved images that are perceptually closer to the ground truth high resolution images than the PnP approach even for images degraded with independent Gaussian noise. The results for the image formation model where images are first downsampled and then deblurred are shown in Table 4.2. Note that `model3` is not trained for this particlar image formation model whereas the PnP approach is based on this model. The same trend can be observed, i.e. `model3` performs better than the PnP approach on realistic noise on the basis of all

metrics and for independent Gaussian noise `model3` outperforms the PnP approach in terms of LPIPS score. It is also important to emphasize that the PnP approach requires an accurate estimate of the noise level as input whereas `model3` operates completely blindly, i.e. it takes only the low resolution image as input. This, combined with the results from our experiments, suggests that the single network approach is overall more robust to unfulfilled assumptions and more flexible than the PnP approach.

Blurring Followed by Downsampling

	Realistic Noise			Gaussian Noise		
Method	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓
`model3`	**26.6646**	**0.7841**	**0.1212**	23.6885	0.6673	**0.1681**
PnP	23.0467	0.6845	0.3115	**25.9489**	**0.7617**	0.1820

TABLE 4.1: Comparison of results on test images first blurred and then downsampled by a scale factor of 2. `model3` achieves better results for realistic noise, whereas the PnP approach achieves better results for additive white Gaussian noise in terms of PSNR and SSIM. However, `model3` achieves better results in terms of LPIPS score.

Downsampling Followed by Blurring

	Realistic Noise			Gaussian Noise		
Method	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓
`model3`	**25.1422**	**0.7342**	**0.1503**	24.4181	0.7029	**0.1438**
PnP	23.3231	0.6706	0.3080	**25.8148**	**0.7408**	0.1743

TABLE 4.2: Comparison of results on test images first downsampled by a scale factor of 2 and then deblurred. `model3` achieves better results for realistic noise, whereas the PnP approach achieves better results for additive white Gaussian noise in terms of PSNR and SSIM. However, `model3` achieves better results in terms of LPIPS score.

4.4.6 Improving on Real World Images

To further improve performance of our method on real world images, we also adopt the perceptual loss first proposed by Ledig et al. [15] and later and improved in [16, 19] (see Chapter 2, for more details) for training with our synthesized data contining real world noise and blur characteristics. We perform tests on a variety of real world images taken from different devices including smartphones and DSLR cameras. The results of our experiment are shown in Figure 4.11. We compare the results of a network trained with only $\mathcal{L}_1$ loss with the results from a network trained with the perceptual loss ($\mathcal{L}_{\text{per}}$). Since the ground truth high resolution images are not available for these real world images, we have to rely on a qualitative comparison.

The results from the network trained with $\mathcal{L}_{\text{per}}$ loss look perceptually more pleasing than the results from the network trained just with the $\mathcal{L}_1$ loss. This is because the network trained with the perceptual loss produces sharper details and more realistic

textures while at the same time effectively suppressing noise in the images. The network trained with $\mathcal{L}_1$ loss can also suppress noise effectively, however, it tends to produce overly smooth images with less details and textures.

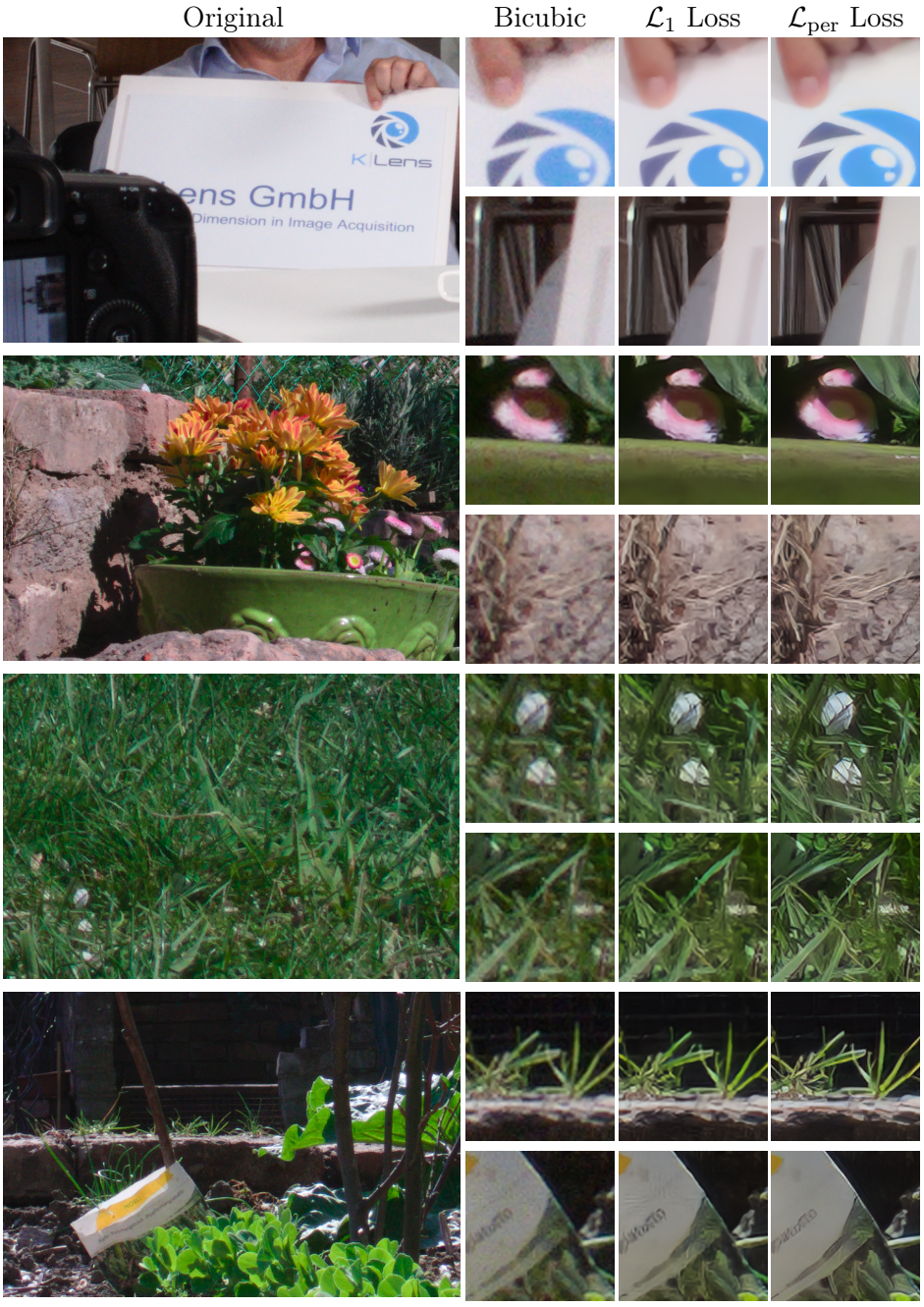


FIGURE 4.11: Qualitative comparison of images generated with networks trained using $\mathcal{L}_{\text{per}}$ loss and $\mathcal{L}_1$ loss trained using identical datasets and other settings. The original images were taken using various DSLR cameras and a K|Lens [69]. The network trained with $\mathcal{L}_{\text{per}}$ generates better super-resolved than the network trained with $\mathcal{L}_1$. This is because the network trained with $\mathcal{L}_{\text{per}}$ can generate images with sharper details and richer textures whereas the network trained with $\mathcal{L}_1$ loss tends to smooth over textures. However, both the networks are effective at suppressing noise in the original images.

4.4.7 Comparison with current state-of-the-art

We perform a comparison of our method with recent state-of-the-art methods for real-world SR on a variety of real-world images taken using different cameras, lenses and under different settings. The methods we use for comparison are ESRGAN [16] which is trained on the default DF2K data set, i.e. with clean LR images. We also use the combination of CycleISP [78] + ESRGAN. CycleISP is the current state-of-the-art method for real-world denoising [74] and we use it to clean up the LR images first. Then, we use ESRGAN to perform SR. This serves as the baseline method. We also compare our method against the method of Fritsche et al. [19], which is the current state of the art in real-world SR [21]. For our method, we use the network trained with $\mathcal{L}_{\text{per}}$ for this experiment. Since ground truth HR images are not available for these real-world images, we have to rely on a qualitative comparison. The results of our experiment are shown in Figures 4.12 and 4.13.

ESRGAN can produce sharp images, however, it also enhances the noise and cannot effectively deal with blurs in real world images. The CycleISP + ESRGAN baseline method, on the other hand, is able to remove some noise but the resulting images are look smooth and lack detailed textures. The results from the method of Fritsche et al. are sharp and contain a good amount of detail. However, it also ehnhances noise in parts of the images very sharply. Lastly, our method produces the best results overall. The noise is effectively suppressed in all parts of the super-resolved image. The generated images are sharp and contain a good amount of texture and details. Thus the images from our method are perceptually the most pleasing out of all the methods compared.

4.5 Summary

In this chapter, we proposed a method for synthesizing datasets to train neural networks for the task of super-resolution. On a high level, our method consists of blurring sRGB images with a variety of low pass filters and downsampling. We then convert the downsampled images to RAW space, inject noise according to the Poisson Gaussian model into the RAW images and then convert the noisy RAW images back to sRGB space. The resulting low resolution images possess realistic noise and blur characteristics. We train convolutional neural networks on the synthesized data and perform experiments to show that the networks can adapt well to a wide range of noise levels without sacrificing performance at lower noise levels. We test our method on variety of real world images. We compare the results of our method with baseline and state-of-the-art methods for SISR. Qualitative assessment of the results shows that our method effectively removes

noise present in the real world images and at the same time produces images that possess sharp edges and fine textures. Therefore, our method produces results which more pleasing perceptually.

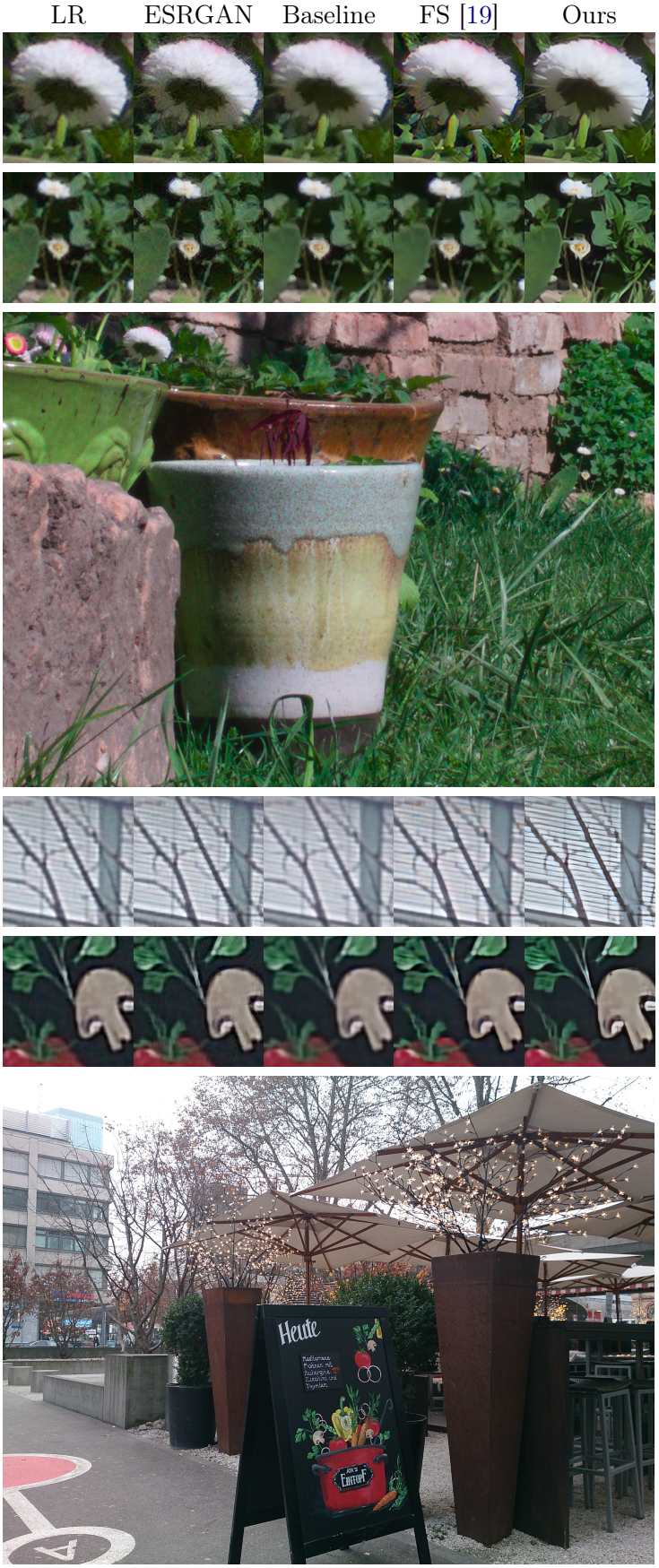


FIGURE 4.12: Comparison of our method with ESRGAN [16], CycleISP [78] + ESRGAN and the method of Fritsche et al. [19] shown under FS. Continued in Figure 4.13.

FIGURE 4.13: Comparison continued. Our method generates the sharpest details while at the same time effectively suppression noise. Thus, our method produces the best results overall.

Chapter 5

Conclusion and Future Work

In the first part of the thesis, we propose a single image super-resolution algorithm for images degraded by arbitrary spatially varying blurs. We derive the algorithm by first formulating the problem of SISR in the MAP framework using the additive white Gaussian noise assumption which results in a minimization problem. We use the variable splitting technique to decouple the data term and the prior term in the minimization problem and then solve it using the ADMM method. Our formulation allows using deep neural networks trained for the task of joint denoising and super-resolution to be used as priors similar to the method of Zhang et al.[17]. We perform experiments on synthetic test data to show that our approach performs better than the baseline approach for a wide range of noise levels and the gain in performance comes from the deblurring step. We also perform experiments on real world images under laboratory settings to show that our approach can generalize to real world settings. The approach requires measurements of blur kernels for optimal performance which can be considered a drawback if we want to super-resolve images in the wild taken from unknown devices. Another drawback of the approach is the independent additive white Gaussian noise assumption. This is because noise in real world images is more complicated than a zero mean Gaussian and can be spatially and chromatically correlated.

To overcome these limitations, we propose an end-to-end learning based approach. Our approach consists of synthesizing realistic datasets which consist of paired low resolution and high resolution images which we use to train convolutional neural networks (CNNs) for the task of super-resolution. CNNs require large amounts of training data and acquiring real world data for SR is known to be a tedious and cumbersome process. This necessitates reliance on synthetic data as large volumes of synthetic data can be easily generated. Furthermore, CNNs deliver optimal performance when the characteristics of training data match the characteristics of test data. Therefore, we aim to synthesize low

resolution images that resemble real world images in terms of blur and noise characteristics by simulating the camera ISP that delivers the final color image. This is achieved by first blurring the high resolution images with a blur kernel obtained from a set of blur kernels and downsampling, which results in a clean but low resolution image. The clean low resolution image is then converted to RAW space and noise according to the Poisson-Gaussian model is injected into the RAW image. We use this model because it captures the noise characteristics of RAW images well and is also easy to work with. The noisy RAW image is then converted back into the sRGB space which finally gives the low resolution image with realistic noise and blur characteristics.

We perform experiments on synthetic data to show that a single network can be trained to adapt to a wide range of noise levels without sacrificing performance at lower levels of noise. We also perform experiments to show that a single model can deal with images degraded with different and spatially varying blurs. We test the model on a variety of real world images taken from different types of devices like DSLRs and smartphone cameras. Qualitative assessments of the results shows that the model can produce images with sharp edges, realistic textures and overall a perceptually pleasing look. With these experiments, we have tried to address the question of how much performance can be squeezed out of a single model. The results suggest that a single model can deal with moderate levels of variation in the data which is sufficient for good performance on real-world images. However, the performance of the model is bound to diminish if the characteristics of the test images differ a lot from the training images. This can occur e.g, if the test images are too blurry, the noise characteristics or the noise levels are different from what the network is trained for. We observe this in the results of our experiments shown in Table 4.1, where the performance of a network trained for realistic noise diminishes for test images degraded with independent Gaussian noise and vice versa even though the noise level of both types of noise is similar.

A single model is also bound to loose expressiveness as it is trained for increasingly general scenarios. As an example, this can occur if we have a test set of images degraded with a specific range of noise levels. If the performance of a model trained for a very wide range of noise levels diminishes compared to a model trained for the noise level range specific to the test images, then we can say that the first model has lost its expressiveness. We did not observe this in our experiments even though we trained the models for a fairly wide range of noise levels.

5.1 Future Work

Results on real world images from the approach proposed in Chapter 3 can be improved by better noise modelling. A straight forward idea to achieve better noise modelling is to work on RAW images instead of color images because the noise characteristics of RAW images are well understood and we plan to cover this in future works. A drawback of this approach is that RAW images are not as readily available as color images. However, as the computational power of onboard computers in digital cameras and smartphones will increase, it will become feasible to deploy advanced super-resolution as part of the software in digital cameras and smartphones.

The end-to-end learning-based approach proposed in the second part of the thesis relies on a single network which has to adapt to various noise levels automatically. On a conceptual level, one can say that the network performs two steps. The first step is to estimate the noise level and the second step is to super-resolve the image based on the noise level of the image. An alternative approach is to have separate implementations of the two steps as well. The first step would involve predicting the noise level in the image and additional information such as ISO level used for the image which can be obtained from image metadata can also be incorporated to improve the noise level estimates. The second step would involve super-resolution of the low resolution image according to the predicted noise levels. As an example, two separate CNNs can be trained to perform the two steps. The first CNN would take as input the low resolution image and optionally other parameters such as ISO and output a noise level map which gives the noise level for each pixel in the low resolution image. The main super-resolution network would take the noise level map as input along with the low resolution image and produce the super-resolved image. Since this approach separates the two conceptual steps, it can result in better rationalization about the factors which affect the quality of results. Another interesting research direction would be to investigate if the techniques used in this thesis to achieve good performance on real world images can also improve robustness and performance in other computer vision problems like optical flow or even high level computer vision tasks like object detection and recognition. This is because neural networks used for these tasks are also trained on "clean" or rendered images and may be less robust to distortions present in real world images. Therefore, training with images containing realistic noise and blur characteristics can result in better and more robust real world performance for different kinds of computer vision tasks.

As far as the question of how much variation can ultimately be captured is concerned, training very large models i.e models with a very large number of learnable parameters

and observing how the performance scales with the number of parameters can help answer this question. However, training such large models is a computationally demanding task. A new area of research called Distributed Deep Learning has recently emerged which aims to address the problem of large scale learning by distributing parts of a very large model over multiple computers and training the models on very large datasets. As an example, Distributed Deep Learning has been successfully applied to train very large language models which give state-of-the-art performance [84] and large scale texture generation [85]. Therefore, similar performance benefits can also be expected for super-resolution suggesting that more variation can be captured by increasing the size of the model.

5.2 Outlook

There have been steady improvements in the performance of SISR methods over the last ten years. Among other factors, these improvements have come largely because of adoption of large scale learning, efficient neural architectures and more recently, perceptually motivated loss functions to train neural networks. We expect the improvements in performance of SR tasks to continue. This is because of several reasons. Firstly, the improvements have been driven by advances in deep learning and deep learning is still a fast growing area of research. Therefore, SR will continue to benefit from advances in deep learning. Secondly, the focus in SR research is now on devising loss functions that promote generation of high quality photo-realistic images i.e. images containing details in the form of textures and content that match real high quality images. The existing perceptual loss functions are based on matching features from pre-trained CNNs. However, recent works have shown that promising gains can by made in performance by incorporating more high-level and semantic information regarding the scene for super-resolution. This is also related to domain specific SR which targets images from a specific domain e.g. images of faces in which high-level information such as pose and geometry of faces is effectively utilized to super-resolve the images. Moreover, focus in SISR research has only now shifted to devise methods that perform well in real-world scenarios. Our work in this thesis is also a contribution in this direction and we have aimed to provide promising methods that can be further explored in future research to push the performance of SISR in real world scenarios. To conclude, the developments in these directions are expected to continue pushing the performance of SR to new levels as well as its adoption in real-world applications.

List of Figures

List of Tables

Bibliography

[1] Hayit Greenspan. "Super-resolution in medical imaging". In: *The Computer Journal* 52.1 (2009), pp. 43–63.

[2] Dinh-Hoan Trinh, Marie Luong, Francoise Dibos, Jean-Marie Rocchisani, Canh-Duong Pham, and Truong Q Nguyen. "Novel example-based method for super-resolution and denoising of medical images". In: *IEEE Transactions on Image processing* 23.4 (2014), pp. 1882–1895.

[3] Jean-Luc Starck, E Pantin, and F Murtagh. "Deconvolution in astronomy: A review". In: *Publications of the Astronomical Society of the Pacific* 114.800 (2002), p. 1051.

[4] Klaus G Puschmann and Franz Kneer. "On super-resolution in astronomical imaging". In: *Astronomy & Astrophysics* 436.1 (2005), pp. 373–378.

[5] Rebecca Willett, IH Jermyn, Robert Nowak, and Josiane Zerubia. "Wavelet-based superresolution in astronomy." In: Astronomical Society of the Pacific, 2003.

[6] Liangpei Zhang, Hongyan Zhang, Huanfeng Shen, and Pingxiang Li. "A super-resolution reconstruction algorithm for surveillance images". In: *Signal Processing* 90.3 (2010), pp. 848–859.

[7] Pejman Rasti, Tonis Uiboupin, Sergio Escalera, and Gholamreza Anbarjafari. "Convolutional neural network super resolution for face recognition in surveillance monitoring". In: *International Conference on Articulated Motion and Deformable Objects*. Springer. 2016, pp. 175–184.

[8] Hilario Seibel, Siome Goldenstein, and Anderson Rocha. "Eyes on the target: Super-resolution and license-plate recognition in low-quality surveillance videos". In: *IEEE Access* 5 (2017), pp. 20020–20035.

[9] William T Freeman, Thouis R Jones, and Egon C Pasztor. "Example-based super-resolution". In: *IEEE Computer Graphics and Applications* 22.2 (2002), pp. 56–65.

[10] Hong Chang, Dit-Yan Yeung, and Yimin Xiong. "Super-resolution through neighbor embedding". In: *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2004. CVPR 2004.* Vol. 1. IEEE. 2004.

[11] Roman Zeyde, Michael Elad, and Matan Protter. "On single image scale-up using sparse-representations". In: *International Conference on Curves and Surfaces*. Springer. 2010, pp. 711–730.

[12] Radu Timofte, Vincent De Smet, and Luc Van Gool. "Anchored neighborhood regression for fast example-based super-resolution". In: *Proceedings of the IEEE International Conference on Computer Vision*. 2013, pp. 1920–1927.

[13] Radu Timofte, Vincent De Smet, and Luc Van Gool. "A+: Adjusted anchored neighborhood regression for fast super-resolution". In: *Asian Conference on Computer Vision*. Springer. 2014, pp. 111–126.

[14] Chao Dong, Chen Change Loy, Kaiming He, and Xiaoou Tang. "Image super-resolution using deep convolutional networks". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 38.2 (2015), pp. 295–307.

[15] Christian Ledig, Lucas Theis, Ferenc Huszár, Jose Caballero, Andrew Cunningham, Alejandro Acosta, Andrew Aitken, Alykhan Tejani, Johannes Totz, Zehan Wang, and Wenzhe Shi. "Photo-realistic single image super-resolution using a generative adversarial network". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2017, pp. 4681–4690.

[16] Xintao Wang, Ke Yu, Shixiang Wu, Jinjin Gu, Yihao Liu, Chao Dong, Yu Qiao, and Chen Change Loy. "ESRGAN: Enhanced Super-Resolution Generative Adversarial Networks". In: *European Conference on Computer Vision*. Springer. 2018, pp. 63–79.

[17] Kai Zhang, Wangmeng Zuo, and Lei Zhang. "Deep plug-and-play super-resolution for arbitrary blur kernels". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2019, pp. 1671–1681.

[18] Eric Kee, Sylvain Paris, Simon Chen, and Jue Wang. "Modeling and removing spatially-varying optical blur". In: *2011 IEEE International Conference on Computational Photography (ICCP)*. IEEE. 2011, pp. 1–8.

[19] Manuel Fritsche, Shuhang Gu, and Radu Timofte. "Frequency separation for real-world super-resolution". In: *2019 IEEE/CVF International Conference on Computer Vision Workshop (ICCVW)*. IEEE. 2019, pp. 3599–3608.

[20] Andrey Ignatov, Radu Timofte, Thang Van Vu, Tung Minh Luu, Trung X Pham, Cao Van Nguyen, Yongwoo Kim, Jae-Seok Choi, Munchurl Kim, Jie Huang, et al. "PIRM Challenge on Perceptual Image Enhancement on Smartphones: Report". In: *ECCV Workshops (5)*. 2018.

[21] Andreas Lugmayr, Martin Danelljan, Radu Timofte, Manuel Fritsche, Shuhang Gu, Kuldeep Purohit, Praveen Kandula, Maitreya Suin, AN Rajagoapalan, Nam Hyung Joon, et al. "Aim 2019 challenge on real-world image super-resolution: Methods and results". In: *2019 IEEE/CVF International Conference on Computer Vision Workshop (ICCVW)*. IEEE. 2019, pp. 3575–3583.

[22] Jianrui Cai, Shuhang Gu, Radu Timofte, Lei Zhang, Xiao Liu, Yukang Ding, Dongliang He, Chao Li, Yi Fu, Shilei Wen, et al. "NTIRE 2019 challenge on real image super-resolution: Methods and results". In: *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*. IEEE. 2019, pp. 2211–2223.

[23] Andreas Lugmayr, Martin Danelljan, and Radu Timofte. "NTIRE 2020 challenge on real-world image super-resolution: Methods and results". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*. 2020, pp. 494–495.

[24] Xiaozhong Ji, Yun Cao, Ying Tai, Chengjie Wang, Jilin Li, and Feiyue Huang. "Real-World Super-Resolution via Kernel Estimation and Noise Injection". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*. 2020, pp. 466–467.

[25] Singanallur V Venkatakrishnan, Charles A Bouman, and Brendt Wohlberg. "Plug-and-play priors for model based reconstruction". In: *2013 IEEE Global Conference on Signal and Information Processing*. IEEE. 2013, pp. 945–948.

[26] Open Stax and Paul Peter Urone. *College Physics*. OpenStax College, Rice University, 2012.

[27] Max Born and Emil Wolf. *Principles of optics: electromagnetic theory of propagation, interference and diffraction of light*. Elsevier, 2013.

[28] Athanasios Papoulis. *Systems and transforms with applications in optics*. McGraw-Hill, 1968.

[29] Mario Bertero and Christine De Mol. "Super-resolution by data inversion". In: *Progress in Optics*. Vol. 36. Elsevier, 1996, pp. 129–178.

[30] R Tsai. "Multiframe image restoration and registration". In: *Advance Computer Visual and Image Processing* 1 (1984), pp. 317–339.

[31] SP Kim, Nirmal K Bose, and Hector M Valenzuela. "Recursive reconstruction of high resolution image from noisy undersampled multiframes". In: *IEEE Transactions on Acoustics, Speech, and Signal Processing* 38.6 (1990), pp. 1013–1027.

[32] Michal Irani and Shmuel Peleg. "Super resolution from image sequences". In: *[1990] Proceedings. 10th International Conference on Pattern Recognition*. Vol. 2. IEEE. 1990, pp. 115–120.

[33] Sina Farsiu, M Dirk Robinson, Michael Elad, and Peyman Milanfar. "Fast and robust multiframe super resolution". In: *IEEE Transactions on Image Processing* 13.10 (2004), pp. 1327–1344.

[34] Xuelong Li, Yanting Hu, Xinbo Gao, Dacheng Tao, and Beijia Ning. "A multi-frame image super-resolution method". In: *Signal Processing* 90.2 (2010), pp. 405–414.

[35] Dennis Mitzel, Thomas Pock, Thomas Schoenemann, and Daniel Cremers. "Video super resolution using duality based tv-l 1 optical flow". In: *Joint Pattern Recognition Symposium*. Springer. 2009, pp. 432–441.

[36] Benedicte Bascle, Andrew Blake, and Andrew Zisserman. "Motion deblurring and super-resolution from an image sequence". In: *European Conference on Computer Vision*. Springer. 1996, pp. 571–582.

[37] Russell C Hardie, Kenneth J Barnard, and Ernest E Armstrong. "Joint MAP registration and high-resolution image estimation using a sequence of undersampled images". In: *IEEE Transactions on Image Processing* 6.12 (1997), pp. 1621–1633.

[38] Daniel Glasner, Shai Bagon, and Michal Irani. "Super-resolution from a single image". In: *2009 IEEE 12th International Conference on Computer Vision*. IEEE. 2009, pp. 349–356.

[39] Manya V Afonso, José M Bioucas-Dias, and Mário AT Figueiredo. "Fast image recovery using variable splitting and constrained optimization". In: *IEEE Transactions on Image Processing* 19.9 (2010), pp. 2345–2356.

[40] Assaf Shocher, Nadav Cohen, and Michal Irani. ""Zero-Shot" super-resolution using deep internal learning". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018, pp. 3118–3126.

[41] Radu Timofte, Rasmus Rothe, and Luc Van Gool. "Seven ways to improve example-based single image super resolution". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2016, pp. 1865–1873.

[42] Jiwon Kim, Jung Kwon Lee, and Kyoung 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.

[43] Chao Dong, Chen Change Loy, and Xiaoou Tang. "Accelerating the super-resolution convolutional neural network". In: *European Conference on Computer Vision*. Springer. 2016, pp. 391–407.

[44] Wenzhe Shi, Jose Caballero, Ferenc Huszár, Johannes Totz, Andrew P Aitken, Rob Bishop, Daniel Rueckert, and Zehan Wang. "Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2016, pp. 1874–1883.

[45] Justin Johnson, Alexandre Alahi, and Fei-Fei Li. "Perceptual losses for real-time style transfer and super-resolution". In: *European Conference on Computer Vision*. Springer. 2016, pp. 694–711.

[46] RW Gerchberg. "Super-resolution through error energy reduction". In: *Optica Acta: International Journal of Optics* 21.9 (1974), pp. 709–720.

[47] Peter Cheeseman, Bob Kanefsky, Richard Kraft, John Stutz, and Robin Hanson. "Super-resolved surface reconstruction from multiple images". In: *Maximum Entropy and Bayesian Methods*. Springer, 1996, pp. 293–308.

[48] David Capel and Andrew Zisserman. "Super-resolution enhancement of text image sequences". In: *Proceedings 15th International Conference on Pattern Recognition. ICPR-2000*. Vol. 1. IEEE. 2000, pp. 600–605.

[49] Peyman Milanfar. *Super-Resolution Imaging*. CRC press, 2017.

[50] Jia-Bin Huang, Abhishek Singh, and Narendra Ahuja. "Single image super-resolution from transformed self-exemplars". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2015, pp. 5197–5206.

[51] Ian Goodfellow, Yoshua Bengio, Aaron Courville, and Yoshua Bengio. *Deep learning*. Vol. 1. MIT press Cambridge, 2016.

[52] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Fei-Fei Li. "Imagenet: A large-scale hierarchical image database". In: *2009 IEEE Conference on Computer Vision and Pattern Recognition*. Ieee. 2009, pp. 248–255.

[53] R. Girshick, J. Donahue, T. Darrell, and J. Malik. "Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation". In: *2014 IEEE Conference on Computer Vision and Pattern Recognition*. 2014, pp. 580–587.

[54] Jonathan Long, Evan Shelhamer, and Trevor Darrell. "Fully convolutional networks for semantic segmentation". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2015, pp. 3431–3440.

[55] Kai Zhang, Wangmeng Zuo, Yunjin Chen, Deyu Meng, and Lei Zhang. "Beyond a gaussian denoiser: Residual learning of deep cnn for image denoising". In: *IEEE Transactions on Image Processing* 26.7 (2017), pp. 3142–3155.

[56] Eddy Ilg, Nikolaus Mayer, Tonmoy Saikia, Margret Keuper, Alexey Dosovitskiy, and Thomas Brox. "FlowNet 2.0: Evolution of Optical Flow Estimation with Deep Networks". In: *CoRR* abs/1612.01925 (2016). arXiv: 1612.01925. URL: http://arxiv.org/abs/1612.01925.

[57] Vinod Nair and Geoffrey E. Hinton. "Rectified Linear Units Improve Restricted Boltzmann Machines". In: *Proceedings of the 27th International Conference on International Conference on Machine Learning*. ICML'10. Madison, WI, USA: Omnipress, 2010, pp. 807–814.

[58] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. "Generative adversarial nets". In: *Advances in Neural Information Processing Systems*. 2014, pp. 2672–2680.

[59] Ian Goodfellow. "NIPS 2016 tutorial: Generative adversarial networks". In: *arXiv preprint arXiv:1701.00160* (2016).

[60] Karen Simonyan and Andrew Zisserman. "Very deep convolutional networks for large-scale image recognition". In: *arXiv preprint arXiv:1409.1556* (2014).

[61] Felix Heide, Mushfiqur Rouf, Matthias B Hullin, Bjorn Labitzke, Wolfgang Heidrich, and Andreas Kolb. "High-quality computational imaging through simple lenses". In: *ACM Transactions on Graphics (TOG)* 32.5 (2013), pp. 1–14.

[62] Michael Hirsch, Suvrit Sra, Bernhard Schölkopf, and Stefan Harmeling. "Efficient filter flow for space-variant multiframe blind deconvolution". In: *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. IEEE. 2010, pp. 607–614.

[63] Yulun Zhang, Kunpeng Li, Kai Li, Lichen Wang, Bineng Zhong, and Yun Fu. "Image super-resolution using very deep residual channel attention networks". In: *Proceedings of the European Conference on Computer Vision (ECCV)*. 2018, pp. 286–301.

[64] Kai Zhang, Luc Van Gool, and Radu Timofte. "Deep unfolding network for image super-resolution". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2020, pp. 3217–3226.

[65] Shady Abu Hussein, Tom Tirer, and Raja Giryes. "Correction filter for single image super-resolution: Robustifying off-the-shelf deep super-resolvers". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2020, pp. 1428–1437.

[66] Stephen Boyd, Stephen P Boyd, and Lieven Vandenberghe. *Convex optimization*. Cambridge University Press, 2004.

[67] Yurii Nesterov. "A method for unconstrained convex minimization problem with the rate of convergence O (1/kˆ 2)". In: *Doklady an USSR*. Vol. 269. 1983, pp. 543–547.

[68] Mahmoud Assran and Michael Rabbat. "On the Convergence of Nesterov's Accelerated Gradient Method in Stochastic Settings". In: *arXiv preprint arXiv:2002.12414* (2020).

[69] Alkhazur Manakov, John Restrepo, Oliver Klehm, Ramon Hegedus, Elmar Eisemann, Hans-Peter Seidel, and Ivo Ihrke. "A Reconfigurable Camera Add-On for High Dynamic Range, Multispectral, Polarization, and Light-Field Imaging". In: *ACM Transactions on Graphics* 32.4 (2013), p. 47.

[70] Samuel W Hasinoff. *Photon, Poisson Noise.* 2014.

[71] Andrew J Blanksby, Marc J Loinaz, DA Inglis, and Bryan D Ackland. "Noise performance of a color CMOS photogate image sensor". In: *International Electron Devices Meeting. IEDM Technical Digest.* IEEE. 1997, pp. 205–208.

[72] Alessandro Foi, Mejdi Trimeche, Vladimir Katkovnik, and Karen Egiazarian. "Practical Poissonian-Gaussian noise modeling and fitting for single-image raw-data". In: *IEEE Transactions on Image Processing* 17.10 (2008), pp. 1737–1754.

[73] Tim Brooks, Ben Mildenhall, Tianfan Xue, Jiawen Chen, Dillon Sharlet, and Jonathan T Barron. "Unprocessing images for learned raw denoising". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition.* 2019, pp. 11036–11045.

[74] Tobias Plotz and Stefan Roth. "Benchmarking denoising algorithms with real photographs". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition.* 2017, pp. 1586–1595.

[75] Tamara Seybold, Özlem Cakmak, Christian Keimel, and Walter Stechele. "Noise characteristics of a single sensor camera in digital color image processing". In: *Color and imaging conference.* Vol. 2014. 2014. Society for Imaging Science and Technology. 2014, pp. 53–58.

[76] Shi Guo, Zifei Yan, Kai Zhang, Wangmeng Zuo, and Lei Zhang. "Toward convolutional blind denoising of real photographs". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition.* 2019, pp. 1712–1722.

[77] Michael D Grossberg and Shree K Nayar. "Modeling the space of camera response functions". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 26.10 (2004), pp. 1272–1282.

[78] Syed Waqas Zamir, Aditya Arora, Salman Khan, Munawar Hayat, Fahad Shahbaz Khan, Ming-Hsuan Yang, and Ling Shao. "CycleISP: Real Image Restoration via Improved Data Synthesis". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2020, pp. 2696–2705.

[79] Vladimir Bychkovsky, Sylvain Paris, Eric Chan, and Frédo Durand. "Learning Photographic Global Tonal Adjustment with a Database of Input / Output Image Pairs". In: *The Twenty-Fourth IEEE Conference on Computer Vision and Pattern Recognition*. 2011.

[80] Eirikur Agustsson and Radu Timofte. "NTIRE 2017 Challenge on Single Image Super-Resolution: Dataset and Study". In: *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*. July 2017.

[81] Bee Lim, Sanghyun Son, Heewon Kim, Seungjun Nah, and Kyoung Mu Lee. "Enhanced Deep Residual Networks for Single Image Super-Resolution". In: *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*. July 2017.

[82] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. "The unreasonable effectiveness of deep features as a perceptual metric". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018, pp. 586–595.

[83] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. "Imagenet classification with deep convolutional neural networks". In: *Advances in Neural information Processing Systems*. 2012, pp. 1097–1105.

[84] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. "Language models are unsupervised multitask learners". In: *OpenAI Blog* 1.8 (2019), p. 9.

[85] Anna Frühstück, Ibraheem Alhashim, and Peter Wonka. "TileGAN: synthesis of large-scale non-homogeneous textures". In: *ACM Transactions on Graphics (TOG)* 38.4 (2019), pp. 1–11.

Appendix A

Derivation of Transpose Operator H^T

Operator H is given by:

$$H = \sum_{r=1}^{p} C_r^T \, \mathcal{F}^{-1} \operatorname{Diag}(\mathcal{F}(\mathbf{h}^{(r)})) \, \mathcal{F}(C_r \operatorname{Diag}(\mathbf{w}^{(r)}))$$

The DFT operator $\mathcal{F}$ can be replaced by the DFT matrix denoted by F because the operations are equivalent and the IDFT operator $\mathcal{F}^{-1}$ can be replaced by the conjugate transpose of DFT matrix denoted by $F^{\mathcal{H}}$ because F is unitary. Therefore, operator H can be equivalently written as

$$H = \sum_{r=1}^{p} C_r^T \, \mathrm{F}^{\mathcal{H}} \operatorname{Diag}(\mathrm{F}\,\mathbf{h}^{(r)}) \, \mathrm{F}\, C_r \operatorname{Diag}(\mathbf{w}^{(r)}) \tag{3.22}$$

Taking transposes on both sides we get

$$H^T = \sum_{r=1}^{p} \operatorname{Diag}(\mathbf{w}^{(r)}) C_r^T F^T \operatorname{Diag}(\mathrm{F}\,\mathbf{h}^{(r)}) (\mathrm{F}^{\mathcal{H}})^T C_r, \tag{A.1}$$

$$\Rightarrow H^T = \sum_{r=1}^{p} \operatorname{Diag}(\mathbf{w}^{(r)}) C_r^T F^T \operatorname{Diag}(\mathrm{F}\,\mathbf{h}^{(r)}) \overline{F} C_r, \tag{A.2}$$

where $\overline{F}$ denotes the complex conjugate of F and $(\mathrm{F}^{\mathcal{H}})^T = \overline{F}$ because $\mathrm{F}^{\mathcal{H}}$ is by definition the conjugate transpose of H. The matrix H is real valued so H^T also has to be real

valued and using Equation A.2 we can write:

$$\begin{aligned}
H^T &= \overline{H^T} \\
\Rightarrow H^T &= \overline{\sum_{r=1}^{p} \text{Diag}(\mathbf{w}^{(r)}) C_r^T F^T \,\text{Diag}(\text{F}\,\mathbf{h}^{(r)}) \overline{F} C_r} \\
\Rightarrow H^T &= \sum_{r=1}^{p} \text{Diag}(\mathbf{w}^{(r)}) C_r^T \overline{F^T} \overline{\text{Diag}(\text{F}\,\mathbf{h}^{(r)})} (\overline{F}) C_r \\
\Rightarrow H^T &= \sum_{r=1}^{p} \text{Diag}(\mathbf{w}^{(r)}) C_r^T \,\text{F}^{\mathcal{H}}\, \overline{\text{Diag}(\text{F}\,\mathbf{h}^{(r)})} F C_r, \qquad \text{(A.3)}
\end{aligned}$$

where $\overline{C_r} = C_r$ because C_r is a real valued matrix. This is also the case for C_r^T and $\text{Diag}(\mathbf{w}^{(r)})$. Finally replacing the FFT F and IFFT matrix $\text{F}^{\mathcal{H}}$ with the corresponding operators $\mathcal{F}$ and $\mathcal{F}^{-1}$ respectively in Equation A.3, we get

$$H^T = \sum_{r=1}^{p} \text{Diag}(\mathbf{w}^{(r)}) C_r^T \,\mathcal{F}^{-1} \left[\overline{\text{Diag}(\mathcal{F}(\mathbf{h}^{(r)}))}\, \mathcal{F}(C_r) \right]. \qquad \text{(A.4)}$$

For completeness, the MVM of the form $H^T\mathbf{x}$ is implemented as:

$$H^T\mathbf{x} = \sum_{r=1}^{p} \text{Diag}(\mathbf{w}^{(r)}) C_r^T \,\mathcal{F}^{-1} \left[\overline{\text{Diag}(\mathcal{F}(\mathbf{h}^{(r)}))}\, \mathcal{F}(C_r\mathbf{x}) \right] \square.$$