COMPUTER VISION AT LOW LIGHT

by

Abhiram Gnanasambandam

A Dissertation

Submitted to the Faculty of Purdue University

In Partial Fulfillment of the Requirements for the degree of

Doctor of Philosophy

School of Electrical and Computer Engineering

West Lafayette, Indiana

August 2022

THE PURDUE UNIVERSITY GRADUATE SCHOOL
STATEMENT OF COMMITTEE APPROVAL

Dr. Stanley H. Chan, Chair

School of Electrical and Computer Engineering

Dr. Charles A. Bouman

School of Electrical and Computer Engineering

Dr. Jan P. Allebach

School of Electrical and Computer Engineering

Dr. Michael D. Zoltowski

School of Electrical and Computer Engineering

Approved by:

Dr. Dimitrios Peroulis

To MSEE 189 (RIP!)

ACKNOWLEDGMENTS

A boy at the beginning of a story has no way of knowing that the story has begun.

-Erin Morgenstern

I came to Purdue University in 2017 to do a Masters in ECE. At that point in time, the only thing I was sure of was that I wanted to work in imaging. I had not worked on imaging before that. The last five years have been as much fun as intellectually gratifying. 2017 me could not have even imagined how nice of a ride these five years have been. Prof. Stanley H. Chan, my advisor, takes most of the credits. Working with him convinced me to switch from a Masters program to a Ph.D. Stanley is always available for a chat whenever you want, whether you get stuck in a problem or you are clueless about how to write your first paper. He gave me the freedom to work on things I wanted to. I have worked on quite a few problems just for fun. Stanley never discouraged it. In fact, sometimes he gets involved too. I am lucky to have had him as my Ph.D. advisor.

I express my appreciation to Prof. Charles A. Bouman, Prof. Jan P. Allebach, and Prof. Michael D. Zoltowski for being a part of my advisory committee and valuable feedback on my work during my preliminary exam and my thesis defense.

I would like to thank everyone in my lab: Nick, Xiangyu, Zhiyuan, Kent, Guanzhe, Yash, Yiheng, Xingguang, and Kevin. I have collaborated with some of these people for a few of my papers, and with others, I have not. Irrespective of that, just being around these talented people and conversing with them has helped me imbibe great qualities and knowledge.

I would like to thank Jiaju Ma, Dakota Starkey, and Saleh Massodian from Gigajot Technology, where I spent two summers. Working at Gigajot, especially under Dr. Jiaju Ma, helped me understand image sensor technology and how a real camera company works. I would also like to thank Dr. Vladlen Koltun for our collaboration.

I want to thank my parents for their support. They were supportive of my decisions and goals, irrespective of how often and how much they kept changing. I also want to thank PK for being there for those long talks whenever I felt down.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABSTRACT

Imaging in low light is difficult because the number of photons arriving at the image sensor is low. This is a major technological challenge for applications such as surveillance and autonomous driving. Conventional CMOS image sensors (CIS) circumvent this issue by using techniques such as burst photography. However, this process is slow and it does not solve the underlying problem that the CIS cannot efficiently capture the signals arriving at the sensors. This dissertation focuses on solving this problem using a combination of better image sensors (Quanta Image Sensors) and computational imaging techniques.

The first part of the thesis involves understanding how the quanta image sensors work and how they can be used to solve the low light imaging problem. The second part is about the algorithms that can deal with images obtained in low light. The contributions in this part include – 1. Understanding and proposing solutions for the Poisson noise model, 2. Proposing a new machine learning scheme called student-teacher learning for helping neural networks deal with noise, and 3. Developing solutions that work not only for low light but also for a wide range of signal and noise levels. Using the ideas, we can solve a variety of applications in low light, such as color imaging, dynamic scene reconstruction, deblurring, and object detection.

1. INTRODUCTION

1.1 A history of cameras

Pictures have been used as a form of communication by humans for thousands of years. Going back to cave paintings as old as 64000 years by Neanderthals, different forms of paintings throughout history, and more recently, film and digital cameras to capture photographs, our modes of visual communication using pictures have been evolving continuously. Over the last few decades, photographs have become an integral part of our lives. We have gone from taking photographs to record significant events to having a camera in our pockets, enabling us to take photographs whenever we want.

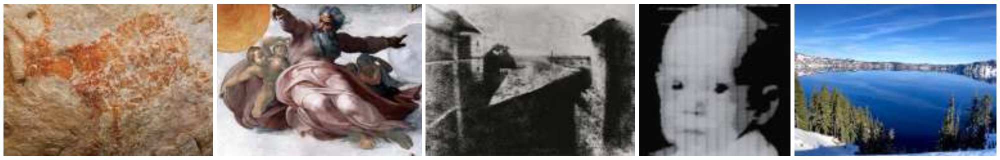


Figure 1.1. History of visual communication using pictures. [Left to right] Depiction of a bovine (40000 BCE) [1], The Creation of the Heavens (1512 CE) [2], The first film photograph (1826 CE) [3], The first digital photograph (1957 CE) [4], A modern digital color photograph (2020 CE).

Cameras have fundamentally changed how we communicate. The ubiquitousness of the cameras comes from the fact that cameras have extremely wide variety of people - Artists, scientists, and journalists. The astounding rate of innovation in the camera technology in the twentieth century has fundamentally shifted how we communicate. For the overwhelming majority of our history, human beings have mainly depended on the verbal or written stories to communicate. Every once in a while we may have a pictoral depiction drawn by someone. Whichever mode it was, the story depended on the teller's perspective. The universal availability of cheap cameras has made photographs the primary mode of communication, which reduces the bias in the story to a great extent, with other modes complimenting the photographs in telling the story.

1.1.1 Early ideas

The first idea for a camera can be traced back to as early as the fifth century BC, in the form of projecting a scene onto a screen. A Chinese philosopher named Mozi described an idea for capturing an image in a dark room using a small hole in the wall. Mozi recorded that by virtue of light traveling in straight line, the image projected in the dark room appears updside down and inverted left to right. This idea was what came to be known as *camera obscura* or a *pinhole image*. Over years, number of different versions of the pinhole cameras were used and the the properties of this system was well studied. The earlier versions were all large dark rooms with a small hole. By 18th century, the size of the pinhole cameras had gone down to just the size of a box and convex lens were added to help with better projection of the images onto the screen.

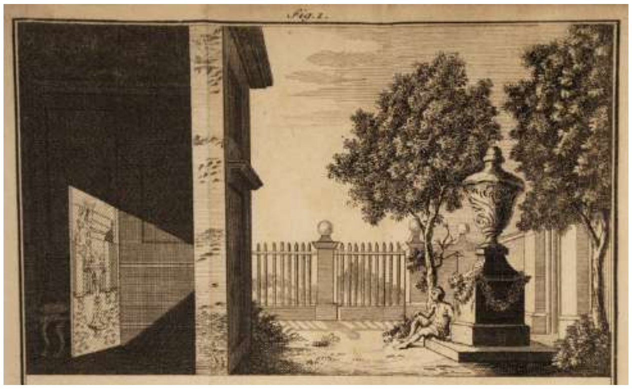


Figure 1.2. Camera Obscura [5]

1.1.2 Film cameras

The pinhole cameras were used to project images, which were then used by artists to draw the scene. However, there was an obvious drawback that the images being projected are not permanent. All this changed in the early nineteenth century, when Joseph Nicephore Niepce placed a photosensitive plate at the back of the pinhole camera to capture the first known permanent photograph (See Figure 1.1 - third image from left). The photographic plate he used was a pewter plate coated with bitumen. Over years the technology kept maturing with better photosenstive materials being developed over years. The photograpahic films that are still in use, use silver halide crystals. The silver halide films can be directly traced to the first flexible photographic film invented by George Eastman in 1888.

While the film cameras in late nineteenth century were maturing, the cameras were still bulky and costly, and were considered a luxury and were used only sporadically by companies of means. The first breakout success for a camera came in 1900 in the form of the Kodak Brownie camera, which was sold for 1$ per camera. It used photographic films that needed to be replaced. The photographic films sold for 15¢, which made the Brownie cameras accessible to a significant number of the population, with 150,000 cameras sold in the first year of its announcement. The Brownie cameras continued to be made till as late as the 1980s.

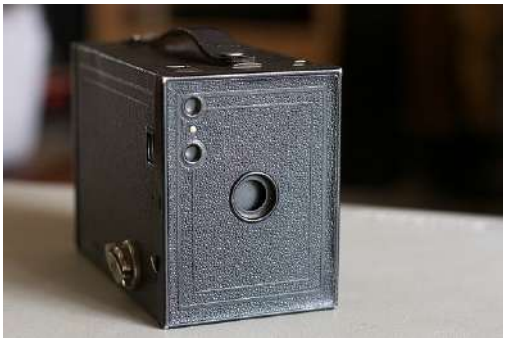


Figure 1.3. Kodak Brownie [6]

The next major revolution in cameras came in the form of instant photos. While cameras in early twentieth century were great, they had a significant drawback. The time taken get the photo in hand was long, especially if one does not take as many photos. One needed to wait till the entire roll of film is done, before sending it off for getting developed. In 1948, Edwand H. Land invented an instant photo camera - Polaroid Land Camera 95, which used a chemical process to develop film inside the camera itself in less than a minute. Although the instant cameras were costly, the idea of getting a developed photo immediately was an instant hit and Polaroid tapped into this and had multiple models by 1960s.

While color photography was developed in tandem with the monochrome film photography, the first tricolor (red, green and blue) films were introduced by Kodak in 1935. Color film was not the go to choice for a long time, as it was always costlier than the monochrome

Figure 1.4. Polaroid [7]

films. This slowly changed over time and nowadays it is hard to even find a monochrome camera.

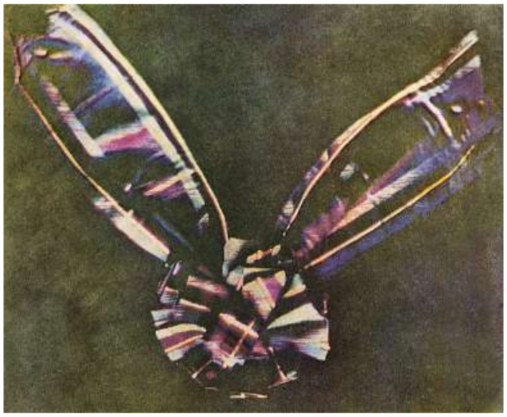


Figure 1.5. First color photo ever taken [8]

1.1.3 Digital cameras

The film cameras were too inflexible in terms of the ability to process them and fix the blemishes which may show up in the captured image. The digital cameras provided a way for separating the mode of capture and the where the digital data is stored. This way the data can be processed before getting stored.

The charge-coupled devices (CCD) that ultimately replaced photographic films were invented in 1969. However, the digital cameras with CCD did not become popular till the 1990s. The first major successful digital camera was Casio QV-10, introduced in 1994. It had a small resolution of just 240 × 320. The quality of the images produced by these cameras was nowhere close to the existing film cameras. However, Casio QV-10 had the built-in feature to display the captured image using an LCD screen, which was an attractive feature for a significant fraction of the population that did not care about the ultimate quality of the captured image as long as it was not too poor.

Figure 1.6. Casio QV-10 [9]

Casio QV-10 started the snowball effect, which led to the digital camera revolution. Tens of billions of dollars worth of digital cameras are being sold every year. The digital cameras got a further boost with the invention of NMOS active pixel sensors by Olympus and the CMOS active pixel sensors by Eric Fossum. The CMOS active pixel sensors have grown leaps and bounds since their invention in 1993. CMOS active pixel sensors are the current go-to choice for digital image sensors.

(a)

Figure 1.7. Famous cameras.

1.1.4 DSLR and cellphone cameras

There is a wide variety of cameras available in consumer camera domain, where we can choose a camera based on the sophistication and the quality we need. There are basic consumer products such as Canon powershot series and there are the highly sophisticated digital single lens reflex (DSLR) cameras used by professionals. The difference between a consumer grade camera and the professional DSLR camera is often attributed to the noise level in the sensors and the optics.

Figure 1.8. iPhone 13

In the last decade, cellphone cameras have almost totally replaced the consumer camera market. Since every smartphone comes with a camera of good enough quality, the consumers do not generally see a need for a separate camera.

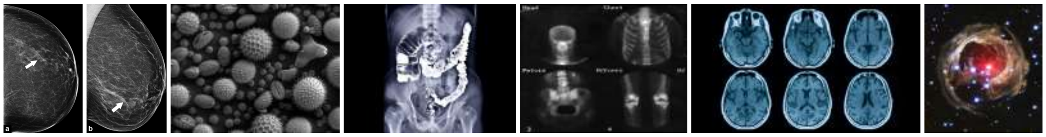


Figure 1.9. Other imaging applications. [Left to Right] Mammography [10], Electron microscopy [11], Fluoroscopy [12], Gamma imaging [13], MRI [14], Imaging the space [15].

1.1.5 Other imaging applications

While the progress was happening in the consumer camera domain, other imaging applications were making giant strides too. In 1913, Albert Solomon invented *mammography* which uses X-ray images to detect early-stage breast cancer [16]. In 1931, Ernst Ruska invented the *electron microscope* that can be used to image view objects as small as the diameter of an atom. Ernst Ruska received the Nobel prize for this invention in 1986. In 1949, Russell Morgan invented an image intensifier that bettered *fluoroscopic vision*, which is used in medical fluoroscopy and military applications to this date. In 1958 Hal Anger invented the gamma camera that enables physicians to detect tumors and diagnose by imaging gamma rays emitted by radioactive isotopes. In the 1970s CAT and MRI scans were invented, which revolutionized the medical imaging domain. In the 1960s and 1970s, space imaging grew significantly. Numerous satellites were sent to image the earth from space. The Hubble space telescope was sent to the Earth's orbit in 1990. Hubble's successor, the James Webb space telescope, was launched in 2021 with improved infra-red imaging capabilities.

1.2 This dissertation

In this dissertation, we are concerned with visible imaging, and we will often deal with photography as the application of choice. As we will notice later in this chapter, the current generation of cameras has difficulty imaging in low light. This can be alleviated to a small degree by using better sensors. With the sensors technology getting better every year, we might soon be reaching as close as we can to a *perfect sensor*, which we define as an image sensor that can detect each and every photon that arrives at the photon. The sensor is free from other noise sources such as read-noise and dark current. Does this mean that we are close to the end of innovations in imaging? The answer is no. Even a perfect sensor cannot overcome the randomness of the photon arrival process, which is intrinsic to imaging in low light. In section (1.3), we look at some of the difficulties of imaging that we face in real life. We will need computational solutions to deal with these problems. Another significant issue is that the term low-light is vaguely defined in the literature. When we see a paper trying to solve a "low-light" problem, it need not necessarily mean that the solution deals with the heavy randomness of the photon arrival. Later in this chapter, we will define the term *photon-limited* and show how it differs from the standard low light that the existing literature deals with.

While we deal only with visible imaging and photography applications, most of the problems we tackle are more widespread in other imaging modalities too. Although the solutions we develop can be transferred to other modalities in most cases, we do not study generalizing the proposed solutions to other modalities.

1.3 Imaging at different light levels

Cameras can be used for still capture or capturing videos. This dissertation looks at some of the major scenarios when the cameras fail while capturing still images. While sometimes we may deal with moving scenes, we are still interested in the still reconstruction of the moving rather than capturing the video itself. We will look at how cameras work in different imaging scenarios and identify the weak points of the current cameras.

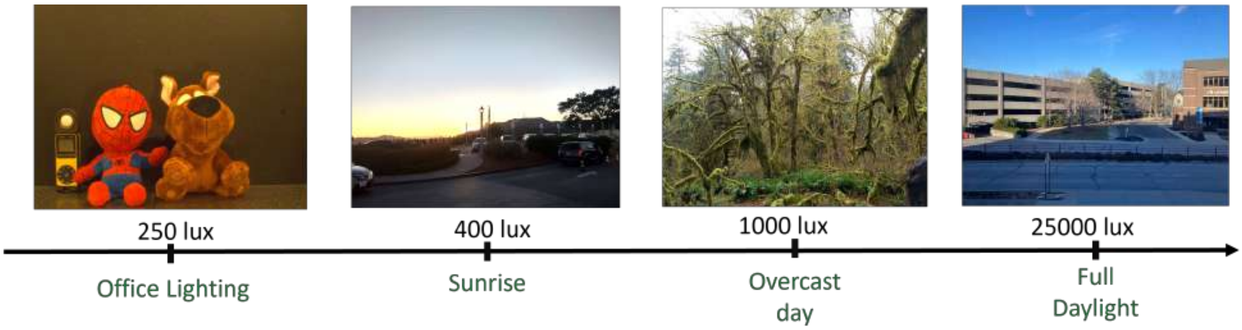


Figure 1.10. Imaging bright scenes. The left most image is captured using a Thorlabs CS165CU camera. The other images are all captued using an iPhone SE (2020). We can see that the cameras work great at these light levels, where enough photons are arriving at the image sensor.

1.3.1 Bright scenes

The existing CMOS image sensors in a DSLR or a cellphone work exceedingly well when enough light emanates from the scene we are interested in capturing. Figure 1.10 shows the wide range of light levels where the existing image sensors work well. The image sensors can do an almost perfect reconstruction of the scene being imaged at these light levels. This is possible because enough photons arrive at the sensor at these light levels. The captured signal has a good enough signal-to-noise ratio, which leads to an excellent reconstruction. These light levels are the expected operating regime for most image sensors.

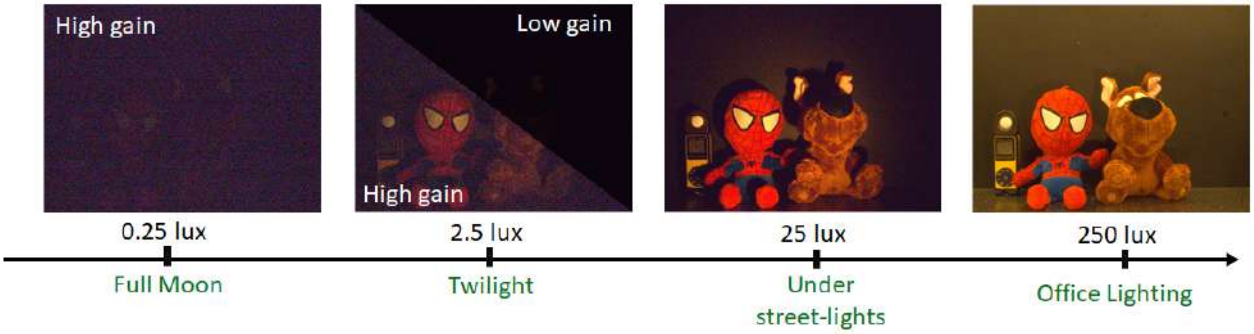


Figure 1.11. Imaging low-light scenes. All the images here are captured using Thorlabs CS165CU camera with integration time of 30ms. As the light level goes down, we can notice that the camera struggles to capture a good enough image.

1.3.2 Low-light scenes

As the scene's brightness keeps decreasing, the image sensors start struggling at reconstructing the scene. We can notice in Figure 1.11 as long as the light level is greater than 25 lux, the camera does not have a problem in imaging the scene. However, when we go even darker to 2.5 lux, we need to use the highest gain (or ISO) setting available on the camera to get any meaningful signal from the sensor. Even this image looks extremely noisy. As the light level goes further down to 0.25 lux, the image we see is so poor even when used at the highest gain.

This is not a special scenario that happens only in a lab setting. Figure 1.12 shows another example where an iPhone SE fails in low light imaging. The image on the left is captured with the lights on, and on the right, the lights are off. One can notice the heavy noise in the image captured with the lights off. It is difficult to get rid of this noise by simply using better hardware since the major noise source here is not the sensor but rather the shot noise arising from the randomness of the arriving photons. This randomness is part of nature, and we cannot get rid of this. Figure 1.13 demonstrates this phenomenon. We

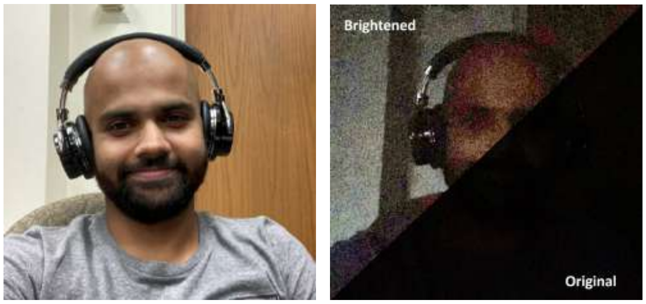
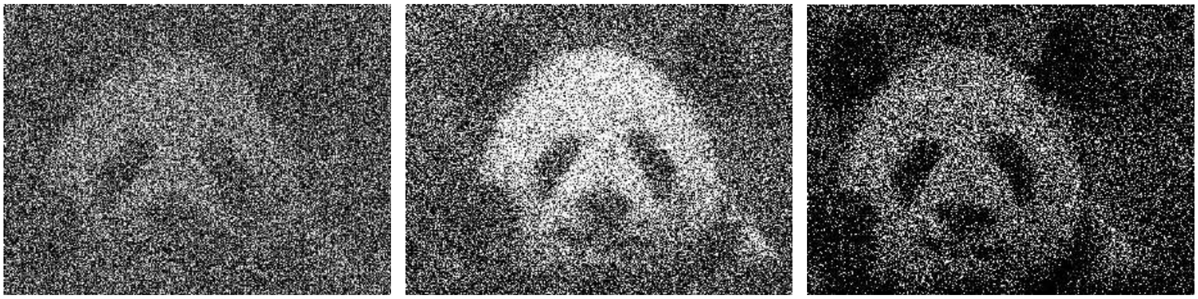


Figure 1.12. Imaging at low light. The same scene is captured with and without external illumination using an iPhone. The image on the right is brightened for better visualization. We can clearly notice the poor signal to noise ratio in the right image.



CIS, real QIS, real Ideal Sensor, simulation

Figure 1.13. An ideal sensor is not enough. The real images are captured at a light level of 0.25 photons per pixel (ppp). The simulation is also done at the same photon level to demonstrate the effect of shot noise on the image. The ideal sensor simulation shows that the imaging in low-light has a fundamental limit which cannot be fixed by using a better sensor.

show images captured using a real CIS, a real QIS, and a simulation of shot noise if captured using an ideal sensor with no other sources of hardware noise. The images are captured at 0.25 photons per pixel (ppp) on average. We can notice a significant improvement in signal-to-noise ratio (SNR) by going from CIS to QIS because of the better sensor noise statistics. Nevertheless, the ideal sensor simulation shows us a fundamental limit to imaging when only a few photons arrive at the sensor. No amount of improvement in sensor technology can fix this problem.

Color Imaging.

Imaging at low light also causes issues with color reconstruction. Cameras generally use

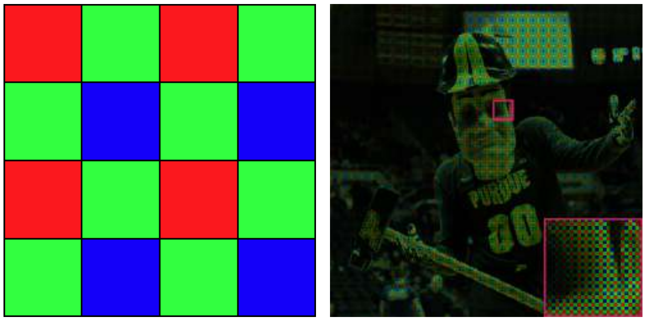


Figure 1.14. Color Filter Array. For color imaging cameras usually place a color filter array (CFA) such as a Bayer pattern CFA seen here on the left. On the right, we can see a simulated 'RAW' image captured by a camera.

Color Filter Arrays (CFA) such as a Bayer pattern CFA for color imaging, where we capture only one color channel per pixel. Figure 1.14 shows the Bayer pattern color filter array and also an example image captured using the CFA placed in front of the sensor. Capturing a single channel image with a CFA and converting it into a three channel RGB image is called demosaicing. Demosaicing by itself is a complex problem since we need to interpolate the missing two channels at each pixel location. When capturing low light images the problem becomes significantly harder since we now need to deal with the noise too.

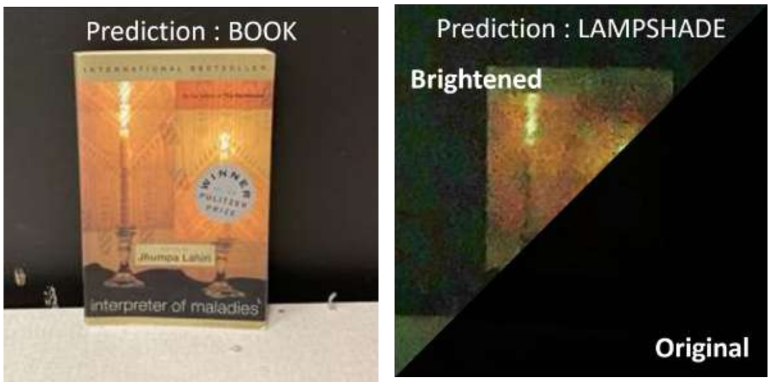


Figure 1.15. Image Classification. Both the images are captrured using an iPhone and classified using Resnet101 [17]. The image on the right is brightened for better visualization.

Object detection.

Low light images cause trouble in downstream high-level computer vision tasks such as image classification and object detection too. Most of the existing object detection and image classification methods work well with photographs taken in good enough illumination. However, existing algorithms fail when the light level goes down, and the captured images become noisy. They are not designed to deal with images with such heavy noise. Figure 1.15 demonstrates this phenomenon using Resnet-101 [17]. Both the images capture the book. The image on the left is well-illuminated. The lights are turned off for the image on the right, leading to a generation of an extremely noisy image. Although the classifier works well on the well-illuminated scene, it fails when the lights are turned off.

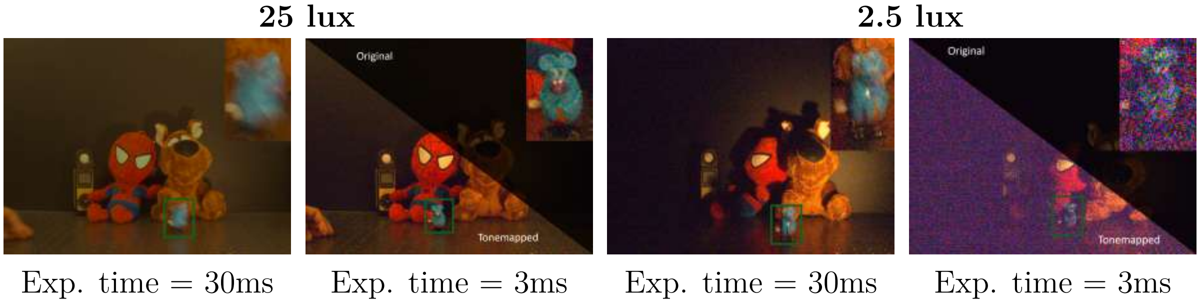


Figure 1.16. Dynamic Scenes. Imaging dynamic scenes at low-light comes with a trade-off. We either obtain a blurred image with higher SNR or sharper image with lower SNR. This phenomenon gets even more significant when the light level goes down.

Dynamic Scenes

Imaging dynamic scenes is a difficult problem because motion in the scenes caused by camera shakes or objects moving the scene introduces blur in the captured image. We can reduce the amount of blur by using a faster shutter, but that would mean that the SNR of the captured image goes down, leading to a noisy image getting captured. However, when the light level is high enough, the short exposure images, albeit a little noisy, can still recover most of the details with simple denoising. Figure 1.16 shows an example of this phenomenon at 25 lux. This problem gets exacerbated at low light because the image at shorter exposure will be extremely noisy to recover any useful information from the data.

1.3.3 High dynamic range (HDR) imaging.

High dynamic range imaging is the process using which we image a scene with pixel intensities larger compared to a standard scenario. Figure 1.17 shows an example of such a

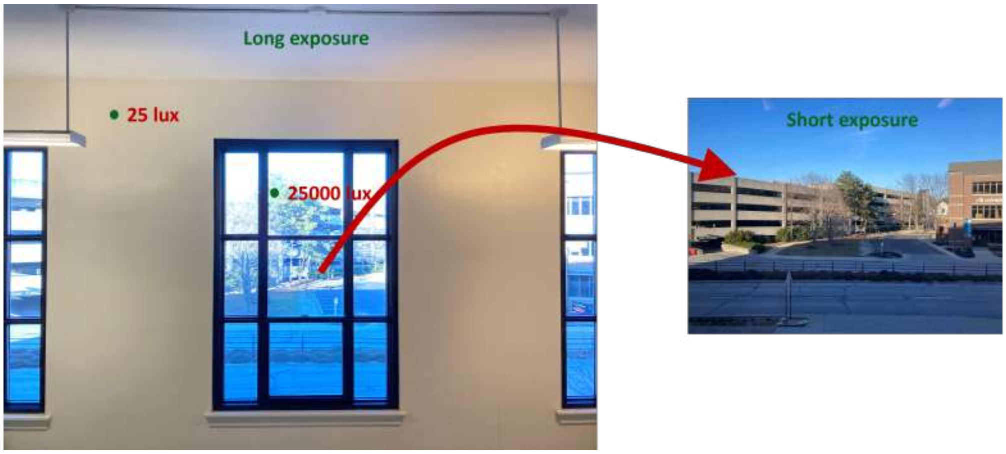
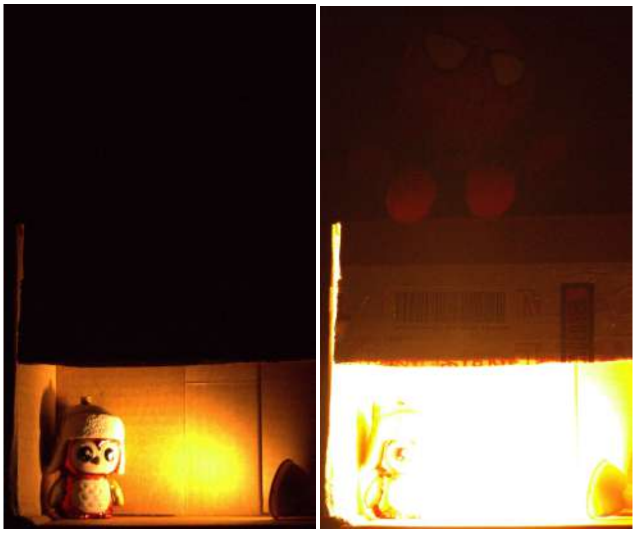


Figure 1.17. High dynamic range. We can see that the camera is capable of capturing all the light levels in this scene, however it is not able to capture them in a single image.

scene. A single short or long exposure CIS image cannot capture the scene completely. A lot of pixels are saturated in the first image, but using a shorter exposure is able to capture the details from the saturated image. However, the indoor details will be lost if we try to reduce the integration time in the first image. We need to use techniques such as exposure bracketing [18] to reconstruct the HDR scene.



Short exposure Long exposure

Figure 1.18. HDR imaging with noise.

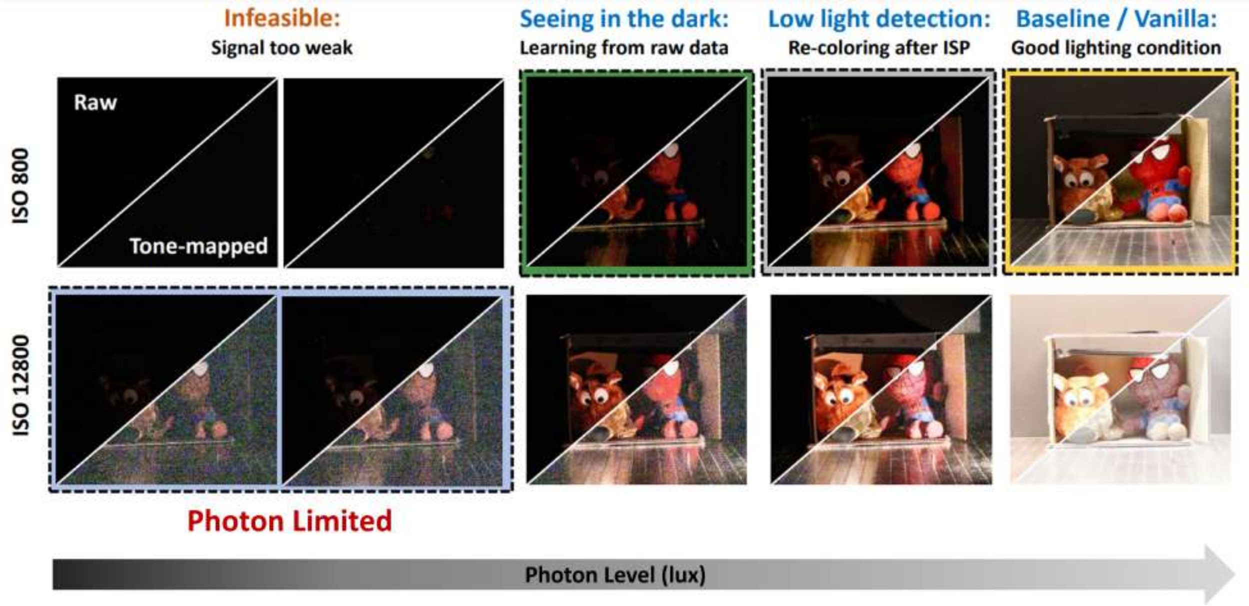


Figure 1.19. What is photon-limited? Most of the existing low light imaging solutions perform simple image enhancement such as recolorization, but do not deal with the photon shot noise. In this dissertation, solutions are proposed for *photon limited* scenarios where simple image enhancement techniques do not work.

However, these standard techniques require capturing multiple images at different exposure times and combining them computationally to get the HDR image. These issues point toward a need for solutions that can capture fast high dynamic range images. The solution should be able to capture the dark regions and also the bright regions simultaneously.

The issue with HDR imaging becomes even more pronounced when the darker parts of the image are in extremely low light. In Figure 1.18, we can notice that even in the long exposure, the details in the darker parts of the image are extremely noisy. We need to capture more frames with even longer exposure in situations like this and then computationally combine them. Capturing more frames will mean more time taken to produce the final image. Additionally, since we need to use a longer exposure time, the captured images are susceptible to motion blur.

1.4 Thesis overview

In the case of both low light and high dynamic range scenarios, we have seen that a standard CMOS image sensor has many drawbacks. To address these issues, in this dissertation, we look at how to intelligently control the sensor to capture the data and further computationally process this data to obtain a good-looking image. The process of jointly controlling how the image is captured and the computational processing of the captured data for obtaining a good final image is called computational imaging. Computational imaging is used for various applications such as microscopy, tomography, and remote sensing. In this dissertation, we will look at computational imaging to solve low-light imaging and high dynamic range imaging problems. Using better sensors to capture as much information about the scene as possible in such scenarios becomes imperative. Therefore, a portion of the work done as part of this dissertation involves using the Quanta Image Sensors (QIS). With its better read noise, dark current, and smaller pitch, QIS provides the necessary features for better low light imaging compared to a regular CMOS image sensor. Other solutions are more generic and could be used for both CMOS image sensors and quanta image sensors. Before we get into an overview of the thesis, let us first try to understand the need for this dissertation and how the contributions compare to the existing methods.

1.4.1 Photon limited imaging

There are a variety of applications, such as surveillance, autonomous cars, and microscopy, where low-light imaging is necessary. In computer vision literature, there are many different methods, such as [19], which deal with low light image enhancement. While these methods deal with low light imaging, they do not work so well in the harder *photon limited* scenarios.

Figure 1.19 demonstrates the difference between low light imaging as usually defined in the literature and photon limited imaging. Most of the existing low light imaging solutions involve simple color and detail enhancement and fail when there is heavy shot noise. Figure 1.20 demonstrates the regime where a standard low light image enhancement method [19] works and how it fails in the presence of photon shot noise. At the light level we call photon limited, only a few photons (as few as only 1 photon per pixel) arrive at each pixel on average. We are forced to use high sensor gain (or equivalently high ISO) to capture any meaningful signal in such scenarios. In this dissertation, we are concerned with imaging in photon-limited and composite scenarios containing photon-limited and bright regions together (e.g., an HDR scene).

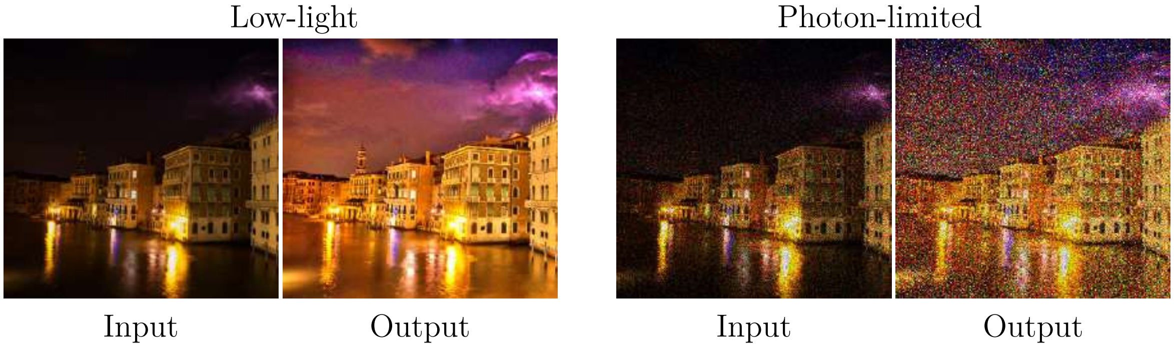


Figure 1.20. Low light image enhancement. We can notice that the low light image enhancement using [19] works well when noise is absent, even though the image appears dark. However, when the scene is corrupted by heavy shot noise, which is what we are interested in this dissertation, the method does not perform that well.

1.4.2 Thesis outline

The rest of this dissertation can be broadly divided into 3 categories - 1. Basics, 2. Low light imaging solutions, and 3. Generalizing to a wide range of light levels.

Basics

Chapters 2 and 3 go through the basics necessary for the rest of the dissertation.

- *Chapter 2.*

In this chapter, through the basics of the semiconductor physics of how image sensors work. The chapter gives a brief history of modern cameras. We will then look at some basic understanding of how image sensors work, get into a bit of semiconductor physics, and finally look at the quanta image sensors and other single-photon devices, which play an integral part in the rest of this dissertation. This chapter will help us understand what makes the

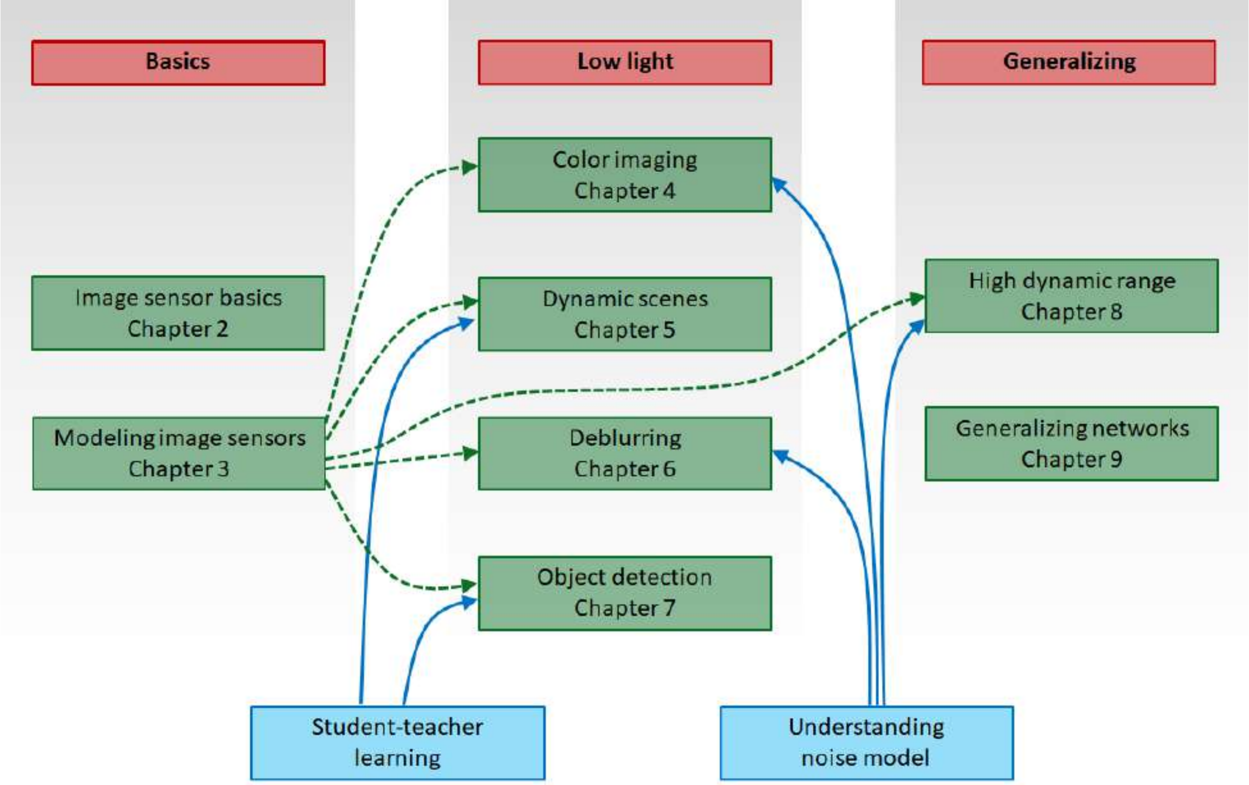


Figure 1.21. Overview of this dissertation.

different image sensor technologies fundamentally different. While the material covered in this chapter directly may not be helpful for the rest of the dissertation, this chapter will help us understand why specific properties such as read noise differ between different sensor technologies.

- *Chapter 3.*

A realistic camera model that considers all the different noise sources is vital for multiple purposes. 1. It helps in simulating artificial data when we cannot access real data firsthand. 2. Some of these noises are fixed-pattern and can easily be corrected if we understand their sources. 3. We need to understand the noise statistics to develop algorithms for attenuating them, which the imaging model helps with.

In addition to helping us generate data, the imaging model is also essential to study and compare different imaging technologies. For example, quanta image sensors generally run in binary mode, and a general CMOS image sensor uses a 14-bit mode. How do we compare these two image sensors? We need to play around with all the different possible settings of the image sensors to understand where the strengths and weaknesses of different image sensors lie.

The first part of the chapter develops the imaging model. The second part comes with a method for calculating the signal-to-noise ratio (SNR) for different image sensors. The second part of the chapter is based on the paper [20].

Low light imaging solutions

This dissertation looks at how we can solve 4 different imaging problems at low light - color imaging, dynamic scene reconstruction, deblurring, and object detection. Some of these problems have multiple solutions, and all the proposed solutions can be broadly categorized into two.

1. Solutions based on understanding the forward imaging and noise model. In this, we understand the noise model in the low light scenarios combined with the forward model of the task at hand to develop a solution that can deal with both the imaging model and the noise.

2. Solutions based on student-teacher learning. In this, we use a novel technique called student-teacher learning, where a pre-trained teacher network is used to distill knowledge into a student network for dealing with heavy noise.

- *Chapter 4.*

This chapter looks at the problem of color imaging at low light using quanta image sensors. We propose two demosaicing solutions to reconstruct color images from photon-limited Bayer pattern mosaic inputs.

The first solution is a traditional demosaicing solution combined with a variance stabilizing transform (VST) for dealing with the Poisson noise. This is based on the paper [21].

The second solution is a neural-network based solution that utilizes the fact that the *luma* channel of the image has good SNR compared to the other two chroma channels. This is based on the paper [22].

- *Chapter 5.*

  In this chapter, we propose to image a dynamic scene at low light using quanta image sensors (QIS). We propose a neural network based solution that is trained using a student-teacher training scheme so that the network can handle two difficult tasks of dealing with noise and motion. This chapter is based on the work [23].

- *Chapter 6.* In this chapter, we propose a deep learning based non-blind deblurring solution which can handle extremely heavy noise. The proposed method unrolls a 3-way split Plug-and-Play ADMM designed to handle Poisson noise. We also collect some real data captured at extremely low light to make sure the proposed method can work in the real world too. This chapter is based on the work [24].

- *Chapter 7.* This chapter looks at how we can solve the computer vision problems of object detection and image classification in low light. We use QIS to improve the performance in low light. The image classification method uses student-teacher learning to deal with the heavy noise in photon-limited scenarios. The object detection method uses student-teacher learning in tandem with non-local feature matching to detect objects in photon-limited scenarios. The classification method is based on the paper [25], and the object detection is based on the paper [26].

Generalizing to a wide range of light levels.

The solutions we have looked at till now deal only with low light images. However, they cannot capture both the low light and bright scenes simultaneously, and the methods proposed do not generalize well to multiple noise levels. We will be forced to train multiple models for different levels. In this part of the thesis, we look at solutions to this problem.

- *Chapter 8.*

  This chapter shows how QIS can help with high dynamic range imaging. We use the SNR expressions developed in chapter 3 to quantitatively address the dynamic range advantage that QIS brings to the table. We propose an optimal HDR reconstruction algorithm to combine multiple QIS images captured using exposure bracketing. This chapter is based on [27].

- *Chapter 9.*

  One of the major flaws of the current neural network solutions is that they need to be trained for the specific noise levels they are being used for. However, when used at a different noise level, they fail. This is not desirable, especially when we want the network to work in low-light scenes where the noise is heavy and bright scenes where the noise is not as strong. When training for multiple noise levels, the performance is good in low light but is usually found lacking in bright scenes. This chapter proposes a training scheme that re-distributes the noise samples over different noise levels so that the trained network performs uniformly well at all noise levels. This chapter is based on [28].

2. DIGITAL IMAGE SENSORS

This chapter has to be the most difficult to write in this dissertation. As a signal processing person, learning how devices work was like learning a new language from scratch. This chapter is also weird because it does not have any direct material output from this dissertation. However, having worked on the signal processing side of the image sensors for close to five years now, the lack of even a basic grasp of the beautiful physics behind making an image sensor work was always a nagging thought. Having this chapter here was a way of alleviating this worry. So, this chapter is one signal processing engineer's understanding of how image sensors work. The rest of this dissertation can be understood without the knowledge of this chapter, so the readers can feel free to skip this chapter. Readers with hardware background - Sorry about the blunders, which there are undoubtedly many.

The chapter starts with a brief history of modern cameras. It will be a biased history, given that this dissertation involves a lot of quanta image sensors. Therefore, the history tries to follow a single thread from the present day's quanta image sensors back to when film photography was invented. Many branches have been missed. Readers can take a look at [29] for a more detailed history of image sensors. We will then look at some basic understanding of how image sensors work, get into a bit of semiconductor physics, and finally look at the quanta image sensors and other single-photon devices, which play an integral part in the rest of this dissertation.

2.1 A brief history of modern cameras

One could argue that the history of imaging started with the discovery that we can use optics to project light rays from an object onto a screen. Wikipedia [30] suggests that it started with the camera obscura ("dark chamber" in Latin) in ancient China, Greek, and Byzantine. However, we will start with the technologies that could reproduce the captured scene or store the captured image in some form. In that sense, the history of the modern camera started with the invention of the photographic film by George Eastman in 1884 [31], which led to the development of the silver halide film technology.

The basic idea behind the silver halide films was that the silver halide reacts to light, and the photons break the bond in the silver halide compound to produce silver. The amount of light dictates the quantity of silver halide compound that gets broken to get silver. Then this film goes through a dedicated development process to get the final image. In silver halide cameras, the film acts as the medium for both the image capture and the storage.

With electronic cameras, the medium of capture and storage were separated. The image sensors captured the scene, and the captured data was stored in a separate memory. The idea for electronic cameras came as early as 1973, with Texas instrument filing the patent for a semiconductor image sensor [32]. There were a few more ideas of these electronic still cameras such as the one by Kodak in 1975 and the Sony Mavica camera in 1981. However, these cameras could not become successful because of the image and video quality. These images, which were stored in a floppy disk, did not have high enough quality to get a good printout of the photos. The first breakout success for a digital camera was the Casio QV-10. Since then, the digital camera market has been growing exponentially.

The early digital cameras were based on charge-coupled device (CCD) technology, invented by George Smith and Willard Boyle at AT&T's Bell Labs in 1969. CCD sensors measure the charge, and the analog-to-digital conversion (ADC) happens for all the pixels outside the sensor. CMOS technology was used for various other logic circuits even by the 1970s. An active pixel device, where photon measurement and amplification happen within the same device, was invented in 1968 by Peter Noble. Image sensors using CMOS technology were developed at Jet propulsion Lab (JPL) and later commercialized by Photobit by combining the active pixel sensor (APS) technology, the CMOS technology, and the process of doing intra-pixel charge transfer. The CMOS image sensors (CIS) combined imaging and the ADC inside the pixels themselves. Initially, CCD image sensors had a significant advantage over the CIS because of better read noise, dark current, and quantum efficiency, even though CIS was much faster and required lesser power consumption. However, over the years, with newer innovations and a better CIS pixel fabrication process, CIS was able to close the gap in terms of circuit noise levels. This led to CIS emerging as the leading imaging technology, with close to 8 billion units of CMOS image sensors being sold in 2021 alone.

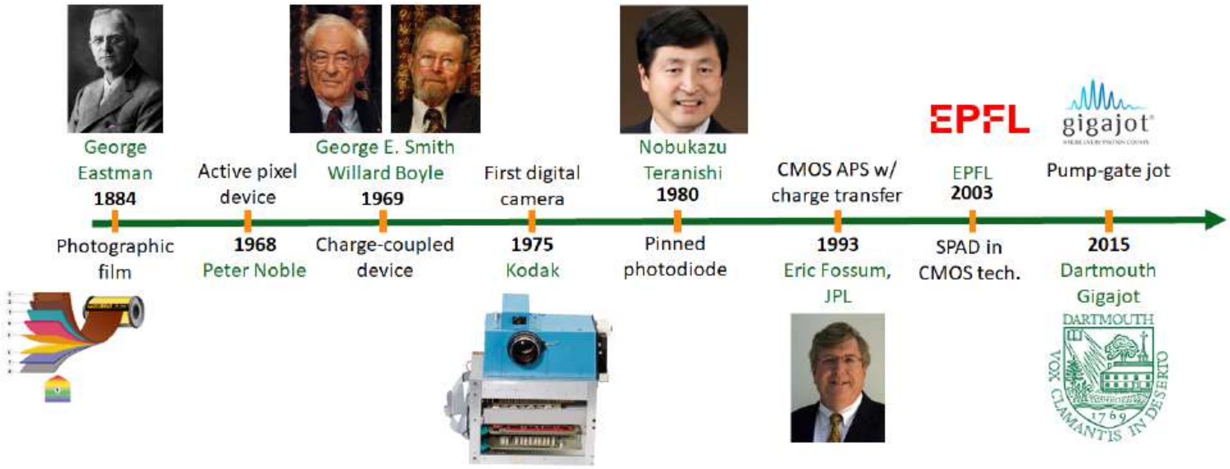


Figure 2.1. A timeline of modern image sensors.

Over the years, image sensors have become a ubiquitous part of our daily life. There is a camera with almost every one of us sitting inside our smartphones. Image sensors in cellphones have many restrictions regarding the size of the sensor and power consumption. Concurrently, we also want the quality not to drop and the resolution to be as large as possible, which has led to an almost constant decline in the image pixel size over the years. However, the smaller pixels come with a drawback in the lower signal-to-noise ratio (SNR) arising from, the lower full-well capacity. With this in mind, when Dr. Eric Fossum was asked about the future of digital cameras, he predicted the emergence of the quanta image sensors (QIS). QIS was envisioned to be a new type of image sensor with sub-diffraction-limit sized pixels, full well capacity as low as just 1 electron, the ability to count individual photons and image at thousands of frames per second [33].

While QIS was envisioned as just an idea in 2005, significant work has been done to make prototype QIS. A pump-gate device, which shares considerable similarities with the CIS technology, was proposed in 2015 by Ma et al., [34] having the photon-counting ability. Since then, the technology has undergone rapid innovations leading to read noise as low as 0.19e$^-$ in the latest iteration of the sensor [35]. The technology is also being commercialized by Gigajot Technology. An alternative technology that has emerged for QIS is single-photon avalanche diode (SPAD) [36]. SPAD achieves single-photon counting using avalanche multiplication, while CIS-based QIS does not. We will look into the differences in more detail in later sections of this chapter.

In the rest of this chapter, we will go into the working of the digital image sensors. The treatment will be at a high level, giving a brief overview to understand the noteworthy differences in the different digital sensor technologies.

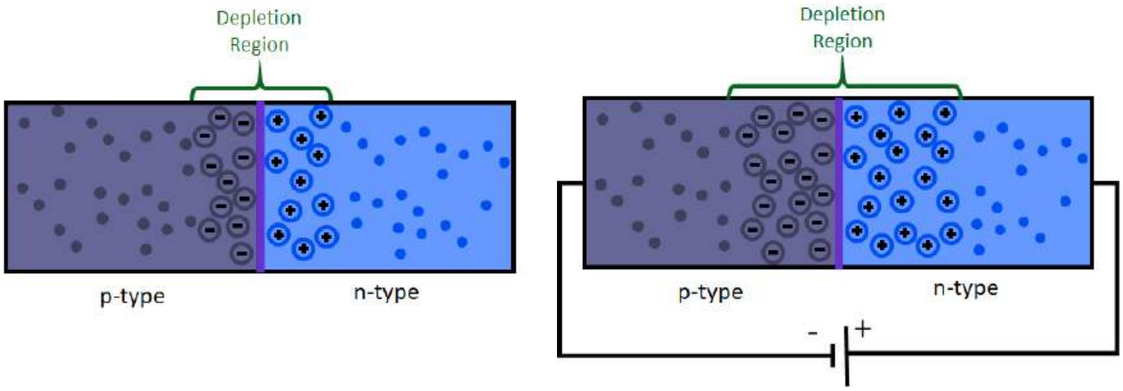
2.2 Semiconductor basics for digital image sensors

Semiconductor innovations are the primary driving force that made cameras possible. In this section, we will look at some semiconductor basics. We will not get into too much detail, just a brief introduction to the essential topics and a rough understanding of the overall idea behind what image sensors are made of.

2.2.1 Reverse-biased p-n junction

A *p-n junction* is formed by putting two types of materials, *p-type* and *n-type* together, such that they share a junction or a boundary. A p-type semiconductor has excessive holes (positive charge), and an n-type semiconductor has excessive electrons (negative charge). p-type semiconductors are produced by doping an intrinsic semiconductor such as silicon with an electron acceptor (e.g., boron). Similarly, n-type semiconductors are doped with an electron donor (e.g., phosphorous).

When a p-type semiconductor and an n-type semiconductor are placed next to each other, it leads to the formation of the *depletion region* as shown in Figure 2.2, which contains the positive and negative ions. The depletion region is formed because of the high concentration of the electrons and holes in the n-type and p-type semiconductors, respectively. The excess electrons in the n-type semiconductor are attracted by the p-type semiconductor holes and move there and vice versa. The size of the depletion region can be increased by reverse biasing the pn-junction. In reverse bias, we connect the positive terminal of the voltage to the n-type semiconductor, and the negative terminal is connected to the p-type semiconductor. This leads to more holes entering the p-type semiconductor and more electrons being drawn from the n-type semiconductor, thus increasing the size of the depletion region. The larger



(a) A p-n junction

(b) Reverse bias

Figure 2.2. A p-n junction. A depletion region forms at the junction of a p-type and n-type semiconductors. The depletion region can be further made larger by reverse biasing the p-n junction.

depletion region gives rise to a potential difference, which creates an electric field at the depletion region.

Remark 2.1. *As one can expect, forward biasing a p-n junction reduces the size of the depletion region.*

2.2.2 Photoelectric effect and photodiodes

Before we get into how pixels work, let us look at the *photoelectric effect*. The photoelectric is a process by which when light waves fall on a material, the photons transfer their energy to electrons in the material, and the electron becomes free. A more detailed discussion on the statistics involved in photons exciting electrons can be found in chapter 3. Here, let us look at the semiconductor physics on what happens during the photoelectric effect.

The amount of energy a photon carries depends on the frequency (or equivalently the wavelength) of the light wave. Each photon with a frequency of ν and wavelength $\lambda = \frac{c}{\nu}$ (c is speed of light in vacuum) carries energy

$$E = h\nu = \frac{hc}{\lambda}. \tag{2.1}$$

A photon needs more energy than the *band-gap*. The band-gap is the amount of energy required to excite an electron from the valence band (still held on to by the atom) to the conduction band (free from the atom). The band-gap for silicon is 1.2 eV. Photons from all the visible spectrum have more energy than the silicon band-gap. Red light ($\lambda = 700$nm), close to the lowest frequency in the visible spectrum, has an energy of 1.77 eV per photon. Each photon can excite at most one electron. However, an electron getting excited is not guaranteed. The probability of electrons getting excited by a photon is called *quantum efficiency* (QE).

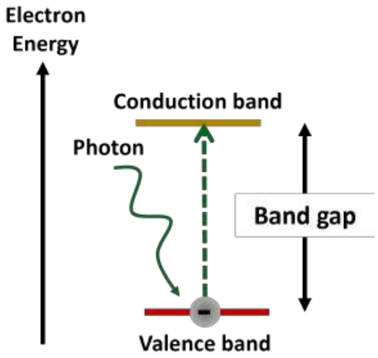


Figure 2.3. Photoelectric effect.A photon should have at the least the energy equal to the band-gap to excite an electron from the valence bond to the conduction band.

The photoelectric effect, in essence, generates a new electron-hole pair. If this happens in a reverse-biased p-n junction, the electric field at the depletion region pulls the holes towards the p-side and the electrons towards the n-side. This could also be represented in terms of the potential energy of the electrons like in Figure 2.4. Since there are more free electrons in n-type semiconductors, the potential energy of the electrons in n-type semiconductors is lower. Note that electrons in the valence band cannot move because they are bonded to their atoms. However, once an electron moves to the conduction band because of an incident photon, they are free to move towards lower potential energy. Therefore, if an electron becomes free in the p-type, it moves towards the n-type.

Notice what happens when the charges move towards their respective sides. The total overall potential difference falls, meaning that only a finite amount of electron-hole pairs that

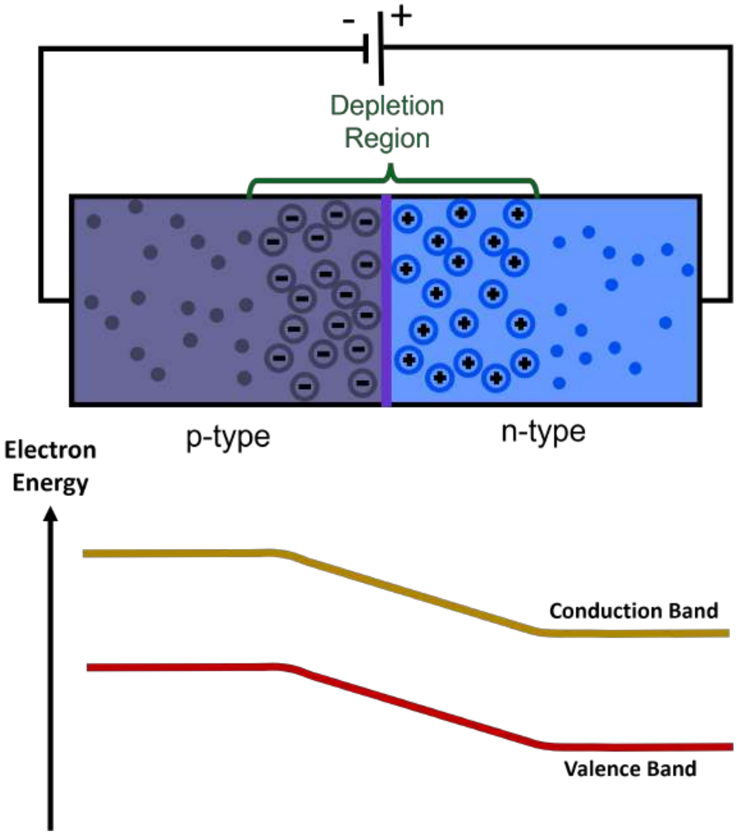


Figure 2.4. Energy band diagram. The lower potential energy for electrons in the n-type means that the electrons in the p-type, that get excited to the conduction band because of the photons, move to the n-type.

are generated will move towards the depletion region. Theoretically, this should equal the number of electrons (or holes) that have moved to the p-type (or n-type) semiconductor at the start. Once the number of photoelectrons matches this, the electric field becomes zero, and photoelectrons generated have no force to move them and may recombine the holes. This phenomenon of having only a finite number of electrons available to be generated is called the *full well capacity* of the pixels. Notice that the full well capacity depends on the reverse bias voltage used: the larger the bias voltage, the larger the full well capacity.

Figure 2.5 shows a simplified version of a *photodiode*. A photodiode is mainly made up of a reverse-biased p-n junction, among other things. Shorter wavelengths are absorbed faster, and longer wavelengths can penetrate deeper. Therefore, a photodiode's depth or thickness is

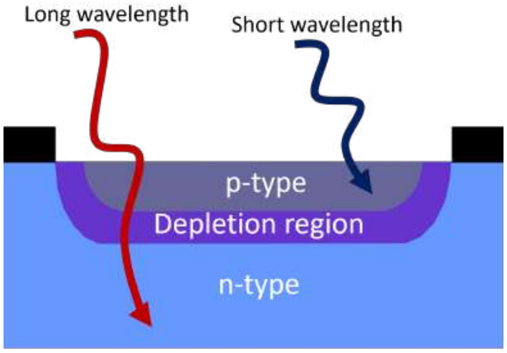


Figure 2.5. A cross section of a photodiode. A very simplified cross section of a photodiode.

an important design consideration. The number of photoelectrons generated can be measured by either transferring the charge outside and measuring it (CCD) or measuring it in the pixel itself (CIS).

Remark 2.2. *The image sensors usually react linearly to the number of photons, i.e., the number of photoelectrons is directly proportional to the number of photons falling on the pixel. However, this linearity does not hold near the full well capacity, as the electric field at the depletion region becomes too weak.*

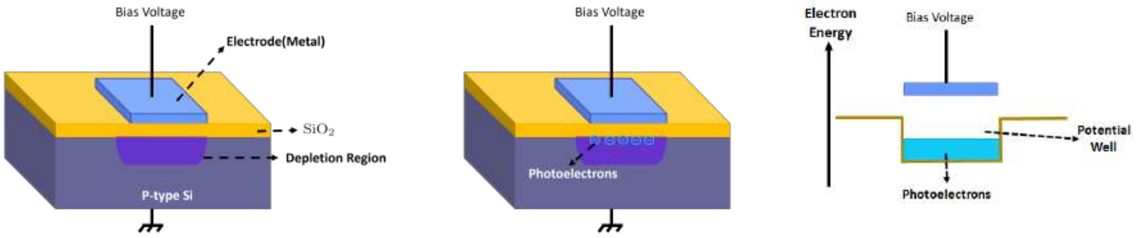
We have seen the basic principles of how a photodiode works. Let us now jump into some of the significant sensor technologies in the next section.

2.3 Charge-coupled device

Two primary functions of a CCD are to 1. accumulate photoelectrons, and 2. Transfer the charges between pixels. Let us first look at how these two are achieved.

2.3.1 MOS capacitors

The basic building block of a charge-coupled device (CCD) image sensor is the metal-oxide-semiconductor (MOS) capacitors. A MOS capacitor has a semiconductor body (or a substrate), an insulator and a metal electrode called a gate. The insulator is the equivalent of the dielectric of a capacitor. The metal acts as an n-type semiconductor in the usual p-n junction (because of the available valence electrons), and the semiconductor body is usually p-type silicon. A positive voltage applied at the gate repels the holes away from the gate, leading to the generation of the depletion region as shown in Figure 2.6 (a). When an electron gets excited in the substrate, it moves towards the gate, but the insulator stops it. The electrons get accumulated at the surface between the substrate and the insulator as shown in Figure 2.6 (b). We usually use a liquid model to visualize the accumulation of photoelectrons in the pixels as shown in Figure 2.6 (c). In some sense, the depletion region acts as a potential well, restricting the total amount of charge that can be accumulated. The accumulated charge is thought of as a liquid filling the well.

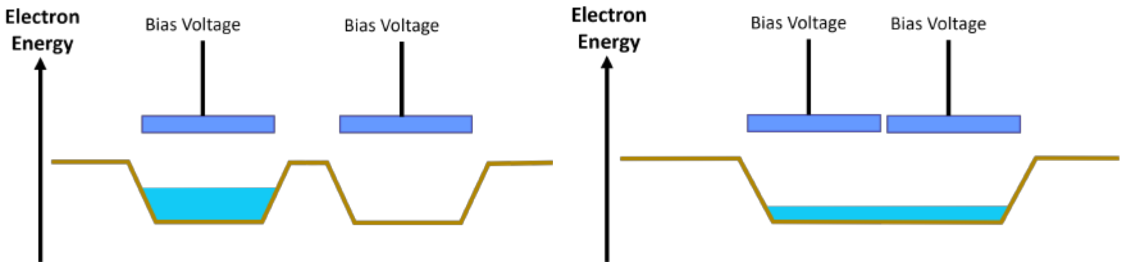


(a) MOS capacitor (b) With photoelectrons (c) Liquid model

Figure 2.6. MOS capacitor. When photoelectrons are generated, they move to the surface between the p-type substrate and SiO$_2$.

2.3.2 Transferring charges

To understand how the charges are transferred using MOS capacitors in CCDs, we start with how two MOS capacitors interact when placed close to one another. Consider the the example illustrated in Figure 2.7. We have two gates, both of which are connected to a positive voltage. A potential well gets created under these gates. Now, consider that one of these gates has accumulated photoelectrons and the other has not. Now, if a significant distance separates the two gates, as shown in Figure 2.7 (a), nothing special happens, and there exist two different potential wells. However, if these two gates are close enough, as shown in Figure 2.7 (b), the two potential wells merge to make one big potential well with the photoelectrons getting distributed across the entire potential well.

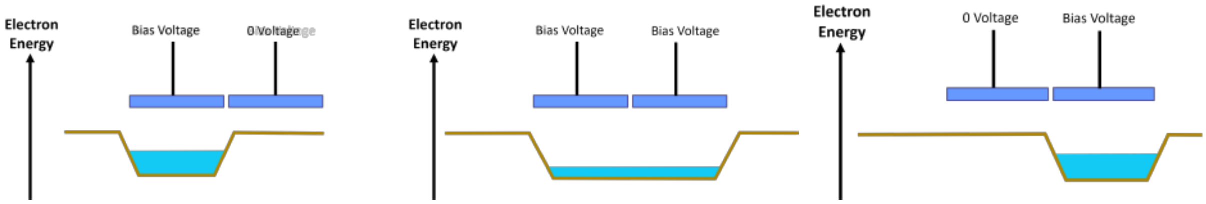


(a) The gates are well separated.

(b) The gates are closely placed.

Figure 2.7. Interaction of two MOS capacitors. When the gates are well separated, the two potential wells stay separate. However, if we reduce the gap between them, the potential wells combine to form one large well.

The above fact, combined with the creation of the potential well only when the bias voltage is on, can be used to transfer charges from one capacitor to a neighboring capacitor. Figure 2.8 illustrates how the charges are transferred from one pixel to another.



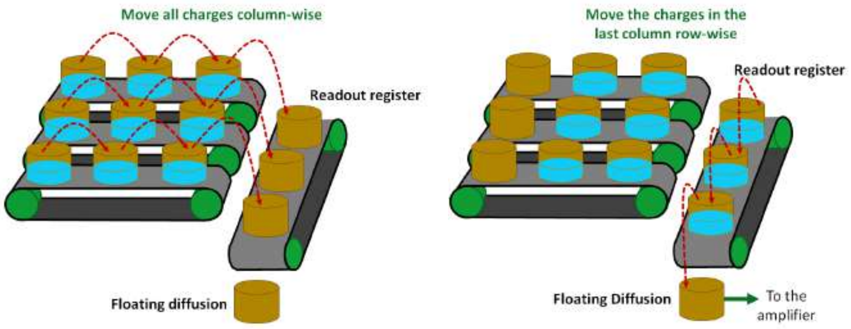
(a)

(b)

(c)

Figure 2.8. Transferring the charges. (a) When the bias voltage is applied on the left gate, there will be a potential well, and the right gate has no voltage applied on it and therefore no potential well. (b) When the right gate also gets the bias voltage, the potential well is created, and the two potential wells merge, and consequently, the photoelectrons get distributed across the large well. (c) If the bias voltage on the left gate gets turned off, all the photoelectrons will move to the right potential well. The charge transfer is now complete.

The next step is understanding how this is done for a large image sensor. It is achieved by employing sequential row and column readout. Figure 2.9 illustrates the idea of how this is done. We start by moving all the charges column-wise to the right. The last column moves into what is called the *readout register*. For a second, assume there exists something called the *floating diffusion* (FD) below the readout register that converts the charge into a digital number. We transfer the charge from the readout register one row down at a time. As the



(a) (b)

Figure 2.9. Bucket brigade. The charges are transferred from the whole column to the next column, and then the last column is read one pixel at a time. This routine is repeated till all pixels are readout.

charges get into the floating diffusion, the data gets readout. So, one by one, we read the amount of charge from each row of the readout register. Once this is done, we move all the remaining charges to the right by one column again and start reading the readout register row by row again. We keep repeating this till all the pixels are read out.

The floating diffusion at the bottom of the right-most column can be considered a capacitor. Once the charges get transferred from a pixel to the floating diffusion, the voltage is read out by first amplifying the signal using two to three voltage followers and sending the amplified analog signal through an analog-to-digital converter (ADC) to get the digital data.

Remark 2.3. *The quality of the signal readout depends on the amplification used for reading out the final signal: more amplification, lesser noise. However, we cannot use arbitrarily large amplification because higher amplification needs high power. The ISO in the digital cameras is nothing but choosing what amplification we want to use.*

Remark 2.4. ***Conversion gain*** *is the voltage signal generated per photoelectron. For CCD image sensors, this is nothing but the amplification used before the ADC.*

2.3.3 The need for a new image sensor technology

We have seen till now the basic working of a CCD image sensor. It is called the surface channel CCD design. Over the years, many innovations were done to make the CCD work better: The buried channel design, where the charges are stored some distance below the surface, instead of the Si - SiO_2 interface - The interline transfer mechanism, which separates the charge collector and the charge transfer device - A vertical overflow bin, which reduces the amount of blooming and smearing in the captured images. Combined with these innovations, CCDs had a few significant advantages that even the CMOS APS that came later did not have. They are

1. CCD image sensors are optimized photodetectors. They do not serve any other purpose, which means they had an extremely high quantum efficiency (QE) and low dark current.

2. The amount of noise introduced is minor. The shifting process is almost perfect.

3. Given that there is only one floating diffusion and amplifier, no non-uniformity exists between the pixels.

However, CCDs had a few disadvantages, which forced the development of the future CMOS APS image sensors. They are

1. CCD image sensors require high power consumption. They also have a high voltage requirement of 10-15V.

2. The read-out mechanism, which requires transferring all the charges one by one, poses a big bottleneck for scaling in terms of speed (frame-rate), especially when the image sensor array is big.

3. Since the pixels are highly optimized for being photodetectors and do not have any other circuitry, it becomes difficult to add any other functionality inside the pixels.

2.4 CMOS image sensors

CMOS refers to Complementary Metal Oxide Semiconductors. CMOS technology has been in use in many analog and digital circuits since the 1970s. The emergence of the CMOS APS image sensors, which are in use today in billions of devices, started with the invention of image sensors that integrated pinned photodiode (PPD) [37] and intra-pixel charge transfer in 1993. With the CMOS APS becoming better over the years, they have almost totally replaced CCD image sensors as the default choice for image sensors [38].

The major advantage that CMOS image sensors offer over the CCD image sensors are these:

1. CMOS APS have in-pixel read-out circuits, which means that the electric charges get amplified and converted to digital numbers in-pixel. The in-pixel read-out circuits combined with X-Y addressing provide the CMOS image sensors with great flexibility in terms of binning pixels and skipping pixels, which could make the image sensors faster in times of need.

2. Their power consumption is smaller than the CCD and can operate at the 3V range.

3. The in-pixel circuit are highly programmable, making the CMOS APS extremely flexible to be modified for specific applications.

2.4.1 Pinned photodiodes

Pinned photodiodes are an essential part of the modern CMOS APS. Figure 2.10 shows a simplified pinned photodiode and how the charge transfer works for it. n+ type and p+ type semiconductors are just doped such that the electrons and holes are available in a higher concentration than the n-type and p-type semiconductors. The pinned photodiodes are doped with n-type, and the floating diffusions are doped with n+ type. The basic idea is that the floating diffusion junction leads to a larger potential well. When the transfer gate (TG) is off, the collected photoelectrons in PPD stay there. The collected charges flow towards FD when TG is on because of the potential difference. The implanted p+ at the top of PPD

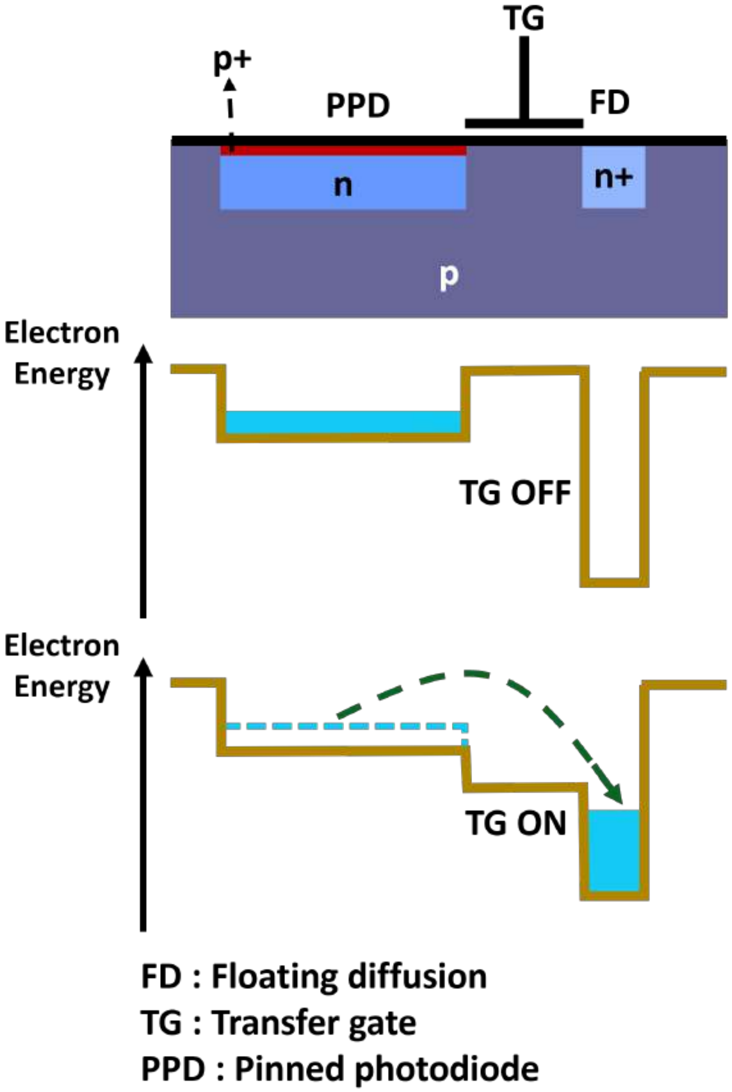


Figure 2.10. Pinned photodiode. The doping and the potential well diagram of the operation of the pinned photodiode. In default mode, TG is off, and the photoelectrons get collected in the photodiode. When TG is turned on, the charges flow into FD.

increases the charge transfer efficiency to the FD [39]. One of the significant advantages of the pinned photodiode design is the ability to do true *correlated double sampling* (CDS). In CDS, we read the voltage at FD twice. Once when TG is off and again after the charge transfer. Then the change in potential ΔV_{FD} is read out as the signal from the pixel. This way, the quality of the signal read is greatly improved.

2.4.2 CMOS APS

Figure 2.11 shows the schematic diagram of four transistor active pixel sensor (APS). The four transistors are - 1. Transfer gate (TG), 2. Reset (RST), 3. Source follower (SF), and 4. Row select (SEL). Once the transfer gate is turned on, the photoelectron charges generated at the photodiode are transferred to the floating diffusion (FD). The change in potential ΔV_{FD} at FD determined by the capacitance C_D of the FD node and the photogenerated charge Q_{Ph} transferred from PPD to FD.

$$\Delta V_{FD} = \frac{Q_{Ph}}{C_D} \tag{2.2}$$

We can see that the number of electrons transferred is simply Q_{Ph}/q_e, where q_e is the charge carried by a single electron. Then, conversion gain (CG) is given by

$$\text{Conversion Gain} = \frac{\Delta V_{FD}}{Q_{Ph}/q_e} = \frac{q_e}{C_D} \tag{2.3}$$

For a general CMOS APS, conversion gain is usually in the range $50\mu V/e^-$ to $100\mu V/e^-$, which translates to a read noise of a few electrons ($> 2e^-$). Once the charge has been transferred to FD, turning the SF on will read the amplified signal. The column bus and SEL are used to select which pixel we want to read the signal from. Note that the signal goes through another amplification (ISO) before getting sent to the ADC.

Remark 2.5. *The conversion gain we have defined here is sometimes called the in-pixel conversion gain. The overall conversion gain is the total amplification, including the ISO amplification.*

The technology is called active pixel sensor (APS) because the job of the photodiode is just to generate photoelectrons. The pixel has a separate structure for dealing with the amplification of signals.

While CMOS image sensors give the advantage of being faster, since each pixel has its own circuit, it suffers from non-uniformities such as photon response non-uniformity (PRNU)

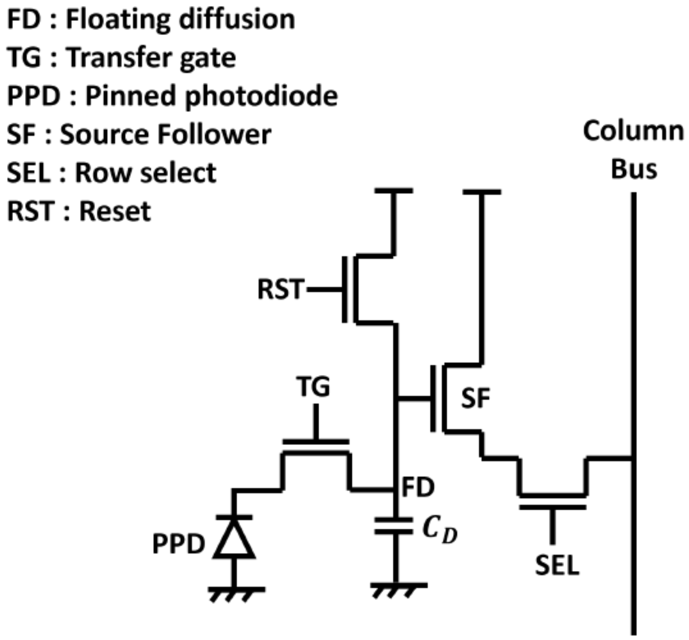


Figure 2.11. **Four transistor active pixel with source follower read-out.**

and dark signal non-uniformity (DSNU). Because the circuit also occupies some pixel area, which in turn affects the fill factor (ratio of photosensitive area to total area), the sensitivity takes a hit. However, the sensitivity issue can be alleviated by techniques such as back-side illumination (BSI) and the usage of the microlens. Table 2.1 shows how CCD and CMOS image sensors compare in some of the essential factors.

Table 2.1. Comparing CCD and CMOS image sensors. Source: [40]

Factor	CCD	CIS
Power	High	Low
Functionality	Off chip	On chip
Cost	Higher	Lower
Speed	Lower	Higher
Sensitivity	Higher	Lower
Dynamic range	Higher	Lower
Image Quality	Higher	Lower
Fill factor	Better	Poor
Quantum efficiency	High	Low

While it may look like CCD are better for higher image quality, CIS has been able to close the gap over years of innovation by incorporating technologies such as buried channel photodiodes, correlated double sampling, and backside illumination. Combined with the fact that CIS offers extensive flexibility for on-chip functionalities, CIS has become the obvious choice for image sensors.

2.5 Single photon counting image sensors

While the CMOS image sensor technology has grown leaps and bounds, they still have a significant issue when dealing with *photon limited imaging*, where the number of photons arriving at the image sensor is limited. It is not desirable to have a read noise of a few electrons at this light level. We want to be able to do photon counting, where the image sensor is sensitive to every available photon to retain as much information as possible from the arriving photons. Counting every photon is the final frontier in image sensing. In this section, let us take a look at three different technologies that strive to achieve this goal :

1. CIS-based quanta image sensor (CIS-QIS),

2. Single-photon avalanche diode (SPAD), and

3. Electron multiplying CCD (EMCCD).

Often, for passive imaging, SPAD and CIS-QIS are grouped into a single family of QIS.

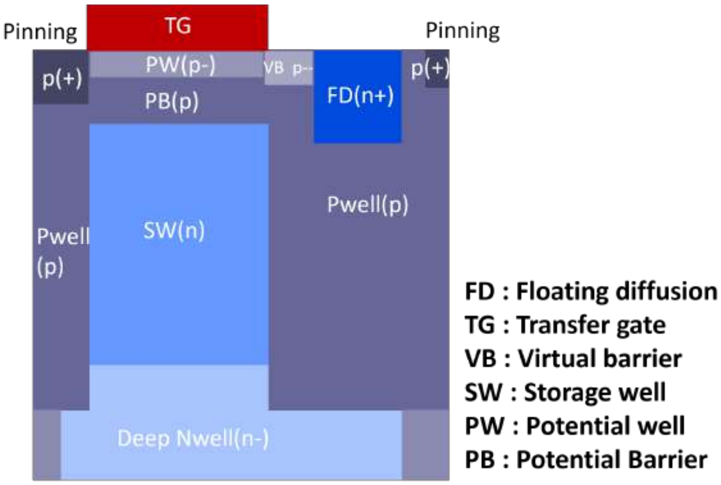


Figure 2.12. Schematic of pump-gate jot doping.

2.5.1 CIS-based quanta image sensor

Pump-gate jot

Pump-gate jot is the basic building block of CIS-based QIS. As we mentioned earlier, the conversion gain in a CIS depends on the floating diffusion capacitance C_D. The in-pixel conversion gain needs to be more than $1000\mu V/e^-$, to achieve less than $0.15e^-$ read noise, which can be considered as good as zero read noise [34]. The pump-gate jot proposed in [34] achieves a conversion gain of $380\mu V/e^-$ with full well capacity of $200e^-$. The actual prototype based on this idea can achieve $0.19e^-$ read noise [35].

To reduce the capacitance of the FD, [34] increases the distance between the TG and FD. They call it the "distal" FD. Given the distance, it becomes difficult to transfer the charge in one go. So, the paper proposes to use a "pump" action, where the photoelectrons are first stored in a potential well when TG is on, and when TG is off, the charges move to FD. Pump-gate also reduces the overlap capacitance between the reset gate and FD by using a tapered reset. Figure 2.12 shows the doping profile of the proposed pump-gates. Figure 2.13 shows the ideal charge transfer during the pump action.

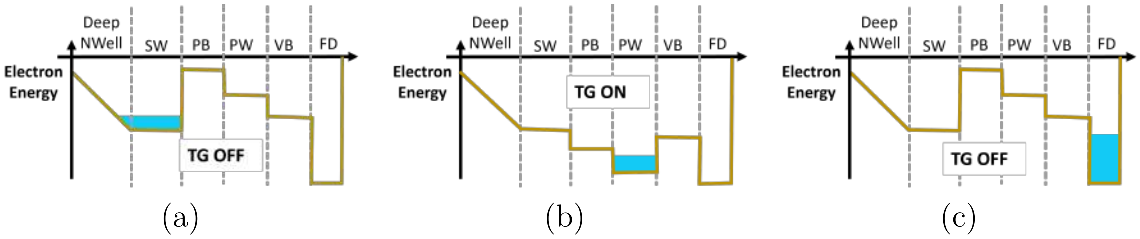


Figure 2.13. Pump action idealized charge transfer diagram. (a) The transfer gate is initially OFF, and Photoelectrons are collected in the storage well. (b)When the transfer gate is ON, photoelectrons are transferred to the potential well. (c) When the transfer gate is OFF again, photoelectrons are transferred to the floating diffusion.

Quanta image sensor

The image sensor which uses pump-gate jot can achieve read noise as low as $0.19e^-$. Figure 2.14 shows real data captured using a prototype CIS-based QIS, where we can clearly resolve the photon numbers. The current version of CIS-based QIS can operate in both single-bit and multi-bit modes. They have a full well capacity of close to 200 electrons, giving the QIS the ability to run in different modes based on the need. We can reduce the number of bits used in the ADC and get an extremely fast QIS, or we can use multiple bits. Both of them may be useful. The jury is still out on understanding the trade-offs and limitations of using QIS at different bit-depths.

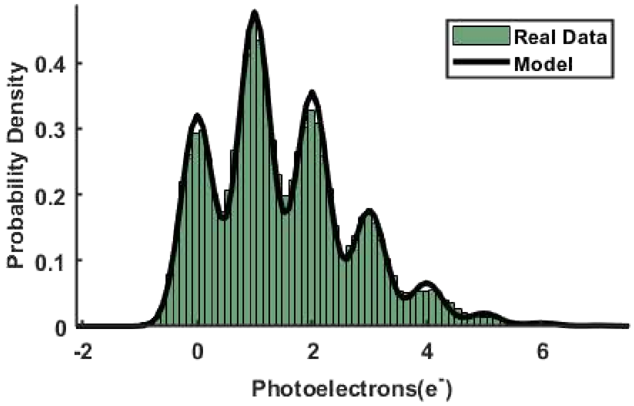


Figure 2.14. Photon counting histogram. CIS-QIS has such a low read-noise that we can resolve each photon arriving at the sensor. First reported in [41].

CIS-based QIS comes with almost all the remarkable properties of a CIS, since they share many similarities: For example, the flexibility in programming. The proposed QIS prototype was able to leverage the advancements in CIS image sensors such as the BSI, CDS, and microlens to achieve exceptional read noise, fill factor, QE, and dark current (Table 2.2).

2.5.2 Single photon avalanche diode

The mode of operation of single-photon avalanche diode is quite different from the image sensors we have seen till now. A doping profile is shown in Figure 2.15. SPAD is reverse biased with voltage greater than the breakdown voltage (Geiger mode), leading to even a single free electron starting an avalanche and consequently generating a huge amount of current. When a photon generates a single photoelectron in this setting, it leads to an avalanche. Recall that CIS-based QIS can keep counting the photons as long as we want. However, that is not the case with SPAD. Once an electron is detected, we need to reset the sensor before using the sensor again. This phenomenon leads to a dead-time while operating SPAD, where the sensor needs to be off to reset the pixels.

SPAD has been in use since the 1970s. SPAD using CMOS technology was invented in 2003 [36], leading to the development of a lot of different prototypes [42].

SPAD needs a large voltage (15-20V) for breakdown, and the power consumption is also higher because of the avalanche. The pixel sizes of SPAD are usually larger than CIS-QIS. However, SPAD can reach more than 100k frames per second [43]. Because of this speed and the ability to figure out precisely when the avalanche is happening, SPAD is better suited for time-resolved imaging such as time-of-flight imaging, where we need to resolve when each photon arrives at the sensor.

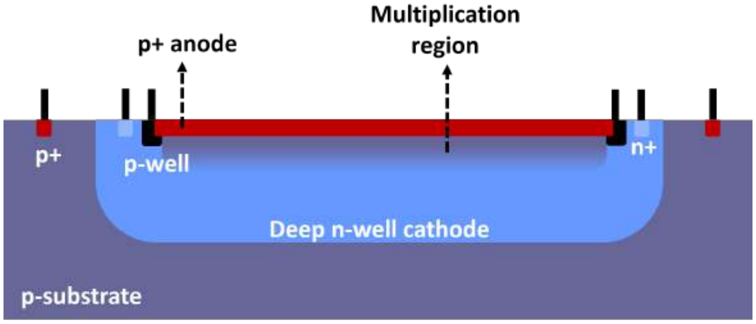


Figure 2.15. Doping profile of a SPAD.

2.5.3 Electron-multiplying CCD

Electron-multiplying charge-coupled device (EMCCD) is fundamentally a CCD but equipped with multiple gain registers between the shift registers and the ADC. Figure 2.16 illustrates this idea. Each gain register tries to do electron multiplication using impact ionization. Impact ionization is very similar to the operation of the avalanche diodes, where a high electric field accelerates the photon charge in hopes of starting an avalanche. However, the probability of a single electron starting an electron multiplication is very low ($< 2\%$) in EMCCD. EMCCD achieves a suitable electron detection probability by having multiple such registers,

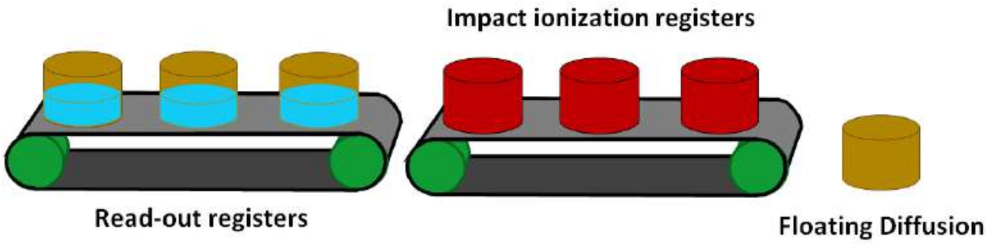


Figure 2.16. EMCCD. In CCD, the charges move directly from the readout registers into the floating diffusion. In contrast, EMCCD makes the charges go through multiple registers with high electric fields to accelerate the charge, leading to electron multiplication.

which will give a significantly high gain when stacked together. Compared to SPAD and CIS-QIS, EMCCD has a very high dark current and must be operated under low temperatures ($-80°\text{C}$) to achieve good imaging results.

2.5.4 Comparing the three technologies

While choosing which of the three single-photon counting technologies to use, we need to consider the pros and cons of different technologies. In Table 2.2 we compare some of the critical properties of the three technologies. While SPAD and CIS-QIS can operate at room temperatures, EMCCD needs to be cooled down for achieving single-photon counting. Among SPAD and CIS-QIS, CIS-QIS has a better dark current, smaller pixels, and a larger resolution. Of course, SPAD has the advantage of having zero read noise, which may play an issue when summing many frames. These technologies are evolving rapidly, and it would be interesting to see the future applications these technologies will be used for.

2.6 Final thoughts

We have looked at the fundamental workings of some of the standard image sensor technologies. As we have noted, the image sensors technologists have systematically identified the existing drawbacks in their technologies and figured out ways to fix them time and again. Given the author's limited knowledge, it would not be appropriate to try and guess the future

Table 2.2. Comparison of the available single photon image sensor technologies.

Properties	CIS-QIS [35]	SPAD [44]	EMCCD [45]
Resolution	16.7 Mpixels	3.2 Mpixels	1Mpixels
Operating temp.	Room temp.	Room temp.	$-100°$ C
Mean dark count	0.086e$^-$/pix/s	$<$ 3e$^-$/pix/s	0.00011e$^-$/pix/s @ $-100°$ C
Max QE	76%	69.4%	$>$ 95%
Pixel pitch	1.1μm	6.39μm	13μm
Frames/sec	1000fps [46]	156 kfps [47]	56 fps
Read noise	0.19e$^-$	0e$^-$	$<$ 1e$^-$(@ $-100°$ C)

direction that image sensors are going to take. However, as a signal processing engineer who works a lot in computational imaging, it will be tremendous to see image sensors with a lot more control from outside than they currently have. In some sense, a *software-defined camera* is every computational imaging engineer's dream.

3. MATHEMATICAL MODELING OF A CAMERA

We live in the deep learning era, and data is the fuel that drives progress. When developing new algorithms, we need real data to train the models and even test and ensure that the model will work when employed in a real case scenario. However, real-world data is hard to come by, especially when we are working on cutting-edge image sensors, where, by the time we have a prototype in our hands, the technology has already progressed. The prototype is not cutting edge anymore. In some cases, we cannot use the real data from the sensors for proprietary reasons. How do we deal with such scenarios?

A solid workaround for this problem is to have a sound understanding of the working of the image sensors and model its image formation process to simulate artificial data whenever we want. Simulating the imaging model has the additional benefit of simulating scenarios where it is impossible to collect data. For example, a prototype sensor may not have the right design to be mounted on a microscope. However, we can simulate the imaging model using existing microscopy images and understand how the camera may work.

A realistic camera model that considers all the different sources of noise is also important because some of these noises are fixed-pattern and can easily be corrected if we understand their sources. On top of that, we need to understand the noise statistics to develop algorithms for attenuating them, which the imaging model helps with.

Understanding all the different noise sources in an image sensor is often unnecessary. In fact, we need not consider any noise source for most of the applications that use images captured in well-lit scenarios. At most, we may need to consider a weak i.i.d. Gaussian noise. A cursory glance at the computer vision literature will make this evident. Works that consider even signal-dependent noise sources are few and far between. Most of the works do not consider anything more than weak Gaussian noise. However, when dealing with *photon-limited imaging*, it becomes imperative to consider all the different sources of noise. At these light levels, methods based on a simple Gaussian model do not perform well.

In addition to helping us generate data, the imaging model is also essential to study and compare different imaging technologies. For example, quanta image sensors generally run in binary mode. A general CMOS image sensor uses a 14-bit mode. How do we compare these two image sensors? We need to play around with all the different possible settings of the image sensors to understand where the strengths and weaknesses of different image sensors lie.

This chapter starts by looking at different noise sources in an image sensor. Then we put all the noise sources together to give a single imaging model. The imaging model can be modified to fit any image sensor we want. We then develop the theory for the signal-to-noise ratio of the image sensors, which can be used to compare different image sensors in different modes. The first part of this chapter, especially the subsection on photon shot noise, borrows a lot of material from [48]. The latter part, where we discuss the signal-to-noise ratio, is based on our work [20].

3.1 Sources of noise

This section will go into all the different possible sources of noise. Some of these noise sources can be fixed using a simple look-up table. For example, once we figure out all the dead pixels in the image sensor, we need to replace the dead pixel values with their neighboring pixels, which can be done in real-time using a look-up table. Other noise sources such as the circuit read noise are not as easy to fix, as we will need a more complicated algorithm to fix it. We will also ignore all the lens aberrations and assume that the lenses we use are all perfect. All the explanations of different noise sources are done from a signal processing perspective. Unless necessary, we will not be going into too many details on the sensor physics that causes all these noise sources. For more detailed discussion on this topic refer [29], [49]

3.1.1 Photon arrival process

The photon arrival is a stochastic process, and this randomness is a direct consequence of the particle nature of the electromagnetic waves. In imaging literature, this randomness is called the *shot noise*. In this subsection, we show some properties of the photon arrival process relevant to us. For a more rigorous treatment of this topic, check Chapter 9 of [48].

Consider a light wave with intensity $I(x, y; t)$, arriving at the image sensor present on the xy plane. We make three assumptions about the photon arrival.

1. The probability of a single photon arriving at a area smaller than the coherence area[1] and shorter than the coherence time[2] (but longer than period of the wave) is proportional to the intensity $I(x, y; t)$ of the light wave. Let K denote the number of photons arriving at the image sensor in a time interval Δt and area ΔA. The assumption says, for some α
$$\mathbb{P}(K = 1; \Delta t, \Delta A) = \alpha\ \Delta t\ \Delta A\ I(x, y; t)). \tag{3.1}$$

2. The probability of more than one photon arriving in the time interval Δt and area ΔA is negigible. i.e. $\mathbb{P}(K > 1; \Delta t, \Delta A) \approx 0$. Therefore, we can write
$$\mathbb{P}(K = 0; \Delta t, \Delta A) = 1 - \alpha\ \Delta t\ \Delta A\ I(x, y; t)). \tag{3.2}$$

3. The photon arrival process in any non-overlapping area or time interval is independent of each other.

Let us look at the probability distribution of the number of photons K arriving at the image sensor at an area of ΔA in some arbitrary time interval $(T, T + \tau + \Delta\tau)$, where $\Delta\tau$ and ΔA satisfy the coherence condition mentioned in assumption 1, and τ is the counting interval we are interested in. Since we are considering a fixed area ΔA around (x, y), we will ignore ΔA, x, and y in the equations for the time being.

$$\begin{aligned}\mathbb{P}(K = k; (T, T + \tau + \Delta\tau)) = \ & \mathbb{P}(K = k - 1; (T, T + \tau)) \cdot \mathbb{P}(K = 1; (T + \tau, T + \tau + \Delta\tau)) \\ & + \mathbb{P}(K = k; (T, T + \tau)) \cdot \mathbb{P}(K = 0; (T + \tau, T + \tau + \Delta\tau)).\end{aligned} \tag{3.3}$$

[1] ↑The largest area such that two pinholes placed anywhere within this area always produces interference.

[2] ↑The maximum time before which the wave can always be considered coherent. Check Section 3.1.2 for more details.

Note that we have used the independence property from assumption 3 to write the above equation. Now, using (3.1) and (3.2), we can rewrite the above equation as

$$\begin{aligned} \mathbb{P}(K = k; (T, T + \tau + \Delta\tau)) &= \mathbb{P}(K = k - 1; (T, T + \tau)) \cdot \alpha \ \Delta t \ I(T + \tau) \\ &+ \mathbb{P}(K = k; (T, T + \tau)) \cdot [1 - \alpha \ \Delta t \ I(t + \tau)]. \end{aligned} \tag{3.4}$$

We can now rearrange the terms in the above equation to get

$$\begin{aligned} &\frac{\mathbb{P}(K = k; (T, T + \tau + \Delta\tau)) - \mathbb{P}(K = k; (T, T + \tau))}{\Delta\tau} \\ &\qquad = \alpha I(t + \tau)[\mathbb{P}(K = k - 1; (T, T + \tau)) \quad - \mathbb{P}(K = k; (T, T + \tau))] \end{aligned} \tag{3.5}$$

Letting $\Delta\tau \to 0$, we get

$$\frac{d\mathbb{P}(K = k; (T, T + \tau))}{d\tau} = \alpha I(T + \tau)[\mathbb{P}(K = k - 1; (T, T + \tau)) - \mathbb{P}(K = k; (T, T + \tau))] \tag{3.6}$$

The following theorem provides the solution to this equation.

Theorem 3.1.1. *Consider the differential equation*

$$\frac{d\mathbb{P}(K = k; (T, T + \tau))}{dt} = G(T + \tau)[\mathbb{P}(K = k - 1; (T, T + \tau)) - \mathbb{P}(K = k; (T, T + \tau))]. \tag{3.7}$$

with the constraint $\sum_{k=0}^{\infty} \mathbb{P}(K = k; (T, T + \tau)) = 1$. *The solution to this differential equation is given by*

$$\mathbb{P}(K = k; (T, T + \tau)) = \frac{\left(\int_T^{T+\tau} G(t)dt\right)^k}{k!} exp\left\{- \int_T^{T+\tau} G(t)dt\right\}, k \in \{0, 1, 2, \dots\} \tag{3.8}$$

Proof. We will prove that (3.8) is a solution to (3.7). The proof that it is the only solution is a little more involved and we will skip it here.

Substitute (3.8) into (3.7). Let $\theta(\tau) = \left(\int_T^{T+\tau} G(t)dt\right)$. We get

$$\begin{aligned}
\frac{d\mathbb{P}(K=k;(T,T+\tau))}{dt} =& \frac{d}{d\tau}\left[\frac{\theta(\tau)^k}{k!} exp\left\{-\theta(\tau)\right\}\right] \\
=& \frac{\theta(\tau)^k}{k!}\frac{d}{d\tau}\left[\exp\left\{-\theta(\tau)\right\}\right] + exp\left\{-\theta(\tau)\right\}\frac{d}{d\tau}\left[\frac{\theta(\tau)^k}{k!}\right] \\
\overset{(a)}{=}& \frac{\theta(\tau)^k}{k!}\left[-G(T+\tau)\exp\{-\theta(\tau)\}\right] + exp\left\{-\theta(\tau)\right\}\left[\frac{1}{k!}k\theta(\tau)^{k-1}G(T+\tau)\right] \\
=& G(T+\tau)\left[-\exp\{-\theta(\tau)\}\frac{\theta(\tau)^k}{k!} + \exp\{-\theta(\tau)\}\frac{\theta(\tau)^{k-1}}{(k-1)!}\right]
\end{aligned}$$

which is same as the right hand side of (3.7). Here, (a) uses the fact that $\frac{d}{d\tau}\theta(\tau) = \frac{d}{d\tau}\int_T^{T+\tau} G(t)dt = G(T+\tau)$. □

By replacing $G(t) = \alpha I(t)$ in (3.8), we get the solution to (3.6). If we let $\theta = \int_T^{T+\tau}\alpha I(t)dt$

$$P(K=k;(T,T+\tau)) = \frac{\mathrm{e}^{-\theta}\theta^k}{k!}, \tag{3.9}$$

which is the well known Poisson PMF distribution.

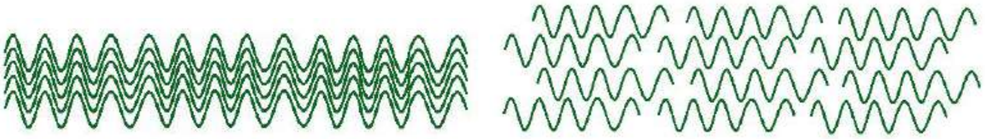
The discussion till now has assumed we have a fixed small area ΔA and a longer time interval τ. Following similar steps, we can also show that the same result holds for a larger area. In a more general version of (3.9), we will have

$$\theta = \iint_{(x,y)\in A}\int_T^{T+\tau} \alpha I(x,y;t)dtdxdy. \tag{3.10}$$

3.1.2 Coherent and incoherent light

We have derived the photon arrival process assuming that we know the intensity $I(x,y;t)$. However, this is not always true. To understand when this is not true, we need to understand the concept of *coherence* of electromagnetic waves.

Two light waves are considered coherent as long as they have a constant phase difference at the same point in space or time. A light source that generates light waves all with the same phase is considered a coherent light source. In contrast, an incoherent light source generates light waves with random phases.Figure 3.1 shows a visualization of coherent and incoherent light waves. For example, lasers are coherent light sources, and the sun is an incoherent light source. In general, thermal light sources such as the sun and light bulbs are incoherent.



(a) Coherent (b) Incoherent

Figure 3.1. Coherence of electromagnetic waves. When all the light waves have the same phase, it is called the coherent light. If the phases are all random, it is called incoherent light.

For a coherent light source, our assumption about knowing the intensity $I(x, y, ; t)$ is true, and therefore it follows a Poisson arrival process. However, for incoherent lights we know only the average intensity and the actual intensity itself is a random process, which turn makes θ in (3.10) a random variable. Then (3.9) is conditioned on θ. So we get

$$\mathbb{P}(K = k \mid \theta) = \frac{\mathrm{e}^{-\theta}\theta^k}{k!}. \tag{3.11}$$

Now we can write $\mathbb{P}(K = k)$ as

$$\begin{aligned}\mathbb{P}(K = k) &= \int_{\theta=0}^{\infty} \mathbb{P}(K = k \mid \theta)\mathbb{P}_\Theta(\theta)d\theta \\ &= \int_{\theta=0}^{\infty} \frac{\mathrm{e}^{-\theta}\theta^k}{k!}\mathbb{P}_\Theta(\theta)d\theta. \end{aligned} \tag{3.12}$$

This equation is usually referred to as Mandel's formula [50].

Incoherent light

This part of the chapter involves a lot of brute force math, so some of the more involved proofs will be skipped. Interested readers can refer to [48] for detailed proofs.

When the light source is incoherent, for any arbitrary counting interval τ it can be shown that the distribution $\mathbb{P}_{\Theta}(\theta)$ follows a Gamma distribution (Chapter 6 of [48])

$$\mathbb{P}_{\Theta}(\theta) = \frac{1}{\alpha}\left(\frac{\alpha\mathcal{M}}{\beta}\right)^{\mathcal{M}} \cdot \frac{\left(\frac{\theta}{\alpha}\right)^{\mathcal{M}-1} \exp\left\{-\mathcal{M}\frac{\theta}{\beta}\right\}}{\Gamma(\mathcal{M})}, \tag{3.13}$$

where $\mathcal{M}$ represents the degrees of freedom in the measurement interval and $\beta = \mathbb{E}[\theta]$. When only temporal degrees of freedom are allowed

$$\mathcal{M} = \left[\frac{1}{\tau}\int_{-\infty}^{\infty} \Lambda\left(\frac{\eta}{\tau}\right) \mid \gamma(\eta) \mid^2 d\eta\right]^{-1}, \tag{3.14}$$

where $\gamma(\eta)$ is the complex degree of coherence (Chapter 5 of [48]) and

$$\Lambda(\tau) = \begin{cases} 1- \mid \tau \mid, & \mid \tau \mid \leq 1 \\ 0, & \text{otherwise} \end{cases}. \tag{3.15}$$

Substituting (3.13) into (3.12) we will get

$$\mathbb{P}(K = k) = \frac{\Gamma(k+\mathcal{M})}{\Gamma(k+1)\Gamma(\mathcal{M})}\left[1 + \frac{\mathcal{M}}{\beta}\right]^{-k}\left[1 + \frac{\beta}{\mathcal{M}}\right]^{-\mathcal{M}}. \tag{3.16}$$

This distribution is referred to as the *negative-binomial* distribution. This distribution is for any arbitrary counting interval τ. We will look at two special cases when τ is very small and long compared to the coherence time.

Short counting interval

When the counting interval τ is smaller than the coherence time, $\mathcal{M}$ is essentially unity, and by substituting $\mathcal{M} = 1$ in (3.13) we can show that θ follows an exponential distribution

$$\mathbb{P}_{\Theta}(\theta) = \frac{1}{\beta}\mathrm{e}^{-\frac{\theta}{\beta}}, \quad \theta \geq 0, \tag{3.17}$$

and substituting it into (3.16), we get

$$\mathbb{P}[K = k] = \frac{1}{1+\beta}\left(\frac{\beta}{1+\beta}\right)^k. \tag{3.18}$$

This is called the *Bose-Einstein* distribution.

Long counting interval

For longer counting interval τ, $\mathcal{M} \to \infty$. To reflect this, let us assume $\mathcal{M} = \beta/\delta$ for arbitrarily small δ. Then (3.16) becomes

$$\mathbb{P}(K = k) = \frac{\Gamma(k+\beta/\delta)}{k!\Gamma(\beta/\delta)}\left[(1+1/\delta)^k\,(1+\delta)^{\frac{\beta}{\delta}}\right]^{-1}. \tag{3.19}$$

Using Stirling's approximations [51], we can show that

$$\Gamma\left(\frac{\beta}{\delta}\right) \approx \sqrt{2\pi}\left(\frac{\beta}{\delta}\right)^{\beta/\delta-0.5} \mathrm{e}^{-\frac{\beta}{\delta}}, \quad \text{and}$$
$$\Gamma\left(k+\frac{\beta}{\delta}\right) \approx \sqrt{2\pi}\left(k+\frac{\beta}{\delta}\right)^{k+\beta/\delta-0.5} \mathrm{e}^{-k-\frac{\beta}{\delta}}.$$

For $\delta \to 0$, we can show

$$\left(1+\frac{1}{\delta}\right)^k \approx \left(\frac{1}{\delta}\right)^k, \qquad \text{and} \qquad (1+\delta)^{\frac{\beta}{\delta}} \approx \mathrm{e}^{\beta}$$

Substituting these into (3.19), we get

$$\mathbb{P}(K = k) = \frac{\mathrm{e}^{-\beta}\beta^k}{k!} \cdot \left[\left(1+\frac{k\delta}{\beta}\right)^{k+\frac{\beta}{\delta}-0.5} \mathrm{e}^{-k}\right]. \tag{3.20}$$

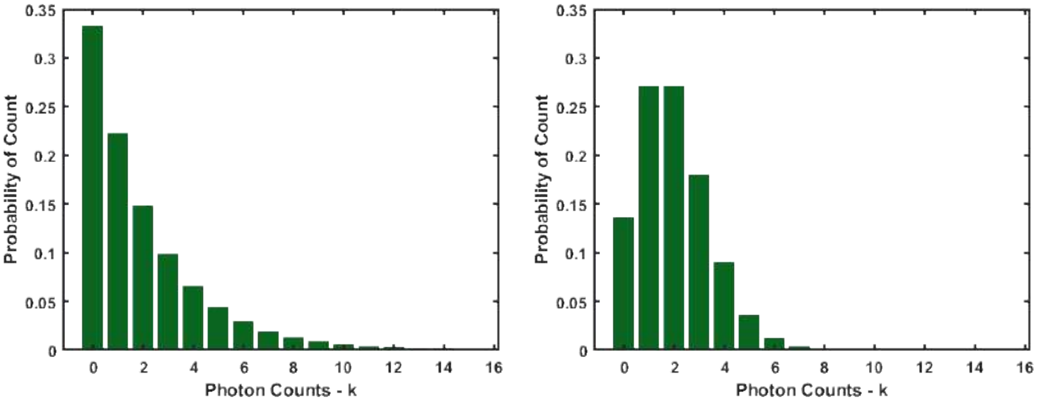
For $\delta \to 0$,

$$\left(1+\frac{k\delta}{\beta}\right)^k \approx \mathrm{e}^k.$$

So, we get

$$\mathbb{P}(K = k) = \frac{\mathrm{e}^{-\beta}\beta^k}{k!}, \tag{3.21}$$

which is the well known *Poisson* distribution.



(a) Bose-Einstein Distribution

(b) Poisson Distribution

Figure 3.2. Photon statistics. PMF of Bose-Einstein and Poisson probability distributions. Both the distributions have a mean of $\beta = 2.0$.

Figure 3.2 shows the PMF of the Bose-Einstein and Poisson probability distributions for mean of $\beta = 2.0$.

Remark 3.1. *According to [48], at a wavelength of 500nm, we need a counting interval of* $10^{-12}s$ *to satisfy the conditions for* (3.18)*, which is too short of an interval. We can safely assume that the photons arriving at the image sensor follow the Poisson distribution for almost all imaging applications we consider in this dissertation.*

3.1.3 Inter-arrival time

In most imaging applications, we are interested in photon counts. However, there are specific applications such as time-of-flight imaging [52] where knowing the time stamp of when the photon is arriving at the image sensor is essential. Recently Ingle et al., [53] showed that inter-arrival timing could be used for passive imaging too. For such cases, it is essential to know the probability distribution of the photon-inter arrival time. The following theorem gives the probability distribution of inter-arrival time when the photon arrival follows a Poisson process.

Theorem 3.1.2. *Let $K(t)$ be a uniform Poisson random process with constant rate $\gamma(t) = \gamma$. Let $\Delta\tau$ denote the time interval between two consecutive Poisson events. Then the random variable $\Delta\tau$ has the probability density function*

$$f_{\Delta\tau}(t) = \gamma \mathrm{e}^{-\gamma t}, \quad t \geq 0. \tag{3.22}$$

Proof. For a time interval t, the pmf of the Poisson random variable K is given by,

$$\mathbb{P}(K = k) = \frac{\mathrm{e}^{-\gamma t}(\gamma t)^k}{k!}. \tag{3.23}$$

Let $\Delta\tau$ be the time interval between two arrivals, which means there are no arrival in the time $\Delta\tau$. So, we can write

$$\begin{aligned} \mathbb{P}(\Delta\tau > t) &= \mathbb{P}(K = 0) \\ &= \mathrm{e}^{-\gamma t} \end{aligned} \tag{3.24}$$

We know that $\mathbb{P}(\Delta\tau > t) = 1 - F_{\Delta\tau}(t)$, where $F_{\Delta\tau}(t)$ is the CDF of the inter-arrival time. Therefore,

$$F_{\Delta\tau}(t) = 1 - \mathrm{e}^{-\gamma t} \tag{3.25}$$

Taking derivative of $F_{\Delta\tau}(t)$ wrt t will give us the required pdf. □

3.1.4 Photons to electrons

We showed in the last subsection that the photon arrival process is a Poisson process. Let us look at a single pixel with some area A, in a counting interval τ. The number of photons arriving at the sensor is a random variable with probability distribution according to (3.21). The photons excite the electrons in the pixel by a process known as the *photoelectric effect*. Figure 3.3 shows an illustration of the process.

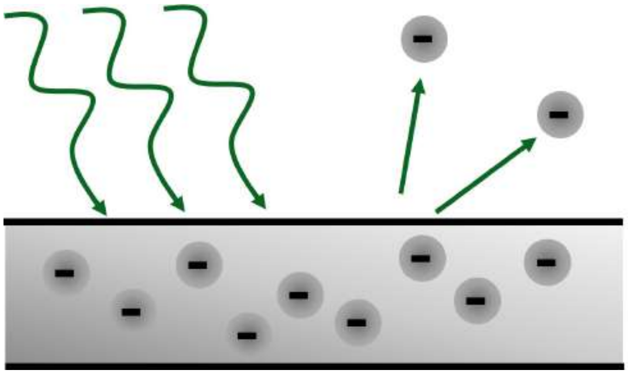


Figure 3.3. **Photoelectric effect.** Phototelectric effect refers to the process by which electrons get excited by light waves.

The photoelectric effect says that one photon can excite only one electron. However, not all photons excite electrons. Each incident photon only excites $0 \leq \eta \leq 1$ electrons on average. η is called the *quantum efficiency* of the image sensor. Image sensors usually have different quantum efficiency at different wavelengths. Most of the image sensors we generally use (e.g., the ones on our cellphones) have high sensitivity to the visible range of the electromagnetic spectrum (400nm – 700nm). For mass-market cameras, the main goal is to reproduce images that look like what a human eye sees. There are other types of image sensors such as the IR image sensors [54] which work in a spectrum different from the visible range. Figure 3.4 shows an example quantum efficiency curve for a camera changing over different wavelengths.

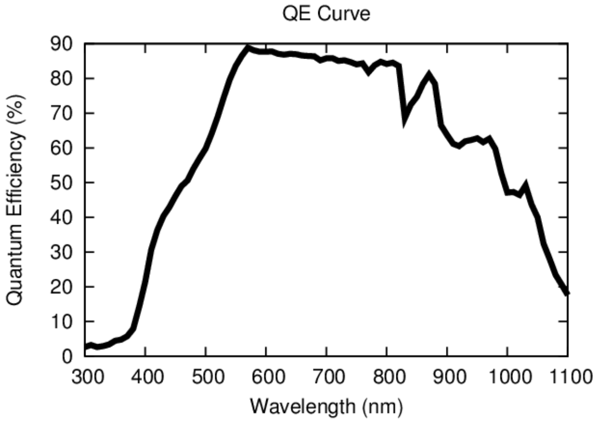


Figure 3.4. **Quantum efficiency.** An example quantum efficiency plot from [55].

Monochromatic light

We will start with a simple case - We assume that the light is monochromatic, where spectrum of the light consists of just a single wavelength λ_0. Let the intensity be $I(x, y, t; \lambda_0)$. We will also assume the photons follow a Poisson process. Let the average number of photons arriving at a particular pixel be $\beta(\lambda_0)$. The number of photons arriving at the sensor k has the probability distribution according to (3.21). We model the photon detection by the pixel as a Bernoulli random variable with probability of detection as $\eta(\lambda_0)$. Then each photon gets detected according to the distribution

$$\mathbb{P}(\text{detection}) = \eta(\lambda_0), \quad \mathbb{P}(\text{miss}) = 1 - \eta(\lambda_0).$$

The number of photons detected (or number of electrons generated) is the sum of $K = k$ Bernoulli random variables. It is well known that the sum of Bernoulli random variables follow a binomial distribution [56]. So, the number of photons detected by the pixel $K_\text{e} = k_\text{e}$ conditioned on the number of photons arriving at the pixel follows the distribution

$$\mathbb{P}\left(K_\text{e} = k_\text{e} \mid K = k\right) = \binom{k}{k_\text{e}} \cdot \eta(\lambda_0)^{k_\text{e}} \cdot (1 - \eta(\lambda_0))^{(k - k_\text{e})}, \quad k_\text{e} \in \{0, 1, \ldots k\}, \tag{3.26}$$

where

$$\binom{k}{k_\text{e}} = \frac{k!}{k_\text{e}!(k - k_\text{e})!}. \tag{3.27}$$

The following theorem gives the expression for the probability distribution $\mathbb{P}(K_\text{e} = k_\text{e})$.

Theorem 3.1.3. *Consider a random variable K_e that has the probability distribution*

$$\mathbb{P}\left(K_\text{e} = k_\text{e} \mid K = k\right) = \binom{k}{k_\text{e}} \cdot \eta^{k_\text{e}} \cdot (1 - \eta)^{(k - k_\text{e})}, \quad k_\text{e} \in \{0, 1, \ldots k\}, \tag{3.28}$$

where K is a Poisson random variable with the distribution

$$\mathbb{P}(K = k) = \frac{\text{e}^{-\beta}\beta^k}{k!}. \tag{3.29}$$

K_e is then a Poisson random variable with distribution

$$\mathbb{P}(K_\mathrm{e} = k_\mathrm{e}) = \frac{\mathrm{e}^{-\eta\beta}(\eta\beta)^{k_\mathrm{e}}}{k_\mathrm{e}!}. \tag{3.30}$$

Proof. We will use moment generating functions to prove this. For some $t \in \mathbb{R}$, the moment generating function of K_e is given by

$$\begin{aligned} M_{K_\mathrm{e}}(t) &= \mathbb{E}_{K_\mathrm{e}}\left[\mathrm{e}^{tK_\mathrm{e}}\right] \\ &= \mathbb{E}_K\left\{\mathbb{E}_{K_\mathrm{e}|K}\left[\mathrm{e}^{tK_\mathrm{e}}\right]\right\}. \end{aligned} \tag{3.31}$$

The moment generating function of a binomial random variable is given by

$$\mathbb{E}_{K_\mathrm{e}|K}\left[\mathrm{e}^{tK_\mathrm{e}}\right] = (1-\eta+\eta\mathrm{e}^t)^K. \tag{3.32}$$

Therefore,

$$\begin{aligned} M_{K_\mathrm{e}}(t) &= \mathbb{E}_K\left[(1-\eta+\eta\mathrm{e}^t)^K\right] \\ &= \mathbb{E}_K\left[\mathrm{e}^{\log(1-\eta+\eta\mathrm{e}^t)K}\right]. \end{aligned} \tag{3.33}$$

Notice that $\mathbb{E}_K\left[\mathrm{e}^{\log(1-\eta+\eta\mathrm{e}^t)K}\right]$ is similar to the moment generating function of a K, $\mathbb{E}_K\left[\mathrm{e}^{sK}\right]$ where $s = \log(1-\eta+\eta\mathrm{e}^t)$. The moment generating function of Poisson random variable is

$$\begin{aligned} M_{K_\mathrm{e}}(t) = \mathbb{E}_K\left[\mathrm{e}^{sK}\right] &= \mathrm{e}^{\beta(\mathrm{e}^s-1)} \\ &= \mathrm{e}^{\beta(\mathrm{e}^{\log(1-\eta+\eta\mathrm{e}^t)}-1)} \\ &= \mathrm{e}^{\beta(1-\eta+\eta\mathrm{e}^t-1)} \\ &= \mathrm{e}^{\eta\beta(\mathrm{e}^t-1)}. \end{aligned} \tag{3.34}$$

By comparing $M_{K_\mathrm{e}}(t)$ with the moment generating function of a Poisson random variable we can clearly see that K_e is Poisson random variable with mean $\eta\beta$. □

We can now write

$$\mathbb{P}(K_{\mathrm{e}} = k_{\mathrm{e}}) = \frac{\mathrm{e}^{-\eta(\lambda_0)\beta(\lambda_0)}(\eta(\lambda_0)\beta(\lambda_0))^{k_{\mathrm{e}}}}{k_{\mathrm{e}}!} \tag{3.35}$$

We have shown that the number of electrons excited by a monochromatic light source follows a Poisson process. However, we do not encounter monochromatic light in real life often, with few exceptions, like when we use a laser as the light source. So, we need to extend the result to non-monochromatic lights too.

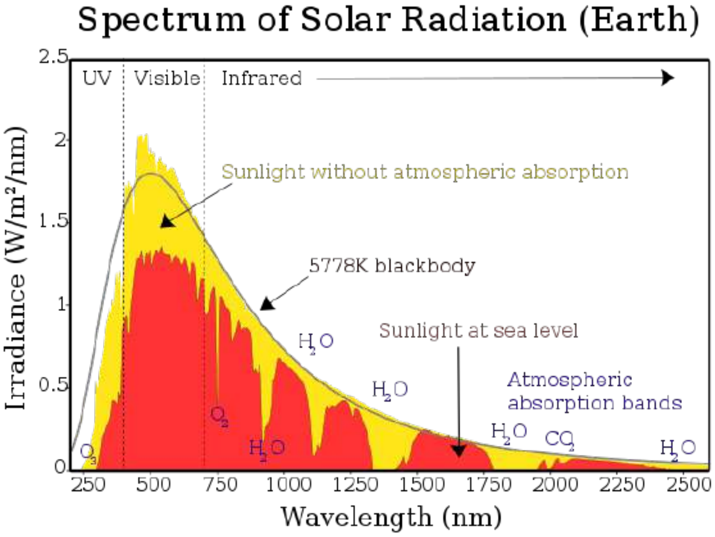


Figure 3.5. Spectrum of a light source. Source: [57]

Non-monochromatic light

Figure 3.5 shows an example spectrum of the sunlight. The irradiance $J(\lambda)$ can easily be converted into into photon rate by using

$$\Phi(\lambda) = \frac{J(\lambda)\lambda}{hc}, \tag{3.36}$$

where h is the Planck's constant, and c is the speed of light. $\Phi(\lambda)$ will have the unit photons/second/unit area/nm. Assuming $\Phi(\lambda)$ remains the same spatially, it can be thought of as the probability distribution of the wavelength to which each photon arriving at the image sensor belongs to. Let

$$f(\lambda) = \frac{\Phi(\lambda)}{\int_0^\infty \Phi(\lambda) d\lambda}. \tag{3.37}$$

Then for each of the K photons arriving at the pixel, the wavelength is a random variable with pdf $f(\lambda)$. Remember that for each wavelength the quantum efficiency of the pixel $\eta(\lambda)$ changes.

So, now we can write the probability distribution of the number of photons detected by the pixel as

$$\mathbb{P}(K_{\mathrm{e}} = k_{\mathrm{e}} \mid K = k, \boldsymbol{\lambda} = \{\lambda_1, \lambda_2, \dots \lambda_k\}) = \sum_{Q \in \mathcal{F}_{k_{\mathrm{e}}}} \left\{ \prod_{\mathrm{i} \in Q} \eta(\lambda_{\mathrm{i}}) \right\} \cdot \left\{ \prod_{\mathrm{i} \in Q^c} [1 - \eta(\lambda_{\mathrm{i}})] \right\}, \tag{3.38}$$

where $\mathcal{F}_{k_{\mathrm{e}}}$ contains all the subsets of $\{1, 2, \dots k\}$ that has k_{e} elements. This distribution is called the Poisson binomial distribution. We assume that all λ_{i} are independent of each other.

The following theorem gives the probability distribution $\mathbb{P}(K_{\mathrm{e}} = k_{\mathrm{e}})$.

Theorem 3.1.4. *Consider a random variable K_{e} has the probability distribution according to*

$$\mathbb{P}(K_{\mathrm{e}} = k_{\mathrm{e}} \mid K = k, \boldsymbol{\lambda} = \{\lambda_1, \lambda_2, \dots \lambda_k\}) = \sum_{Q \in \mathcal{F}_{k_{\mathrm{e}}}} \left\{ \prod_{\mathrm{i} \in Q} \eta(\lambda_{\mathrm{i}}) \right\} \cdot \left\{ \prod_{\mathrm{i} \in Q^c} [1 - \eta(\lambda_{\mathrm{i}})] \right\}, \tag{3.39}$$

where λ_{i} are all independent random variable that has some probability distribution $f(\lambda)$, and K is a Poisson random variable with the distribution

$$\mathbb{P}(K = k) = \frac{\mathrm{e}^{-\beta} \beta^k}{k!}. \tag{3.40}$$

K_{e} is then a Poisson random variable with distribution

$$\mathbb{P}(K_{\mathrm{e}} = k_{\mathrm{e}}) = \frac{\mathrm{e}^{-\bar{\eta}\beta} (\bar{\eta}\beta)^{k_{\mathrm{e}}}}{k_{\mathrm{e}}!}, \tag{3.41}$$

where $\bar{\eta} = \int_0^\infty \eta(\lambda) f(\lambda) d\lambda$.

Proof. We will again use the moment generating function to prove this.

$$M_{K_\mathrm{e}}(t) = \mathbb{E}_K \left\{ \mathbb{E}_{\boldsymbol{\lambda}|K} \left\{ \mathbb{E}_{K_\mathrm{e}|K,\boldsymbol{\lambda}} \left[\mathrm{e}^{tK_\mathrm{e}} \right] \right\} \right\} \tag{3.42}$$

$K_\mathrm{e} \mid K, \boldsymbol{\lambda}$ is a Poisson binomial distribution. It is nothing but sum of K independent Bernoulli random variables with different probability of success each time. Hence, the moment generating function is

$$\mathbb{E}_{K_\mathrm{e}|K,\boldsymbol{\lambda}} \left[\mathrm{e}^{tK_\mathrm{e}} \right] = \prod_{\mathrm{i}=1}^{K} \left(1 - \eta(\lambda_\mathrm{i}) + \eta(\lambda_\mathrm{i}) \mathrm{e}^t \right) \tag{3.43}$$

Now,

$$\begin{aligned} \mathbb{E}_{\boldsymbol{\lambda}} \left\{ \mathbb{E}_{K_\mathrm{e}|K,\boldsymbol{\lambda}} \left[\mathrm{e}^{tK_\mathrm{e}} \right] \right\} &= \mathbb{E}_{\boldsymbol{\lambda}} \left[\prod_{\mathrm{i}=1}^{K} \left(1 - \eta(\lambda_\mathrm{i}) + \eta(\lambda_\mathrm{i}) \mathrm{e}^t \right) \right] \\ &\overset{(a)}{=} \prod_{\mathrm{i}=1}^{K} \mathbb{E}_{\lambda} \left[1 - \eta(\lambda) + \eta(\lambda) \mathrm{e}^t \right] \\ &= \left[1 + \bar{\eta}(\mathrm{e}^t - 1) \right]^K, \end{aligned} \tag{3.44}$$

where (a) is possible because λ_i are all independent, and $\bar{\eta} = \mathbb{E}_{\lambda}(\eta(\lambda))$. We can now write the moment generating function of K_e as

$$\begin{aligned} M_{K_\mathrm{e}}(\mathrm{e}^{tK_\mathrm{e}}) = \mathbb{E}_K \left\{ \mathbb{E}_{\boldsymbol{\lambda}} \left\{ \mathbb{E}_{K_\mathrm{e}|K,\boldsymbol{\lambda}} \left[\mathrm{e}^{tK_\mathrm{e}} \right] \right\} \right\} &= \mathbb{E}_K \left\{ \left[1 + \bar{\eta}(\mathrm{e}^t - 1) \right]^K \right\} \\ &= \mathbb{E}_K \left\{ \mathrm{e}^{\left[\log\left(1 + \bar{\eta}(\mathrm{e}^t - 1) \right) K \right]} \right\}. \end{aligned} \tag{3.45}$$

From here we can follow the steps similar to the proof for Theorem 3.1.3 to complete the proof. □

Thus we have proven that irrespective of whether the light is monochromatic or not, the number of photons detected by the image sensor follows a Poisson distribution. The results in Theorems 3.1.3 and 3.1.4 hold for coherent light, and incoherent light with longer counting interval.

Remark 3.2. *A color filter array is placed in front of the image sensor in most modern image sensors to detect different colors. Hence, the quantum efficiency curve $\eta(\lambda)$ will vary for each pixel.* Figure 3.6 *shows one such example. The QE curve will vary depending on whether the pixel has a blue, green, or red filter.*

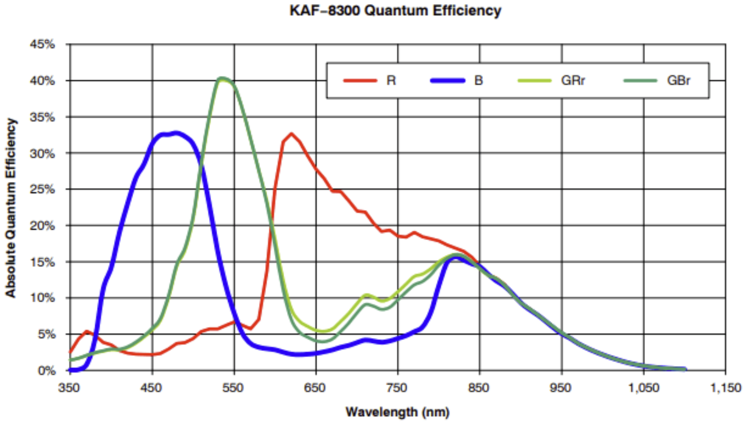
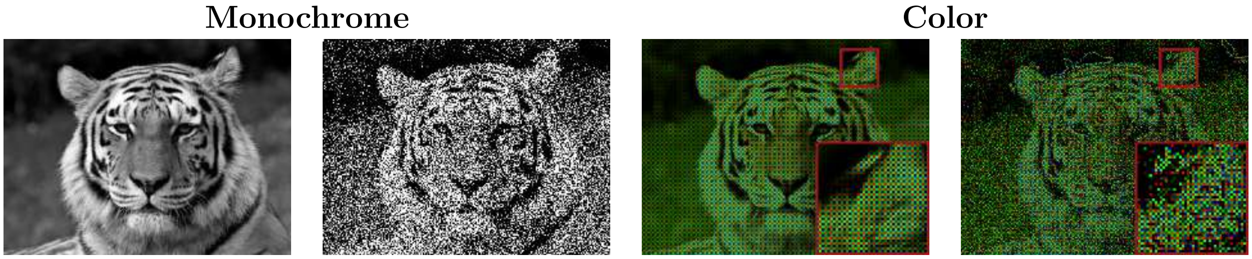


Figure 3.6. Quantum Efficiency with color filter arrays. Source : [58]



Mean electrons $\bar{\eta}\beta$ Realization K_e Mean electrons $\bar{\eta}\beta$ Realization K_e

Figure 3.7. Effect of shot noise. The average number of photoelectrons per pixel is 0.5. The color images assume that a Bayer pattern CFA is used.

In Figure 3.7, we show an example of shot noise simulated in an image. We take a clean image and then normalize it such that the average intensity is 0.5. And then use MATLAB's *poissrnd* function to generate the image labelled K.

3.1.5 Dark current

Dark current refers to the current generated in the pixels in the absence of light. It is undesirable as it hinders the actual number of photons we want to measure. Dark current reduces the dynamic range, i.e., the range of photons that can be detected (formally defined in chapter 8), as the total number of electrons that can be stored in the pixel is limited. The dark current creates many issues in low light imaging when the number of photons from the scene becomes comparable to the spurious electrons generated by the dark current. There are multiple different sources for dark current. We will not examine in detail all these different sources. Interested readers can refer to **janesick2001_ScientificCharge**, [29] for more details. The dark current is a function of the integration and the operating temperature. The electrons collected in the pixel due to the dark current, K_d, can be modeled as a Poisson random variable.

$$\mathbb{P}(K_d = k) = \frac{\mathrm{e}^{-\beta_d}(\beta_d)^k}{k!}, \tag{3.46}$$

where $\beta_d = \gamma_d \tau$ is the mean number of electrons accumulated due to the dark current. Here, τ is the integration time and γ_d is the mean number of electrons generated because of the dark current per unit time.

The total number of electrons accumulated K in the pixel is the sum of the two random variables $K = K_\mathrm{e} + K_d$ [3]. So, suppose β_e is the mean number of electrons excited by the arriving photons, and β_d is the mean number of electrons accumulated due to the dark current. The total number of electrons accumulated in a pixel follows the Poisson distribution with mean $\beta_\mathrm{tot} = \beta_\mathrm{e} + \beta_d$.

$$\mathbb{P}(K = k) = \frac{\mathrm{e}^{-(\beta_\mathrm{e}+\beta_d)}(\beta_\mathrm{e} + \beta_d)^k}{k!}. \tag{3.47}$$

[3] ↑For simplicity, we call the total number of electron K too, not to be confused the number of photons arriving. Since we will concern only with the number of photoelectrons, we can safely use the notation without causing any confusion.

Remark 3.3. *In most modern CMOS image sensors, the dark current can virtually be ignored for normal photography applications. For example, [35] reports a dark current of 0.086* e$^-$*/pix/s. Considering the common imaging integration time of* 30 *ms, the average number of electrons accumulated due to dark current will be* $\beta_d = 0.003$*, which we can comfortably ignore.*

3.1.6 Read noise

Once the charges are accumulated in the pixels, these charges need to be readout. Read noise refers to the noise generated by the readout electronics of the pixels, and this noise is generally modeled as additive Gaussian noise,

$$\eta_{\text{read}} \sim \mathbb{N}(0, \sigma^2_{\text{read}}), \tag{3.48}$$

where σ_{read} is the standard deviation of the noise referred to as the read noise. The probability distribution is given by

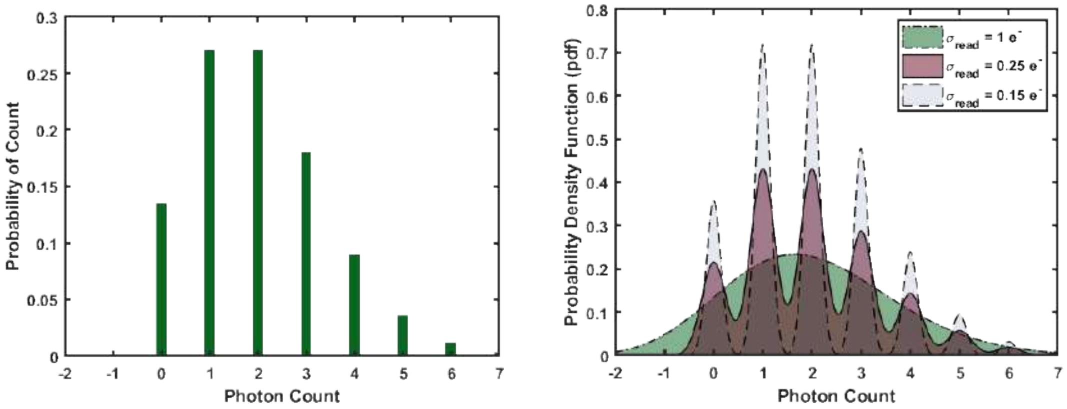
$$\mathbb{P}(\eta_{\text{read}} = n) = \frac{1}{\sqrt{2\pi\sigma^2_{\text{read}}}} \exp\left\{ -\frac{1}{2}\left(\frac{n}{\sigma_{\text{read}}}\right)^2 \right\}. \tag{3.49}$$

The analog signal that is read out of the sensor is then given by

$$y = G \cdot (K + \eta_{\text{read}}), \tag{3.50}$$

where G is the conversion gain of the sensor which maps the electron number to the voltage read out of the sensor. Y is referred to as a Poisson-Gaussian random variable. The probability distribution of Y is a convolution of the Poisson and the Gaussian distributions given by

$$\mathbb{P}(Y = y) = \sum_{\ell=0}^{\infty} \left(\frac{\beta^{\ell}_{\text{tot}}}{\ell!} \mathrm{e}^{-\beta_{\text{tot}}} \frac{1}{\sqrt{2\pi G^2 \sigma^2_{\text{read}}}} \exp\left\{ -\frac{1}{2}\left(\frac{y - G\ell}{G\sigma_{\text{read}}}\right)^2 \right\} \right). \tag{3.51}$$



(a) Without read noise

(b) With read noise

Figure 3.8. Read noise. We plot the probability distribution function of Y when $G = 1$ and $\beta_{\text{tot}} = 2$. On the left is the Poisson pmf. On the right is Poisson-Gaussian pdf for different σ_{read}. We can see that when $\sigma_{\text{read}} = 1.0$, the photon counts are not recognizable. At $\sigma_{\text{read}} = 0.25$, we can see the individual photons, but there is an overlap between neighboring photon counts. At $\sigma_{\text{read}} = 0.15$, there is almost no overlap between neighboring photon counts that we can get plot on the right by just rounding off the values to the nearest integer.

Figure 3.8 shows how the the probability distribution function of Y looks like without and with different read noise strengths. We use $\beta_{\text{tot}} = 2$. We can notice that with $\sigma_{\text{read}} = 1$, we cannot see the different peaks distinctly anymore. However at $\sigma_{\text{read}} = 0.25$, we can notice we can notice the peaks for each individual photons, albeit with some overlap. At $\sigma_{\text{read}} = 0.15$, there is almost no overlap between the photon counts. For any $\sigma_{\text{read}} < 0.15$, it is therefore considered as good as $\sigma_{\text{read}} = 0$.

In Figure 3.9, we validate the accuracy of the model in (3.51) using data from a QIS pixel. A total of 50,000 repeated measurements from a single pixel is used to construct a photon-counting histogram (PCH). Each measurement has an integration time of 50μs. The average photon count is 1.48 photons per pixel (ppp). The ADC uses a bit-depth of 14 bits. The least significant bit is 0.05e$^-$. Because the ADC uses 14 bits, the resulting histogram is close to a continuum.

To plot the theoretical model, we assume that the read noise level is 0.25e$^-$. The dark current is assumed to be 0.0068e$^-$ per second [46]. β_{tot} is chosen such that the mean squared error between the histogram and the theoretical curve is minimized. Since the integration

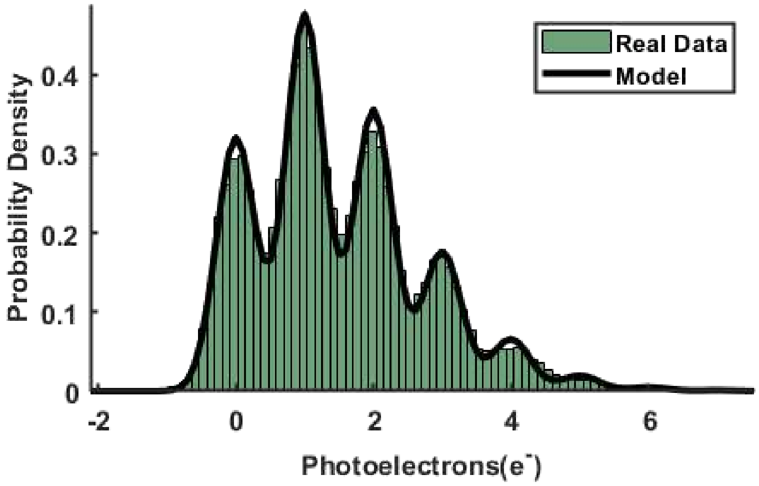


Figure 3.9. Validation of the Model. First reported in [41]. We compute the photon counting histogram of a real QIS sensor and compare it with our theoretical model. Note the similarity between the two.

time is only 50μs, we safely neglect the dark current. Putting these together, we obtain the black curve as shown in Figure 3.9. As we can see, the theoretical model fits the real data well.

In Figure 3.10, we visualize the effect of different strengths of read noise on photon-limited images using simulated data. We simulate three images with an average signal level of 0.5e$^-$ per pixel. We assume that the dark current is zero and $G = 1$. The first image has a read noise of $\sigma_{\text{read}} = 0$. The only source of noise is the shot noise. In the second image, we use $\sigma_{\text{read}} = 0.25$. This image does not look too different from the first image with zero read noise. However, in the third image, when we use $\sigma_{\text{read}} = 1.5$, the details of the image are almost completely destroyed.

Remark 3.4. *The gain factor G depends on the amplifier gain used in the image sensor. The read noise σ_{read} depends on the amplification used. Usually larger the gain G, the smaller the read noise in terms of electrons, but the higher the absolute noise witnessed in Y. When operating cameras at higher ISO, will the image will look noisier. Nevertheless, the signal-to-noise ratio with respect to the input signal is higher at high ISO than lower ISO.*

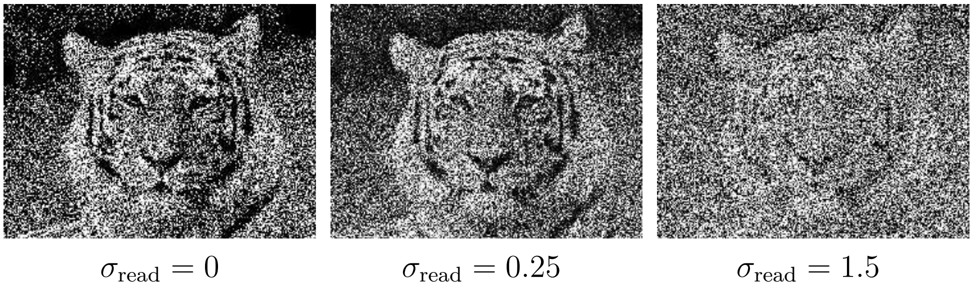

$\sigma_{\text{read}} = 0$
$\sigma_{\text{read}} = 0.25$
$\sigma_{\text{read}} = 1.5$

Figure 3.10. Effect of read noise in imaging. We simulate imaging at an average light level of 0.5 photoelectrons per pixel. Notice that a read noise of $\sigma_{\text{read}} = 0.25$ does not affect the visual quality that much, however $\sigma_{\text{read}} = 1.5$ almost completely destroys all the information in the image.

Remark 3.5. *An iPhone XS has read noise as low as* 1.7e^- *[59]. CIS-based quanta image sensors have read noise of* 0.19e^- *[35]. Single-Photon Avalanche Diodes have zero read noise.*

3.1.7 Fixed pattern noise

Until now, we have looked at temporal noise sources, i.e., they change over time, and the noise realization is different each time. There are other sources of noise that do not change over time. They are called fixed pattern noise (FPN). Two of the most common sources of FPN are 1. Photon Response Non-Uniformity (PRNU), and 2. Dark Signal Non-Uniformity (DSNU).

PRNU is the randomness in the photon detection efficiency $\bar{\eta}$ and the conversion gain G of different pixels. PRNU arises from the randomness in the fabrication process, which leads to small variations among the pixels of the same sensor. Therefore, each pixel will have a different mean voltage readout when illuminated by a uniform light source. PRNU is generally a function of the illumination and therefore is calculated at different light levels. PRNU becomes prominent at low light levels. For example, the range of PRNU can range from 0.1% at 15000e^- signal to 6% at 10e^- [61]. DSNU is a similar variation in dark current β_d. DSNU is independent of illumination, though they depend on integration time and the operating temperature. Figure 3.11 shows example histograms of the distribution of PRNU

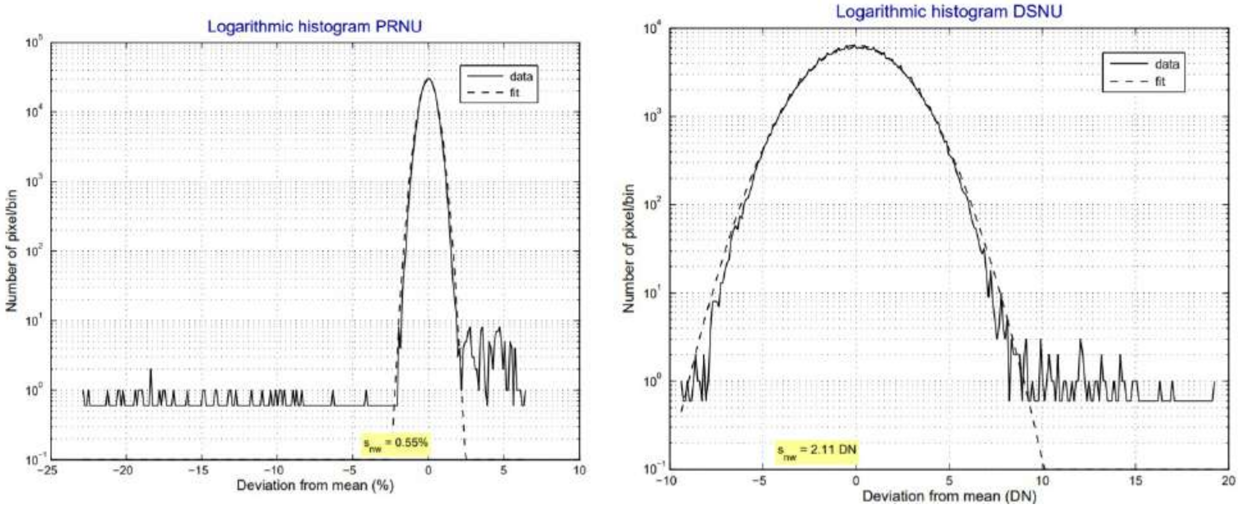
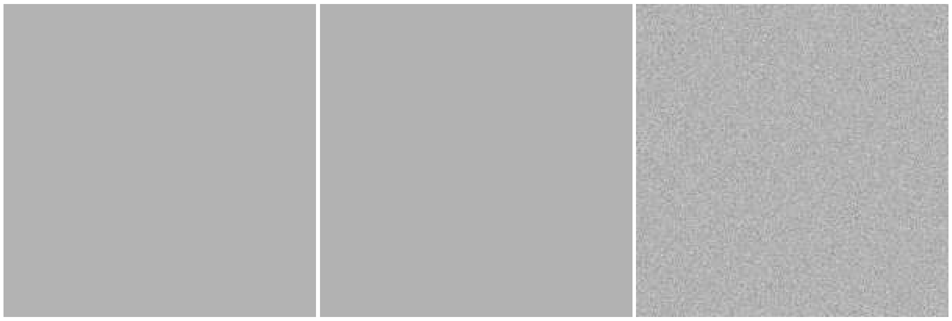


Figure 3.11. Example logarithmic histogram for PRNU and DSNU. s_{nw} is the standard deviation of the Gaussian distribution used to model the behavior. Source : [60]

and DSNU on an image sensor. The distribution of PRNU and DSNU are usually modeled as Gaussian distribution. Figure 3.12 shows an example of what the average of many captured images will look like at different levels of PRNU distribution.



Ideal 0.5% Non-uniformity 5% Non-uniformity

Figure 3.12. PRNU. This figure shows an example of what the average of many captured images will look like at two different levels of non-uniformity. At 0.5%, we almost cannot see the difference, but the difference becomes apparent at 5%.

CMOS image sensors historically have had more FPN than the preceding CCD image technology, with column FPN patterns looking like Figure 3.13. CMOS image sensors suffer because of pixel transistors and column or row amplifiers. Mathematically this could be modeled as varying gain G for every pixel. Over the years, the technology has become better, and innovations like correlated double sampling (CDS) [62] has helped in the reduction of FPN.

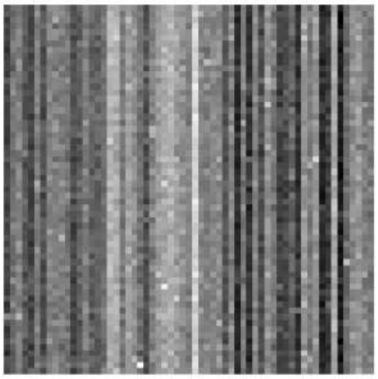


Figure 3.13. Column FPN in CMOS image sensors. Source: [63]

Remark 3.6. *PRNU is unique for each sensor. This fact can be leveraged to identify the source camera of every photo. This process of identifying the cameras based on their PRNU is called* ***camera fingerprinting*** *[64].*

Dead pixels

In image sensors, sometimes pixels stop responding, and we call these the dead pixels. They give zero or very large outputs, irrespective of the illumination. These pixels are in some sense special cases of PRNU, where the multiplying factor for these images is just zero or a huge constant.

3.1.8 Analog-to-digital conversion

We have looked at the analog signal/voltage that the image sensor generates. For storing and further processing image data, we want the signals to be converted to digital data. The analog signal thus is sent to an analog-to-digital converter (ADC), which gives rise to what is called the 'quantization noise.' This process is modeled as

$$Z = \mathrm{ADC}(Y+O) = \begin{cases} 0, & \lceil Y+O \rfloor \leq 0 \\ \lceil Y+O \rfloor, & 0 < \lceil Y+O \rfloor < L_{\mathrm{ADC}} \\ L_{\mathrm{ADC}}, & \lceil Y+O \rfloor \geq L_{\mathrm{ADC}} \end{cases}, \tag{3.52}$$

where $\lceil \cdot \rfloor$ is the function that rounds off the real-valued numbers to the nearest integer, O is an offset added to the signal often so that we can access the negative signal values, and L_{ADC} is the largest integer that the ADC can count. L depends on the bit-depth used by the ADC. For example, if we use a 10-bit ADC, $L = 2^{10} - 1 = 1023$.

The bit-depth of the image sensor affects the speed at which the camera can operate because the bit rate at which the data gets readout of the sensor acts as a bottleneck for the camera speed. Therefore, different cameras and imaging technologies use different bit-depths for their ADC. For example, a 'Canon T6i Rebel' has a 14 bit ADC. A mobile phone camera usually uses only 10 bits [49]. Quanta image sensors use a single bit, or very few bits (up to 3 bits) [65].

Full well capacity

Full well capacity is a pixel property and should have been discussed after learning about photoelectrons. However, the ADC bit-depth also plays a significant role in the full-well capacity of the image sensors in this dissertation. So, its discussion has been delayed till now.

Usually, a finite amount of electrons are available in a pixel to be excited, limiting the total amount of charge accumulated in the pixel. So, the accumulated charge will follow a thresholded Poisson instead of the traditional Poisson as mentioned in (3.47). The total charge accumulated cannot exceed the full well capacity. For a pixel with a full well capacity of L_{fw}, the total charge accumulated in the pixel will follow the distribution

$$\mathbb{P}(K=k) = \begin{cases} \frac{e^{-(\beta_e+\beta_d)}(\beta_e+\beta_d)^k}{k!}, & k < L_{\text{fw}} \\ \sum_{i=L_{\text{fw}}}^{\infty} \frac{e^{-(\beta_e+\beta_d)}(\beta_e+\beta_d)^i}{i!}, & k \geq L_{\text{fw}} \end{cases}. \tag{3.53}$$

The full well capacity L_{fw} depends on the size of the pixel. Larger pixels tend to have higher full well capacity and smaller pixels lower. Figure 3.14 shows the dependence using data from different cameras.

At higher ISO in cameras, usually, the full well capacity of the pixel itself does not affect the maximum number of photoelectrons that can be read. Rather the maximum digital number that the ADC can read out decides the full well capacity. Since the gain factor G is a large number at higher ISO, the maximum number of photoelectrons that can be read out in the digital data becomes limited. In such cases, the full well capacity ends up being L_{ADC}/G. Single-bit QIS often has a full-well capacity of just one electron.

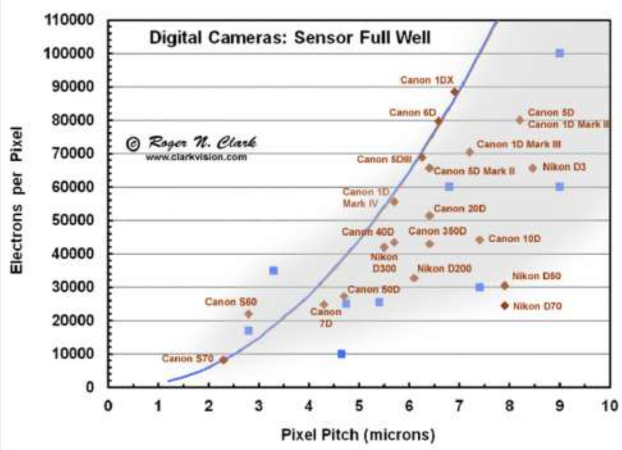


Figure 3.14. Full well capacity vs. pixel pitch. Source: [66]

Remark 3.7. *In some digital cameras, the cameras stop doing analog amplification after a certain ISO level and start doing digital amplification. Digital amplification does* *not provide any helpful gain that is not present before the amplification. So, 'beware' of these fake ISO gains.*

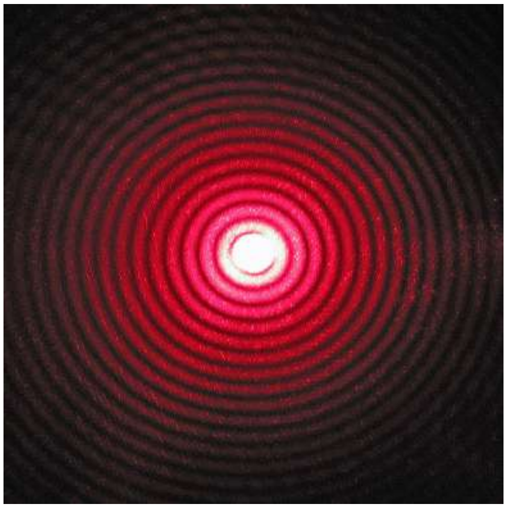


Figure 3.15. Diffraction pattern. Diffraction pattern when a red laser passes through a circular aperture. Source: [67]

3.1.9 Other sources of noise

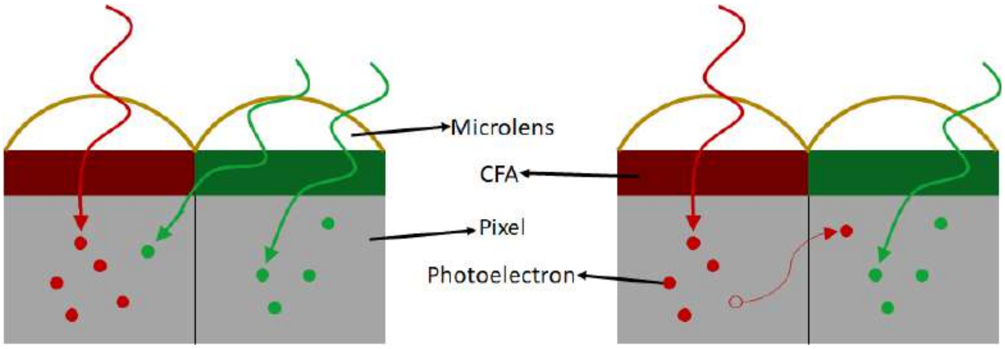
Other noise sources may be necessary for some specific applications, but we will not be used in this dissertation. We will, however, take a look at some of them here.

Diffraction limit

Diffraction is the physical phenomenon where the light bends when it passes through an aperture. Figure 3.15 shows an example diffraction pattern seen on a plate when red light from a laser passes through a small hole in another plate. When passing through a circular aperture, the diffraction pattern is also circular and therefore is called *Airy disks* (named after George Biddell Airy). The size of the Airy disk depends on the wavelength of the light and the size of the aperture. Smaller the aperture, the larger the size of the Airy disk. The diameter of the Airy disks is given by

$$x \approx 2.44\lambda\frac{f}{d}, \tag{3.54}$$

where $\frac{f}{d}$ is the f-number of the lens, and λ is the wavelength of the light. Take, for example, a lens with an f/8 setting, imaging a green light at $550nm$. At this light level, the diameter of the Airy disk is $\sim 10\mu m$. So, if the pixel pitch is smaller than $10\mu m$, there will be blurring due to diffraction in the captured image, which we may need to deal with in post-processing. However, suppose the pixel pitch is smaller than 25% of the Airy disk diameter. In that case, the two points become unresolvable, i.e., making the pixel pitch smaller than $2.5\mu m$ does not add any resolution gain. Note that we can make the Airy disk smaller by using a larger aperture. However, doing so will introduce an out-of-focus blur—the larger the aperture, the smaller the depth of field, and the larger the out-of-focus blur radius.



(a) Optical Crosstalk (b) Electrical Crosstalk

Figure 3.16. Types of crosstalk. Optical crosstalk occurs when a photon supposed to fall on a particular pixel ends up on a neighboring pixel. Electrical crosstalk occurs when the charge generated in a particular pixel diffuses to a neighboring pixel.

Crosstalk

Crosstalk refers to the process where a photon that is supposed to be incident on a particular pixel ends up generating a photoelectron in a neighboring pixel. There are two different processes by which crosstalk may occur [68]. 1. The photon incident on a particular pixel may end up on a neighboring pixel. This is called *optical* crosstalk. Recent innovations such as backside illumination sensors (BSI) [69], and microlens [70] have mitigated this type of crosstalk to a large extent. However, as the pixels become smaller than the diffraction limit, optical crosstalk becomes inevitable. 2. The charge generated in a pixel may diffuse to a neighboring pixel. This type of crosstalk is called *electrical* crosstalk that can be fixed only by careful hardware design of the image sensors. Figure 3.16 shows a visualization of the two types of crosstalk in an image sensor.

Crosstalk results in images captured losing the resolution because of the blur introduced, and it also reduces the color signals of each pixel and more considerable overlap between color channels. Crosstalk complicates the color reconstruction and makes the recovered image fade out in terms of color, which in turn needs to be dealt with during the image signal processing pipeline. Recent works such as [71], [72] are tackling this problem by designing color filter arrays that mitigate the effect of crosstalk in color imaging.

3.2 Simulating a camera

We have looked at the different noise sources in an image sensor and how to model them. Let us now put it all together and simulate a camera imaging model. Figure 3.17 shows a compact version of all the different sources of noise we have seen till now.

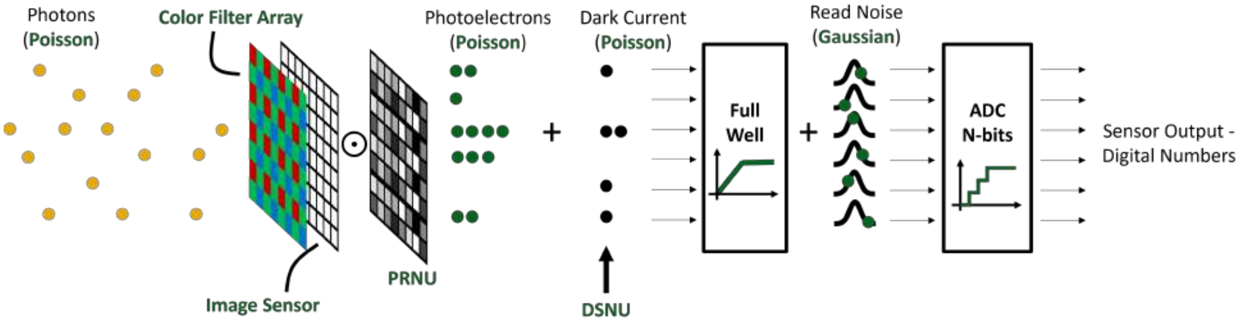


Figure 3.17. Imaging Model.

For a highly realistic simulation, we should start with the light source's spectrum and the reflectance of the object being imaged to predict the wavefront reaching the image sensor. However, this will require data that is extremely difficult to collect. So, we will start with any color image. We will assume that we use Bayer pattern CFA. We will also assume that each pixel value is proportional to the mean number of photoelectrons that is generated at the image sensor pixel. Note that we can play around with the proportionality constant to simulate the sensor's exposure time. Then we can follow the noise models we have developed to simulate the captured digital image.

The following MATLAB code snippets explain how the camera model is simulated.

```
function y = imaging_model(x, ppp, prnu, m_dark, dsnu, fwc, ...
                           gain, sigma_r, adc_offset, n_bits)
% Function simulating a camera
```

The function takes 10 inputs.

1. `x` - `x` is the input image. It could be a color image or Monochrome image. It is a 2D or 3D array.

2. `ppp` - `ppp` expands to photons per pixel. It is the average photons per pixel that we need to simulate the image for. It is a scalar.

3. `prnu` - PRNU map for the image. It is a 2D array. This can calibrated for a given camera or could be generated as a random variable using as shown in section 3.1.7.

4. `m_dark` - Average number of electrons generated due to dark current. It is a scalar. Note that this will change usually linearly with the choice of the integration time.

5. `dsnu` - DSNU map for the image. It is a 2D array. Again, this can be either calibrated or realized as a random variable.

6. `fwc` - Full well capacity of the image sensor. Scalar.

7. `gain` - The gain G of the image sensor. Scalar.

8. `sigma_r` - The read noise standard deviation. Scalar.

9. `adc_offset` - The offset O in Equation (3.52)

10. `n_bits` - Number of bits in the ADC.

We start by first converting the color images to Bayer pattern images, if the input is a color image. Here we assume a Bayer pattern CFA with ’RGGB’ format. For other CFAs, this code has to change correspondingly.

```
% If color image get the Bayer pattern image
if size(x,3) == 3
    x_Bayer = zeros(size(x,1), size(x,2));

    x_Bayer(1:2:end,1:2:end) = x(1:2:end,1:2:end,1);
    x_Bayer(1:2:end,2:2:end) = x(1:2:end,2:2:end,2);
    x_Bayer(2:2:end,1:2:end) = x(2:2:end,1:2:end,2);
    x_Bayer(2:2:end,2:2:end) = x(2:2:end,2:2:end,3);

    x = x_Bayer;
end
```

We then convert the given image to average number of photoelectrons per pixel, for this we first make the mean of the image is `ppp`. We then multiply the given image with `prnu` pointwise to simulate the effect of PRNU.

```
% Convert given image to avg. no. of photoelectrons
mean_photoelectrons = x / mean(x(:)) * ppp .* prnu;
```

Now, we will add the average electrons added due to dark current. We obtain this by multiplying the mean number of electrons from dark current with the DSNU.

```
% Add the Dark current to get avg. electrons generated for each
   pixel
mean_electrons = mean_photoelectrons + m_dark * dsnu;
```

The next step is to get a random realization of the total number of electrons generated. This is obtained by simulating a Poisson random variable.

```
% Simulate single realization of electrons generated.
electrons = poissrnd(mean_electrons);
```

However, sometimes simulating Poisson random variable is too costly and takes a lot of time, especially when ppp is too large. In such cases, we can use the Gaussian approximation to the Poisson random variable. We call this the affine approximation for Poisson distribution. If β is the mean number of electrons getting generated then the the number of electrons is given by

$$\begin{aligned} k &= \lceil \beta + \eta \rfloor_+ \\ \eta &\sim \mathcal{N}(0, \beta), \end{aligned} \tag{3.55}$$

where $\lceil \cdot \rfloor_+$ rounds off the real valued number to the nearest non-negative integer.

```
% Simulate single realization of electrons generated.
electrons = round(electrons + sqrt(electrons).* ...
                              randn(size(electrons)));
electrons(electrons<0) = 0;
```

We now have to enforce the number of electrons generated to the full well capacity. This is done by simply clamping the number of electrons generated to $[0, \texttt{fwc}]$.

```
% Constrain the number of electrons to full well capacity
electrons(electrons > fwc) = fwc;
```

We now have to convert the charges collected to voltage by adding the read noise and multiplying the conversion gain.

```
% Add read noise and multiply gain factor
analog_signal = gain * (electrons + sigma_r * randn( ...
                                    size(x,1), size(x,2)));
```

Now, all that is left to do is send in the analog signal we have generated through the ADC to get the digital image.

```
% Quantization using ADC
y = round(analog_signal + adc_offset);
```

```
y(y<0) = 0;
y(y>2^n_bits-1) = 2^n_bits - 1;
```

In the rest of this dissertation we will use some version of this simulation code for generating realistic images from a camera.

3.3 Modelling the performance of a camera

Let us switch gears a little bit here. Until now, we have looked at how we can model any camera. Now, let us take a look at how we can analyze the performance of a camera. For example, let us say we have two cameras with us. 1. A traditional CMOS image sensor with 14 bit ADC. 2. A Quanta Image Sensor with 1-bit ADC. There are, of course, more differences between the sensors, such as full-well capacity and frame rate. We have to decide which of these two sensors is better for our application. To be able to decide this, we need to quantify the performance of the two cameras. Signal-to-noise ratio (SNR) is a metric that we can use to quantify this performance. The rest of this chapter is about understanding SNR and deriving mathematical expressions that we can use to quantify the performance of a camera.

3.3.1 Signal-to-noise ratio

The signal-to-noise ratio (SNR) characterizes an imaging device's performance when acquiring, transmitting, and processing raw data in the presence of noise. In as early as 1949, when Claude Shannon derived the information capacity of a noisy Gaussian channel, the concept of SNR was already presented [73]. As the name suggests, the SNR is the ratio between the signal power and the noise power

$$\mathrm{SNR} = \frac{\text{signal power}}{\text{noise power}}, \tag{3.56}$$

which is sometimes expressed in the logarithmic scale via $10 \log_{10} \mathrm{SNR}$. In digital image sensors, since the measured pixel values are results of the analog-to-digital conversion of the voltage (instead of power), a more commonly seen definition is the ratio between the mean and the standard deviation of the raw measurement:

$$\mathrm{SNR}_{\mathrm{out}} = \frac{\mathrm{E}\left[Y\right]}{\sqrt{\mathrm{Var}\left[Y\right]}}, \tag{3.57}$$

where, in this equation, Y is a random variable denoting the measurement generated by the sensor, $\mathrm{E}\left[\,\cdot\,\right]$ denotes the statistical expectation, and $\mathrm{Var}\left[\,\cdot\,\right]$ denotes the statistical variance. The resulting SNR is known as the *output*-referred SNR because it measures directly what a sensor outputs.

The output-referred SNR is convenient to calculate. In the most straightforward setting where we have access to the sensor's analog data, and the sensor's output is mainly influenced by the photon shot noise and the electronic read noise, Y will follow the Poisson-Gaussian distribution [29], [74]

$$Y \sim \mathrm{Poisson}(\beta) + \mathrm{Gaussian}(0, \sigma_{\mathrm{read}}^2),$$

where β denotes the flux integrated over the surface area and the exposure time, and σ_{read} is the standard deviation of the read noise. Assuming that the full-well capacity is at the *infinity* so that the measurement Y will never saturate, the expectation of Y is $\mathrm{E}\left[Y\right] = \beta$ and the variance is $\mathrm{Var}\left[Y\right] = \beta + \sigma_{\mathrm{read}}^2$. Thus, $\mathrm{SNR}_{\mathrm{out}}$ can be computed via

$$\mathrm{SNR}_{\mathrm{out}}(\beta) = \frac{\beta}{\sqrt{\beta + \sigma_{\mathrm{read}}^2}}. \tag{3.58}$$

If the read noise is negligible such that $\sigma_{\mathrm{read}} = 0$, one can recover an even simpler expression $\mathrm{SNR}_{\mathrm{out}}(\beta) = \sqrt{\beta}$. This widely adopted equation says that as the scene becomes brighter, the gain in the signal will override the random fluctuation of the noise, and hence the SNR will increase.

Limitations of output-referred SNR

The problem arises when the full-well capacity of the sensor is *finite*. Certainly, the $\mathrm{SNR}_{\mathrm{out}}$ in Equation (3.57) is still valid when the exposure β is much smaller than the full-well capacity. However, if β reaches the full-well and goes beyond it, the mean $\mathrm{E}\left[Y\right]$ will stop growing with

β as illustrated in Figure 3.18. The variance $\text{Var}[Y]$ will gradually drop to zero because Y cannot go beyond the full-well capacity. As a result, SNR_{out} according to Equation (3.57) will eventually go to infinity (because $\text{Var}[Y] \to 0$.) However, realistically speaking, this cannot be true because the SNR beyond saturation must be poor.

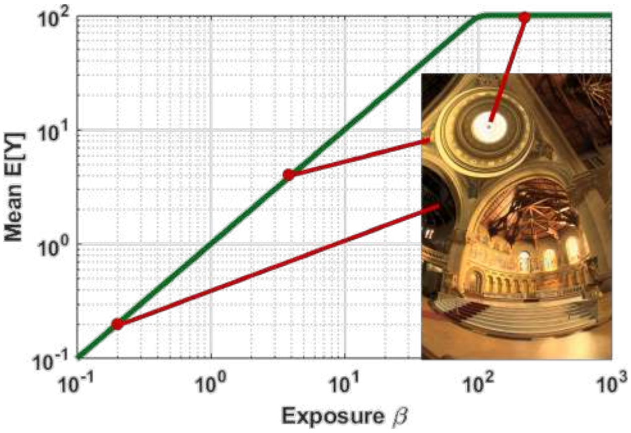


Figure 3.18. Expected value of the camera response. With a finite full-well capacity, the mean $\text{E}[Y]$ will stop growing when the exposure β exceeds the full-well capacity $L = 10^2$.

The SNR_{out} going to infinity is artificially caused by the inability of SNR_{out} to capture the behavior near and beyond the full-well capacity. The common wisdom here is then to create a special case by declaring a zero SNR_{out} [75]:

$$\text{SNR}_{\text{out}}(\beta) = \begin{cases} \frac{\beta}{\sqrt{\beta + \sigma^2_{\text{read}}}}, & \beta < L, \\ 0, & \beta \geq L, \end{cases} \tag{3.59}$$

where L denotes the full-well capacity. The definition in Equation (3.59) is adequate for image sensors with a sufficiently large full-well capacity. More importantly, it is convenient for signal processing. Before the saturation, the SNR grows linearly (in the log-log plot). After that, the SNR is zero[4].

[4] ↑A subtle point here is that when $\beta > L$, the *mean* $\text{E}[Y]$ is saturated, but an instantaneous observation Y may still be unsaturated. Thus, technically speaking, the SNR does not drop to zero instantly, but it will have a finite transient period. The exposure-referred SNR is derived to describe the transient mathematically.

However, the pixel pitch of image sensors over the past decade has shrunk significantly. The full-well capacity L is becoming smaller and smaller. For example, in a 1-bit quanta image sensor (QIS) with a threshold at q photons [33], [76], [77], the measurement Y is a binary random variable

$$Y = \begin{cases} 0, & X < q, \\ 1, & X \geq q. \end{cases} \tag{3.60}$$

This, in turn, is a special case of the more general ℓ-bit digital (CCD or CMOS) image sensors with a full-well capacity $L = 2^\ell - 1$ where the measurement follows the equation [78]

$$Y = \begin{cases} X, & X < L, \\ L, & X \geq L. \end{cases} \tag{3.61}$$

Here, $X \sim \text{Poisson}(\beta) + \text{Gaussian}(0, \sigma^2_{\text{read}})$ is the actual voltage measured before the analog-to-digital converter. For these small pixels, the nonlinearity is missing and hence the approximation in Equation (3.59) is invalid.

Exposure-referred SNR

When the full-well capacity L is small, little is known about how to derive the SNR because most models assume a large full-well capacity. In 2013, Fossum analyzed the single-bit, and multi-bit quanta image sensor [65]. In that paper, he argued that instead of using the output-referred SNR, one could consider the so-called *exposure*-referred SNR

$$\text{SNR}_{\text{exp}}(\beta) = \frac{\beta}{\sqrt{\text{Var}\left[Y\right]}} \cdot \frac{d\mu}{d\beta}, \tag{3.62}$$

where $\mu = \text{E}\left[Y\right]$.

Remark 3.8. *Tracing back the history, Mitsunaga and Nayar had the same equation (3.62) in 1999, although they did not give it a name [79]. In the sensor literature, El Gamal called the calibrated noise the "input-referred noise" in his Stanford EE 392B lecture note (2004-2015) [63].*

The intuition of the exposure-referred SNR was documented in a supplementary report of the paper by Elgendy, and Chan [80]. They argued that the derivative $d\beta/d\mu$ could be considered the "transfer function" of a black box system that takes the output μ and maps it back to the input β. Thus $d\beta/d\mu$ is the gain of such a transfer function that scales the noise from $\sqrt{\mathrm{Var}[Y]}$ to $\sqrt{\mathrm{Var}[Y]}\frac{d\beta}{d\mu}$. It is also explained that if β is beyond the full-well capacity L, the derivative $d\mu/d\beta$ will become zero because of any change in the exposure β will no longer affect μ. Hence, the issue of $\mathrm{SNR}_{\mathrm{out}}$ going to infinity is resolved because when a pixel saturates, $d\mu/d\beta = 0$ and so the SNR will go to zero.

The above intuition is certainly not rigorous. In this section, we will try to fill this theoretical gap by answering four questions:

(i) What is *the* correct way of defining the SNR and how to theoretically derive $\mathrm{SNR}_{\mathrm{exp}}(\beta)$ from the first principle?

(ii) What is the relationship between $\mathrm{SNR}_{\mathrm{out}}(\beta)$ and $\mathrm{SNR}_{\mathrm{exp}}(\beta)$? Under what condition would the former become a special case of the latter?

(iii) For complex noise models where closed-form expressions are unavailable, how can numerically predict the SNR via Monte-Carlo sampling techniques?

(iv) What utilities can $\mathrm{SNR}_{\mathrm{exp}}(\beta)$ offer to improve the sensor's imaging capabilities?

Tools in statistical estimation theory will be utilized to answer these questions.

3.3.2 Some mathematical tools

The purpose of this subsection is to elaborate on two sets of mathematical tools that will be useful later. For simplicity of the notations, most theoretical results will be derived for the Poisson random variable that accounts for the shot noise.

Truncated Poisson and the incomplete Gamma function

Let $X \sim \mathrm{Poisson}(\beta)$ be a Poisson random variable with a parameter β that represents the total exposure integrated over the sensor area and the exposure time. The Poisson random variable X is subject to a finite full-well capacity L (a positive integer), beyond which X

will stay at the saturation level. This leads to a truncated Poisson variable Y as defined in Equation (3.61). The probability mass function of Y is given by

$$p_Y(y) = \begin{cases} \frac{\beta^y}{y!}\mathrm{e}^{-\boldsymbol{\eta}}, & y < L, \\ \sum_{\ell=L}^{\infty} \frac{\beta^\ell}{\ell!}\mathrm{e}^{-\beta}, & y = L. \end{cases} \tag{3.63}$$

By construction, the random variable Y will never take a value greater than L. The probability that $Y = L$ is given by the sum of the Poisson tail, which can be conveniently expressed via the incomplete Gamma function as shown in Figure 3.19.

Definition 3.3.1 (Incomplete Gamma function)**.** *The upper incomplete Gamma function is defined as* $\Psi_L : \mathbf{R}_+ \to [0,1]$, *with*

$$\Psi_L(\beta) = \frac{1}{\Gamma(L)}\int_\beta^\infty t^{L-1}\mathrm{e}^{-t}dt = \sum_{\ell=0}^{L-1}\frac{\beta^\ell \mathrm{e}^{-\beta}}{\ell!}, \tag{3.64}$$

for $\beta > 0, L \in \mathbf{N}$ *where* $\Gamma(L) = (L-1)!$ *is the standard Gamma function.*

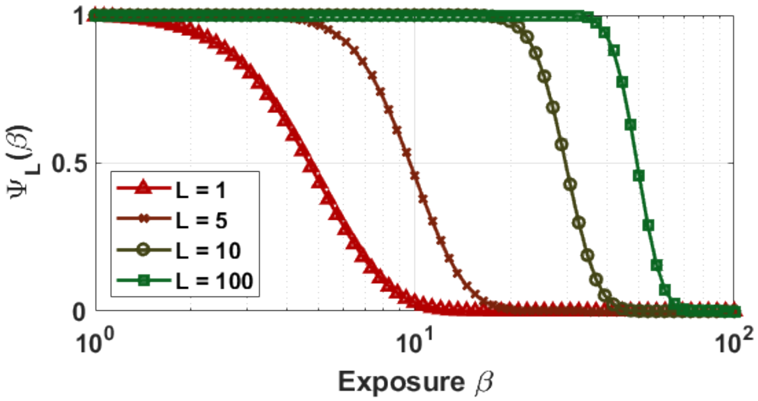


Figure 3.19. Incomplete Gamma function $\Psi_L(\beta)$ as a function of β.

The first-order derivative of $\Psi_L(\beta)$ is [81]

$$\Psi_L'(\beta) = -\frac{\beta^{L-1}\mathrm{e}^{-\beta}}{(L-1)!} < 0, \quad \text{for all } \beta, \tag{3.65}$$

which means that $\Psi_L(\beta)$ is a strictly decreasing function in β. The steepest slope can be determined by analyzing the curvature

$$\Psi_L''(\beta) = -(L-1)\beta^{L-2}\mathrm{e}^{-\beta} + \mathrm{e}^{-\beta}\beta^{L-1}.$$

Equating this to zero will yield $\beta^* = L-1$. At this critical point and assume $L \gg 1$, Stirling's formula implies that[5]

$$\Psi_L'(\beta^*) \approx -\frac{1}{\sqrt{2\pi\beta^*}} \exp\left\{-\frac{(\beta^* - (L-1))^2}{2\beta^*}\right\}. \tag{3.66}$$

Therefore, $\Psi_L'(\beta^*) = -\frac{1}{\sqrt{2\pi(L-1)}}$. Hence, the slope of the incomplete Gamma function reduces as L increases.

Remark 3.9. *Most papers in the image sensor literature plot curves with respect to* $\log\beta$ *instead of* β*, like the one shown in Figure 3.19. The* x*-axis compression caused by* $\log\beta$ *will make the curves to* ***appear*** *steeper during the transient. This is expected because for any function* $f(\beta)$*, the slope in the* $\log\beta$ *space is determined by* $\frac{d}{d\log\beta}f(\beta) = \beta f'(\beta)$*. So for large* β*, the slope will appear steeper.*

Delta method

The second mathematical tool is a special case of the *Delta Method* in statistics. It approximates the variance when a random variable undergoes a nonlinear transformation.

Lemma 3.1 (Delta Method)**.** *Let* X *be a random variable with mean* $\mathrm{E}[X] = \mu$ *and let* f *be a continuously differentiable function within a small neighborhood of* μ*. Then*

$$\mathrm{E}\left[(f(X) - f(\mu))^2\right] \approx [f'(\mu)]^2\,\mathrm{Var}[X]. \tag{3.67}$$

Proof. Consider the Taylor expansion

$$f(X) \approx f(\mu) + f'(\mu)(X-\mu).$$

[5] ↑The proof of this result is given in the appendix A

Taking the expectation of the $(f(X) - f(\mu))^2$ will yield

$$\begin{aligned} \mathrm{E}\left[(f(X) - f(\mu))^2\right] &= \mathrm{E}\left[f'(\mu)^2 (X - \mu)^2\right] \\ &= [f'(\mu)]^2 \,\mathrm{Var}\,[X], \end{aligned}$$

thus proving the result. □

The validity of the approximation depends on the second-order term, which is assumed to be small when the random variable X is sufficiently close to μ. One way for this to hold is that the random variable X is the sample average of N independent random variables such that $X = (1/N) \sum_{n=1}^{N} Y_n$ where $\mathrm{E}\left[Y_n\right] = \mu$ for all n. For large enough N, X will concentrate around μ, so the Delta Method is valid.

3.3.3 SNR: A statistical definition

Mean invariance property

When defining the SNR, it is important to clarify the signal formation process. In most of the imaging problems, the underlying signal is the scene exposure β. This is the *signal*. The observations are random samples drawn from a certain distribution $p_Y(y; \beta)$ parameterized by β. For example, if $Y \sim \mathrm{Poisson}(\beta)$ then $p_Y(y; \beta) = \beta^y \mathrm{e}^{-\beta}/y!$ is the distribution.

Reconstruction of the signal β from Y is often based an *estimator* $\widehat{\beta}(\cdot)$. An estimator can be *any* mapping that maps Y to an estimate $\widehat{\beta}(Y)$. The estimator $\widehat{\beta}(\cdot)$ can be the maximum-likelihood estimator, the maximum-a-posteriori estimator, or other mappings as long as they make sense. In the context of SNR, a technical requirement of the estimator is that it has to satisfy the mean invariance property.

Definition 3.3.2 (Mean invariance property)**.** *Let Y be a random variable drawn from a distribution $p_Y(y; \beta)$, with a mean $\mathrm{E}\left[Y\right] = \mu(\beta)$ for some function $\mu(\cdot)$. An estimator $\widehat{\beta}(\cdot)$ of β is said to satisfy the mean invariance property if*

$$\widehat{\beta}(\mu(\beta)) = \beta. \tag{3.68}$$

That is, the estimator evaluated at the sample mean $\mathrm{E}[Y]$ *will return the true estimate* β*.*

One should not confuse the mean invariance property and the unbiasedness of an estimator. An estimator $\widehat{\beta}(Y)$ is unbiased if $\mathrm{E}[\widehat{\beta}(Y)] = \beta$. This is different from mean invariance which requires $\widehat{\beta}(\mathrm{E}[Y]) = \beta$. An estimator satisfying the latter does not necessarily satisfy the former, and vice versa. One of the exceptions is that $\widehat{\beta}(\cdot)$ is linear.

At first glance, it would seem that the mean invariance property is constructed for some technical convenience. However, a careful examination would confirm that the property holds for many estimators. The two examples below shall illustrate this point.

Example 3.1. *Let* $Y \sim$ *Gaussian*(β, σ^2_{read}) *where* β *is the unknown parameter. It can be shown that the maximum-likelihood estimator is*

$$\widehat{\beta}(Y) = \underset{\beta}{argmax}\ \frac{1}{\sqrt{2\pi\sigma^2_{read}}} \exp\left\{-\frac{(Y-\beta)^2}{2\sigma^2_{read}}\right\} = Y.$$

Notice that since Y *is Gaussian, the mean is* $\mu = \mathrm{E}[Y] = \beta$*. Therefore, the mean invariance property holds:*

$$\widehat{\beta}(\mu) \overset{(a)}{=} \mu = \beta,$$

where (a) *is due to the fact that* $\widehat{\beta}(Y) = Y$*.* □

Example 3.2. *Let* Y *be the 1-bit quanta image sensor with a threshold* $q = 1$ *in* (3.60)*. In this case, the distribution of* Y *is a Bernoulli such that* $Y \sim$ *Bernoulli*$(1 - \mathrm{e}^{-\beta})$*. Then the maximum-likelihood estimator is*[a]

$$\widehat{\beta}(Y) = \underset{\beta}{argmax}\ (1-\mathrm{e}^{-\beta})^Y(\mathrm{e}^{-\beta})^{1-Y} = -\log(1-Y).$$

Since the mean of Y is $\mu = \mathrm{E}[Y] = 1 - \mathrm{e}^{-\beta}$, it follows that

$$\widehat{\beta}(\mu) = -\log(1-\mu) = -\log(1-(1-\mathrm{e}^{-\beta})) = \beta.$$

Again, the mean invariance property is satisfied. □

[a] ↑In practice, Y is often the sample average of N independent measurements so that $Y = (1/N)\sum_{n=1}^{N} Y_n$. For sufficiently large N, the chance that $Y = 1$ is negligible and hence this will prevent $\log(1-Y)$ from going to negative infinity.

If it is not difficult for estimators to satisfy the mean invariance property, why bother introducing this concept? From a practical point of view, the distribution of the measurement Y may come with a complicated expression. Thus, constructing an estimator $\widehat{\beta}(Y)$ to maximize the likelihood is not always easy. On the other hand, determining the mean $\mathrm{E}[Y]$ is much easier. Even if one cannot analytically derive an expression for $\mathrm{E}[Y]$, a Monte Carlo sampling would be sufficient to numerically generate it. Once the mean $\mathrm{E}[Y]$ is determined, the estimator $\widehat{\beta}(Y)$ can be defined by citing the mean invariance property: Let $\mu(\beta) = \mathrm{E}[Y]$, then the property will give

$$\widehat{\beta} = \mu^{-1}. \tag{3.69}$$

Going back to Examples 3.1 and 3.2, the above argument provides a procedure to *construct* an estimator that would satisfy the mean invariance principle.

Follow up of Example 3.1. Let $Y \sim \text{Gaussian}(\beta, \sigma^2_{\text{read}})$ where β is the unknown parameter. Since the mean is $\mathrm{E}[Y] = \beta$, it follows that $\mu(\beta) = \beta$. This μ is the identity mapping, and so the inverse mapping is $\mu^{-1}(s) = s$. Thus, one can define an estimator as $\widehat{\beta}(Y) = \mu^{-1}(Y) = Y$, and it is the same result as Example 1. Moreover, $\widehat{\beta}(Y)$ satisfies the mean invariance property because $\widehat{\beta}$ is constructed in that way. □

Follow up of Example 3.2. Let Y be the 1-bit quanta image sensor with a distribution $Y \sim \text{Bernoulli}(1 - \mathrm{e}^{-\beta})$. The mean is $\mu(\beta) = 1 - \mathrm{e}^{-\beta}$, and so the inverse is $\mu^{-1}(s) =$

$-\log(1-s)$. Therefore, one can define the estimator as $\widehat{\beta}(Y) = -\log(1-Y)$, and this $\widehat{\beta}(Y)$ satisfies the mean invariance property. The result is identical to Example 2. □

The advantage of the mean invariance property is that it bypasses the complication of solving an optimization (as in maximum likelihood). On the other hand, because of how $\widehat{\beta}$ is constructed, it is guaranteed to satisfy the mean invariance property.

Based on the analysis so far, it would seem natural to conjecture that any maximum-likelihood estimator would satisfy the mean invariance property. That is, $\widehat{\beta}_{\mathrm{ML}}(\mathrm{E}[Y]) = \beta$ for all reasonably smooth likelihoods. Proving (or giving conditions for) the conjecture would be valuable. A completely arbitrary estimator $\widehat{\beta}(\cdot)$ does not work for SNR. For example, $\widehat{\beta}(Y) = 0$ for all Y is an estimator but it is useless. Therefore, the mean invariance property can be considered as a sufficient condition to guarantee a meaningful SNR. However, whether it is a necessary condition is another open question that would be valuable to explore.

Defining the SNR

After clarifying the assumption of the estimator, it is necessary to discuss the *noise* in SNR. First, notice that the estimator $\widehat{\beta}(Y)$ is random because it is a function of Y. Thus $\widehat{\beta}(Y)$ fluctuates relative to the true deterministic parameter β. The randomness defines the noise, which is technically the mean squared error:

$$\text{noise} = \mathrm{E}\,[(\widehat{\beta}(Y) - \beta)^2]. \tag{3.70}$$

The SNR is then defined as follows.

Definition 3.3.3 (SNR, formal definition)**.** *Let Y be a random variable with a distribution $p_Y(y;\beta)$ where β is the underlying parameter to be estimated. Construct an estimator $\widehat{\beta}(Y)$ which satisfies the mean invariance property. Then the signal-to-noise ratio (SNR) is defined as*

$$SNR(\beta) \overset{\text{def}}{=} \frac{\beta}{\sqrt{\mathrm{E}\,[(\widehat{\beta}(Y) - \beta)^2]}}. \tag{3.71}$$

To convince readers that the formal definition of the SNR is valid, consider the two examples below.

Example 3.3 (Poisson)**.** *Let $Y \sim$ Poisson(β), and consider the maximum-likelihood estimator $\widehat{\beta}(Y) = Y$. It is relatively straightforward to show that $\mathrm{E}[Y] = \beta$ and that $\widehat{\beta}(\mathrm{E}[Y]) = \mathrm{E}[Y] = \beta$; so the estimator satisfies the mean invariance property. Since the estimator is $\widehat{\beta}(Y) = Y$, it follows that*

$$\mathrm{E}[(\widehat{\beta}(Y) - \beta)^2] = \mathrm{E}[(Y - \beta)^2] = \beta,$$

where the second equality holds because the variance of a Poisson is β. Therefore, SNR$(\beta) = \sqrt{\beta}$, which is consistent with Equation (3.58) when $\sigma_{read} = 0$. □

Example 3.4 (Poisson + Gaussian)**.** *Let $Y \sim$ Poisson(β) + Gaussian$(0, \sigma_{read}^2)$. Then the maximum-likelihood estimator is $\widehat{\beta}(Y) = Y$. The mean invariance property holds because $\mathrm{E}[Y] = \beta$ and $\widehat{\beta}(\mathrm{E}[Y]) = \mathrm{E}[Y] = \beta$. It then follows that*

$$\mathrm{E}[(\widehat{\beta}(Y) - \beta)^2] = \mathrm{E}[(Y - \beta)^2] = \beta + \sigma_{read}^2,$$

where the second equality holds because the variance of a Poisson-Gaussian is the sum of the two variances. Therefore, SNR$(\beta) = \beta/\sqrt{\beta + \sigma_{read}^2}$. This result is consistent with Equation (3.58) for a general σ_{read}. □

Exposure-referred SNR

A natural question to ask now is how does SNR(β) compare to SNR$_{\text{exp}}(\beta)$ and SNR$_{\text{out}}(\beta)$. It turns out that the SNR(β) is the same as SNR$_{\text{exp}}(\beta)$ up to the approximation inherent from the Delta Method. With that, one can explain where the "magical" derivative $d\mu/d\beta$ in Equation (3.62) comes from.

Theorem 3.3.1. *Let Y be a random variable with a probability density function $p_Y(y;\beta)$. Let $\mathrm{E}[Y]=\mu$, and let $\widehat{\beta}(Y)$ be an estimator such that $\widehat{\beta}(\mu)=\beta$. Then the SNR defined in Equation (3.71) is related to the SNR$_{out}$ and SNR$_{exp}$ as follows:*

$$SNR(\beta) \approx SNR_{exp}(\beta) = SNR_{out}(\beta)\cdot\frac{\beta}{\mu}\cdot\frac{d\mu}{d\beta}. \tag{3.72}$$

Proof. By the delta method, the mean squared error can be approximated by

$$\begin{aligned}\mathrm{E}\left[\left(\widehat{\beta}(Y)-\beta\right)^2\right] &= \mathrm{E}\left[\left(\widehat{\beta}(Y)-\widehat{\beta}(\mu)\right)^2\right]\\ &\approx \left[\widehat{\beta}'(\mu)\right]^2 \mathrm{Var}[Y].\end{aligned}$$

Since $\widehat{\beta}(\mu)=\beta$, it follows that $\frac{d\widehat{\beta}(\mu)}{d\mu}=\frac{d\beta}{d\mu}$. So,

$$\mathrm{E}\left[\left(\widehat{\beta}(Y)-\beta\right)^2\right] = \left[\frac{d\beta}{d\mu}\right]^2 \mathrm{Var}[Y].$$

Using the fact that $\frac{d\beta}{d\mu}=1/\frac{d\mu}{d\beta}$, the SNR can be written as

$$\mathrm{SNR}(\beta) = \frac{\beta}{\sqrt{\mathrm{E}\left[\left(\widehat{\beta}(Y)-\beta\right)^2\right]}} = \underbrace{\frac{\beta}{\sqrt{\mathrm{Var}[Y]}}\cdot\frac{d\mu}{d\beta}}_{\mathrm{SNR}_{\mathrm{exp}}(\beta)}.$$

To show the second relationship, notice that

$$\begin{aligned}\frac{\beta}{\sqrt{\mathrm{Var}[Y]}}\cdot\frac{d\mu}{d\beta} &= \frac{\mu}{\sqrt{\mathrm{Var}[Y]}}\cdot\frac{\beta}{\mu}\cdot\frac{d\mu}{d\beta}\\ &= \underbrace{\frac{\mathrm{E}[Y]}{\sqrt{\mathrm{Var}[Y]}}}_{=\mathrm{SNR}_{\mathrm{out}}(\beta)}\cdot\frac{\beta}{\mu}\cdot\frac{d\mu}{d\beta}.\end{aligned}$$

This completes the proof. □

As one can see from the proof of the theorem, what makes SNR$_{\mathrm{exp}}(\beta)$ and SNR$_{\mathrm{out}}(\beta)$ different is the derivative term $d\mu/d\beta$. The derivative changes the *output*-referred noise

$\sigma_{\text{out}} \overset{\text{def}}{=} \sqrt{\text{Var}\,[Y]}$ to the *exposure*-referred noise, as noted by Elgendy and Chan [80] via the "transfer function" perspective:

$$\sigma_{\text{exp}} \overset{\text{def}}{=} \sqrt{\text{Var}\,[Y]} \cdot \frac{d\beta}{d\mu}. \tag{3.73}$$

Thus, while $\text{SNR}_{\text{out}}(\beta) = \mu/\sigma_{\text{out}}$, the exposure-referred SNR is $\text{SNR}_{\text{exp}}(\beta) = \beta/\sigma_{\text{exp}}$. For saturated pixels, σ_{exp} will have a large value.

Illustrating the SNR via 1-bit QIS

To elaborate on the difference between $\text{SNR}_{\text{exp}}(\beta)$ and $\text{SNR}_{\text{out}}(\beta)$, it would be instructive to consider the statistics of a 1-bit quanta image sensor. Let $X \sim \text{Poisson}(\beta)$ and let Y be a random variable following Equation (3.60).

First, consider the case where $q = 1$. Since $Y \sim \text{Bernoulli}(1 - \mathrm{e}^{-\beta})$, the mean is $\mathrm{E}\,[Y] = 1 - \mathrm{e}^{-\beta}$. Define the mean as $\mu = \mathrm{E}\,[Y] = 1 - \mathrm{e}^{-\beta}$. As shown in Example 3.1, the maximum-likelihood estimate of β is $\widehat{\beta}(Y) = -\log(1-Y)$ and it satisfies the mean invariance property. The derivative $d\mu/d\beta$ is

$$\frac{d\mu}{d\beta} = \frac{d}{d\beta}\left[1 - \mathrm{e}^{-\beta}\right] = \mathrm{e}^{-\beta}.$$

Substituting into Theorem 3.3.1, it can be shown that

$$\text{SNR}_{\text{exp}}(\beta) = \frac{\beta}{\sqrt{\text{Var}\,[Y]}} \cdot \frac{d\mu}{d\beta} = \beta \cdot \sqrt{\frac{\mathrm{e}^{-\beta}}{1 - \mathrm{e}^{-\beta}}}.$$

For cases where $q > 1$, one can utilize the incomplete Gamma function for the truncated Poisson random variable such that

$$p_Y(y;\beta) = \begin{cases} 1 - \Psi_q(\beta), & y = 1, \\ \Psi_q(\beta), & y = 0. \end{cases}$$

It then follows that $\mathrm{E}\,[Y] = 1 - \Psi_q(\beta)$ and the estimator can be chosen such that $\widehat{\beta}(Y) = -\Psi_q^{-1}(1-Y)$. The mean invariance property is therefore validated. The derivative $d\mu/d\beta$ is

$$\frac{d\mu}{d\beta} = \frac{d}{d\beta}(1 - \Psi_q(\beta)) = \frac{\beta^{q-1}\mathrm{e}^{-\beta}}{(q-1)!}.$$

Hence, the SNR is

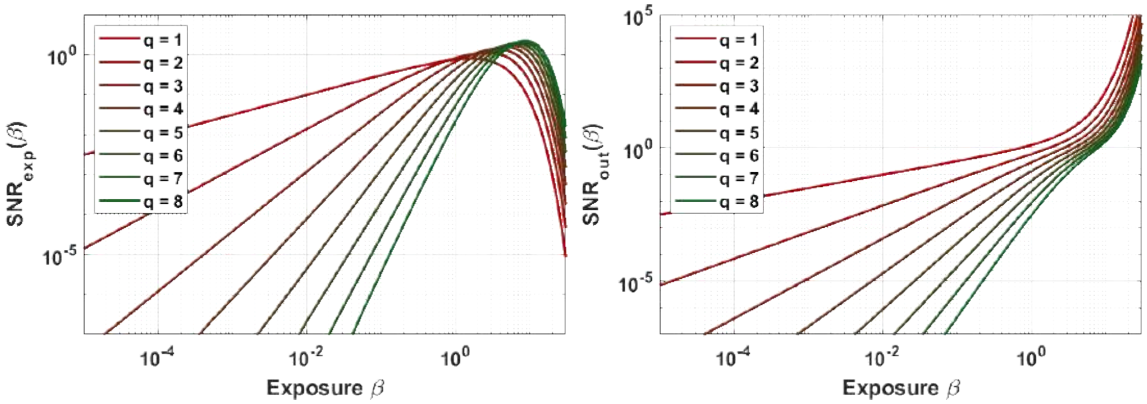
$$\mathrm{SNR}_{\mathrm{exp}}(\beta) = \frac{\beta}{\sqrt{\Psi_q(\beta)(1-\Psi_q(\beta))}} \cdot \frac{\beta^{q-1}\mathrm{e}^{-\beta}}{(q-1)!}, \tag{3.74}$$

of which the visualization is shown in Figure 3.20 (a).

Unlike $\mathrm{SNR}_{\mathrm{exp}}(\beta)$, the output-referred SNR goes to infinity when β grows. For the same 1-bit statistics, the output-referred SNR is simply the ratio between $\mathrm{E}[Y]$ and $\mathrm{Var}[Y]$, which is

$$\mathrm{SNR}_{\mathrm{out}}(\beta) = \frac{\mathrm{E}[Y]}{\sqrt{\mathrm{Var}[Y]}} = \sqrt{\frac{1-\Psi_q(\beta)}{\Psi_q(\beta)}}. \tag{3.75}$$

As shown in Figure 3.20 (b), $\mathrm{SNR}_{\mathrm{out}}(\beta)$ grows indefinitely as β grows which is known to be false because when β grows beyond the threshold q, the measurement Y will stay at $Y = 1$ more likely. The signal degrades and hence eventually the SNR should drop to zero.



(a) $\mathrm{SNR}_{\mathrm{exp}}(\beta)$ (b) $\mathrm{SNR}_{\mathrm{out}}(\beta)$

Figure 3.20. Exposure-referred and Output-referred SNR for a 1-bit quanta image sensor. Y is defined in Equation (3.60). As β goes beyond the threshold q, $\mathrm{SNR}_{\mathrm{exp}}(\beta)$ starts to drop as it should be. However, $\mathrm{SNR}_{\mathrm{out}}(\beta)$ continues to grow because of the inability of $\mathrm{SNR}_{\mathrm{out}}(\beta)$ to handle pixel saturation.

3.3.4 $\mathbf{SNR_{exp}}(\beta)$ for finite full well capacity

Let us now look at how we can get the SNR for image sensors with finite full well capacity. Of particular interest are image sensors with low bit-depth ADC, which we will discuss in the rest of this dissertation.

$\mathbf{SNR_{exp}}(\beta)$ for truncated Poisson

We will start the derivation for a truncated Poisson distribution defined in Equation (3.61). Extension to the more complex noise model will be analyzed later.

Theorem 3.3.2 ($\mathrm{SNR}_{\mathrm{exp}}(\beta)$ for truncated Poisson)**.** *Consider the truncated Poisson statistics defined in Equation (3.61). Let $\widehat{\beta}(\cdot)$ be an estimator satisfying the mean invariance property, i.e., $\widehat{\beta}(\mathrm{E}[Y]) = \beta$. Then the exposure-referred SNR is*

$$SNR_{exp}(\beta) = \frac{\beta}{\sqrt{\mathrm{Var}[Y]}} \cdot \frac{d\mu}{d\beta}, \tag{3.76}$$

where

$$\begin{aligned}
\mathrm{E}[Y] &= \beta\Psi_{L-1}(\beta) + L(1-\Psi_L(\beta)) \overset{\text{def}}{=} \mu, \\
\mathrm{Var}[Y] &= \beta^2\Psi_{L-2}(\beta) + \beta\Psi_{L-1}(\beta) + L^2(1-\Psi_L(\beta)) - \mu^2, \\
\frac{d\mu}{d\beta} &= \beta\Psi'_{L-1}(\beta) + \Psi_{L-1}(\beta) - L\Psi'_L(\beta).
\end{aligned} \tag{3.77}$$

Proof. Recall the probability density function of Y:

$$p_Y(y) = \begin{cases} \frac{\beta^y}{y!}\mathrm{e}^{-\beta}, & y < L, \\ \sum_{k=L}^{\infty} \frac{\beta^k}{k!}\mathrm{e}^{-\beta} = 1 - \Psi_L(\beta), & y \geq L, \end{cases}$$

where $\Psi_L(\beta)$ is the incomplete Gamma function. The mean of Y can be shown as

$$\begin{aligned}
\mu = \mathrm{E}\left[Y\right] &= \sum_{k=0}^{L-1} k \cdot \frac{\beta^k}{k!}\mathrm{e}^{-\beta} + L \cdot \left(\sum_{k=L}^{\infty} \frac{\beta^k}{k!}\mathrm{e}^{-\beta}\right) \\
&= \sum_{k=1}^{L-1} \frac{\beta^k}{(k-1)!}\mathrm{e}^{-\beta} + L \cdot (1 - \Psi_L(\beta)) \\
&= \beta \sum_{k=0}^{L-2} \frac{\beta^{k-1}}{(k)!}\mathrm{e}^{-\beta} + L \cdot (1 - \Psi_L(\beta)) \\
&= \beta\Psi_{L-1}(\beta) + L \cdot (1 - \Psi_L(\beta)).
\end{aligned}$$

The derivative $d\mu/d\beta$ is therefore

$$\begin{aligned}
\frac{d\mu}{d\beta} &= \frac{d}{d\beta}\left\{\beta\Psi_{L-1}(\beta) + L \cdot (1 - \Psi_L(\beta))\right\} \\
&= \beta\Psi'_{L-1}(\beta) + \Psi_{L-1}(\beta) - L \cdot \Psi'_L(\beta).
\end{aligned}$$

For the variance, since $\mathrm{Var}\left[Y\right] = \mathrm{E}\left[Y^2\right] - \mu^2$, it remains to determine $\mathrm{E}\left[Y^2\right]$.

$$\begin{aligned}
\mathrm{E}\left[Y^2\right] &= \sum_{k=0}^{L-1} k^2 \cdot \frac{\beta^k}{k!}\mathrm{e}^{-\beta} + L^2 \cdot \left(\sum_{k=L}^{\infty} \frac{\beta^k}{k!}\mathrm{e}^{-\beta}\right) \\
&= \sum_{k=1}^{L-1} k\frac{\beta^k}{(k-1)!}\mathrm{e}^{-\beta} + L^2 \cdot (1 - \Psi_L(\beta)) \\
&= \sum_{k=1}^{L-1} (k-1+1)\frac{\beta^k}{(k-1)!}\mathrm{e}^{-\beta} + L^2 \cdot (1 - \Psi_L(\beta)) \\
&= \sum_{k=2}^{L-1} \frac{\beta^k}{(k-2)!}\mathrm{e}^{-\beta} + \sum_{k=1}^{L-1} \frac{\beta^k}{(k-1)!}\mathrm{e}^{-\beta} \\
&\qquad\qquad + L^2 \cdot (1 - \Psi_L(\beta)) \\
&= \beta^2\Psi_{L-2}(\beta) + \beta\Psi_{L-1}(\beta) + L^2(1 - \Psi_L(\beta)).
\end{aligned}$$

This completes the proof. □

To illustrate the predicted $\mathrm{SNR}_{\mathrm{exp}}(\beta)$ as a function of β, Figure 3.21 shows several curves evaluated at different full-well capacity L. As is consistent with the 1-bit QIS example, the exposure-referred SNR for a truncated Poisson random variable also demonstrates a drop

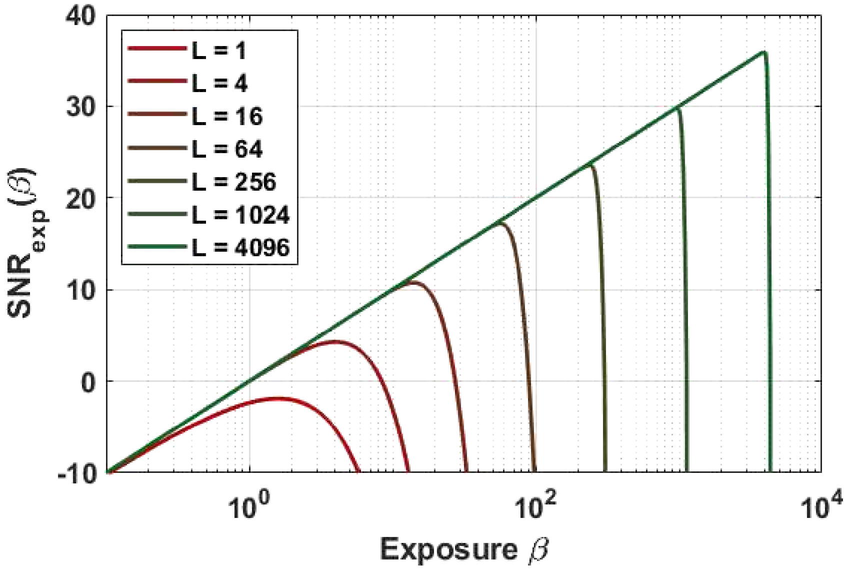


Figure 3.21. Exposure-referred SNR for a digital image sensor with a full-well capacity of L electrons.

in $\text{SNR}_{\text{exp}}(\beta)$ after the pixel saturates. What is more interesting is that as L increases, $\text{SNR}_{\text{exp}}(\beta)$ becomes a straight line in the log-log plot with a sharp decay after saturation. This is reminiscent to the heuristic definition of the $\text{SNR}_{\text{out}}(\beta)$ in Equation (3.59). However, for small L, the smooth transition is something that was not predicted by Equation (3.59).

The rapid drop after the saturation is attributed to two reasons. First, the log-log plot compresses the x-axis so that the slope is amplified with β. If one plots the x-axis in the linear scale (instead of the log scale), the sharp cutoff will appear in a smoother transition. However, the exposure is always shown in the log scale in practice. Hence, what is shown in Figure 3.61 is valid. The second reason for the drop after the saturation is due to the limiting behavior of the incomplete Gamma function. As L increases, the incomplete Gamma function in the log-log plot will have an increasingly sharp transient as shown in Figure **3.19**.

A corollary of the theorem is the case where there are N i.i.d. observations $Y_1, \ldots, Y_N$ instead of a single measurement. In this case, $\text{SNR}_{\text{exp}}(\beta)$ will grow linearly with respect to the square root of the number of observations $\sqrt{N}$.

Corollary 3.1. *Consider the same setting as Theorem 3.3.2, but assume a sequence of i.i.d. random variables* $Y_1, \ldots, Y_N$*. Define the average* $Y = (1/N)\sum_{n=1}^{N} Y_n$*. The exposure-referred SNR for* Y *is*

$$\mathit{SNR}_{\mathit{exp}}(\beta) = \sqrt{N} \cdot \frac{\beta}{\sqrt{\operatorname{Var}[Y_1]}} \cdot \frac{d\mu}{d\beta}. \tag{3.78}$$

Proof. Since $Y = (1/N)\sum_{n=1}^{N} Y_n$, the mean $\mathrm{E}[Y] = \mathrm{E}[Y_1]$. It then follows that the derivative $d\mu/d\beta$ remains unchanged. For the variance, it is easy to show that $\operatorname{Var}[Y] = \operatorname{Var}[Y_1]/N$. Substituting these results into Equation (3.76) would yield

$$\begin{aligned} \mathrm{SNR}_{\mathrm{exp}}(\beta) &= \frac{\beta}{\sqrt{\operatorname{Var}[Y]}} \cdot \frac{d\mu}{d\beta} \\ &= \frac{\beta}{\sqrt{\operatorname{Var}[Y_1]/N}} \cdot \frac{d\mu}{d\beta}, \end{aligned}$$

which is the desired result. □

Limiting Case

Figure 3.61 shows that as the full-well capacity L increases, $\mathrm{SNR}_{\mathrm{exp}}(\beta)$ becomes more linear in the log-log plot. Such a behavior can be theoretically derived by analyzing the limiting cases of the incomplete Gamma function.

Recall in **Figure 3.19** the incomplete Gamma function is a monotonically decreasing function with a transient located around L. Suppose that the width of the transient is δ, then there exists an interval $\{\beta \mid L - \delta/2 \leq \beta \leq L + \delta/2\}$ such that the lower and the upper limits are

$$\Psi_L(\beta) \approx \begin{cases} 1, & \beta \leq L - \delta/2, \\ 0, & \beta \geq L + \delta/2. \end{cases} \tag{3.79}$$

Here, the approximation "≈" can be defined based on a confidence, e.g., a 99% confidence. Under these two limiting cases, the exposure-referred SNR can be derived accordingly.

Corollary 3.2. *Consider the same conditions as in Theorem 3.3.2 but with $L \gg 1$. Let $\Psi = \Psi_L(\beta)$. Under the limiting assumption of $\Psi_L(\beta)$ described in Equation (3.79), it holds that when $\beta < L - \delta/2$,*

$$\mu = \beta, \quad \mathrm{Var}\left[Y\right] = \beta, \quad \frac{d\mu}{d\beta} = 1,$$

and when $\beta > L + \delta/2$,

$$\mu = \beta\Psi + L, \quad \mathrm{Var}\left[Y\right] = \beta\Psi(\beta + 1 - 2L), \quad \frac{d\mu}{d\beta} = \Psi.$$

Consequently,

$$SNR_{\exp}(\beta) = \begin{cases} \sqrt{\beta}, & \beta \le L - \delta/2, \\ 0, & \beta \ge L + \delta/2. \end{cases} \tag{3.80}$$

Proof. When L is large, $\Psi_L(\beta)$ and $\Psi_{L-1}(\beta)$ are close enough that they can be considered approximately equal. Denote the value $\Psi_L(\beta)$ as Ψ. Then by Equation (3.79) it holds that $\Psi \to 0$ for $\beta \ge L + \delta/2$ and $\Psi \to 1$ for $\beta \le L - \delta/2$. In either case, since Ψ is a constant, it follows that the derivative $\Psi'_L(\beta) = 0$ as long as $\beta \ge L + \delta/2$ or $\beta \le L - \delta/2$. Therefore, the two cases can be derived as follows.

When $\beta \le L - \delta/2$, it holds that

$$\begin{aligned} \mu &= \beta\Psi + L(1 - \Psi) = \beta, \\ \mathrm{Var}\left[Y\right] &= \beta^2\Psi + \beta\Psi + L^2(1 - \Psi) - \mu^2 \\ &= \beta^2 \cdot 1 + \beta \cdot 1 + L^2 \cdot 0 - \mu^2 = \beta, \\ \frac{d\mu}{d\beta} &= \Psi - (L - \beta)\Psi'_{L-1}(\beta) \\ &= 1 - (L - \beta) \cdot 0 = 1. \end{aligned}$$

So, the overall SNR for $\beta \leq L - \delta/2$ is

$$\mathrm{SNR}_{\mathrm{exp}}(\beta) = \frac{\beta}{\sqrt{\mathrm{Var}\left[Y\right]}} \cdot \frac{d\mu}{d\beta} = \sqrt{\beta}.$$

When $\beta \geq L + \delta/2$, $\Psi \to 0$. Therefore,

$$\begin{aligned}
\mu &= \beta\Psi + L(1 - \Psi) \approx \beta\Psi + L, \\
\mathrm{Var}\left[Y\right] &= \beta^2\Psi + \beta\Psi + L^2(1 - \Psi) - \mu^2 \\
&= \beta^2\Psi + \beta\Psi + L^2(1 - \Psi) - (\beta\Psi + L)^2 \\
&= \beta^2\Psi + \beta\Psi + L^2 - \beta^2\Psi^2 - 2\beta\Psi L - L^2 \\
&= \beta^2\Psi + \beta\Psi - 2\beta\Psi L \\
&= \beta\Psi(\beta + 1 - 2L), \\
\frac{d\mu}{d\beta} &= \beta\Psi'_{L-1}(\beta) + \Psi_{L-1}(\beta) - L\Psi'_L(\beta) \\
&= \beta \cdot 0 + \Psi - L \cdot 0 = \Psi.
\end{aligned}$$

By taking the limit that $\Psi \to 0$, it follow that

$$\begin{aligned}
\lim_{\Psi\to 0} \mathrm{SNR}_{\mathrm{exp}}(\beta) &= \lim_{\Psi\to 0} \frac{\beta}{\sqrt{\mathrm{Var}\left[Y\right]}} \cdot \frac{d\mu}{d\beta} \\
&= \lim_{\Psi\to 0} \frac{\beta}{\sqrt{\beta\Psi(\beta + 1 - 2L)}} \cdot \Psi \\
&= \lim_{\Psi\to 0} \frac{\sqrt{\beta\Psi}}{\sqrt{(\beta + 1 - 2L)}} = 0.
\end{aligned}$$

Combining with the case where $\beta \leq L - \delta/2$, the overall SNR is proved. □

The corollary implies that as L increases, plotting $\mathrm{SNR}_{\mathrm{exp}}(\beta)$ in the log-log plot will give a linear response followed by an abrupt transition. This is exactly what is happening in the output-referred SNR equation shown in Equation (3.59). Therefore, Theorem 3.3.2 is a generalized version of the output-referred SNR curves reported in the literature. For practical algorithms such as those for high dynamic range imaging, Equation (3.80) is very common, for example, used in [75].

SNR$_{\text{exp}}(\beta)$ for truncated Poisson-Gaussian

We have found the SNR expression for truncated Poisson. Let us make the situation a little more realistic by adding the read noise and the quantization, which is nothing but the digital numbers we got in Equation (3.52). We assume that the conversion gain $G = 1$ and the offset $O = 0$. In such a case, the observation Z can be written as

$$Z = \text{ADC}(Y + \eta_{\text{read}}) = \begin{cases} 0, & \lceil Y + \eta_{\text{read}} \rfloor \leq 0 \\ \lceil Y + \eta_{\text{read}} \rfloor, & 0 < \lceil Y + \eta_{\text{read}} \rfloor < L \\ L, & \lceil Y + \eta_{\text{read}} \rfloor \leq L \end{cases}, \tag{3.81}$$

where $Y = \text{Poisson}(\beta)$, and $\eta_{\text{read}} \sim \mathcal{N}(0, \sigma^2_{\text{read}})$ and $\lceil \cdot \rfloor$ is a function that rounds the real numbers to the nearest integers. The following theorem gives the expression for the SNR if data is drawn according to Equation (3.81).

Theorem 3.3.3 (SNR$_{\text{exp}}(\beta)$ for truncated Poisson Gaussian)**.** *Consider the truncated Poisson Gaussian statistics defined in Equation (3.81). Let $\widehat{\beta}(\cdot)$ be an estimator satisfying the mean invariance property, i.e., $\widehat{\beta}(\mathbb{E}[Z]) = \beta$. Then the exposure-referred SNR is*

$$SNR_{exp}(\beta) = \frac{\beta}{\sqrt{\text{Var}[Z]}} \cdot \frac{d\mu}{d\beta}, \tag{3.82}$$

where

$$\begin{aligned} \mathbb{E}[Z] = \mu &= \beta\Psi_{L-1}(\beta) + L(1 - \Psi_L(\beta)) + \Delta_\mu(\beta), \\ \text{Var}[Z] &= \beta^2\Psi_{L-2}(\beta) + \beta\Psi_{L-1}(\beta) + L^2(1 - \Psi_L(\beta)) - \mu^2 + \Delta^2_\sigma(\beta), \end{aligned}$$

and the quantities $\Delta_\mu(\beta)$ *and* $\Delta_{\sigma^2}(\beta)$ *are respectively*

$$\begin{aligned}\Delta_\mu(\beta) = \sum_{k=-\infty}^{\infty} p_k\bigg(\sum_{q=[k]_+}^{L-1} \bigg(\frac{\mathrm{e}^{-\beta}\beta^{q-k}}{(q-k)!} - \frac{\mathrm{e}^{-\beta}\beta^q}{q!}\bigg)q \\ + L(\Psi_L(\beta) - \Psi_{[L-k]_+}(\beta))\bigg) \qquad (3.83)\end{aligned}$$

$$\begin{aligned}\Delta_{\sigma^2}(\beta) = \sum_{k=-\infty}^{\infty} p_k\bigg(\sum_{q=[k]_+}^{L-1} \bigg(\frac{\mathrm{e}^{-\beta}\beta^{q-k}}{(q-k)!} - \frac{\mathrm{e}^{-\beta}\beta^q}{q!}\bigg)q^2 \\ + L^2(\Psi_L(\beta) - \Psi_{[L-k]_+}(\beta))\bigg), \qquad (3.84)\end{aligned}$$

where $[\,\cdot\,]_+ = \max(\cdot, 0)$ *returns the positive value, and*

$$p_k = \int_{k-0.5}^{k+0.5} \frac{1}{\sqrt{2\pi\sigma_{read}^2}}\mathrm{e}^{-\frac{x^2}{2\sigma_{read}^2}}dx \qquad (3.85)$$

is the error probability due to read noise. The derivative $d\mu_Z/d\beta$ *is*

$$\begin{aligned}\frac{\partial\mu_Z}{\partial\beta} = \Psi_{L-1}(\beta) - \beta\frac{\mathrm{e}^{-\beta}\beta^{[L-2]_+}}{[L-2]_+!} + L\frac{\mathrm{e}^{-\beta}\beta^{[L-1]_+}}{[L-1]_+!} + \sum_{k=-\infty}^{\infty} p_k\bigg(\sum_{q=[k]_+}^{L-1}\bigg(-\frac{\mathrm{e}^{-\beta}\beta^{q-k}}{(q-k)!} + \frac{\mathrm{e}^{-\beta}\beta^q}{q!}\bigg)q \\ + \sum_{q=[k]_+}^{L-2}\bigg(\frac{\mathrm{e}^{-\beta}\beta^{q-k}}{(q-k)!} - \frac{\mathrm{e}^{-\beta}\beta^q}{q!}\bigg)(q+1) + L\Big(-\frac{\beta^{(L-1)}\mathrm{e}^{-\beta}}{(L-1)!} + \frac{\beta^{([L-k-1]_+)}\mathrm{e}^{-\beta}}{[L-k-1]_+!}\Big)\bigg).\end{aligned}$$

Proof. Check Appendix B. □

3.3.5 Monte-Carlo simulation

We can see that the expressions for SNR start becoming messy when we are making the noise model realistic. As we make the models more and more realistic, the analytic expressions would be significantly more challenging. A more reasonable approach is to resort to numerical schemes to estimate the approximate SNR.

General Principle

To compute $\text{SNR}_{\text{exp}}(\beta)$ for any given distribution, the more viable approach is to sample the the distribution defined by the forward model:

$$Y_m = \text{forward model}\,(\beta \;\mid\; \beta_{\text{dark}}, \sigma_{\text{read}}, L)\,, \tag{3.86}$$

for $m = 1, \ldots, M$, where M denotes the number of Monte Carlo samples drawn to compute the SNR. The notation in Equation (3.86) means that the mth sample Y_m is drawn from any given forward model. The sample Y_m is a function of the underlying signal β, among with other model parameters. When running the Monte Carlo simulation, for each β a set of $\{Y_1, \ldots, Y_M\}$ will be drawn to compute the mean and variance.

Specifically, for every β, the sample average is an estimate of $\mathrm{E}\,[Y]$ and the sample variance is an estimate of $\mathrm{Var}\,[Y]$:

$$\widehat{\mu}(\beta) = \frac{1}{M}\sum_{m=1}^{M} Y_m, \quad \text{and} \quad \widehat{\sigma}^2(\beta) = \frac{1}{M}\sum_{m=1}^{M} (Y_m - \widehat{\mu})^2.$$

Once $\widehat{\mu}(\beta)$ has been determined for every β, the derivative $d\mu/d\beta$ can be approximated by

$$\frac{d\widehat{\mu}}{d\beta} = \frac{\widehat{\mu}(\beta_{k+1}) - \widehat{\mu}(\beta_k)}{\beta_{k+1} - \beta_k},$$

where $\{\beta_k \mid \beta_k < \beta_{k+1},\; k = 1, \ldots, K\}$ is the discrete set of exposures used to evaluate the mean and variance. Consequently, $\text{SNR}_{\text{exp}}(\beta)$ can be approximately estimated by

$$\widehat{\text{SNR}}_{\text{exp}}(\beta) = \frac{\beta}{\widehat{\sigma}} \cdot \frac{d\widehat{\mu}}{d\beta}. \tag{3.87}$$

MATLAB code

The MATLAB code below illustrates the Monte-Carlo simulation of how $\text{SNR}_{\text{exp}}(\beta)$ is generated for a truncated Poisson distribution. Adding other factors to the forward model can be done by modifying the random variable Y.

```
N  = 100000;
L  = 10;
```

```
theta_set = logspace(-2,3,100);
mu      = zeros(1,100);
sigma = zeros(1,100);
for i=1:100
    theta      = theta_set(i);
    Theta      = theta*ones(N,1);
    Y          = poissrnd(Theta);
    Y(Y>L)     = L;
    mu(i)      = mean(Y);
    sigma(i) = std(Y);
end
dmu_dt   = [diff(mu)./diff(theta_set)  1];
SNR       = theta_set./sigma.*dmu_dt;
loglog(theta_set, SNR);
```

For plotting the theoretical $\mathrm{SNR}_{\mathrm{exp}}(\beta)$, one just needs to call the incomplete Gamma function.

```
theta  = logspace(-2,3,100);
Psi    = gammainc(theta,L,'upper');
Psi1   = gammainc(theta,L-1,'upper');
Psi2   = gammainc(theta,L-2,'upper');
dPsi   = -theta.^(L-1).*exp(-theta)  ...
         /gamma(L);
dPsi1  = -theta.^(L-2).*exp(-theta)  ...
         /gamma(L-1);
mu     = theta.*Psi1 + L.*(1 - Psi);
sigma  = sqrt(theta.^2 .* Psi2 + ...
         theta.*Psi1 + L^2*(1-Psi) - ...
         mu_theory.^2);
dmu_dt = theta.*dPsi1 + Psi1 - L*dPsi;
```

```
SNR      = theta./sigma.*dmu_dt;
loglog(theta, SNR);
```

Visualizing the Impacts of β_{dark} and σ_{read}

With the Monte Carlo simulation technique, complex forward models can be visualized. To illustrate the utility of the simulation, it would be useful to consider two scenarios as follows.

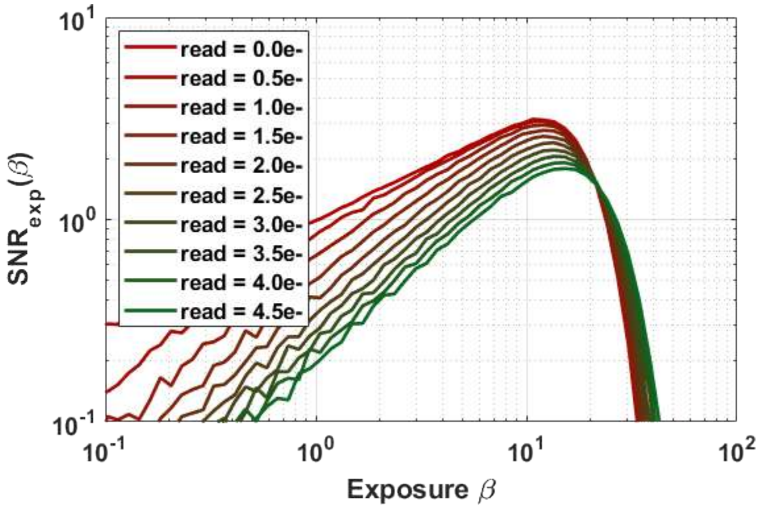


Figure 3.22. Influence of read noise. Exposure-referred SNR for a digital image sensor by considering different levels of read noise.

Example 3.5 (Influence of Read Noise)**.** *The first scenario considers a fixed dark current, full-well capacity, and A/D converter, but a varying read noise level. Let beta*$_d = 0.016e$*- (which is consistent with the quanta image sensor [46]), and a 4-bit A/D converter. The read noise level* σ_{read} *varies from 0e- to 5e- with a step interval of 0.5e-. By using* $M = \times 10^6$ *Monte-Carlo samples, the numerically simulated* $SNR_{exp}(\beta)$ *is plotted in Figure 3.22.*

As one can observe in this example, increasing the read noise leads to a reduced SNR for all β *before saturation. After saturation, the presence of the read noise will occasionally move a saturated measurement back to an unsaturated state because the Gaussian noise can take a negative value. See that the purple curves on the right-*

hand side of the plot is higher than the green curves. Therefore, for large β, there is a minor but noticeable gain in SNR especially when the read noise is high. This is not necessarily a better outcome because the increased read noise would require a more powerful denoising algorithm which is not an easy problem either.

The small fluctuation towards the tail in Figure 3.22 is due to the randomness in the Monte-Carlo simulation. As M goes to infinity, the random estimate will approach the expectation by the law of large number. □

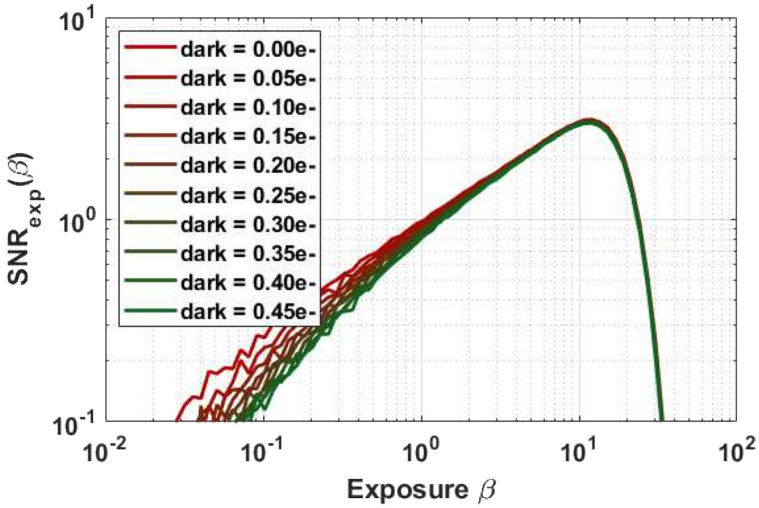


Figure 3.23. Influence of dark current. Exposure-referred SNR for a digital image sensor by considering different levels of read noise. The small fluctuation towards the tail on the left-hand side is due to randomness in the Monte-Carlo simulation.

Example 3.6 (Influence of dark current). *The second scenario considers a fixed read noise, full-well capacity, and A/D converter, but a varying dark current. To be consistent with the literature, the dark current β_{dark} is assumed to vary from 0e- to 0.5e- with a step interval of 0.05e-. The read noise level is fixed at 0.2e- based on [46]. The full-well capacity is $L = 15$ electrons, and a 4-bit A/D converter is used. Same as Example 5, $M = 10^6$ Monte Carlo samples are used to numerically generate the $SNR_{exp}(\beta)$ plot in Figure 3.23.*

Unlike Example 5 where the read noise has a substantial influence to the SNR, an increased dark current will only show its impact for small β. This should not be a surprise because when the true signal β is strong, the influence of β_{dark} will be negligible considering the small magnitude it usually has. For small β, the impact of β_{dark} is more prominent. A smaller dark current indeed leads to a higher SNR as expected. □

The utility of the Monte Carlo simulation is that it bypasses the complication of seeking for an analytic expression of $\text{SNR}_{\text{exp}}(\beta)$. To account for even more difficult modelings such as the pixel response non-uniformity, conversion gain, and exposure time, etc, one just needs to modify the forward image formation model.

3.3.6 Alternatives to SNR?

While SNR is a natural choice for analyzing the performance of an image sensor, it is by no means the *only* option. Especially for 1-bit devices such as the quanta image sensor, there are other ways to characterize the performance.

Entropy

As far as 1-bit measurements are concerned, the entropy is a natural substitute of the SNR. If Y is binary with $p_Y(1) = 1 - \Psi_q(\beta)$ and $p_Y(0) = \Psi_q(\beta)$, the entropy is

$$\begin{aligned} H(Y) &= -p_Y(1)\log_2 p_Y(1) - p_Y(0)\log_2 p_Y(0) \\ &= -(1-\Psi_q(\beta))\log_2(1-\Psi_q(\beta)) \\ &\qquad - \Psi_q(\beta)\log_2 \Psi_q(\beta). \end{aligned} \tag{3.88}$$

It is relatively easy to show that the derivative of the entropy with respect to $\Psi_q(\beta)$ is

$$\frac{d}{d\Psi_q(\beta)} H(Y) = -\log\left(\frac{1-\Psi_q(\beta)}{\Psi_q(\beta)}\right).$$

Setting it to zero will yield $\Psi_q(\beta) = \frac{1}{2}$. Therefore, the entropy is maximized when $\mathrm{E}[Y] = 1 - \Psi_q(\beta) = \frac{1}{2}$. Since $\mathrm{E}[Y]$ is the expected value of the measurement, $\mathrm{E}[Y] = \frac{1}{2}$ means that the entropy is maximized when there are 50% one's and 50% zero's in a set of independent measurements. So, if the application goal is to identify a threshold q such that the performance of the sensor is maximized, then instead of optimizing for the SNR, the alternative is to optimize the entropy.

Bit Error Rate (BER)

In the presence of read noise, the bit error rate is another commonly used criterion to evaluate the performance of a sensor. For 1-bit quanta image sensor, the BER measures the probability of making a wrong decision (i.e., declaring a 0 as a 1, or declaring a 1 as a 0). It can be readily computed as

$$\begin{aligned}\mathrm{BER}(\beta) &= p_Y(0) \cdot \int_q^{\infty} \frac{1}{\sqrt{2\pi\sigma_{\text{read}}^2}} e^{-\frac{t^2}{2\sigma_{\text{read}}^2}} dt \\ &\quad + p_Y(1) \cdot \int_{-\infty}^{q} \frac{1}{\sqrt{2\pi\sigma_{\text{read}}^2}} e^{-\frac{(t-1)^2}{2\sigma_{\text{read}}^2}} dt \\ &= \frac{1}{2}\mathrm{erfc}\left(\frac{q}{\sigma_{\text{read}}\sqrt{2}}\right)\Psi_q(\beta) \\ &\quad + \frac{1}{2}\mathrm{erfc}\left(\frac{1-q}{\sigma_{\text{read}}\sqrt{2}}\right)(1-\Psi_q(\beta)). \end{aligned} \tag{3.89}$$

Therefore, if $q = 1/2$, the BER is simplified to

$$\mathrm{BER}(\beta) = \frac{1}{2}\mathrm{erfc}\left(\frac{1}{\sigma_{\text{read}}\sqrt{8}}\right), \tag{3.90}$$

which does not depend on β. If $\mathrm{BER}(\beta)$ can be empirically measured, then by inverting Equation (3.90) one can estimate the read noise σ_{read}. For a fixed β, one can also optimize Equation (3.89) by finding an appropriate q.

3.4 Final thoughts

We have developed a very generic camera model, which could fit any application. Many of the remaining chapters use the image formation model we have developed here to understand the physics or generate data for training a machine learning model. In the latter part of the chapter, we developed the theory for calculating the signal-to-noise ratio for any image sensor. Chapter 8 uses the SNR expression we have derived here to compare the CMOS image sensors and the quanta image sensors. The SNR expression is also used to develop an HDR reconstruction algorithm.

3.4.1 Where to, from here?

Until now, we have looked at how to model the image formation process for existing cameras, which could be used for developing algorithms and analyzing the cameras. However, most of the literature seems to miss a critical aspect of developing such a tool. The imaging model could help us identify the weak points of the existing cameras, using which computational imaging engineers could develop a feedback loop with the device engineers. Based on the analysis, the computational imaging engineers could advise the hardware engineers on the weak points that need immediate attention to get the most significant benefit from a new modification to the existing sensors. Another obvious application is to develop image sensors with real-time control, where we can modify the camera setting in real-time based on the SNR that we get for the scene.

4. COLOR IMAGING WITH QUANTA IMAGE SENSORS

Color imaging is often achieved by placing a *color filter array* (CFA) in front of the image sensor, such as the Bayer pattern CFA, which has only one color channel per pixel. The captured image will be a mosaic of different colors like in Figure 4.1. Converting this single-channel mosaic image into a three-channel RGB image is called demosaicing. Demosaicing is a difficult task because we are essentially doing a type of super-resolution, where we are going from one color channel per pixel (different color for different pixels) to three color channels. We need to deal with aliasing in color channels, which requires careful consideration. *Photon-limited* conditions further complicate the matter because of the need to handle demosaicing and also heavy noise at the same time.

Traditional CIS demosaicing algorithms usually emphasize the mitigation of the aliasing (aka the *Moirè* artifacts), for example, using advanced edge-aware demosaicing methods [82] or using a post-processing module to remove the demosaicing artifacts [83], [84]. However, aliasing is not a big problem when the pixels are small (Figure 4.1) because of the *diffraction limit*, which can be thought of as a slight blur introduced by diffraction. Because of diffraction, aliasing is not dominant anymore. However, note that the image will be blurred due to diffraction, requiring further processing to be deblurred. We are not worried about this blur in this chapter. If we ignore the blur, we can use a simple method for demosaicing, possibly some traditional methods, such as linear demodulation followed by denoising, to reconstruct the color.

To elaborate on the aliasing problem, we show three illustrations in Figure 4.1. In Figure 4.1(a), we plot the Nyquist sampling limit of the color filter array as a function of the pixel pitch and illustrate two-color spectra associated with the sampling limits. Nyquist limit defines the lowest spatial sampling frequency required to prevent aliasing. As the pixels become small, we effectively oversample the scene, increasing the Nyquist limit. Using a $f/5.6$ optical system as an example, the maximum pixel pitch we can afford is 1.6μm, which is safely above the 1.1μm pitch of the current CIS-QIS [35]. This argument is further justified by looking at the synthetic data shown in Figure 4.1(b) and (c), where we show the raw Bayer

CFA data and the corresponding color spectrum. It can be seen that because of the small pixel pitch, QIS can potentially offer a much better spectrum.

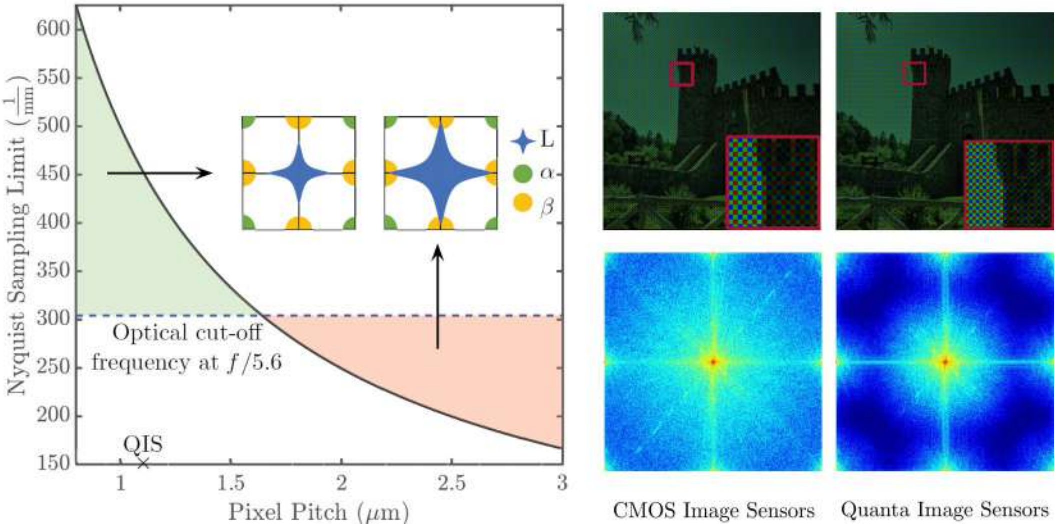


Figure 4.1. Aliasing is not a problem for small pixels. [Left] Nyquist limit decreases with pixel pitch. At f-number value of $f/5.6$, QIS is diffraction-limited since Nyquist limit exceeds the optical cut-off frequency. [Top Right] An average of 10 QIS frames at photon level of 5.5e$^-$. The CFA is a Bayer pattern. [Bottom Right] Fourier spectrum of the color. Notice that interference between base-band luminance and chrominance components at $(\pi, 0)$, $(0, \pi)$ and (π, π) is minimal.

4.1 Related Work

Classical demosaicing algorithms. Classical demosaicing methods are developed for CIS with well-illuminated images. The demosaicing algorithm must also have denoising capabilities under low-light conditions where the noise is heavy. It is important to note that the order of denoising and demosaicing matters [85]–[88]. If demosaicing is performed first, then the interpolation will destroy the spatial independence of the noise, which will substantially complicate the denoising process [89]. If denoising is performed first, then most of the image priors cannot be used because the mosaicked images do not have natural image statistics [86], [90]–[93]. Joint demosaicing and denoising methods are better options here. However, most of the joint demosaicing and denoising methods are iterative [87], [94]–[99].

Deep neural network based demosaicing. State-of-the-art demosaicing algorithms are largely based on deep neural networks. The idea is to modify a generic deep neural network by adding a space-to-depth layer that converts the raw Bayer image into 4 Bayer channels with quarter resolution. The network processes these down-sampled channels then upscales them to a full-resolution color image using a depth-to-space layer. For example, the Demosaic-Net by Gharbi et al. [90] uses a residual network and a customized dataset within a curriculum training approach. Dong et al. [91] use generative adversarial training. Tan et al. [98], Cui et al. [83], Niu-Ouyang [84] use multi-phase approaches. Ehret et al. [93] study burst reconstruction without ground truth. Wu et al. [100], and Kiku et al. [101] use a guided filter for chrominance reconstruction. However, they do not consider the effect of noise.

We propose two solutions in this chapter - one classical and one deep learning-based demosaicing method. Both the methods consider the physics behind the noise and the color imaging.

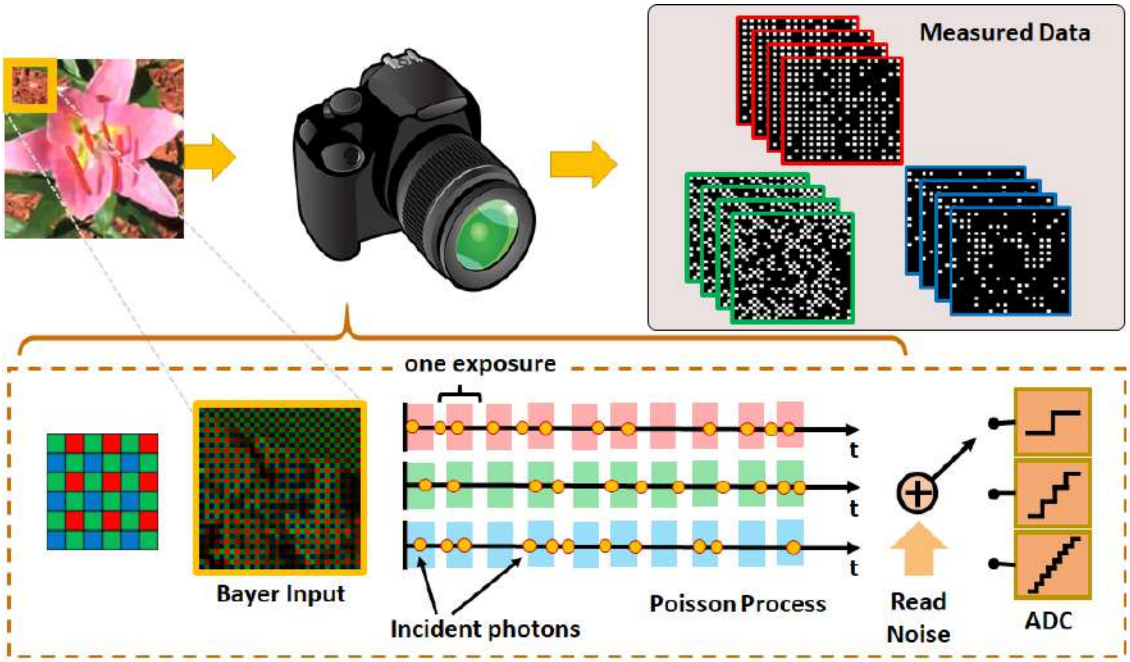


Figure 4.2. Color imaging model.

4.2 Plug-and-play ADMM based color reconstruction

4.2.1 Modified imaging model

We start by introducing some small changes to the imaging model we introduced in chapter 3. Recall that in the absence of dark noise and other non-uniformities, and , if we assume that the scene is not changing over the integration time, we can write the imaging model of a camera as

$$\boldsymbol{Y} = \mathrm{ADC}\left\{G.\left(\mathrm{Poisson}(\alpha \cdot \boldsymbol{\beta}) + \mathcal{N}(\mathbf{0}, \sigma_{\mathrm{read}}^2 \mathbb{I})\right)\right\}, \tag{4.1}$$

where $\boldsymbol{\beta} \in \mathbb{R}^N$ is the underlying ground-truth intensity we want to estimate (N is the total number of pixels), α is proportional to the integration time and decides the average photon level of the scene and σ_{read} is the read noise. We assume that the offset O for the ADC used in chapter 3 is zero.

For color imaging, we argued that $\boldsymbol{\beta}$ contains different color channels for each pixel. This however is not a convenient notation here, because we want to recover all three channels for each pixel. So we make a small tweak in the imaging model as follows.

$$\boldsymbol{Y} = \mathrm{ADC}\left\{\mathrm{Poisson}(\boldsymbol{\beta} = \alpha \cdot \boldsymbol{S}\boldsymbol{\theta}) + \mathcal{N}(\mathbf{0}, \sigma_{\mathrm{read}}^2 \mathbb{I})\right\}, \tag{4.2}$$

where $\boldsymbol{\theta} \in \mathbb{R}^{3N}$ contains 3 channels (R,G and B) per pixel. $\boldsymbol{S} \in \mathbb{R}^{M \times 3M}$ is a fat matrix with 1's and 0's which decides the color channel that corresponds to each pixel based on the color filter array (CFA) used.

$$\boldsymbol{S} \overset{\mathrm{def}}{=} \{\boldsymbol{S}_r, \boldsymbol{S}_g, \boldsymbol{S}_b\}, \tag{4.3}$$

where $\boldsymbol{S}_r$, $\boldsymbol{S}_g$, and $\boldsymbol{S}_b$ are all diagonal matrices with 1's and 0's along the diagonal, and $\boldsymbol{S}_r + \boldsymbol{S}_g + \boldsymbol{S}_b = \mathbb{I}$. Note that in this case, we want to recover $\boldsymbol{\theta}$.

In this section, we deal with both single-bit ADC and multi-bit. For multi-bit imaging, we assume that the conversion gain $G = 1$. For single-bit however, we use any *integer* conversion gain $G \in \mathbb{Z}$.

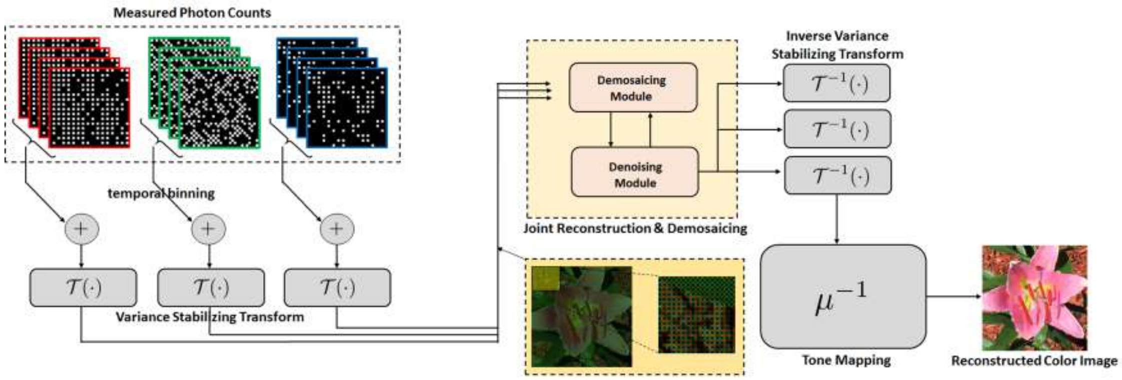

Measured Photon Counts
temporal binning
$\mathcal{T}(\cdot)$
$\mathcal{T}(\cdot)$
$\mathcal{T}(\cdot)$
Variance Stabilizing Transform
Demosaicing Module
Denoising Module
Joint Reconstruction & Demosaicing
Inverse Variance Stabilizing Transform
$\mathcal{T}^{-1}(\cdot)$
$\mathcal{T}^{-1}(\cdot)$
$\mathcal{T}^{-1}(\cdot)$
μ^{-1}
Tone Mapping
Reconstructed Color Image

Figure 4.3. The image reconstruction pipeline consists of (i) a temporal binning step to sum the input frames, (ii) a variance stabilizing transform $\mathcal{T}$ to transform the measurement so that the variance is stabilized, (iii) a joint reconstruction and demosaicing algorithm to recovery the color, (iv) an inverse transform to compensate the forward transform, and (v) a tone mapping operation to correct the contrast.

4.2.2 Joint denoising and demosaicing

The task of image reconstruction is to recover color scene $\boldsymbol{\theta}$ from the measurements $\mathcal{Y} = \{\boldsymbol{Y}_1, \boldsymbol{Y}_2, \dots \boldsymbol{Y}_T\}$, where T is the number of frames captured. In the gray-scale setting, we can formulate the problem as maximum-likelihood and solve it using convex optimization tools [80], [102]–[104]. We can also use learning-based methods, e.g., [105]–[108] to reconstruct the signal. The method we present here is based on the transform-denoise approach by Chan *et al.* [76]. Transform-denoise because is a physics-based approach and is robust to different sensor configurations. For example, in learning-based approaches, if we change the number of frames to sum, we need to train a different model or neural network like in chapter 5.

4.2.3 Reconstruction pipeline

The pipeline of the proposed reconstruction algorithm is shown in Fig. **4.3**. Given the measurements $\mathcal{Y} = \{\boldsymbol{Y}_1, \boldsymbol{Y}_2, \dots \boldsymbol{Y}_T\}$ we first averagew the frames to generate a single image $\boldsymbol{Z}$:

$$\boldsymbol{Z} = \frac{1}{T}\sum_{t=1}^{T} \boldsymbol{Y}_t. \tag{4.4}$$

This step can be integrated into the hardware of the camera, if we want to, so that the output of the camera will be the average of multiple frames.

If we assume that there is no processing of the bits besides temporal average, then as shown in chapter Equation (3.69), using the mean invariance property, we can write the estimator $\widehat{\beta}$ as

$$\widehat{\beta} = \mu^{-1}(Z) \tag{4.5}$$

where $\mu(\beta)$ is the mean function which maps the underlying parameter β to the corresponding mean of the the variable Z. Check chapter 3 for more details on how to obtain this estimator.

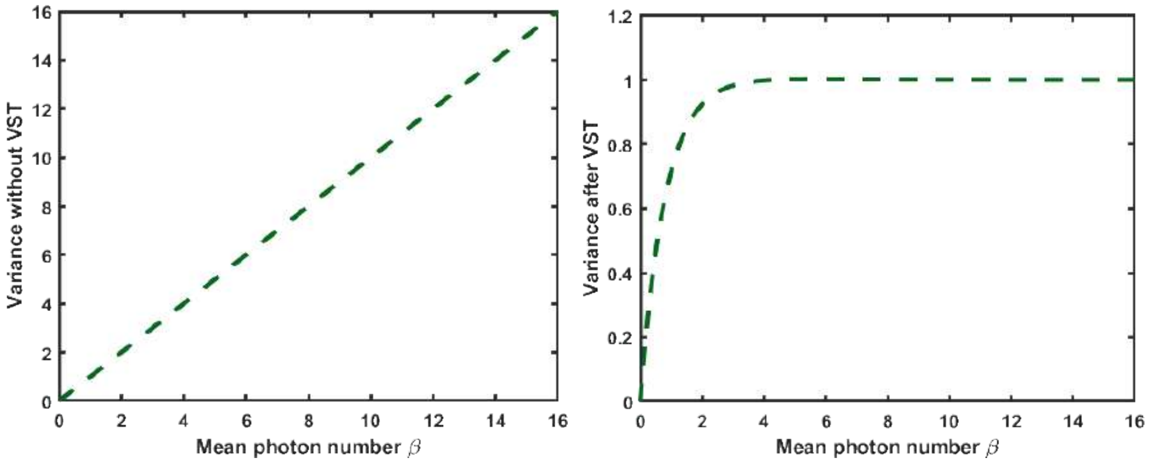
Variance stabilizing transform

Now, we want to process the data besides just averaging the frames. However, in either single-bit or multi-bit mode, the Binomial or the Poisson statistics are not easy as the variance changes with the mean. This prohibits the use of any off-the-shelf algorithms that are based on i.i.d. Gaussian assumptions. To use any of these methods developed for Gaussian noise with the assumption that the noise strength is uniform throughout the image, we need to have a technique that stabilizes the noise to the same strength across the image. Such a technique is called the *variance stabilizing transform* (VST) [109]–[111]. Figure 4.4 demonstrates the effect of VST on Poisson data.

The VST for single bit, irrespective of the gain G is simple, as the random variable is a binomial random variable. As suggested by [76], we use the corresponding Ancombe transform.

$$\mathcal{T}_{\text{single-bit}}(\boldsymbol{z}) \overset{\text{def}}{=} \sqrt{T + \frac{1}{2}} \sin^{-1} \sqrt{\frac{\boldsymbol{z} + \frac{3}{8}}{T + \frac{3}{4}}}. \tag{4.6}$$

For multi-bit, ideally we should either use [113], [114] to find the corresponding VST. In this section, we take a lazy route, and assume that the pixels do not saturate in the multi-bit mode and use the VST corresponding to simple Poisson random variable. It is given by



Without VST After VST

Figure 4.4. Variance Stabilization Transform. We look at the effect VST has on Poisson data. We use the Anscombe transform [112] for stabilizing the transform. We can see that the variance which is directly proportional to the mean β stabilizes to a value of 1 after VST.

$$\mathcal{T}_{\text{multi-bit}}(\boldsymbol{z}) \overset{\text{def}}{=} \sqrt{\boldsymbol{z}+\frac{3}{8}}. \tag{4.7}$$

So now if we apply the appropriate VST $\tau(\cdot)$ on the sum of frames $\boldsymbol{z}$, the noise will stabilized, and we can use any Gaussian methods for demosaicing and denoising. The problem can be formulated as

$$\hat{\boldsymbol{v}} = \underset{\boldsymbol{v}}{\operatorname{argmin}} \ \ ||\boldsymbol{S}\boldsymbol{v} - \mathcal{T}(\boldsymbol{z})||^2 + \lambda g(\boldsymbol{v}), \tag{4.8}$$

where $\boldsymbol{v} \in \mathbb{R}^3 N$ is the RGB image we are interested in recovering, g is some regularization function controlling the smoothness of $\boldsymbol{v}$. Note that Equation (4.8) is a standard demosaicing-denoising problem assuming i.i.d. Gaussian noise.

Joint reconstruction and demosaicing

The optimization problem in Equation (4.8) is a standard least-squares with regularization function g. Thus, most convex optimization algorithms can be used as long as g is convex. This section adopts a variation of the alternating direction method of multiplier (ADMM) by replacing g with an off-the-shelf image denoiser. Such algorithm is coined the name Plug-and-Play (PnP) [115] (and different versions thereafter [116]).

For the particular problem in Equation (4.8), the PnP ADMM algorithm iteratively updates the following two steps:

Demosaicing Module:

$$\boldsymbol{x}^{(k+1)} = (\boldsymbol{S}^T\boldsymbol{S} + \rho\boldsymbol{I})^{-1}(\boldsymbol{S}^T\mathcal{T}(\boldsymbol{z}) + \rho(\boldsymbol{v}^{(k)} - \boldsymbol{u}^{(k)})), \tag{4.9}$$

Denoising Module:

$$\boldsymbol{v}^{(k+1)} = \mathcal{D}_{\rho/\lambda}(\boldsymbol{x}^{(k+1)} + \boldsymbol{u}^{(k)}), \tag{4.10}$$

and updates the Lagrange multiplier by $\boldsymbol{u}^{(k+1)} = \boldsymbol{u}^{(k)} - (\boldsymbol{x}^{(k+1)} - \boldsymbol{v}^{(k+1)})$. Readers interested in the detailed derivation can consult, e.g., [116]. Here, ρ is an internal parameter that controls the convergence. The operator $\mathcal{D}$ is an off-the-shelf image denoiser, e.g., Block-matching and 3D filtering (BM3D) or deep neural network denoisers. The subscript ρ/λ denotes the denoising strength, i.e., the hypothesized "noise variance". Since $\boldsymbol{S}^T\boldsymbol{S}$ is a diagonal matrix, the inversion is pointwise.

Non-iterative algorithm

The 1.1μm pixel pitch of the proposed QIS can potentially lead to a spatial resolution as high as or even higher than a conventional CMOS sensor. When this happens, in certain applications we can trade-off the color reconstruction efficiency and the resolution. For example, instead of using one jot for one pixel, we can use four jots for one pixel as shown in Fig. **4.5**.

Using four jots for one pixel allows us to bypass the iterative ADMM steps because there is no more missing pixel problem. In this case, the matrix $\boldsymbol{S} \in \mathbf{R}^{M\times 3M}$ will become $\boldsymbol{S} = \text{diag}\{\frac{1}{4}\boldsymbol{I}, \frac{1}{2}\boldsymbol{I}, \frac{1}{4}\boldsymbol{I}\} \in \mathbf{R}^{3M\times 3M}$, and hence Equation (4.8) is simplified to a denoising problem with different noise levels for the three channels. In particular, the green channel has half of the variance of the red and the blue. For implementation, we can modify a denoiser, e.g., BM3D to accommodate the different noise variances. Since there is no more ADMM iteration, the algorithm is significantly faster. While we have used BM3D to demonstrate the

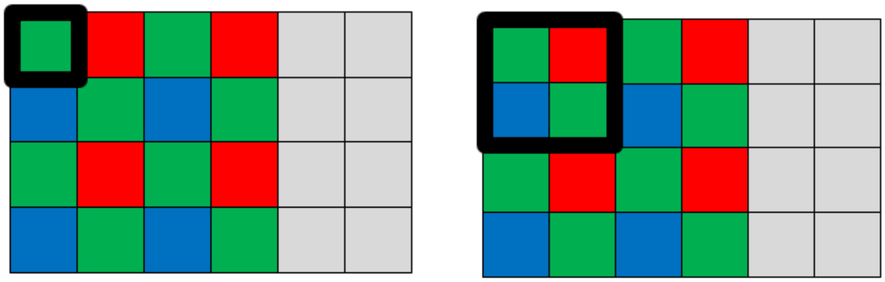


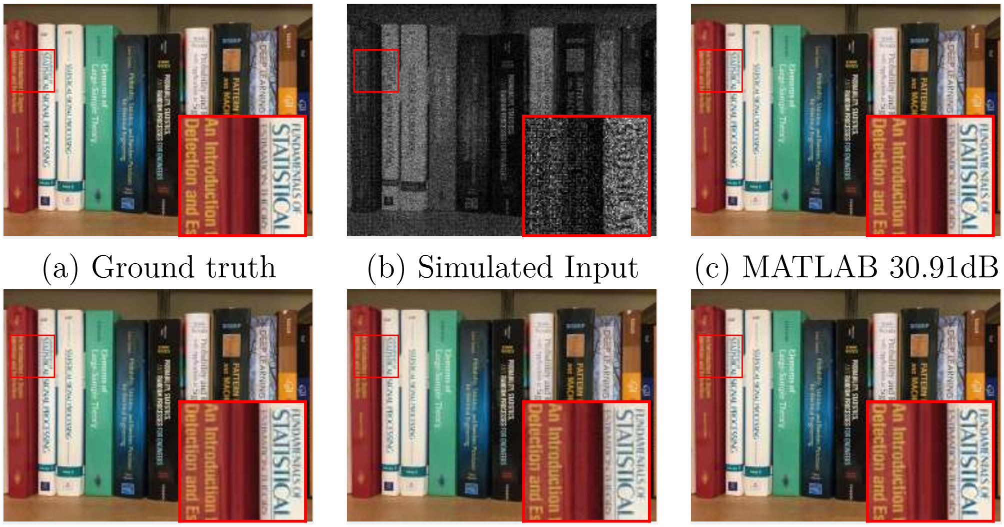
Figure 4.5. Non-iterative solution. QIS can achieve higher spatial resolution, we can use four color jots to reconstruct one pixel. In this case, we can bypass the iterative ADMM algorithm and use a one-shot denoising method.

results, any off the shelf denoiser which is used in CIS based cameras can be used to denoise the image for the four-jot to one-pixel method. We would also like to stress on the fact that both the Anscombe transform and the transform $\mathcal{M}$ can be implemented as a look up table. So, this method can be as fast a current denoiser being used in a CIS based camera.

4.2.4 Experimental results

Comparisons

As we will see later in the chapter, in terms of performance, neural networks are miles ahead. However, traditional methods do not put too much demand on hardware needed for reconstruction. To be fair, in this section we compare the proposed methods only with non-neural network solutions. We compare the proposed method with several existing methods on a synthetic dataset shown in Fig. 4.6. We simulate the raw color QIS data by assuming a Bayer pattern and using the image formation pipeline described in the previous section. We demosaic the images using: (a) a baseline method using MATLAB's demosaic preceded by gray-scale BM3D denoising of R, G_1, G_2 and B channels and followed by color BM3D denoising; (b) Least-squares luma-chroma demultiplexing (LSLCD) method [117], which has a built-in BM3D denoiser; (c) Hirakawa's PSDD method [118], which does joint denoising and demosaicing for Poisson noise; and (d) the proposed method using BM3D with $(\lambda, \rho) = (0.001, 5)$. We apply variance stabilizing transform, except for PSDD which is designed for Poisson noise. The results show that the proposed method has a significantly better performance, both in terms of Peak Signal to Noise Ratio (PSNR) and visual quality.



(a) Ground truth (b) Simulated Input (c) MATLAB 30.91dB

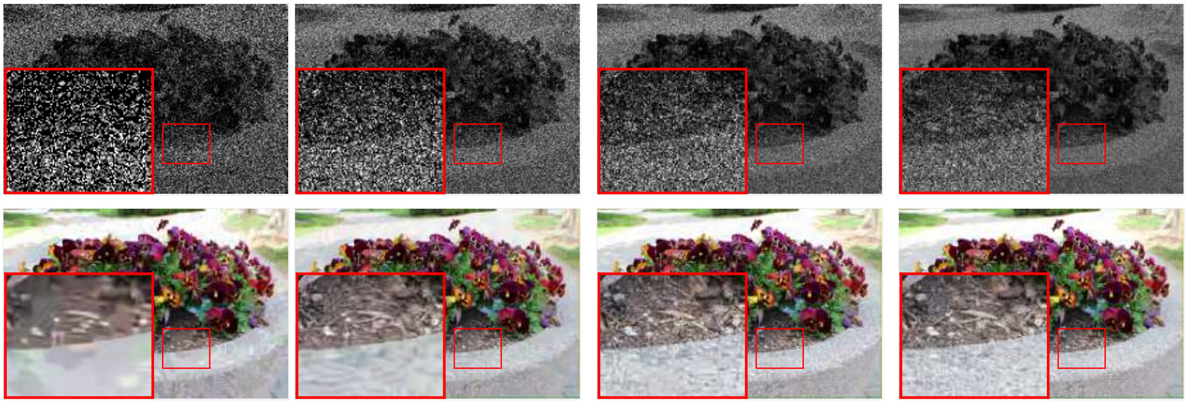
(d) [117] 30.24dB (e) [119] 29.57dB (f) Ours 31.51dB

Figure 4.6. Simulated QIS experiment. The goal of this experiment is to compare the proposed iterative algorithm with existing methods. We assume the observed Bayer RGB image is from a 3-bit QIS sensor. (a) Ground Truth; (b) One 3-bit QIS frame; (c) MATLAB demosaic preceded and followed by BM3D; (d) LSLCD[117]; (e) Hirakawa's PSDD method [119], with a built-in wavelet shrinkage denoiser; (f) Proposed method with BM3D.

Synthesized QIS data

We conduct a synthetic experiment to provide a quantitative evaluation of the performance of the proposed algorithm. To this end, we simulate the image formation pipeline by passing through color images to generate the QIS raw input data, with different number of bits. Figure 4.7 shows one example.

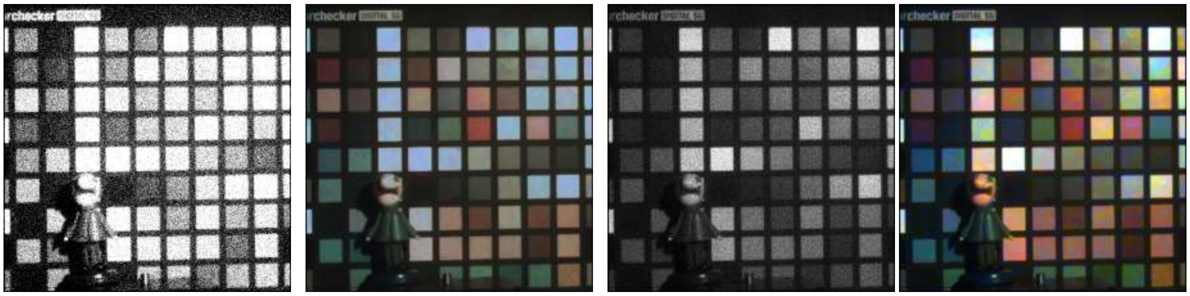
In our simulation, we assume that the number of QIS frames is $T = 4$, and the average number of photon per pixel is 0.28, 0.85, 1.98 and 4.23 photons / frame for 1-bit, 2-bit, 3-bit and 4-bit QIS, respectively. On the measurement side, we generate single-bit and multi-bit data by thresholding the raw sensor output. To reconstruct the image, we use the proposed method with PnP and BM3D. The parameters are set as $\rho = 1$ and $\lambda = 0.007, 0.003, 0.002$ and 0.0007 for 1-bit, 2-bit, 3-bit and 4-bit QIS, respectively. With as low as 1-bit, the reconstructed image in Fig. 4.7 is already capturing most of the features. As the number of bits increases, the visual quality improves.



1-bit, 22.67dB 2-bit, 23.86dB 3-bit, 25.37dB 4-bit, 26.73dB

Figure 4.7. Synthetic experiment for quantitative evaluation. [Top row]: One frame of the QIS measurements using different number of bits. [Bottom row]: Reconstructed images using the proposed method with 20 frames of QIS data. The average photon counts per pixel are 0.25, 0.75, 1.75 and 3.75 for 1-bit, 2-bit, 3-bit and 4-bit QIS, respectively.

Real QIS data

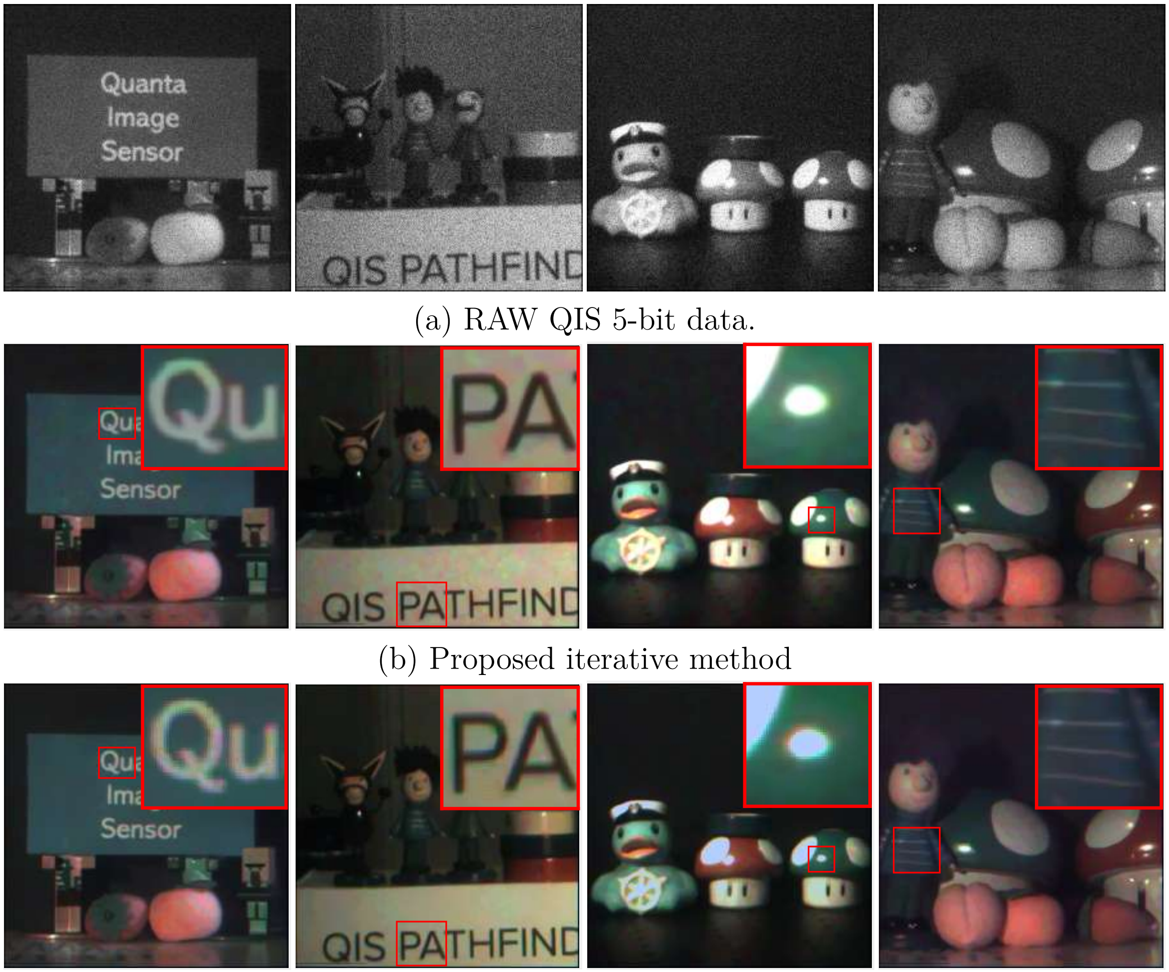


(a) Raw input, 1-bit (b) Processed from (a) (c) Raw input, 5-bit (d) Processed from (c)

Figure 4.8. (a) One 1-bit frame. (b) Reconstructed color image using 50 frames of 1-bit input with threshold $q = 4$. (c) One 5-bit frame. (d) Reconstructed color image using 10 frames of 5-bit input. The average number of photons per frame is 5.

We first show the reconstruction of an image of the Digital SG ColorChecker chart. We generate two sets of measurements: (a) a set of 50 one-bit frames, quantized with a threshold $q = 4$, and (b) a set of 10 five-bit frames. We use PnP and BM3D with $(\lambda, \rho) = (0.15, 10)$

and $(0.01, 10)$ for the 1-bit and 5-bit data, respectively. After reconstruction, the results are multiplied by a 3×3 color correction matrix to mitigate any color cross-talk. This matrix is generated by linear least square regression of the 24 Macbeth color patches that lies inside the SG ColorChecker chart. The results in Fig. 4.8 suggest that while 1-bit mode has color discrepancy with the ground truth, the 5-bit mode is producing a reasonably-high color accuracy.



(a) RAW QIS 5-bit data.

(b) Proposed iterative method

(c) Proposed non-iterative method

Figure 4.9. Real QIS image reconstruction. The exposure time for each frame is 50 μs. The average number of photons per frame is 4.2, 3.0, 1.9, and 2.9 for each image respectively. Both methods use 4 frames for reconstruction. The raw data has a resolution of 1024×1024 pixels. The ADMM method retains the resolution, whereas the non-iterative method reduces the resolution to 512×512. Reconstruction using both the methods are shown at the same size for easier visual comparison. Notice that the non-iterative algorithm is able to achieve a visual quality almost similar to the ADMM method.

Next, we show the result of imaging real scenes. See Fig. 4.9. In this experiment, the exposure time for each frame is set to 50μs. The average number of photons per pixel is approximately 4.2 photons for the ”QIS” sign image, 3 photons for the Pathfinder image, 1.9 photons for the duck image, and 2.9 photons for the mushroom image. For all images, we collect the data using a 5-bit QIS.

We demonstrate two algorithms: (i) the proposed transform-denoise framework using PnP + BM3D, and (ii) assuming four-jots to one-pixel scenario by trading half of the spatial resolution. The typical runtime on an unoptimized MATLAB code is approximately 4 minutes for PnP, and 10 seconds for the four-jot to one-pixel method. An interesting observation is that even in the lower-resolution case, the details are not significantly deteriorated unless we zoom-in. However, the speed up we get is substantial.

4.3 Learning based demosaicing

We have seen a classical demosaicing method in the last section. In this section, we will take a look at deep learning based demosaicing method that still takes into consideration the physics of color imaging.

4.3.1 Innovations

The method proposed in this section makes two major innovations

1. Frequency-selection. The proposed demosaicing algorithm is rooted in the classical theory of color filter arrays. We develop a color processing module to *demodulate* the color channels by selecting the known carrier frequencies of the color filter array. Existing deep learning-based solutions using generic convolutional neural networks largely do not use the physics of the color filter arrays.

2. Guided reconstruction. The proposed algorithm leverages the physics that the luma channel has a much better signal-to-noise ratio than the chroma channels. As such, signal details that are preserved by the luma channel can be used to guide the filtering

of the chroma channels. Existing generic convolutional neural networks do not exploit these characteristics of the data.

4.3.2 Frequency selection demosaicing network

The proposed method leverages the classical frequency selection with new modifications. In this section, we first provide a background on frequency selection. Afterward, we present our modifications of learned demosaicing low-pass filters and guided filtering of chroma channels using the luma channel (III.B). We also present the loss functions and the training process.

Frequency selection

Consider a color image $\boldsymbol{y}_{rgb} \in \mathbf{R}^{H \times W \times 3}$. We denote the normalized light intensities in the red, green and blue channels at the pixel (m,n) as

$$\boldsymbol{y}_{rgb}(m,n) = \Big[\boldsymbol{y}_r(m,n), \quad \boldsymbol{y}_g(m,n), \quad \boldsymbol{y}_b(m,n)\Big], \tag{4.11}$$

where $m = 0, \ldots, H-1$, $n = 0, \ldots, W-1$. The color image $\boldsymbol{y}_{rgb} \in \mathbf{R}^{H \times W \times 3}$ is sub-sampled by the color filter array (CFA) to create a mosaicked image $\boldsymbol{y}_{\text{CFA}} \in \mathbf{R}^{H \times W}$. Assuming that the CFA follows the standard Bayer pattern, it can be shown that the pixel (m,n) of the mosaicked image takes the form (See [120], [121]):

$$\begin{aligned} \boldsymbol{y}_{\text{CFA}}(m,n) = \boldsymbol{y}_L(m,n) + \boldsymbol{y}_\alpha(m,n) \left(\mathrm{e}^{\mathrm{j}\pi m} + \mathrm{e}^{\mathrm{j}\pi n}\right) \\ + \boldsymbol{y}_\beta(m,n)\mathrm{e}^{\mathrm{j}\pi(m+n)}, \end{aligned} \tag{4.12}$$

where the components $\boldsymbol{y}_L$, $\boldsymbol{y}_\alpha$ and $\boldsymbol{y}_\beta$ are defined as a linear transformation of the latent RGB color pixels:

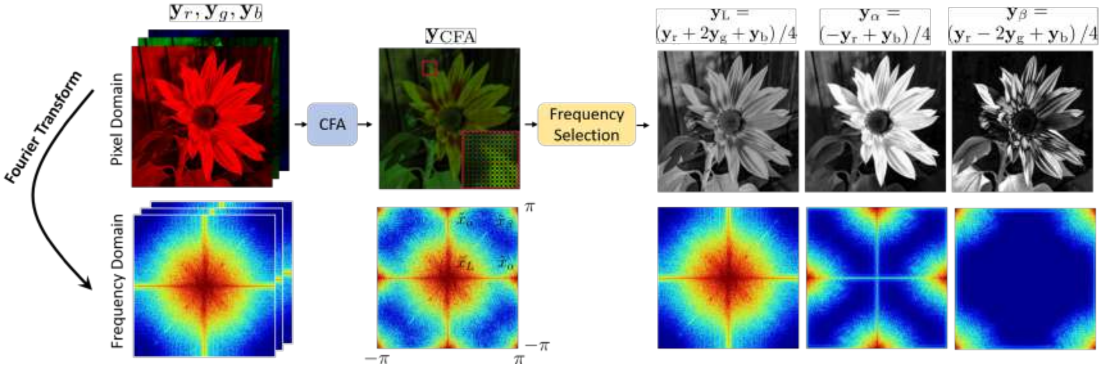
$$\begin{bmatrix} \boldsymbol{y}_L(m,n) \\ \boldsymbol{y}_\alpha(m,n) \\ \boldsymbol{y}_\beta(m,n) \end{bmatrix} = \underbrace{\begin{bmatrix} 1/4 & 1/2 & 1/4 \\ -1/4 & 0 & 1/4 \\ 1/4 & -1/2 & 1/4 \end{bmatrix}}_{\overset{\text{def}}{=} \boldsymbol{T}} \begin{bmatrix} \boldsymbol{y}_r(m,n) \\ \boldsymbol{y}_g(m,n) \\ \boldsymbol{y}_b(m,n) \end{bmatrix}. \tag{4.13}$$

Note that Equation Equation (4.12) is a forward model. That is, given the luma and chroma components $(\boldsymbol{y}_L, \boldsymbol{y}_\alpha, \boldsymbol{y}_\beta)$ we can determine $\boldsymbol{y}_{\text{CFA}}$. The inverse problem, which is the demosaicing problem, is to determine $(\boldsymbol{y}_L, \boldsymbol{y}_\alpha, \boldsymbol{y}_\beta)$ from $\boldsymbol{y}_{\text{CFA}}$.

The starting point of frequency selection is to inspect the Fourier spectrum of $\boldsymbol{y}_{\text{CFA}}$. If we take the 2D discrete Fourier transform of $\boldsymbol{y}_{\text{CFA}}$, we can show that the frequency representation of $\boldsymbol{y}_{\text{CFA}}$ is given by

$$\begin{aligned}\widetilde{\boldsymbol{y}}_{\text{CFA}}(\mu,\nu) = \underbrace{\widetilde{\boldsymbol{y}}_L(\mu,\nu)}_{\text{base-band}} + \underbrace{\widetilde{\boldsymbol{y}}_{\alpha_1}(\mu-\pi,\nu) + \widetilde{\boldsymbol{y}}_{\alpha_2}(\mu,\nu-\pi)}_{\text{horizontal/vertical side-band}} \\ + \underbrace{\widetilde{\boldsymbol{y}}_\beta(\mu-\pi,\nu-\pi)}_{\text{diagonal side-band}},\end{aligned} \tag{4.14}$$

where μ and ν are the 2D angular frequencies and $\widetilde{(\cdot)}$ denotes the Fourier transform of the argument. Equation Equation (4.14) suggests that the spectrum of the mosaicked image $\widetilde{\boldsymbol{y}}_{\text{CFA}}$ comprises a linear combination of a luma channel $\widetilde{\boldsymbol{y}}_L$, two alpha chroma channels $\widetilde{\boldsymbol{y}}_{\alpha_1}$ and $\widetilde{\boldsymbol{y}}_{\alpha_2}$, and the beta chroma channels $\widetilde{\boldsymbol{y}}_\beta$. Figure 4.10 illustrates the ideas of the frequency analysis. Given a color image, we can inspect the image generated by the color filter array. In the frequency domain, the luma channel occupies the center of the spectrum, whereas the chroma channels are located on the sides of the spectrum.



$\mathbf{y}_r, \mathbf{y}_g, \mathbf{y}_b$
Fourier Transform
Pixel Domain
Frequency Domain
CFA
$\mathbf{y}_{\text{CFA}}$
$\widetilde{x}_\alpha$
$\widetilde{x}_\beta$
$\widetilde{x}_L$
$\widetilde{x}_\alpha$
π
$-\pi$
$-\pi$
π
Frequency Selection
$\mathbf{y}_{\text{L}} = (\mathbf{y}_{\text{r}} + 2\mathbf{y}_{\text{g}} + \mathbf{y}_{\text{b}})/4$
$\mathbf{y}_\alpha = (-\mathbf{y}_{\text{r}} + \mathbf{y}_{\text{b}})/4$
$\mathbf{y}_\beta = (\mathbf{y}_{\text{r}} - 2\mathbf{y}_{\text{g}} + \mathbf{y}_{\text{b}})/4$

Figure 4.10. Objective of classical frequency selection. Given a color filter array (CFA) image $\boldsymbol{y}_{\text{CFA}}$, the frequency selection method is a Fourier domain operation that extracts the corresponding frequency components of the luma $\boldsymbol{y}_L$ and chroma $\boldsymbol{y}_\alpha, \boldsymbol{y}_\beta$ channels from the image.

The Fourier spectrum in Equation Equation (4.14) indicates that the CFA is effectively *modulating* the color channels. As such, a natural solution for demosaicing is to *demodulate* $\boldsymbol{y}_{\text{CFA}}$ so that we can retrieve $(\boldsymbol{y}_L, \boldsymbol{y}_\alpha, \boldsymbol{y}_\beta)$. Demodulation is feasible here because we know the CFA and also its *carrier frequency* from basic principles of sampling theorem in 2D domains. Denoting the carrier frequencies for $\boldsymbol{y}_{\alpha_1}$, $\boldsymbol{y}_{\alpha_2}$ and $\boldsymbol{y}_\beta$ are $\boldsymbol{\omega}_{\alpha_1}$ and $\boldsymbol{\omega}_{\alpha_2}$, $\boldsymbol{\omega}_\beta$ respectively, then the carriers are defined as (using α_1 as an example):

$$c_{\alpha_1}(m,n) = A_{\alpha_1} \cos\left(\boldsymbol{\omega}_{\alpha_1}^T \begin{bmatrix} m \\ n \end{bmatrix} + \theta_{\alpha_1}\right), \tag{4.15}$$

where A_{α_1} and θ_{α_1} are the amplitude and the phase offset of the carriers. To demodulate the color, we multiply $\boldsymbol{y}_{\text{CFA}}(m,n)$ with the carriers $c_{\alpha_1}(m,n)$, followed by convolving with a predefined lowpass filter $g(m,n)$:

$$\boldsymbol{y}_{\alpha_1}(m,n) = (\boldsymbol{y}_{\text{CFA}}(m,n) \times c_{\alpha_1}(m,n)) \circledast g(m,n) \tag{4.16}$$

The computation for $\boldsymbol{y}_{\alpha_2}$ and $\boldsymbol{y}_\beta$ is performed in a similar manner. For simplicity, we combine the two α channels by simple averaging:

$$\boldsymbol{y}_\alpha(m,n) = \boldsymbol{y}_{\alpha_1}(m,n) + \boldsymbol{y}_{\alpha_2}(m,n). \tag{4.17}$$

The base-band luma component is recovered by subtracting the re-modulated (i.e., shifted to their original positions) $\boldsymbol{y}_{\alpha_1}$, $\boldsymbol{y}_{\alpha_2}$ and $\boldsymbol{y}_\beta$ components from the input CFA image:

$$\begin{aligned} \boldsymbol{y}_L(m,n) = \boldsymbol{y}_{\text{CFA}}(m,n) - \boldsymbol{y}_{\alpha_1}(m,n) \times c_{\alpha_1}(m,n) \\ - \boldsymbol{y}_{\alpha_2}(m,n) \times c_{\alpha_2}(m,n) - \boldsymbol{y}_\beta(m,n) \times c_\beta(m,n). \end{aligned} \tag{4.18}$$

The demodulation process is pictorially illustrated in Figure 4.11. The input CFA image is first multiplied by the carriers. In the Fourier domain (the red curves shown at the bottom), the spectrum is shifted according to the carrier frequency. Since we know the CFA, the carrier frequency is deterministic. We then pass the signal through a lowpass filter. Afterward, we multiply the signal with the carriers again to un-shift the spectrum. The whole pipeline is reminiscent of the classical sinusoidal demodulation problem in, e.g., [122, Chatper 8].

To customize frequency selection for our problem, we make the lowpass filters trainable. Specifically, we use three layers of convolutional kernels of size 7×7 to reconstruct the vertical, horizontal, and diagonal chroma channels. During training, filters are regularized by enforcing the ℓ_1 norm of the filter coefficients such that it is equal to unity. This filter learning approach is more flexible than the minimum mean-squared error (MMSE) filter estimation such that those used in the classical literature [123] since it performs the filter estimation jointly with the luma and chroma denoising in an end-to-end training approach.

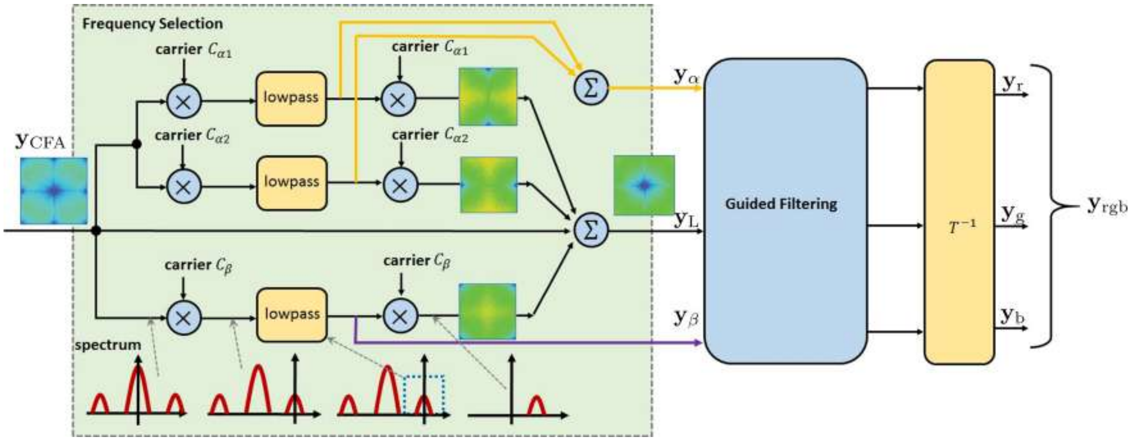


Figure 4.11. Implementation of the classical frequency selection. Given the input spectrum $\boldsymbol{x}_{\mathrm{CFA}}$, our goal is to remove the unwanted side-bands. By adopting the classical demodulation scheme, i.e., carrier+lowpass+carrier, we can recover the main lobe. The demodulated signals are $\boldsymbol{y}_{\alpha}$, $\boldsymbol{y}_{\beta}$ and $\boldsymbol{y}_{L}$. After post-processing (typically the luma-denoising) and coordinate transform $\boldsymbol{T}$, we retrieve the RGB signal.

Guided Filtering

The output of the frequency selection stage consists of the luma signal $\boldsymbol{y}_L$, and the two color signals $\boldsymbol{y}_\alpha$ and $\boldsymbol{y}_\beta$. All signals are corrupted by noise because during the frequency selection process, we have only decoupled the colors from the input and have not aggressively removed the noise. The objective of the guided filtering step is to denoise.

The rationale behind the guided filtering step is the different signal-to-noise ratios of $\boldsymbol{y}_L$, $\boldsymbol{y}_\alpha$, and $\boldsymbol{y}_\beta$. The luma signal $\boldsymbol{y}_L$, by definition, is the average of the RGB signals, i.e.,

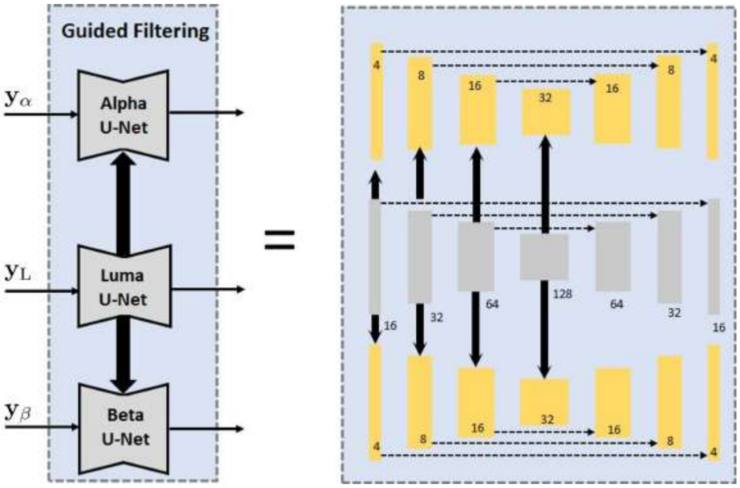


Figure 4.12. The proposed guided filtering. Guided filtering step consists of three UNets: The luma UNet, and two chroma UNets. Each UNet has a residual connection. Across the networks, we transfer knowledge from the luma channel to the chroma channels by concatenating features.

$\boldsymbol{y}_L = (\boldsymbol{y}_r + 2\boldsymbol{y}_g + \boldsymbol{y}_b)/4$. This averaging process suppresses the noise more than that of the color channels. Classical papers in color processing have long recognized this phenomenon, e.g., [124], [125]. Instead of independently denoising the three channels or jointly denoising the channels by treating them as a spectral volume, it is more promising to first denoise the luma channel, then use the recovered luma signal to guide the filtering of the chroma signals.

Building upon this intuition, we use three deep networks as shown in Figure 4.12. The luma denoising network is a standard UNet [126] with the number of layers as shown in the middle of Figure 4.12. On top of this luma network, we introduce two smaller UNets for the chroma channels. The size of the chroma channel UNet is 4 times less than that of the luma channel UNet. When training the networks, we pull the features generated by the encoder of the luma UNet, and concatenate with corresponding features generated by the chroma UNets. The chroma UNets are benefited from this feature sharing since they can use the high frequency information such as edges and textures from the luma denoiser. The layers of all the three UNets are convolutional, with a kernel size of 3×3. Following [127], [128], the number of feature channels at all image scales is kept fixed to avoid unnecessary enlargement of the network size. We replace the trainable transposed convolution layers in the standard UNet with non-trainable bilinear upsampling layers to reduce the total number of parameters [129].

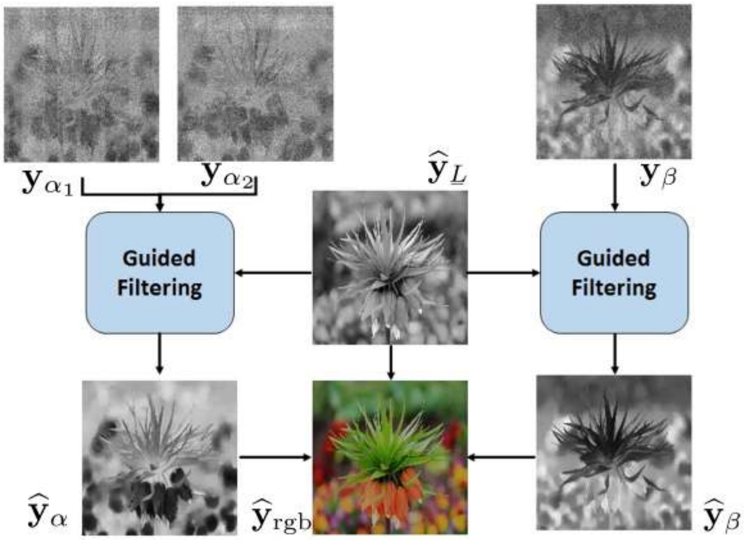


Figure 4.13. Visualization of the proposed guided filtering. Given the input $\boldsymbol{y}_{\alpha_1}$, $\boldsymbol{y}_{\alpha_2}$, $\boldsymbol{y}_\beta$ and the luma estimate $\widehat{\boldsymbol{y}}_L$, the guided filter leverages the luma estimate to reconstruct the chroma channels which are otherwise very difficult to recover. The resulting image is $\widehat{\boldsymbol{y}}_{\text{rgb}}$.

The effectiveness of the proposed guided filtering step can be visualized in Figure 4.13. In this example, we generate the input signal using a QIS imaging model with a mean photon arrival rate of 10 photoelectrons per pixel. Given the inputs $\boldsymbol{y}_{\alpha_1}$, $\boldsymbol{y}_{\alpha_2}$ and $\boldsymbol{y}_\beta$, straight-forward denoising on each of these channels independently will be extremely difficult because there is not much signal in the input. With guided filtering, since $\boldsymbol{x}_L$ preserves most of the details of the image, it serves as a strong prior to the chroma channels. As we can observe in Figure 4.13, the reconstructed chroma channels are substantially better than without the guided filtering.

Loss Functions

The overall system is trained end-to-end. To perform the training, we define the overall training loss as

$$\mathcal{L}(f_{\mathbf{\Theta}}) = \mathcal{L}_{\text{RGB}}(f_{\mathbf{\Theta}}) + \eta_1 \mathcal{L}_{\text{luma}}(f_{\mathbf{\Theta}}) + \eta_2 \mathcal{L}_{\text{chroma}}(f_{\mathbf{\Theta}}), \tag{4.19}$$

where $f_{\boldsymbol{\Theta}}$ denotes the model parameterized by $\boldsymbol{\Theta}$. There are three terms contributing to the overall loss.

The RGB loss is defined as the sum of a *mean absolute error* (MAE) loss and the *perceptual* loss. The MAE loss is the ℓ_1 difference between the predicted image $f_{\boldsymbol{\Theta}}(\boldsymbol{y}_{\text{CFA}})$ and the ground-truth image $\boldsymbol{x}_{rgb}$, whereas the perceptual loss is the ℓ_2 difference between the feature embedding using a pre-trained network [130]. The intuition is that if our network is performing well, then the features of the reconstructed image should be close to that of the clean image. Here, the features are obtained from a VGG-19 network, although other embeddings can also be used.

The luma and the chroma losses are defined as the ℓ_1 loss between the predicted luma and the ground truth luma, and the predicted chroma and the ground truth chroma, respectively. The hyper-parameters η_1, η_2 are chosen empirically to minimize the validation loss.

Implementation

The training of the proposed model is based on the WED database [131] which contains 4744 high-quality color images of natural scenes. Color images are mosaicked using Bayer CFA. We simulate an average of 1 to 10 photons per pixel (ppp) randomly for every patch. We also simulate readout noise by a zero-mean Gaussian with the standard deviation set to 0.25e^{-}. Images are then normalized to the range $[0, 1]$ by dividing by thrice the average photon count and clipping to $[0, 1]$.

In every training epoch, 128×128 patches are randomly cropped from each image, and data augmentation is performed by random flipping in the horizontal and vertical directions. Pairs of clean and noisy patches are fed into the network with batch size 64 for training. The network is trained for a total of 1000 epochs in mixed precision using NVIDIA GeForce RTX 2080 GPU with 8GBs of dedicated memory. Adam is used for optimization with a learning rate of 10^{-4} and 10^{-5} for the first and second 500 epochs, respectively.

Training hyper-parameters are $\eta_1 = 1$, $\eta_2 = 1$. The trainable lowpass filter $g(m, n)$ is modeled using a 7×7 kernel. The perceptual loss is based on MSE loss computed at 8th and 35th layers of the VGG-19 network. For validation during training, we compute the average PSNR and SSIM on the 18 color images of McMaster dataset [132] every 50 epochs.

4.3.3 Experiment

Evaluation Metric

We use the WED [131] database for training, the McMaster dataset [132] for validation, and 24 images from the Kodak dataset and 100 validation images of Div2k dataset [133] for testing. This ensures no overlap between the three different tasks. We simulate noisy images at signal levels $\{1, 2, \ldots, 10\}$ ppp, and we assume that the read noise is 0.25e$^-$ rms.

Since no single image quality metric can address all questions, we use the following suite of evaluation metrics:

- Peak signal to noise ratio (PSNR), which is the negative log of the mean squared error between the estimated image and the ground truth image. The mean squared error is computed per pixel per color. The is the usual PSNR extended to three color channels.

- Structure Similarity Index Metric (SSIM) [134], used to quantify the visual difference between two images.

- CIE 2000 color error metric, which measures the mean square color difference in the CIELAB color space as suggested by the CIE 2000 standard [135]. CIE 2000 metric captures better the color difference and is used in previous literature in color filter arrays [71], [72]. We use the implementation of skimage color module in [136] to compute the CIE 2000 metric.

- Learned Perceptual Image Patch Similarity (LPIPS) metric, proposed by Zhang et al. [137]. This metric measures the perceptual difference between 2 images using deep features. According to [137], this metric shows a good correlation with human perception. To compute the deep features, we use AlexNet.

Competing Methods

We compare the proposed method with the following classical and deep learning-based methods. We acknowledge other approaches proposed in the past and recent years. However, due to limited space, we have chosen to focus on a few representative ones.

- Classical frequency selection (FS) by Condat [123], [138]. This method uses a linear demodulation scheme by decorrelating the luma and chroma from the color data. The denoising step uses a follow up work by Condat and Mosaddegh [94] which applies total variation minimization. We apply Anscombe transform from [76] before the reconstruction to stabilize the noise variance.

- Classical optimization-based algorithm, using the alternating direction method of multiplier (ADMM). We use the implementation in [21]. The backbone optimization algorithm is the Plug-and-Play ADMM (PnP-ADMM) where we use BM3D as the denoiser [125].

- Demosaic-Net by Gharbi et al. [90]. This is a generic deep neural network solution. The method has a standard convolutional structure that takes a raw CFA color image and converts to a full RGB image. We used the code provided by the authors of [90], and trained the network from scratch using our noisy data.

- MM-Net by Kokkinos and Lefkimmiatis [99]. The method combines an explicit forward degradation model with a deep neural network. The algorithm iterates like classical optimization approaches, but each iteration is executed by a deep network. We used the code provided by the authors of [99], and trained from scratch using our noisy data and the pre-trained denoiser provided by the author.

Results of Synthetic Experiments

In this section, we report the results of synthetic experiments. We first offer some visual comparisons. Figure **4.14** shows a few snapshots of several testing images taken from the Div2K dataset. A few observations we can see from these images:

- Details: It is evident from the first three rows of the images that the classical FS [94] and the ADMM approach [21] have hallucinated the details whereas deep learning approaches tend to over smooth the details. The proposed method gives a more faithful recovery of details.

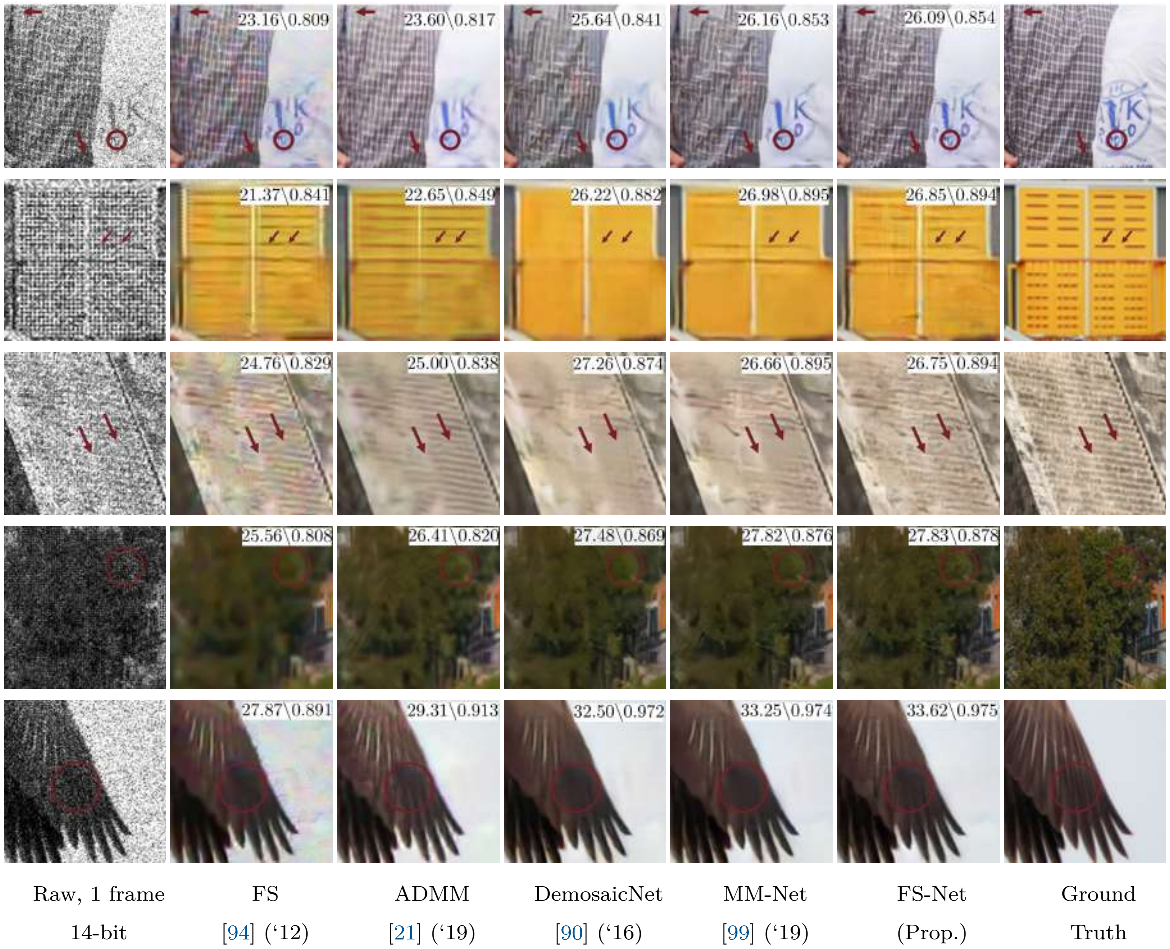
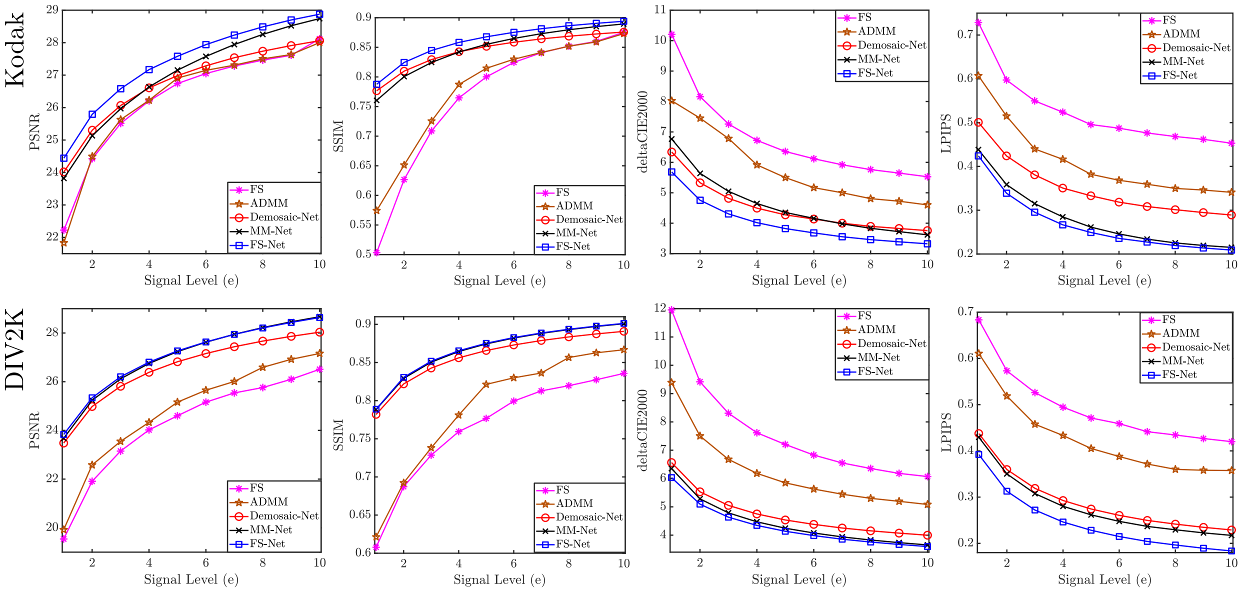


Figure 4.14. Synthetic experiments using data from the DIV2k dataset. We simulate the QIS data at an average of 10 photoelectrons per pixel. Quantitative image quality metrics PSNR\SSIM are shown on the top right of every image.

- Color: The proposed method has less false colors than classical methods. This is especially obvious when we compare the stair-case image and the feather image, where color bleeding of the classical methods is severe.

A quantitative comparison is shown in Figure 4.15, where we compared the PSNR, SSIM, CIE 2000, and LPIPS curves as a function of the photon level. We separately consider the Kodak dataset and the Div2K dataset so that we can see the generalization capabilities of the methods. We make some observations:

First of all, the performance of the competing methods is consistent for both datasets. In particular, we observe that learning-based methods are generally better than classical



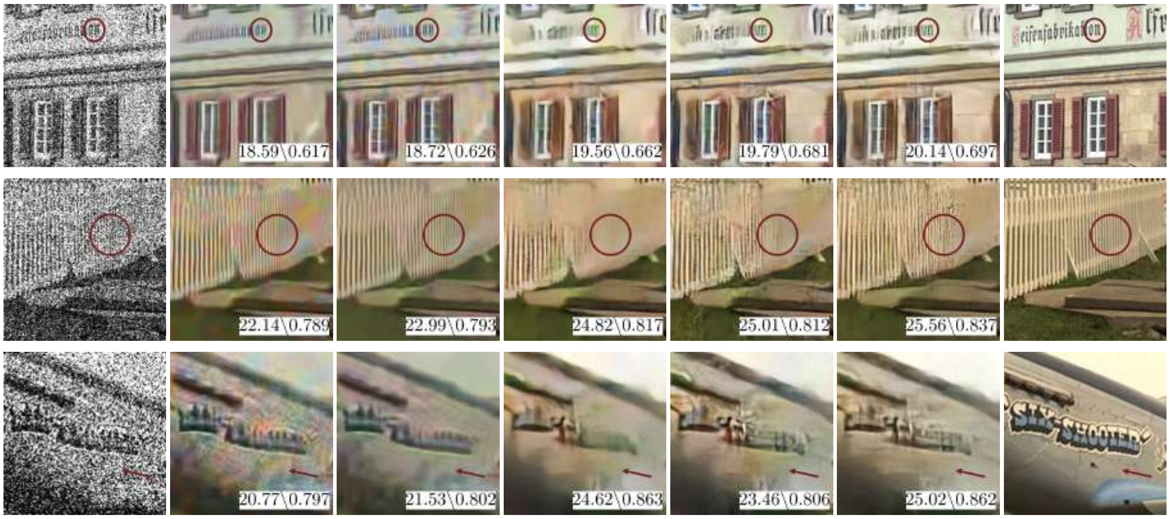
(a) PSNR (b) SSIM (c) CIE 2000 (d) LPIPS

Figure 4.15. Performance of synthetic QIS data with read noise = 0.25e$^-$ at different signal levels. The first row shows the results of using 24 images in the Kodak dataset, whereas the second row shows 100 images of the DIV2K dataset.

methods across different photon levels. The gap appears smaller at stronger photon levels (10 ppp). Learning-based methods are very competitive, especially between MM-Net and the proposed method, using PSNR and SSIM. However, as we have seen in the visual comparisons in Figure 4.14, similar PSNR values can be drastically different visual appearance. Further evidence can be seen from the CIE 2000 metric and the LPIPS metric. In both metrics, the proposed method has a more obvious gap compared to MM-Net and Demosaic-Net.

Second, as photon level drops, the gap between the proposed method and the competing methods becomes larger. This is particularly evident in the Kodak dataset where the proposed method is almost 0.5dB higher than MM-Net and Demosaic-Net in terms of PSNR. The visual comparison in Figure 4.16 confirms these numbers. The proposed FS-Net has a better recovery of both details and colors.

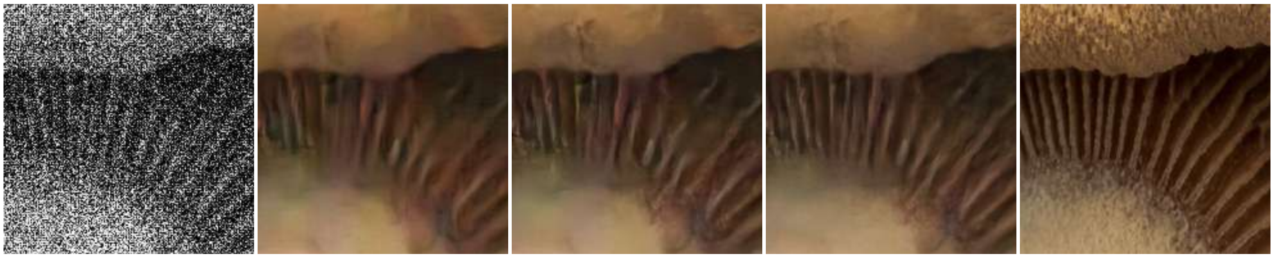
We also simulated a higher readout noise with standard deviation of 2e$^-$, which is similar to what we can get from a standard CIS, and an average signal level of 2 ppp. We ran the same networks used in the previous experiment: Demosaic-Net, MM-Net, and FS-Net on a



Raw, 1 frame 14-bit | FS [94] (2012) | ADMM [21] (2019) | DemosaicNet [90] (2016) | MM-Net [99] (2019) | FS-Net (Proposed) | Ground Truth

Figure 4.16. Visual evaluation using the Kodak synthetic dataset. The data is simulated at 2 ppp and read noise of 0.25 e^-. Observe the strong color noise in classical methods such as FS, and over-smoothing in learning-based methods. Quantitative image quality metrics PSNR\SSIM are shown on the top right of every image.

sample image from Div2K dataset. Figure 4.17 shows that FS-Net can still give more details and less color noise compared to other networks.



Raw, 1 frame 14-bit | Demosaic-Net [90] (2016) | MM-Net [99] (2019) | FS-Net (Proposed) | Ground Truth

Figure 4.17. Visual evaluation of using the Div2K synthetic dataset. The data is simulated at 2 ppp average signal level and $2e^-$ read noise.

As shown in Figure 4.18, we show the spatial and spectral representation of the isotropic Gaussian initialization for the low-pass filters as well as the learned filters for the three chrominance channels $\alpha 1$, $\alpha 2$ and β. We notice that the network tends to maximize the spectral support of the chrominance filters to make sure that no chrominance information is lost. Although a wider spectrum implies more noise, this noise is removed afterward in the guided denoising phase. At the same time, the filter response is zeroed-out at the positions of the luminance channel to avoid aliasing.

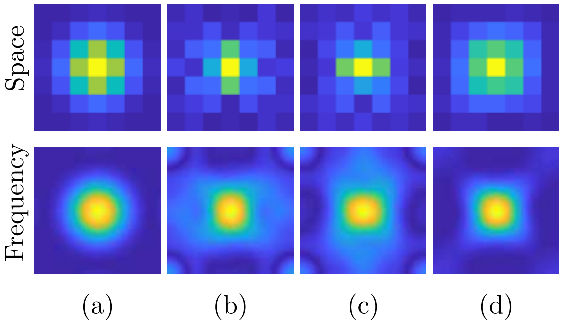


Figure 4.18. Low-pass filters learnt by FSNet/(a) Initial Gaussian estimate of low-pass filters and the learned filters by FSNet: (b) $g_{\alpha 1}$, (c) $g_{\alpha 2}$ and (d) g_{β}. First and second rows show the spatial and frequency representations, respectively.

Real QIS Data

In this section, we report our experimental results on real QIS data. To conduct the experiments, a QIS camera module developed by Gigajot Tech Inc was used. The sensor has a resolution of 1024 × 1024 and is equipped with a Bayer color filter array. During the experiments, short exposure times of 74us were used to limit the number of photoelectrons per pixel per frame. Eight (8) "ground truth" images were captured with a longer exposure time of 740us to minimize the impact of photon shot noise, i.e., the average signal level is approximately 10 times the short-exposure images. We then perform the standard linear demosaicing algorithm in [82] to recover the color, and then a simple BM3D denoising algorithm to remove the residual noise. We chose to use the same camera (instead of using a DSLR camera) to obtain the ground truths because this can minimize the impact generated by sensor characteristics and focus the comparison on the color reconstruction.

Figure 4.19 shows the snapshots of two scenarios. We only compare Demosaic-Net and MM-Net because the synthetic experiments showed that they are better than classical methods. When compared with these methods, we observe that they have more color reconstruction error. For example, in the "Expo" image, Demosaic-Net and MM-Net have severe color bleeding occurring near the cap of the green marker (See the zoom-in). They also have

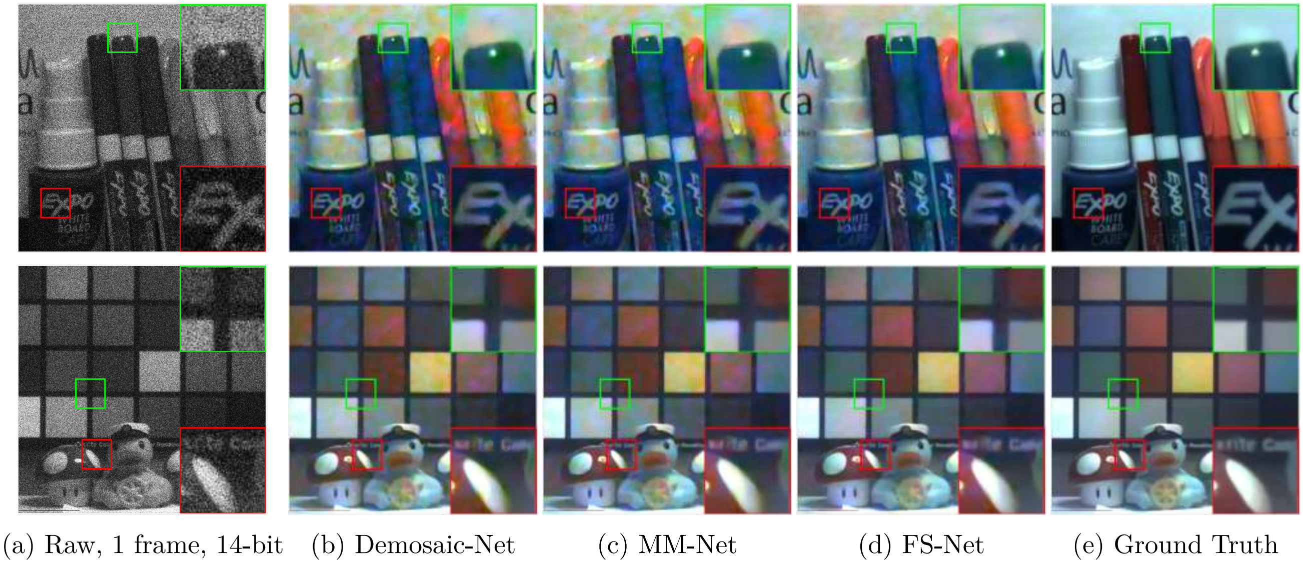


Figure 4.19. Real data reconstruction results. (a) Input, collected at 1.8 ppp (1st row) and 5.5 ppp (2nd row), (b) Demosaic-Net, (c) MM-Net, (d) Proposed FS-Net, and (e) Ground truth, obtained by very long-exposure, with post-processing using demosaicing and denoising.

more noise issues showing up in the text regions. In the "Duck" image, we observe similar problems where the color bleeding is severe in the background color chart.

To further evaluate the performance of the methods, we captured a real resolution chart using a QIS and reconstruct the image. Figure 4.20 shows the visual comparison. It is evident from the figures that Demosaic-Net and MM-Net have more false colors. Besides, the resolution that can be recovered by these two methods is less than that of the proposed FS-Net.

Finally, in Figure 4.21 we show the results using 2-bit and 3-bit data. As we can see, the performance of the proposed method remains competitive in these low bit-depth situations.

Computational Complexity To evaluate the network complexity, we compute the theoretical total number of multiply-add operations (MAC) for each network using code in [139]. We also compare the total number of parameters for each network to evaluate the required memory for the network weights.

Table 4.1 shows the comparison between Demosaic-Net, MM-Net with 2 iterations, and FS-Net. We notice that DemosiacNet has the least complexity since it processes a down-

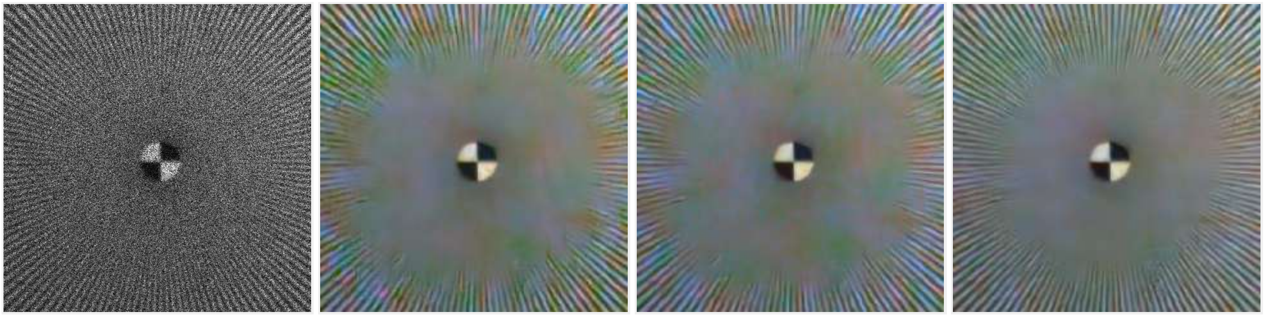


Figure 4.20. Comparison with resolution chart. Grayscale resolution chart captured with Integration time 224 μsec, lens aperture f/1.8 and average signal level per frame is 8.4 ppp. For ideal demosaicing, the result should not have any false colors.

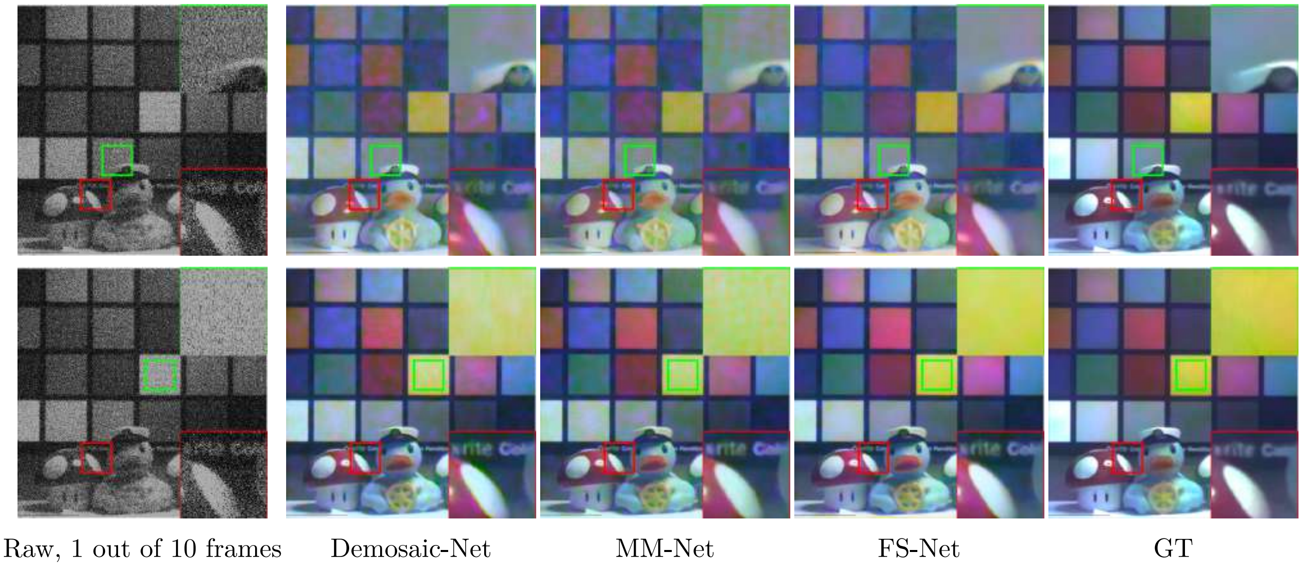


Figure 4.21. First row: 10 frames of 2-bit QIS data, average signal per frame: 1.86 ppp. 2nd Row: 10 frames of 3-bit QIS data, average signal per frame:3.8 ppp

scaled version of the image, and then do up-sampling at the end of the network. However, this down-scaling leads to a loss of resolution as we saw in visual comparison. MM-Net has the least number of parameters since it reuses the same network for multiple iterations. However, it has the highest complexity. FS-Net has comparable complexity and network size to Demosaic-Net while achieving superior image quality.

Table 4.1. Comparisons of network complexity and size

Method	Demosiac-Net	MM-Net	FS-Net
Complexity (GMAC)	2.20	12.44	2.76
# Parameters	524k	380k	546k

4.4 Final thoughts

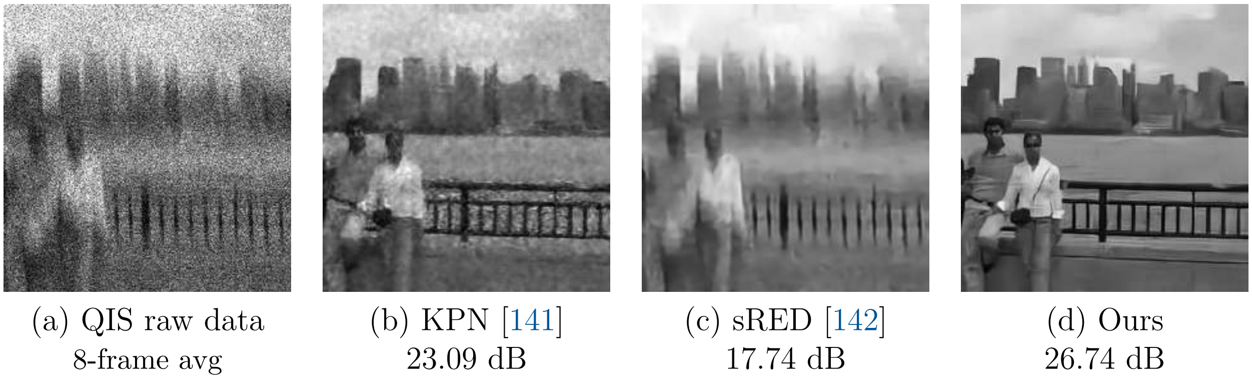
We have proposed two methods for demosaicing - one classical and one deep learning based. Both the methods take the physics in terms of noise and the color imaging using color filter arrays into consideration. We have seen that within the classes of methods they belong to, these methods perform better than their state-of-the-art counterparts.

4.4.1 Where to, from here?

We were able to leverage the small pixel sizes in this chapter for color imaging. However, smaller pixels come with a few drawbacks which we haven't dealt with. As we mentioned earlier in this chapter, one of them is the blur that diffraction introduces. To deal with diffraction, we need to think about how we can do super resolution based on the blur introduced. Another typical issue that shows in smaller pixels is crosstalk, which we defined in chapter 3. Crosstalk will only become worse as we start manufacturing even smaller pixels. It is important to understand how crosstalk works and come up with ways to fix. One way of mitigating crosstalk is to design color filter arrays (for e.g., [72]) which will inherently lead to better processing later. It will also be interesting to see how the demosacing algorithms look like for these new CFA. Furthermore, we need to co-design color imaging with other applications such as imaging motion, high dynamic range and deblurring which we will see later in this dissertation.

5. BURST RECONSTRUCTION WITH QUANTA IMAGE SENSORS

Imaging a moving scene is a difficult task. Burst imaging is one of the standard solutions that cameras utilize to solve this problem. The idea is that if we take multiple frames with a short integration time, each frame will be sharp but noisy. We can then deal with the noise by coming up with intelligent ways to combine multiple frames. The algorithm's performance in combining the burst into a single image generally takes a hit when the frames are too noisy. So, there is a tradeoff between capturing noisy images, which could be combined using an intelligent algorithm, and a blurred image, which could be deblurred [140].


(a) QIS raw data
8-frame avg

(b) KPN [141]
23.09 dB

(c) sRED [142]
17.74 dB

(d) Ours
26.74 dB

Figure 5.1. The dilemma of noise and motion. (a) A simulated QIS sequence at 2 photons per pixel (ppp), averaged over 8 frames. (b) Result of Kernel Prediction Network (KPN) [141], a burst photography method that handles motion. (c) Result of a single-frame image denoiser sRED [142] applied to the 8-frame avg. (d) Result of our proposed method.

When imaging a dynamic scene with a quanta image sensor (QIS), especially if we use low bit depth, the problem of dealing with noise and motion is unavoidable. So, we need to figure out solutions that can deal with noise and motion together. The dilemma here is that they are intertwined. To remove noise in a dynamic scene, we often need to either align the frames or construct a steerable kernel over the space-time volume. The alignment step is roughly equivalent to estimating optical flow [143], whereas constructing the steerable kernel is equivalent to non-local means [144], [145] or kernel prediction [141]. However, if the images are contaminated by noise, both optical flow and kernel prediction will fail. When this step fails, denoising will be difficult because we will not easily find neighboring patches for filtering.

There are QIS burst solutions that take hundreds of frames to make a reconstruction [146]. However, it is unclear how we can reconstruct using fewer frames ($<$ 10 frames). Existing algorithms in the denoising literature can usually only handle noise or motion. For example, the kernel prediction network (KPN) [141] can extract motion information from a dynamic scene, but its performance drops when noise becomes heavy. Similarly, the residual encoder-decoder networks REDNet [142] and DnCNN [147] are designed for static scenes. In Figure **5.1**, we show the results of a synthetic experiment. The results illustrate the limitations of the motion-based KPN [141] and the single-frame REDNet (sRED) [142]. Our goal is to leverage the strengths of both and develop a solution that can deal with fewer frames and still reconstruct an image with good enough quality.

5.1 Method

5.1.1 Student teacher learning

We will look at the working of student-teacher learning for classification and object detection in chapter 7. In this chapter, we are going to use it for image reconstruction. The underlying idea is to use teacher networks that can do more manageable tasks than the task at hand to distill knowledge into the network we want to be trained in. That is, we ask the specific question - *A kernel prediction network can handle clean image sequences well. A denoising network can handle static image sequences well. Is there a way we can leverage their strengths to address the dynamic low-light setting?*

Figure **5.2** describes our method. There are three players in this training protocol: a teacher for motion (based on kernel prediction), a teacher for denoising (based on image denoiser networks), and a student, which is the network we will eventually use. The two teachers are individually pretrained using their respective imaging conditions. For example, the motion teacher is trained using sequences of clean and dynamic contents, whereas the denoising teacher is trained using sequences of noisy but static contents. During the training step, the teachers will transfer their knowledge to the student. During testing, only the student is used.

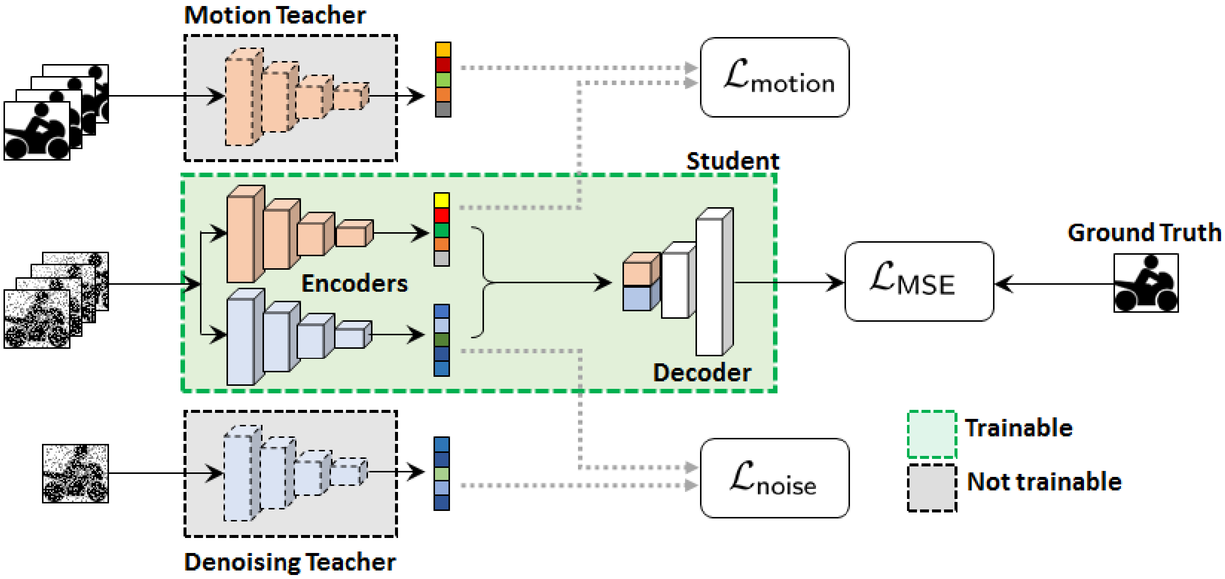


Figure 5.2. Overview of the proposed method. The proposed student-teacher setup consists of two teachers and a student. The motion teacher shares motion features, whereas the denoising teacher shares denoising features. To compare the respective feature differences, perceptual losses $\mathcal{L}_{\text{noise}}$ and $\mathcal{L}_{\text{motion}}$ are defined. The student network has two encoders and one decoder. The final estimates are compared with the ground truth using the MSE loss $\mathcal{L}_{\text{MSE}}$.

In order to transfer knowledge from two teachers, the student is designed to have two branches - one branch duplicating the architecture of the motion teacher and another branch duplicating the architecture of the denoising teacher. When training the student, we generate three versions of the training samples. The motion teacher sees training samples that are clean and only contain motion, $\boldsymbol{x}_{\text{motion}}$. The denoising teacher sees a training sample containing no motion but corrupted by noise, $\boldsymbol{x}_{\text{noise}}$. The student sees the noisy dynamic sequence $\boldsymbol{x}_{\text{motion+noisy}}$.

Because the student has identical branches to the teachers, we can compare the features extracted by the teachers and the student. Specifically, if we denote $\phi(\cdot)$ as the feature extraction performed by the motion teacher, $\widehat{\phi}(\cdot)$ the student motion branch, $\varphi(\cdot)$ the denoising teacher, and $\widehat{\varphi}(\cdot)$ the student denoising branch, then we can define a pair of *perceptual similarities*: the motion similarity

$$\mathcal{L}_{\text{motion}} = \|\underbrace{\widehat{\phi}(\boldsymbol{y}_{\text{motion+noisy}})}_{\text{motion student}} - \underbrace{\phi(\boldsymbol{y}_{\text{motion}})}_{\text{motion teacher}}\|^2 \tag{5.1}$$

and the denoising similarity

$$\mathcal{L}_{\text{noise}} = \|\underbrace{\widehat{\varphi}(\boldsymbol{y}_{\text{motion+noisy}})}_{\text{denoising student}} - \underbrace{\varphi(\boldsymbol{y}_{\text{noisy}})}_{\text{denoising teacher}}\|^2. \tag{5.2}$$

Intuitively, this pair of equations ensure that the features extracted by the student branches are similar to those extracted by the respective teachers. These are the features that can be extracted in good conditions. If this can be achieved, we will have a good representation of the noisy dynamic sample, and hence we can do a better reconstruction.

The two student branches can be considered as autoencoders that convert the input images to codewords. As shown on the right side of Figure 5.2, we have a "decoder" which translates the concatenated codewords back to an image. The loss function of the decoder is given by the standard mean squared error (MSE) loss:

$$\mathcal{L}_{\text{MSE}} = \|f(\boldsymbol{y}_{\text{motion+noisy}}) - \boldsymbol{y}_{\text{true}}\|^2, \tag{5.3}$$

where f is the student network and so $f(\boldsymbol{x}_{\text{motion+noisy}})$ denotes the estimated image. The overall loss function is the sum of these losses:

$$\mathcal{L}_{\text{overall}} = \mathcal{L}_{\text{MSE}} + \lambda_1 \mathcal{L}_{\text{motion}} + \lambda_2 \mathcal{L}_{\text{noise}}, \tag{5.4}$$

where λ_1 and λ_2 are tunable parameters. Training the network is equivalent to finding the encoders $\widehat{\phi}$ and $\widehat{\varphi}$, and the decoder f.

5.1.2 Choice of teacher and student networks

The proposed student-teacher framework is quite general. Specific to this paper, the two teachers and the student are chosen as follows.

The motion teacher is the kernel prediction network (KPN) [141]. We modify it by removing the skip connections to maintain the information kept by the encoder. In addition, we remove the pooling layers and the bilinear upsampling layers to maximize the amount of information being fed to the feature layer. With these changes, the KPN becomes a fully convolutional-deconvolutional network.

The denoising teacher we use is a modified version of REDNet [142], which is also used in another QIS reconstruction method [106]. To differentiate this single-frame REDNet and another modified version (to be discussed in the experiment section), we refer to this single-frame REDNet denoising teacher as sRED. Like the motion teacher, we remove the residual connections since they hurt the feature transfer in student-teacher learning.

The student network has two encoders and a decoder. The encoders have the same architecture as the teachers. The decoder is a stack of 15 layers where each layer is a 128-channel up-convolution. The entrance layer is used to concatenate the motion and denoising features.

5.2 Experiments

5.2.1 Setting

Training data. The training data consists of two parts. The first part is for *global motion*. We use the Pascal VOC 2008 dataset [148] which contains 2000 training images. The second part is for *local motion*. We use the Stanford Background Dataset [149] which contains 715 images with segmentation. We randomly crop patches of size 64×64 from the images to serve as ground truth for both datasets. An additional 500 images are used for validation. We shift the patches according to a continuous random camera motion to create global motion. The number of pixels traveled by the camera ranges from 7 to 35 across 8 consecutive frames. This is approximately 1 m/s. We fix the background and shift the foreground using translations and rotations for local motion. The translation implementation is the same as that of the global motion but applied to foreground objects. The rotation is implemented by rotating the object with an angle ranging from 0 to 15 degrees.

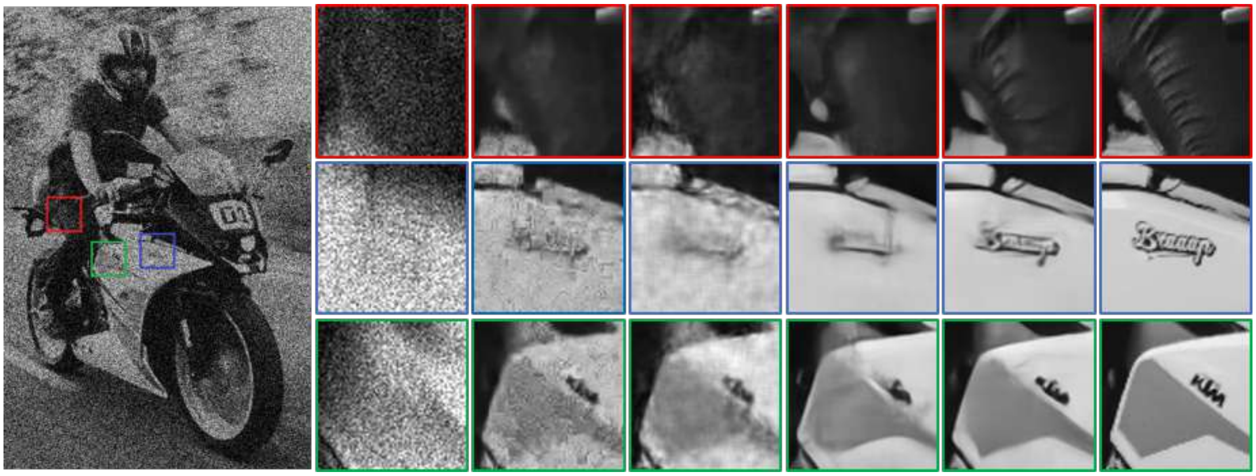
Training the teachers. The motion teacher is trained using a set of noise-free and dynamic sequences. The loss function is the mean squared error (MSE) loss suggested by [141]. The network is trained for 200 epochs using the dataset described above. The denoising teacher is trained using a set of noisy but static images. Therefore, for every ground-truth sequence, we generate a triplet of sequences: A noise-free dynamic sequence for the motion teacher, a noisy static image for the denoising teacher, and a noisy dynamic sequence for the student. We remark that such a data synthesis approach works for our problem because the simulated QIS data matches the statistics of real measurements.

Baselines. We compare the proposed methods with three existing dynamic scene reconstruction methods: (i) BM4D [150], (ii) Kernel Prediction Network (KPN) [141], and (iii) a modified version of REDNet [142]. Our modification generalizes REDNet to multi-frame inputs by introducing a 3D convolution at the input layer to pool the features. We refer to the modified version as multi-frame RED (mRED). Note that mRED has residual connections while sRED (denoising teacher) does not. We consider mRED a more fair baseline since it takes an input of 8 consecutive frames rather than a single frame. For KPN, the original method [141] suggested using a fixed kernel size of $K = 5$; we modify the setting by defining K as the maximum number of pixels traveled by the motion.

Implementation. All networks are implemented using Keras [151] and TensorFlow [152]. The student-teacher training is done using a semi-annealing process. Specifically, the regularization parameters λ_1 and λ_2 are updated once every 25 epochs such that λ_1 and λ_2 decay exponentially for the first 100 epochs. For the next 100 epochs, λ_1 and λ_2 are set to 0 and the overall loss function becomes $\mathcal{L}_{\text{overall}} = \mathcal{L}_{\text{MSE}}$.

5.2.2 Synthetic experiments

We begin by conducting synthetic experiments. We first visually compare the reconstructed images of the proposed method and the competing methods. Figure 5.3 shows some

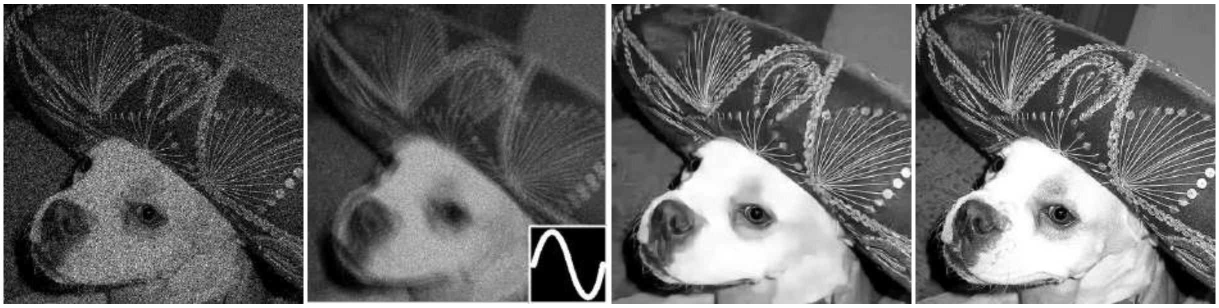


(a) QIS raw data, 1 frame (b) Avg of 8 frames (c) BM4D 23.04 dB (d) KPN 25.45 dB (e) mRED 26.42 dB (f) Ours 29.39 dB (g) Ground Truth

Figure 5.3. Simulated QIS data with linear global motion. (a) The raw QIS image is simulated at 2 ppp, with a global motion of 28 pixels uniformly spaced across 8 frames. (b) An average 8 QIS raw frames. (c) BM4D [150]. (d) KPN [141]. (e) mRED, a modification of REDNet [142]. (f) Proposed method. (g) Ground truth.

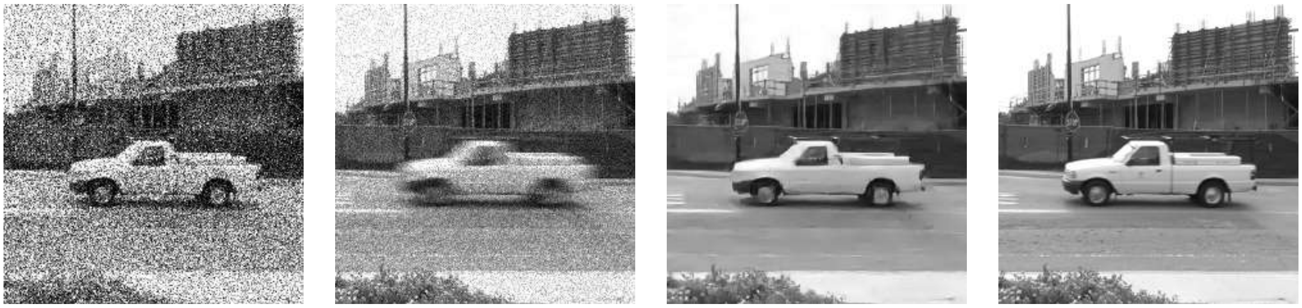
results using global translation. The motion magnitude is 28 pixels across 8 frames, at 2 ppp. Figure 5.4 shows some results using arbitrary global motion, at 4 ppp. The motion trajectory is shown in the inset in the figure. Figure 5.5 shows some results of local motion. We simulate QIS data with a real motion video of 30 fps. The photon level is 1.5 ppp. The average inference time of KPN on a 512 × 512 patch is 0.0886 seconds using an NVIDIA GeForce RTX 2080 Ti graphics card. For the same testing setting, mRED takes 0.0653 seconds, and the proposed method takes 0.1943 seconds. The average time for BM4D (MATLAB version) is 23.6985 seconds.

To quantitatively analyze the performance, we use the global linear motion to plot two sets of curves as shown in Figure 5.6. In the first plot, we show PSNR as a function of the motion magnitude. The magnitude of the motion is defined as the number of pixels traveled along the dominant direction over 8 consecutive frames. As shown in Figure 5.6(a), the proposed method has a consistently higher PSNR compared to the three competing methods, ranging from 1.5 dB to 3 dB. The higher PSNR suggests that the presence of both teachers has provided a positive impact on solving the motion and noise dilemma, which is



(a) QIS raw (b) avg 8 frames (c) Ours (d) Ground truth

Figure 5.4. Simulated QIS data with arbitrary global motion. (a) QIS raw data simulated at 4 ppp. The motion trajectory is shown in the inset. (b) Average of 8 frames. (c) Proposed method. (d) Ground truth.



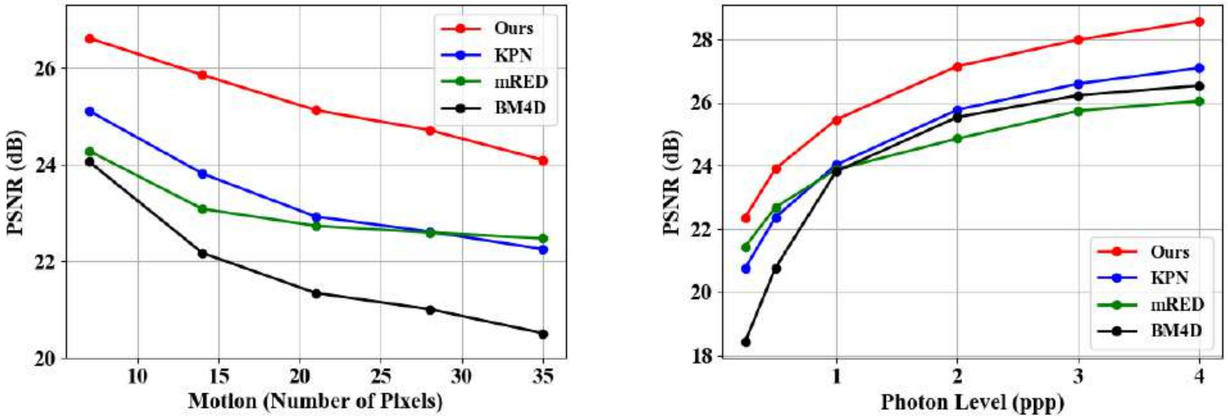
(a) QIS raw (b) avg 8 frames (c) Ours (d) Ground truth

Figure 5.5. Simulated QIS data with local motion. In this example, only the car moves. The background is static. (a) Raw QIS frame assuming 1.5 ppp. (b) The average of 8 QIS frames. (c) Proposed method. (d) Ground truth.

difficult for both KPN and mRED. The second set of curves is shown in Figure 5.6(b) and reports PSNR as a function of the photon level. The curves in Figure 5.6(b) suggest that for the photon levels we have tested, the performance gap between the proposed method and the competing methods is consistent, which provides additional evidence of the effectiveness of the proposed method.

5.2.3 Real experiments

We verify the results using real QIS data. The real data is collected using a prototype Gigajot PathFinder camera [46]. The camera has a spatial resolution of 1024 × 1024. The integration time of each frame is 75 μs. Each reconstruction is based on 8 consecutive QIS



(a) PSNR vs. Motion
at photon level of 2 ppp

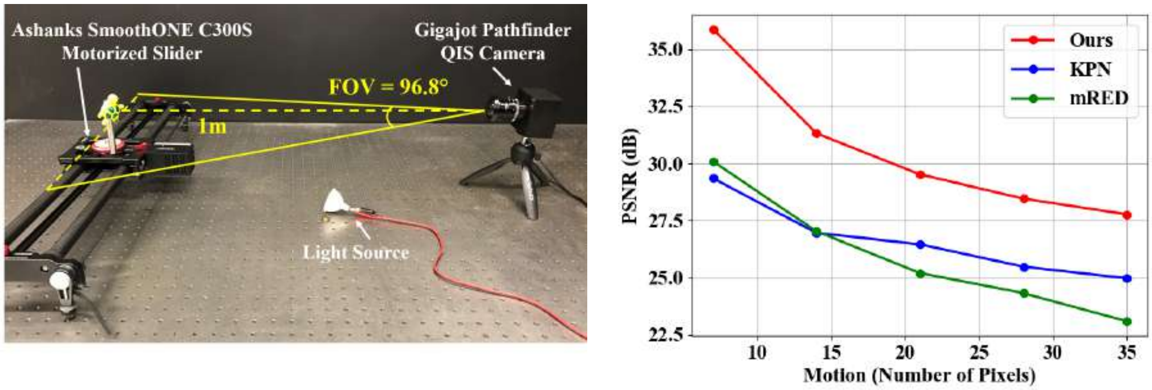
(b) PSNR vs. Photon Level
at motion magnitude of 4 pixels

Figure 5.6. Quantitative analysis using synthetic data. (a) PSNR as a function of the motion magnitude, at a photon level of 2 ppp. The magnitude of the motion is defined as the number of pixels traveled along the dominant direction, over 8 consecutive frames. (b) PSNR as a function of photon level. The motion magnitude is fixed at 4 pixels, but the photon level changes. Our method consistently outperforms BM4D [150], KPN [141], and mRED (a modified version of [142]).

frames. When this experiment was conducted, the readout circuit of this camera was still a prototype that was not optimized for speed. Thus, instead of demonstrating a real high-speed video, we capture a slowly moving real dynamic scene where the motion is continuous but slow. We make the exposure period short so that it is equivalent to a high-speed video. We expect the problem to be solved in the next generations of QIS.

The physical setup of the experiment is shown in Figure 5.7(a). We put the camera approximately 1 meter away from the objects, and a light source controls the photon level. The objects are mounted on an Ashanks SmoothONE C300S motorized camera slide to create motion, which allows us to control the location of the objects remotely. The "ground truth" (reference images) in this experiment is obtained by capturing a static scene via 8 consecutive QIS frames. Since these static images are noisy (due to photon shot noise), we apply mRED to denoise the images before using them as references.

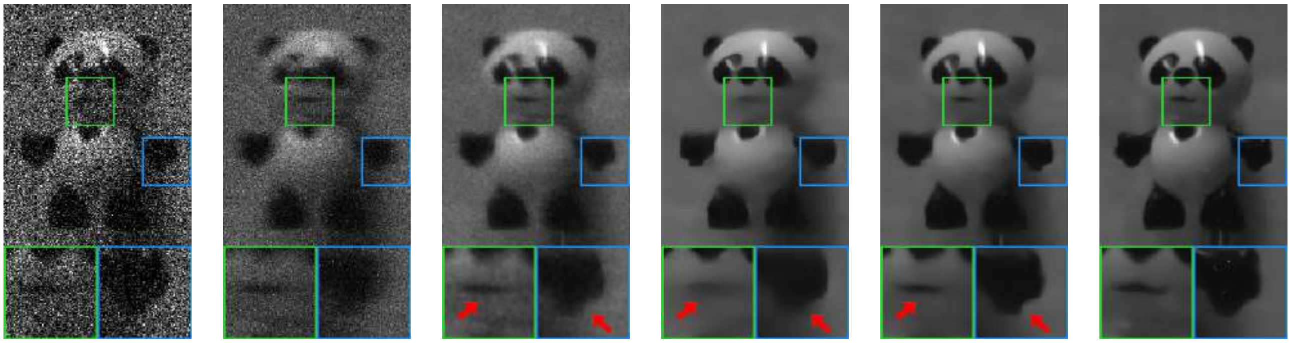
A visual comparison for this experiment is shown in Figure 5.8. The quantitative analysis is shown in Figure 5.7(b), where we plot the PSNR curves as functions of the number of



(a) Experimental Setup

(b) PSNR vs. motion (pixels)

Figure 5.7. **(a) Setup of QIS data collection**. The QIS camera is placed 1 meter from the object attached to a motorized slider. The lens's horizontal field of view (FOV) is 96.8°. The motion is continuous but slow. **(b) Quantitative analysis on real data**. The plot shows the PSNR values as a function of the motion magnitude under a photon level of 0.5 ppp. The "reference" in this experiment is determined by reconstructing an image using a stack of static frames of the same scene. The reconstruction method is based on [106]



(a) QIS raw 1 frame

(b) Average 8 frames

(c) KPN 25.08 dB

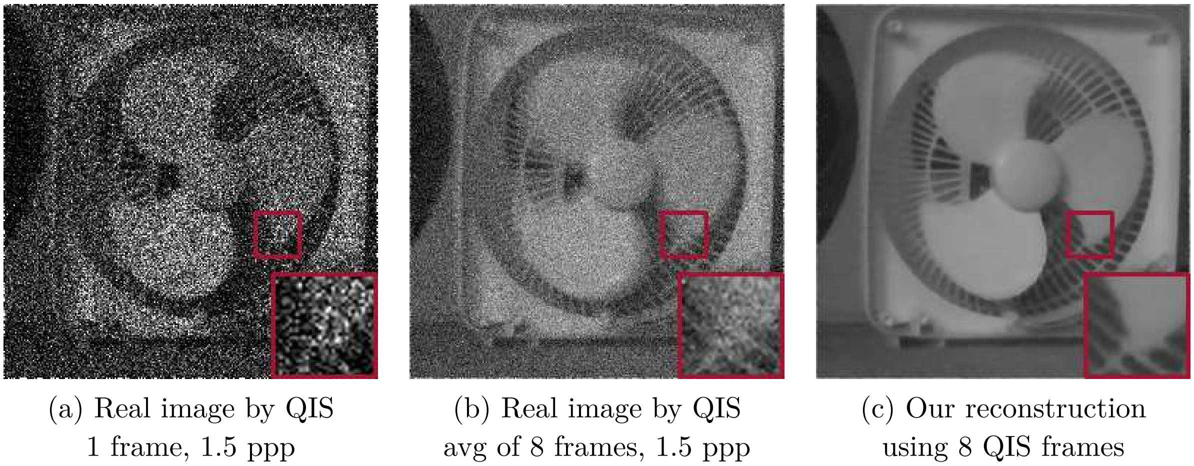
(d) mRED 25.33 dB

(e) Ours 30.97 dB

(f) Reference

Figure 5.8. **Real QIS data**. (a) A snapshot of a real QIS frame captured at 2 ppp per frame. The number of pixels traveled by the object over the 8 frames is 28 pixels. (b) The average of 8 QIS frames. Note the blur in the image. (c) Reconstruction result of KPN [141]. (d) Reconstruction result of mRED, a modification of [142]. (e) Our proposed method. (f) Reference image is a static scene denoised using mRED.

pixels traveled by the object. As we can see, the performance of the proposed method and the competing methods are similar to those reported in the synthetic experiments. The gap appears to be consistent with the synthetic experiments. An additional real data experiment is shown in Figure 5.9, where we use QIS to capture a rotating fan scene.


(a) Real image by QIS
1 frame, 1.5 ppp

(b) Real image by QIS
avg of 8 frames, 1.5 ppp

(c) Our reconstruction
using 8 QIS frames

Figure 5.9. Real QIS data with rotational motion. The image is captured at 1.5 ppp. Note the motion blur in the 8-frame average.

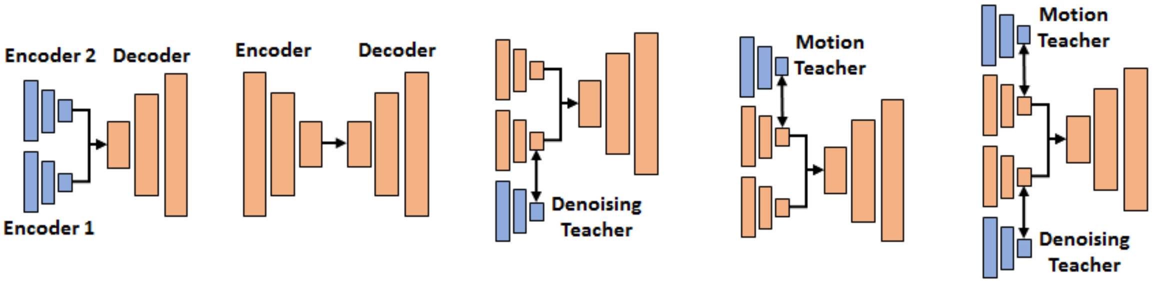
5.2.4 Ablation study

We conducted an ablation study to evaluate the significance of the proposed student-teacher training protocol. Figure 5.10 summarizes the 5 configurations we study. Config A is a vanilla baseline where the denoising and motion teachers are pretrained. Config B uses a single encoder instead of two encoders. Ours-I uses a student-teacher set up to train the denoising encoder. Ours-II is similar to Ours-I, but we use the motion teacher in place of the denoising teacher. Ours-full uses both teachers. All networks are trained using the same set of noisy and dynamic sequences. The experiments are conducted using synthetic data at a photon level of 1 ppp and a motion of 28 pixels across 8 frames. The results are summarized in Table 5.1.

Is student-teacher training necessary? Configurations A and B do not use any teacher. Compared to Ours-full, the PSNR values of Config A and Config B are worse by more than 1dB. Even if we compare with a single teacher, e.g., Ours-I, it is still 0.8dB ahead of Config B, which implies that the student-teacher training protocol positively impacts performance.

Table 5.1. Ablation study results. This table summarizes the influence of different teachers on the proposed method. The experiments are conducted using synthetic data, at a photon level of 1 ppp and a motion of 28 pixels along the dominant direction.

Configuration	# of Encoders	Which Teacher?	Test PSNR
A	2	None	21.51 dB
B	1	None	22.74 dB
Ours-I	2	Denoising	23.53 dB
Ours-II	2	Motion	23.65 dB
Ours-full	2	Both	23.87 dB



(a) Config A (b) Config B (c) Ours-I (d) Ours-II (e) Ours-full

Figure 5.10. Configurations for ablation study. (a) Config A: Uses pretrained teachers. (b) Config B: Uses a single encoder instead of two smaller encoders. (c) Ours-I: Uses denoising teacher only. (d) Ours-II: Uses motion teacher only. (e) Our-full: The complete model. In this figure, blue layers are pretrained and fixed. Orange layers are trainable.

Do teacher encoders extract meaningful information? Config A uses two pretrained encoders and a trainable decoder. The network achieves 21.51dB, which means that some features are helpful for reconstruction. However, compared with Ours-full, it is substantially worse (23.87dB compared to 21.51dB). Since the network architectures are identical, the performance gap is likely caused by the training protocol, indicating that the student-teacher setup is better for transferring knowledge from teachers to a student network.

Which teacher to use? Configurations Ours-I and Ours-II both use one teacher. The results suggest that if we only use one teacher, the motion teacher has a slight gain (0.1dB) over the denoising teacher. However, if we use both teachers as the proposed method, we observe another 0.2dB improvement. Thus, the presence of both teachers is helpful.

5.3 Final thoughts

We have proposed a deep learning based burst image processing algorithm that can take 8 QIS frames as input and produce a clean denoised image. The method uses student-teacher learning. Using student-teacher learning, we were able to bring together the best of two worlds - 1. noiseless burst reconstruction and 2. static denoising. We use teachers that are trained for each of these specific tasks separately. The teachers distill the knowledge into the student network to deal with both motion and noise at the same time. We have shown that the proposed method can outperform the existing state-of-the-art.

5.3.1 Where to, from here?

We have demonstrated that even with as few as 8 frames, we can reconstruct a good image when the scene is moving and at low light. However, the proposed method is too rigid and will not work for other settings. It is essential to make the proposed method adaptive to any number of frames and bit-depths. It will be interesting to see how we can combine the dynamic scene reconstruction with the color reconstruction we saw in chapter 4. Another critical aspect of the problem is that these low bit-depth reconstruction methods need fast output rates, for which we need to make the methods lightweight to improve the inference time. The student-teacher training scheme we have introduced here is an exciting idea. We will later see how we can utilize it for classification and object detection in chapter 7. Based on [153], the student-teacher learning also can shrink networks which could be used to achieve the goal of reducing the inference time.

6. LOW LIGHT NON-BLIND DEBLURRING

Image deblurring is a classical restoration problem where the goal is to recover a clean image from an image corrupted by a blur due to motion, camera shake, or defocus. In the simplest setting assuming a spatially invariant blur, the forward image degradation problem is

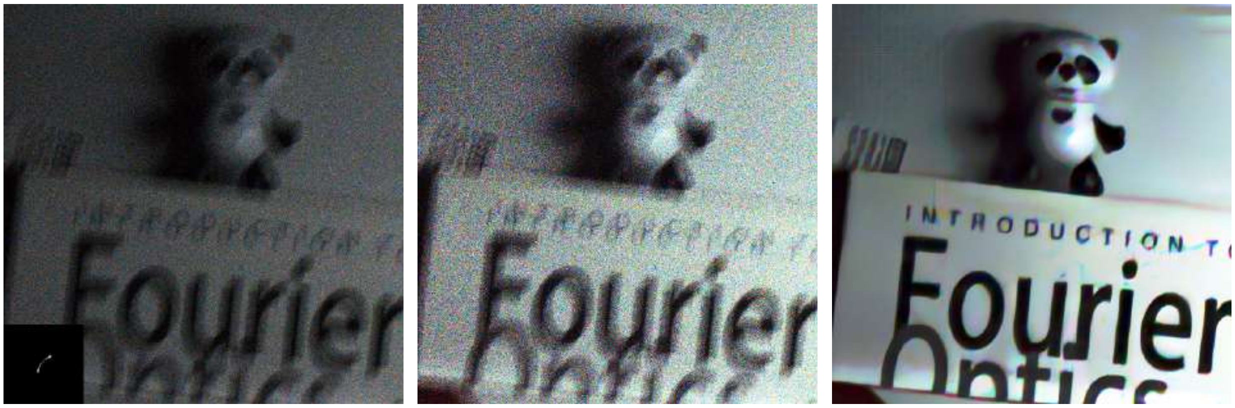
$$\boldsymbol{y} = \boldsymbol{h} * \boldsymbol{x} + \boldsymbol{\eta}, \tag{6.1}$$

where $\boldsymbol{x} \in \mathbf{R}^N$ is the clean image to be recovered from the corrupted image $\boldsymbol{y} \in \mathbf{R}^N$, the vector $\boldsymbol{h} \in R^d$ denotes the blur kernel, $\boldsymbol{\eta} \in \mathbf{R}^N$ denotes the additive i.i.d Gaussian noise, and "$*$" denotes the convolution operator. The deblurring problem can be further classified as *non-blind* and *blind*. A non-blind deblurring problem assumes that the blur kernel $\boldsymbol{h}$ is known whereas a blind-deblurring problem do not make such an assumption. In this work, we focus on the non-blind case.

While non-blind deblurring methods are abundant [154]–[159], the majority are designed for well-illuminated scenes where the noise is i.i.d. Gaussian and the noise level is not too high. However, as one pushes the photon level low enough that the photon shot noise dominates, the deblurring task is no longer as simple. As illustrated in Figure 6.1 which is a real low-light example we captured using a Canon T6i camera at a photon level approximately 5 lux, the observed image is not only dark but is strongly contaminated by photon shot noise (which is visible in the histogram equalized image). Given the blur kernel, our goal is to recover the image.

The operating regime of the proposed method is illustrated in Figure 6.2 we use another real low-light image to compare this work and other mainstream deblurring algorithms. We highlight the raw sensor capture (shown in the bottom left of each sub-figure) and the tone-mapped image (shown in the top right of each sub-figure) at different illumination levels. Our algorithm is specifically designed for an illumination level of 1 lux or lower.

Under such a severe lighting condition, state-of-the-art algorithms have a hard time working. In Figure 6.3 we use the deep Wiener deblurring network [158] to deblur the image. When the illumination is strong, the method performs well. But when the illumination is



(a) Raw camera image (b) Histogram equalized image (c) Our reconstruction

Figure 6.1. Overview. The goal of this work is to present a new algorithm that reconstructs images from blur at a photon-limited condition.

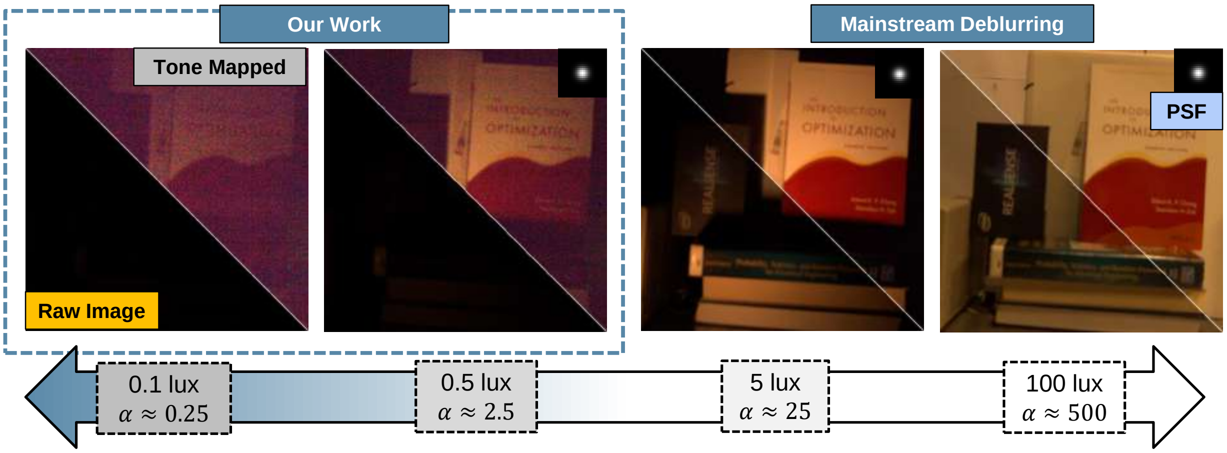


Figure 6.2. Comparison of photon-limited scenes (Left) with relatively well illuminated scenes (Right). Raw images and their tone mapped versions taken in different illuminations and blurred by defocus are shown in the figure. As illumination of the scene decreases, the photon shot noise becomes more dominant, making the deblurring problem substantially more difficult - as shown in Figure 6.3. In this work, we address the problem of non-blind deblurring in a *photon-limited* setting i.e. when the number of photons captured by the sensor is low leading to corruption of images by the photon shot noise.

weak, the algorithm performs poorly. We remark that this observation is common for many mainstream deblurring algorithms.

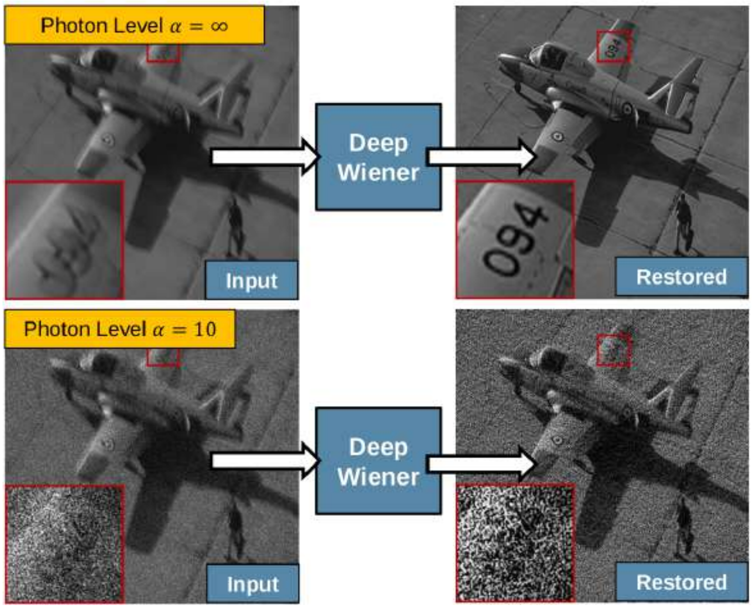


Figure 6.3. Limitation of existing image deblurring algorithms when applied to low-light images. In this example we use the pre-trained neural network [158] to recover a well-illuminated scene and a poorly-illuminated scene. The method fails because of the noise, even though the deblurring in a well-illuminated scene is satisfactory.

The present problem is best described as *photon limited* non-blind deblurring. It is a common problem for a variety of applications such as microscopy [160] and astronomy [161]. One should note that photon-limited imaging is a problem even if we use a perfect sensor with zero read noise and 100% quantum efficiency. The photon shot noise still exists due to the stochasticity of the photon arrival process [48]. Therefore, the solution presented in this work is pan-sensor, meaning that it can be applied to the standard CCD and CMOS image sensors and the more advanced quanta image sensors (QIS) [21], [23], [26].

6.0.1 Problem formulation

Consider a monochromatic image $\boldsymbol{x} \in \mathbf{R}^N$ normalized to $[0, 1]$. We write the blurred image as $\boldsymbol{Hx}$ where $\boldsymbol{H} \in R^{N \times N}$ represents the blur kernel $\boldsymbol{h}$ in the matrix form. In photon-limited conditions, the observed image is given by

$$\boldsymbol{y} = \text{Poisson}(\alpha \cdot \boldsymbol{Hx}), \tag{6.2}$$

where Poisson$(\cdot)$ denotes the Poisson process, and α is a scalar to be discussed. The likelihood of the observed image $\boldsymbol{y}$ follows the Poisson probability distribution:

$$p(\boldsymbol{y} \mid \boldsymbol{x}; \alpha) = \prod_{\mathrm{j}=1}^{N} \frac{[\alpha \boldsymbol{Hx}]_{\mathrm{j}}^{\boldsymbol{y}_{\mathrm{j}}} \mathrm{e}^{-[\alpha \boldsymbol{Hx}]_{\mathrm{j}}}}{\boldsymbol{y}_{\mathrm{j}}!}, \tag{6.3}$$

where $[\,\cdot\,]_{\mathrm{j}}$ denotes the jth element of a vector. The scalar α represents the photon level. It is a function of the sensor's properties (e.g. quantum efficiency), camera settings (exposure time, aperture), and illumination level of the scene. For a given illumination, the photon level α can be increased by increasing the exposure time or the aperture. To give readers a better idea of the photon level α, we give a rough estimate of the photon flux (measured in terms of lux level) in Table 6.1 under a few typical imaging scenarios.[1]

Table 6.1. Lighting condition and illumination level

Lighting condition	Illumination (lux)
Sunset	400
Dimly-lit Street	20-50
Moonlight	1
$\alpha = 5$ (This work)	1

[1] ↑To estimate the photon level α from the photon flux level, we set the scene illumination to 1 lux (measured using a light meter) and measure the corresponding photons-per-pixel from the image sensor data captured using a Canon EOS Rebel T6i.

6.0.2 Contributions and scope

Photon-limited non-blind deblurring is a special case of the Poisson linear inverse problem. We limit the scope to deblurring so that we can demonstrate the algorithm using real low-light data.

Existing photon-limited deblurring methods are mostly deterministic [162]–[164]. To overcome the limitation of these methods, in this work we present a deep-learning solution. We make two contributions:

1. We propose an unrolled plug-and-play (PnP [115], [165]) algorithm for solving the non-blind deblurring problem in *photon-limited* conditions. Unlike existing work such as [166] which uses an inner optimization to solve the Poisson proximal map, we use a three-operator splitting technique to turn all the sub-routines differentiable. This allows us to train the unrolled network end-to-end (which is previously not possible), and hence makes us the first unrolled network for Poisson deblurring.

2. We overcome the difficulty of collecting *real* photon-limited motion blur kernels and images for algorithm evaluation. A dataset containing 30 low-light images and the corresponding blur kernels are produced. We make this dataset publicly available.

6.1 Related Work

6.1.1 Poisson deconvolution

Poisson deconvolution has been studied for decades because of its important applications [167]. One of the earliest and the most cited works is perhaps the Richardson-Lucy (RL) algorithm [163], [164]. The method assumes a known blur kernel and derives an iterative scheme which converges to the maximum-likelihood estimate (MLE) of the deconvolution problem. The RL algorithm was applied to problems such as emission tomography [168] and confocal microscopy [169], [170]. However, since the prior is not used, the quality of reconstruction is limited.

Another class of iterative methods is based on maximum-a-posteriori (MAP) estimation by using a signal prior. For example, PIDAL-TV [171] solves a MAP cost function with the total-variation (TV) regularization using an augmented Lagrangian framework. Similarly, the sparse Poisson intensity reconstruction algorithm (SPIRAL) [172] looks for sparse solutions in an orthonormal basis, whereas [173] solves a MAP cost function with multiscale prior using the expectation-maximization algorithm.

Shrinkage based approaches such as PURE-LET [162] assume the deconvolution output to be a linear combination of elementary functions and minimize the expected mean squared error under a joint Poisson-Gaussian noise model. This boils down to solving a linear system of equations and has been also used to solve denoising, deblurring processes under Gaussian noise assumptions [174], [175].

Denoising under Poisson noise conditions can be viewed as a special case of the deblurring problem. One of the widely used techniques for Poisson denoising is the variance stabilizing transforms (VST) which applies the Anscombe transform [112] to stabilize the spatially varying noise variance. A standard denoising method is then used, followed by the inverse Anscombe transform. In [176], it was shown that an optimal inverse transform can outperform other standard Poisson denoising methods such as [177], [178]. The method in [110] provides an iterative version of the denoising via VST scheme by treating last iteration's denoised image as scaled Poisson data.

6.1.2 Plug-and-play

The Plug-and-play (PnP) framework was first introduced in [115] as a general purpose method to solve inverse problems by leveraging an off-the-shelf denoiser. Since then, the framework has been applied to different problems like bright field electron tomography [179] and magnetic resonance imaging (MRI) [180]. Using the same principle but with the half-quadratic splitting scheme, [181] demonstrated the use of a single denoiser for different image restoration tasks such as super-resolution, deblurring, and inpainting. Variations of PnP have also been used for Poisson deblurring [166], [182] and non-linear inverse problems [183]. A stochastic version of the scheme (PnP stochastic proximal gradient method) has been proposed for inverse problems with prohibitively large datasets [184]. Using the consensus equilibrium (CE) framework [185], the scheme can be extended to fuse multiple signal and sensor models.

The convergence of the Plug-and-Play scheme has been studied in detail. For example, [165] provided a variation of the scheme which was provably convergent under the assumptions of a bounded denoiser and its performance was analysed under assumptions of a graph filter denoiser in [186]. [187] showed that if a denoiser satisfies certain Lipshitz conditions, the corresponding Plug-and-Play scheme can be shown to converge. Furthermore, the authors proposed real-spectral normalization as a way to impose the conditions on deep-learning based denoisers.

A closely related method which provides a framework to solve inverse problems using denoisers is REgularization by Denoising (RED) [188], [189]. The framework poses the cost function for an inverse problem as sum of a data term and image-adaptive Laplacian regularization term. This allows the resulting iterative process to be written as a series of denoising steps. In [190], it was mentioned that for RED to be valid the denoiser needs to have a symmetric Hessian.

6.1.3 Algorithm unrolling

The difficulty of running PnP and RED is that they need to iteratively use a deep network denoiser. An alternative way to implement the algorithm was proposed by Gregor and LeCun in 2010 [191] to unroll an iterative algorithm and train it in a supervised manner. For example, one can unroll the iterative shrinkage threshold algorithm (ISTA) for the purpose of approximating sparse codes of an image. The idea of unrolled networks has been employed in various image restoration tasks such as super-resolution [192], deblurring [193], [194], compressive sensing [195], and haze removal [196]. For a more extensive review of algorithm unrolling, we refer the reader to [197]. More recently, there are new attempts to relax the fixed iteration structure of unrolling by analyzing the equilibrium of the underlying operators [198] .

As stated in [197], unrolling iterative algorithms provide multiple advantages compared to generic deep learning architectures. For example, the unrolled networks provide greater interpretability and are often parameter efficient compared to their counterparts such as the U-Net [126]. Since the networks are unrolled version of iterative algorithms, they are less susceptible to problem of overfitting.

6.2 Method

6.2.1 Algorithm unrolling

The proposed solution for the Poisson deblurring problem is to unroll the iterative PnP algorithm. The idea is that we start by deriving the PnP iterative update steps. In the "unrolled" version of the iterative algorithm, each iteration is treated as a computing block. Each computing block has its own set of trainable parameters. The blocks are concatenated in series with each other. The output at the end of the last block is used as the target for a supervised loss to fine-tune the trainable parameters.

Before describing the iterative algorithm we aim to unroll, we briefly describe the underlying cost function. Most inverse problem algorithm aim to determine the MAP estimate of the underlying signal $\boldsymbol{x}$ by maximizing the log-posterior

$$\boldsymbol{x}^* = \underset{\boldsymbol{x}}{\operatorname{argmax}} \Big[\log p(\boldsymbol{y} \mid \boldsymbol{x}) + \log p(\boldsymbol{x}) \Big], \tag{6.4}$$

where $p(\boldsymbol{x})$ denotes the natural image prior. Plugging (6.3) in (6.4) and taking the negative of the cost function, the maximization becomes

$$\boldsymbol{x}^* = \underset{\boldsymbol{x}}{\operatorname{argmin}} \Big[\alpha \mathbf{1}^T \boldsymbol{H}\boldsymbol{x} - \boldsymbol{y}^T \log(\alpha \boldsymbol{H}\boldsymbol{x}) - \log p(\boldsymbol{x}) \Big], \tag{6.5}$$

where $\mathbf{1}$ represents the all-one vector. Note that the factorial term $\log \boldsymbol{y}!$ has been dropped since it is independent of $\boldsymbol{x}$. The prior $p(\boldsymbol{x})$ has not been explicitly specified yet and this issue will be addressed through the use of a denoiser in the next subsection.

6.2.2 Conventional PnP for Poisson inverse problems

Now we describe how the Plug-and-Play method can be applied to the Poisson deblurring problem. We start with the alternate direction of method of multipliers (ADMM) [199] formulation – where we convert the unconstrained optimization problem to a constrained optimization problem by performing the variable splitting $\boldsymbol{x} = \boldsymbol{z}$

$$\begin{aligned} \{\boldsymbol{x}^*, \boldsymbol{z}^*\} &= \underset{\boldsymbol{x},\boldsymbol{z}}{\operatorname{argmin}} \Big[-\log p(\boldsymbol{y} \mid \boldsymbol{x}) - \log p(\boldsymbol{z}) \Big], \\ &\text{subject to } \boldsymbol{x} = \boldsymbol{z}, \end{aligned} \tag{6.6}$$

At the minimum of the above optimization problem, the constraint $\boldsymbol{x}^* = \boldsymbol{z}^*$ must be satisfied and hence the constrained optimization solution is equivalent to the unconstrained solution in (6.5).

The augmented Lagrangian associated with the constrained problem in (6.6) is

$$\begin{aligned} \{\boldsymbol{x}^*, \boldsymbol{z}^*, \boldsymbol{u}^*\} = \underset{\boldsymbol{x},\boldsymbol{z}}{\operatorname{argmin}} \Big[&\alpha \mathbf{1}^T \boldsymbol{H}\boldsymbol{x} - \boldsymbol{y}^T \log(\alpha \boldsymbol{H}\boldsymbol{x}) \\ &- \log p(\boldsymbol{z}) + \frac{\rho}{2}\|\boldsymbol{x} - \boldsymbol{z} + \boldsymbol{u}\|^2 - \frac{\rho}{2}\|\boldsymbol{u}\|^2 \Big], \end{aligned} \tag{6.7}$$

where $\boldsymbol{u}$ denotes the scaled Lagrange multiplier corresponding to the constraint $\boldsymbol{x} = \boldsymbol{z}$, and ρ denotes the penalty parameter. The corresponding iterative updates are:

$$\boldsymbol{x}^{k+1} = \underbrace{\underset{\boldsymbol{x}}{\operatorname{argmin}} \left[\alpha \mathbf{1}^T \boldsymbol{H}\boldsymbol{x} - \boldsymbol{y}^T \log(\alpha \boldsymbol{H}\boldsymbol{x}) + \frac{\rho}{2}\|\boldsymbol{x} - \widetilde{\boldsymbol{x}}^k\|^2 \right]}_{\text{Proximal operator for the negative log-likelihood}}, \tag{6.8a}$$

$$\boldsymbol{z}^{k+1} = \underbrace{\underset{\boldsymbol{z}}{\operatorname{argmin}} \left[-\log p(\boldsymbol{z}) + \frac{\rho}{2}\|\boldsymbol{z} - \widetilde{\boldsymbol{z}}^k\|^2 \right]}_{\text{Proximal operator for the negative-log-prior}}, \tag{6.8b}$$

$$\boldsymbol{u}^{k+1} = \boldsymbol{u}^k + (\boldsymbol{x}^{k+1} - \boldsymbol{z}^{k+1}), \tag{6.8c}$$

with $\widetilde{\boldsymbol{x}}^k \overset{\text{def}}{=} \boldsymbol{z}^k - \boldsymbol{u}^k$ and $\widetilde{\boldsymbol{z}}^k \overset{\text{def}}{=} \boldsymbol{x}^k + \boldsymbol{u}^k$. In the Plug-and-Play framework [76], [115], the $\boldsymbol{z}$ update in (6.8b) is implemented by an image denoiser.

The difficulty of solving the above problem is that the $\boldsymbol{x}$-update in (6.8a) does not have a closed form expression for the Poisson likelihood. Thus (6.8a) needs to be solved using an inner-loop optimization method such as L-BFGS [200]. Unrolling this inner-loop optimization solver can be inefficient as it may not be differentiable. Hence unrolling the PnP scheme for the Poisson inverse problem using the existing framework is infeasible. To be more specific, while the $\boldsymbol{z}$-update in (6.8b) can be implemented as a neural network and hence is differentiable, the same cannot be said for $\boldsymbol{x}$-update in (6.8a). As shown in Figure **6.4**, when (6.8a) is solved using another iterative method such as L-BFGS (for e.g. in [166]), it is not differentiable. As a result, training the unrolled network via backpropagation is not possible unless (6.8a) can be made differentiable.

6.2.3 Three-operator splitting for Poisson PnP

As explained in the previous subsection, the current framework does not allow for algorithm unrolling. To circumvent this issue, we use an alternate three-operator formulation of the PnP-framework. Through this reformulation of Plug-and-Play, we derive a series of iterative updates where each step can be implemented as a single-step that is differentiable. The three-operator splitting strategy we use here has been used in context of Poisson deblurring in [171], [201] and [182] using a TV and BM3D denoiser respectively.

In this scheme, instead of a two-operator splitting strategy for conventional PnP in Equation (6.6), we use three-operator splitting to form the corresponding constrained optimization problem. Specifically, in addition to splitting the variable as $\boldsymbol{x} = \boldsymbol{z}$, we introduce a third variable $\boldsymbol{v}$ corresponding to blurred image $\boldsymbol{Hx}$ and hence the constraint $\boldsymbol{Hx} = \boldsymbol{v}$.

$$\{\boldsymbol{x}^*, \boldsymbol{z}^*, \boldsymbol{v}^*\} = \underset{\boldsymbol{x},\boldsymbol{z},\boldsymbol{v}}{\operatorname{argmin}} \Big[-\boldsymbol{y}^T \log(\alpha \boldsymbol{v}) + \alpha \mathbf{1}^T \boldsymbol{v} + \log p(\boldsymbol{z}) \Big],$$
$$\text{subject to } \boldsymbol{x} = \boldsymbol{z}, \quad \text{and} \quad \boldsymbol{Hx} = \boldsymbol{v}. \tag{6.9}$$

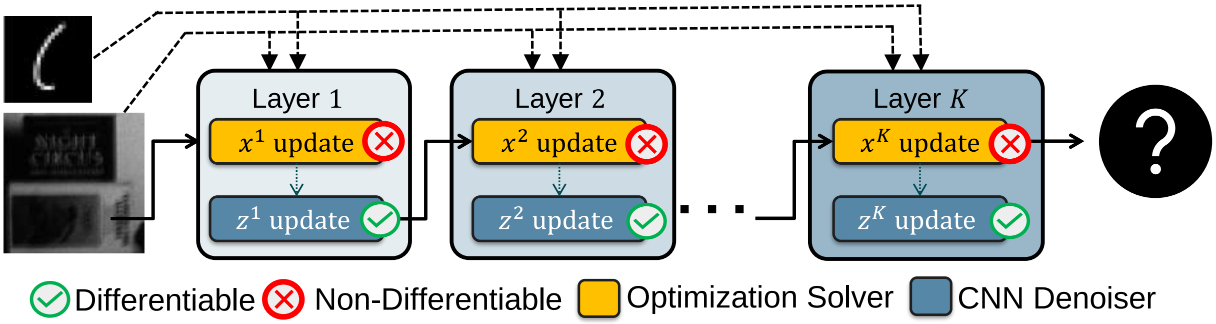


Figure 6.4. Conventional two-operator splitting Plug-and-Play. Conventional Plug-and-Play applied to the Poisson deblurring problem using equations (6.8a) and (6.8b). While (6.8b) is implemented as an image denoiser and hence differentiable, $\boldsymbol{x}$-update i.e. (6.8a) is implemented as a convex optimization solver and hence not differentiable. This makes the conventional PnP infeasible for fixed iteration unrolling and hence end-to-end training.

After forming the corresponding augmented Lagrangian, we arrive at the following iterative updates:

$$\boldsymbol{x}^{k+1} = \underset{\boldsymbol{x}}{\text{argmin}} \left[\frac{\rho_1}{2} \|\boldsymbol{x} - \tilde{\boldsymbol{x}}_0^k\|^2 + \frac{\rho_2}{2} \|\boldsymbol{H}\boldsymbol{x} - \tilde{\boldsymbol{x}}_1^k\|^2 \right], \tag{6.10a}$$

$$\boldsymbol{z}^{k+1} = \underset{\boldsymbol{z}}{\text{argmin}} \left[-\log p(\boldsymbol{z}) + \frac{\rho_1}{2} \|\boldsymbol{z} - \tilde{\boldsymbol{z}}_1^k\|^2 \right], \tag{6.10b}$$

$$\boldsymbol{v}^{k+1} = \underset{\boldsymbol{v}}{\text{argmin}} \left[-\boldsymbol{y}^T \log(\alpha \boldsymbol{v}) + \alpha \mathbf{1}^T \boldsymbol{v} + \frac{\rho_2}{2} \|\boldsymbol{v} - \tilde{\boldsymbol{v}}^k\|^2 \right], \tag{6.10c}$$

$$\boldsymbol{u}_1^{k+1} = \boldsymbol{u}_1^k + \boldsymbol{x}^{k+1} - \boldsymbol{z}^{k+1}, \tag{6.10d}$$

$$\boldsymbol{u}_2^{k+1} = \boldsymbol{u}_2^k + \boldsymbol{H}\boldsymbol{x}^{k+1} - \boldsymbol{v}^{k+1}, \tag{6.10e}$$

where $\tilde{\boldsymbol{x}}_0^k \overset{\text{def}}{=} \boldsymbol{z}^{k+1} - \boldsymbol{u}_1^k$, $\tilde{x}_1^k \overset{\text{def}}{=} \boldsymbol{v}^{k+1} - \boldsymbol{u}_2^k$, $\boldsymbol{v}^k \overset{\text{def}}{=} \boldsymbol{H}\boldsymbol{x}^k + \boldsymbol{u}_2^k$, and $\tilde{\boldsymbol{z}}^k \overset{\text{def}}{=} \boldsymbol{x}^k + \boldsymbol{u}_1^k$. Similar to the PnP formulation described in last subsection, the vectors $\boldsymbol{u}_1, \boldsymbol{u}_2$ denote the scaled Lagrangian multipliers for the constraints $\boldsymbol{x} - \boldsymbol{z} = 0$ and $\boldsymbol{H}\boldsymbol{x} - \boldsymbol{v} = 0$ respectively. The scalars ρ_1, ρ_2 denote the corresponding penalty parameters.

Each of the subproblems defined in (6.10a, 6.10b, 6.10c) have a closed form solution and are described below:

***x*-subproblem**: (6.10a) is a least squares minimization problem, whose solution can be explicitly given as follows:

$$\boldsymbol{x}^{k+1} = (\boldsymbol{I} + (\rho_2/\rho_1)\boldsymbol{H}^T\boldsymbol{H})^{-1}(\widetilde{\boldsymbol{x}}_0^k + (\rho_2/\rho_1)\boldsymbol{H}^T\widetilde{\boldsymbol{x}}_1^k). \tag{6.11}$$

Since $\boldsymbol{H}$ represents a convolutional operator, the operation can be performed without any matrix inversions using Fourier Transforms.

$$\boldsymbol{x}^{k+1} = \mathcal{F}^{-1}\Big[\frac{\mathcal{F}(\widetilde{\boldsymbol{x}}_0^k) + (\rho_2/\rho_1)\overline{\mathcal{F}(\boldsymbol{h})}\mathcal{F}(\widetilde{\boldsymbol{x}}_1^k)}{1 + (\rho_2/\rho_1) \mid \mathcal{F}(\boldsymbol{h}) \mid^2}\Big], \tag{6.12}$$

where $\mathcal{F}(\cdot)$ represents the discrete Fourier transform of the image or blur kernel implemented using the Fast Fourier Transform after appropriate boundary padding. We refer to it as the *deblurring operator.*

***z*-subproblem**: (6.10b) is a proximal operator for the negative log prior term. Using the insight provided in Plug-and-Play scheme, (6.10b) can be viewed as a denoising operation

$$\boldsymbol{z}^{k+1} = D(\widetilde{\boldsymbol{z}}^k), \tag{6.13}$$

where $D(\cdot)$ is any image denoiser. For end-to-end training, we require $D(\cdot)$ to be differentiable and trainable – a property satisfied by all convolutional neural network denoisers.

***v*-subproblem**: (6.10c) is a convex optimization problem but can be solved without an iterative procedure. Separating out each component of the vector minimization and setting the gradient equal to zero gives the following equation

$$-\frac{[\boldsymbol{y}]_\mathrm{i}}{[\boldsymbol{v}^{k+1}]_\mathrm{i}} + \alpha + \rho_2([\boldsymbol{v}^{k+1}]_\mathrm{i} - [\widetilde{\boldsymbol{v}}^k]_\mathrm{i}) = 0, \tag{6.14}$$

for $\mathrm{i} = 1, 2, \cdots, N$. Solving the resulting quadratic equation and ignoring the negative solution gives the following update step

$$\boldsymbol{v}^{k+1} = \frac{(\rho_2\widetilde{\boldsymbol{v}}^k - \alpha) + \sqrt{(\rho_2\widetilde{\boldsymbol{v}}^k - \alpha)^2 + 4\rho_2\boldsymbol{y}}}{2\rho_2}, \tag{6.15}$$

Since the optimization problem in (6.10c) is a sum of the the negative log-likelihood for Poisson noise and a quadratic penalty term, we refer to this update as *Poisson proximal operator*.

Algorithm 1 Three-Operator Splitting for Poisson PnP

```
**Input**: Blurred and Noisy Image $\boldsymbol{y}$, kernel $\boldsymbol{h}$, Photon level $\alpha$
Initialize $\boldsymbol{x}^0$ using (6.16)
$\boldsymbol{z}^0 \leftarrow \boldsymbol{x}^0$, $\boldsymbol{v}^0 \leftarrow \boldsymbol{y}$ $\boldsymbol{u}_1^0 \leftarrow 0$, $\boldsymbol{u}_2^0 \leftarrow 0$
**for** $k = 1, 2, \cdots, K$ **do**
    Update $\boldsymbol{x}^k$ using Eq. (6.12)
    Update $\boldsymbol{z}^k$ using Eq. (6.13)
    Update $\boldsymbol{v}^k$ using Eq. (6.15)
    $\boldsymbol{u}_1^k \leftarrow \boldsymbol{u}_1^{k-1} + \boldsymbol{x}^k - \boldsymbol{z}^k$
    $\boldsymbol{u}_2^k \leftarrow \boldsymbol{u}_2^{k-1} + \boldsymbol{H}\boldsymbol{x}^k - \boldsymbol{v}^k$
**end for**
return $\boldsymbol{x}^K$
```

The convergence of Algorithm 1 has been derived in [171]. It was shown that as long as $\boldsymbol{G} = [\boldsymbol{H}^T, \boldsymbol{I}]^T$ has a full column rank, the three-operator splitting scheme converges. Furthermore, assuming the denoiser D is continuously differentiable and $\nabla D(\cdot)$ is symmetric with eigenvalues in $[0, 1]$, convergence results in [179] show that the corresponding negative-log prior, i.e., $-\log(p(\cdot))$ is closed, proper and convex. Combined with the result from [171], it can be shown that the three-operator PnP scheme in Algorithm 1 converges.

6.2.4 Unfolding the three-operator splitting

With an end-to-end trainable iterative process, we can now describe the unfolded iterative network. The Plug-and-Play updates described in Algorithm 1 are now unfolded for $K = 8$ iterations and the entire differentiable pipeline is trained in a supervised manner, as summarized in Figure **6.5**.

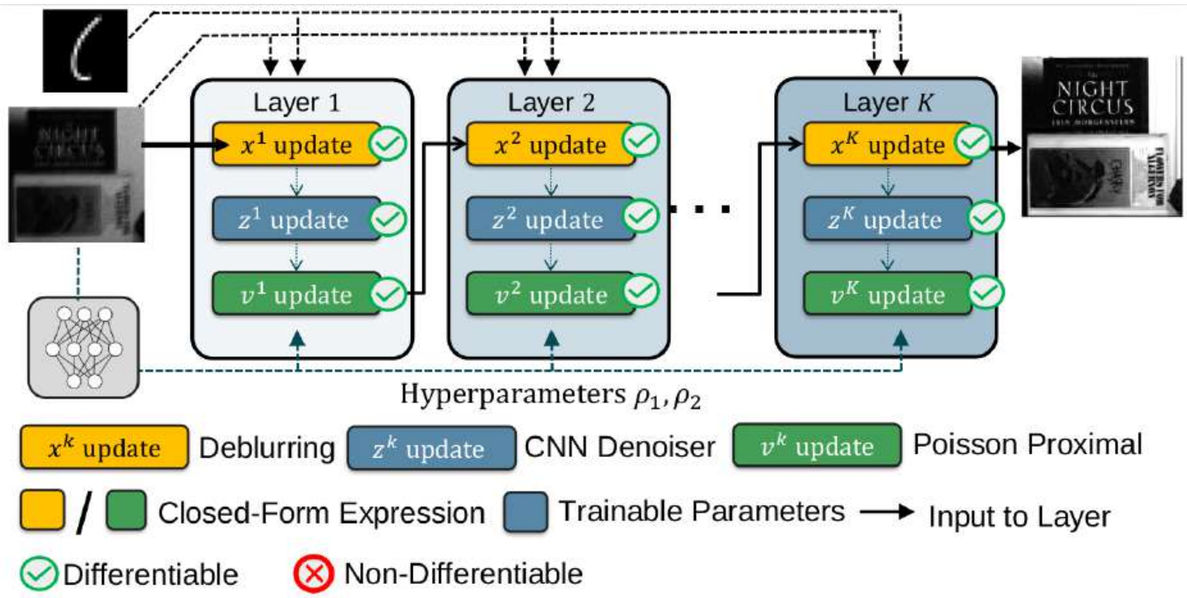


Figure 6.5. Proposed unrolled Plug-and-Play for deblurring. For conventional PnP, the data sub-problem cannot be solved in a single step and instead requires convex optimization solvers. This stops us from unrolling the iterative procedure and training it end-to-end via back-propagation. Through the three-operator splitting formulation of the problem, each sub-module in an iteration is in closed form and more importantly, differentiable. This allows for end-to-end training which was not possible in conventional PnP. The network below the input represents the hyperparameter network which predicts ρ_1 and ρ_2 using the blur kernel and the photon level.

Initialization: To initialize the variable $\boldsymbol{x}^0$, we use the Wiener filtering step (not to be confused with [158]) :

$$\boldsymbol{x}^0 = \frac{1}{\alpha}\mathcal{F}^{-1}\left\{\frac{\overline{\mathcal{F}(\boldsymbol{h})}\mathcal{F}(\boldsymbol{y})}{1/\alpha+ \mid \mathcal{F}(\boldsymbol{h}) \mid^2}\right\}, \tag{6.16}$$

where the constant factor $1/\alpha$ in the denominator represents the inverse of the signal-to-noise ratio of the blurred measurements. Note that this step can be derived as an ℓ_2 regularized solution of the deconvolution problem as well.

Hyperparameters: The parameters used in updates (6.10a), (6.10c) – ρ_1, ρ_2 are changed for each iteration and determined in one-shot by the blurring kernel $\boldsymbol{h}$ and photon level α as they control the degradation of the image. The kernel $\boldsymbol{h}$ is used as input to 4 convolutional layers, flattened to a vector of length 1024. Along the with the photon level α, the flattened vector is used as an input to a 3-layer fully connected network which output two set of vectors i.e. $\{\rho_1^1, \rho_1^2, ..., \rho_1^K\}$ and $\{\rho_2^1, \rho_2^2, ..., \rho_2^K\}$. We refer the readers to the supplementary document for further architectural details.

Note that there is no ground-truth assumed for parameters ρ_1, ρ_2 as the hyperparameter network described above is trained simultaneously as rest of the parameters of the network.

Denoiser: For the denoiser used in (6.13), we use the architecture provided in [192] which introduces skip connections in a U-Net architecture known as ResUNet. Like a standard U-Net, there are four downsampling operations followed by 4 upsampling operations with skip connections between the upsampling and downsampling operators. For further details of the architecture we refer the readers to [192] or the supplementary document. Note that in our implementation of the architecture, we do not concatenate the denoiser input $\tilde{\boldsymbol{z}}^k$ with a noise level.

6.3 Experiments

6.3.1 Training

We train the network described in section 6.2 using ℓ_1-loss function. We use images from the Flickr2K [202] dataset to train the network. The dataset contains a total of 2650 images of which we partition using a 80/20 split for training and validation. All images are converted to gray-scale, scaled to a size of 256×256, and are blurred using motion kernels generated from [203] and Gaussian blur kernels. Due to memory limits of GPU, random patches of size 128×128 were cropped and used as inputs for the network during training.

For training, a combination of 60 motion kernels generated from [203] and 10 isotropic gaussian blur kernels with σ varying from $\left[0.1, 2.5\right]$ were used. All the kernels were pre-generated prior to training and were randomly selected during training. Entries of the blur kernel are non-negative and sum to 1. Photon Shot noise is synthetically added to the blurred image according to (6.2). The photon level α is uniformly sampled from the range $\left[1, 60\right]$.

The inputs to the network consist of the blurred and corrupted image $\boldsymbol{y}$, the normalized blur kernel $\boldsymbol{h}$, and the photon noise level α. The output from the network is the reconstructed

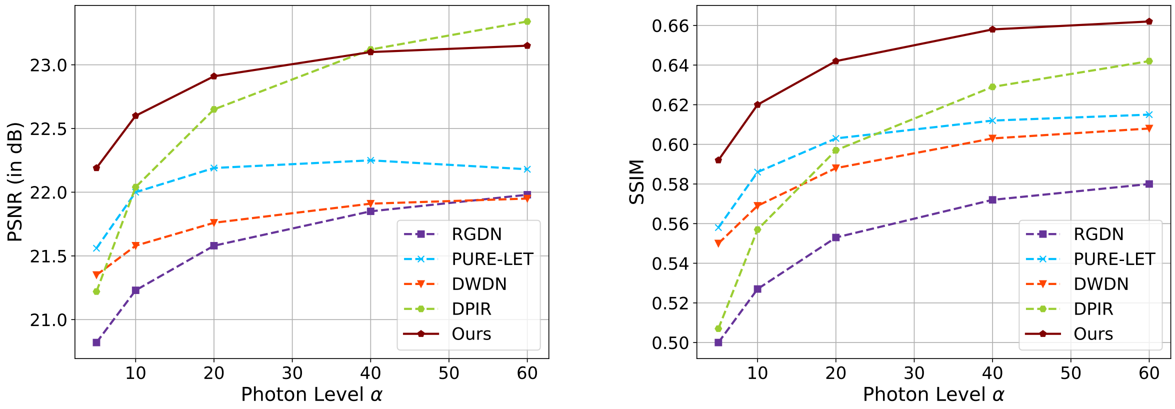


Figure 6.6. Quantitative evaluation. Comparison of PSNR and SSIM of the different methods on Levin et. al. dataset [204]. The dataset consists of 32 blurred images generated by blurring 4 images by 8 motion kernels and average PSNR/SSIM for all images and kernels plotted for different photon levels. The images were corrupted by Poisson noise at photon levels $\alpha = 5, 10, 20, 40$ and 60.

image $\boldsymbol{x}^K$ where K denotes the number of iterations for which the scheme is unrolled for. We set the the number of iterations in our implementation to $K = 8$. Using the ℓ_1-loss function, we train the network with Adam optimizer [205] using a learning rate 1×10^{-4} and batch size of 5 for 100 epochs. All the parameters of the network are initialized using Xavier initialization [206] and is implemented in Pytorch 1.7.0. For training, we use an NVIDIA Titan Xp GP102 GPU and it takes approximately 20 hours for training to complete.

6.3.2 Choice of Deblurring Methods for Comparison

Before describing the results of quantitative evaluation, we briefly discuss the other deblurring approaches we compare our method with. The methods, namely **RGDN** [154], **DWDN** [158], **DPIR** [181], and **PURE-LET** [162], were chosen because they give state-of-the-art results on the deblurring problem *and* because they represent different contemporary approaches to solving the non-blind deconvolution problem.

RGDN (Recurring Gradient Descent Network) is an unrolled optimization method. More specifically, the authors take the deconvolution cost function $\|\boldsymbol{y} - \boldsymbol{k} * \boldsymbol{x}\|^2 + \Omega(\boldsymbol{x})$

and provide a gradient descent iterative scheme for it. The second term in the cost functions represents image prior and the corresponding gradient term $\nabla\Omega(\boldsymbol{x})$ is estimated using a convolutional neural network and the network, after being unrolled for fixed iterations, is trained end-to-end.

Deep-Weiner Deconvolution (DWDN) can be viewed as a hybrid deconvolution/denoising method. As a U-Net denoiser converts an image into a smaller feature space and then reconstructs the image using a decoder, DWDN first extracts features, performs Weiner deconvolution in that feature space, and then followed by decoding to a clean image. Through this architecture choice, they are able to perform denoising through the encoder-decoder structure but also deblur the image using Weiner deconvolution.

DPIR (Deep Plug-and-Play Image Restortation) uses a pre-trained denoiser in a half-quadratic splitting scheme and represents a state-of-the-art method which can be used for general purpose linear inverse problems like super-resolution and deblurring. Like our approach, it also boils down to a iterative series of denoising and deblurring steps.

PURE-LET (Poisson Unbiased Risk Estimate - Linear Expansion of Thresholds) proposes the solutions as a linear combination of basis function whose weights are determined by minimizing the unbiased estimate of the mean squared loss under given noise conditions. While not a deep-learning method, it performs surprisingly competitively and can incorporate both Poisson shot noise and Gaussian read noise explicitly.

Table 6.2. Different features of methods used in this work for Poisson deblurring. We classify the methods based on three criteria - iterative/non-iterative, end-to-end trainability and whether the model explicitly incorporates the fact that the images are corrupted by Poisson shot noise.

Method	Iterative?	End-to-End Trainable?	Handles Poisson Noise?
RGDN [154]	✓	✓	×
PURE-LET [162]	×	×	✓
DWDN [158]	×	✓	×
DPIR [181]	✓	×	×
Ours	✓	✓	✓

6.3.3 Quantitative Evaluation

The results are summarized in Figure 6.6. We evaluate our method using synthetically generated noisy blurred images on 100 images from the BSDS300 dataset [207], from now on referred to as *BSD100*. We evaluate the performance on different photon levels ($\alpha = 5, 10, 20, 40$) representing various levels of degradation in terms of signal-to-noise ratio. We test the methods for different blur kernels - specifically 4 isotropic Gaussian kernels, 4 anisotropic Guassian kernels, and 4 motion kernels, as illustrated in Figure 6.7. Note that the top-left kernel's width is very small - this can be viewed as an identity operator and hence equivalent to evaluating the method's performance on denoising (as opposed to deblurring).

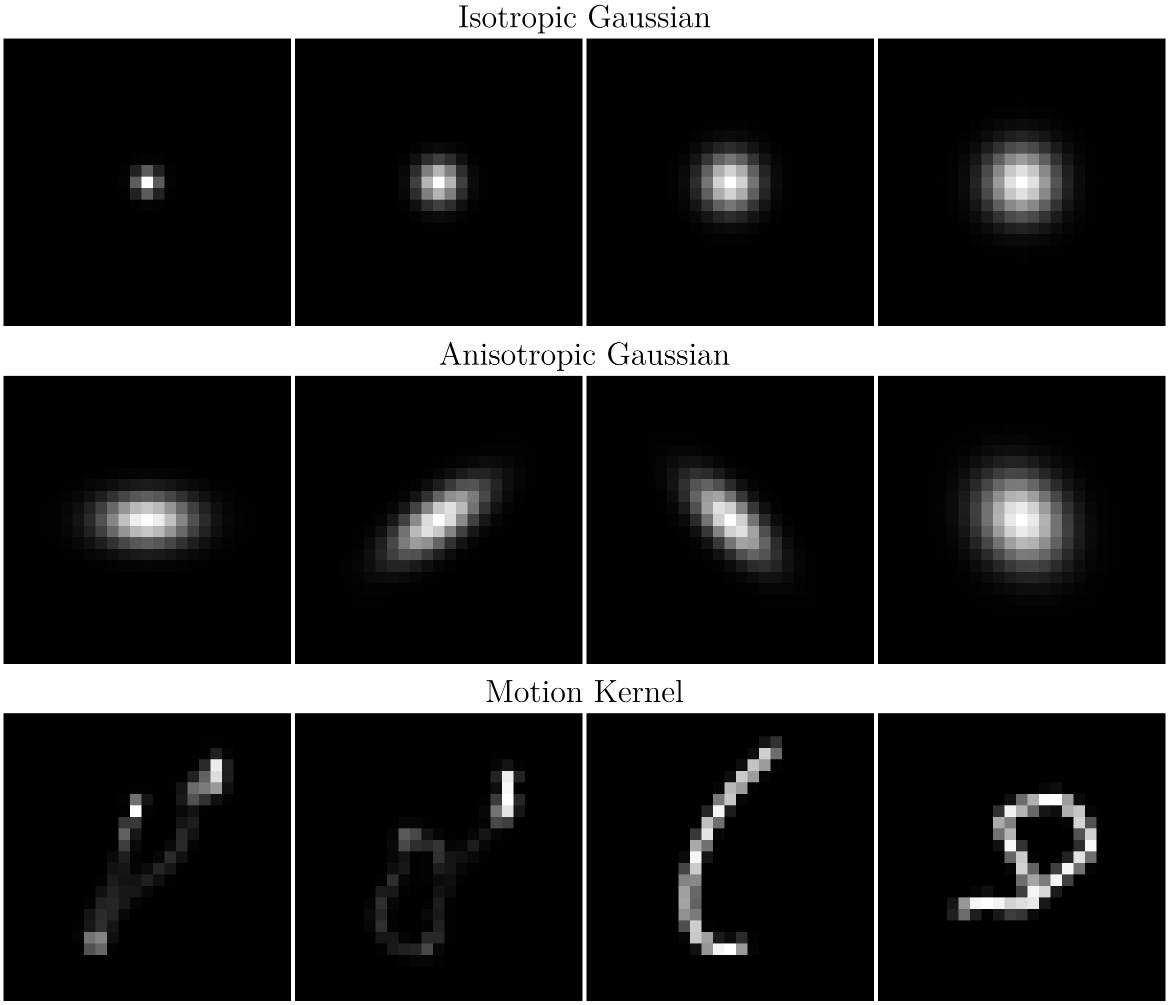


Figure 6.7. Kernels used for evaluation on BSD100 dataset.

As described in the previous subsection, we compare our method with the following deblurring methods - **RGDN**, **PURE-LET**, **DWDN**, and **DPIR**. Different features of the abovementioned deconvolution approaches have been summarized in Table 6.2 for reader's convenience. For the sake of a fair comparison, the end-to-end trainable methods RGDN and DWDN were retrained using the same procedure as that of our method.

Table 6.3. Comparison of proposed method with other competing approaches on BSD100 dataset

Photon Level	Kernel		RGDN [154]	PURE-LET [162]	DWDN [158]	DPIR [181]	Ours
$\alpha = 5$	Isotropic	PSNR (dB)	21.77	22.78	22.50	22.33	**23.46**
	Gaussian	SSIM	0.440	0.502	0.493	0.431	**0.531**
	Anisotropic	PSNR (dB)	21.62	22.22	22.19	21.92	**22.70**
	Gaussian	SSIM	0.427	0.463	0.464	0.409	**0.491**
	Motion	PSNR (dB)	21.14	21.49	21.54	21.35	**22.12**
		SSIM	0.377	0.419	0.413	0.377	**0.433**
$\alpha = 10$	Isotropic	PSNR (dB)	22.57	23.54	22.86	23.17	**24.24**
	Gaussian	SSIM	0.491	0.549	0.527	0.476	**0.576**
	Anisotropic	PSNR (dB)	22.30	22.81	22.56	22.60	**23.28**
	Gaussian	SSIM	0.466	0.501	0.494	0.448	**0.525**
	Motion	PSNR (dB)	21.51	22.07	21.94	21.98	**22.80**
		SSIM	0.399	0.454	0.443	0.411	**0.475**
$\alpha = 20$	Isotropic	PSNR (dB)	23.11	24.27	23.16	23.98	**24.96**
	Gaussian	SSIM	0.528	0.594	0.558	0.522	**0.621**
	Anisotropic	PSNR (dB)	22.78	23.34	22.86	23.20	**23.83**
	Gaussian	SSIM	0.494	0.536	0.522	0.485	**0.557**
	Motion	PSNR (dB)	21.82	22.70	22.27	22.65	**23.47**
		SSIM	0.418	0.494	0.475	0.448	**0.515**
$\alpha = 40$	Isotropic	PSNR (dB)	23.47	25.00	23.35	24.76	**25.68**
	Gaussian	SSIM	0.555	0.638	0.582	0.569	**0.663**
	Anisotropic	PSNR (dB)	23.10	23.82	23.10	23.74	**24.36**
	Gaussian	SSIM	0.515	0.569	0.545	0.520	**0.589**
	Motion	PSNR (dB)	22.07	23.38	22.52	23.36	**24.20**
		SSIM	0.436	0.538	0.502	0.488	**0.564**

In addition to the BSD100 dataset, we also evaluated these methods on the blurring dataset provided in Levin et. al [204]. This dataset contains a set of 32 blurred images generated by blurring 4 different clean images by 8 different motion kernels. We synthetically corrupt the blurred images with Poisson noise at different illumination levels.

The results for these evaluations are provided in Table (6.3) and Figure **6.6**. For qualitative comparison on grayscale and colour reconstructions, one can refer to Figure **6.8**. On the BSDS100 dataset, our method outperforms the competing methods on all blurring kernels

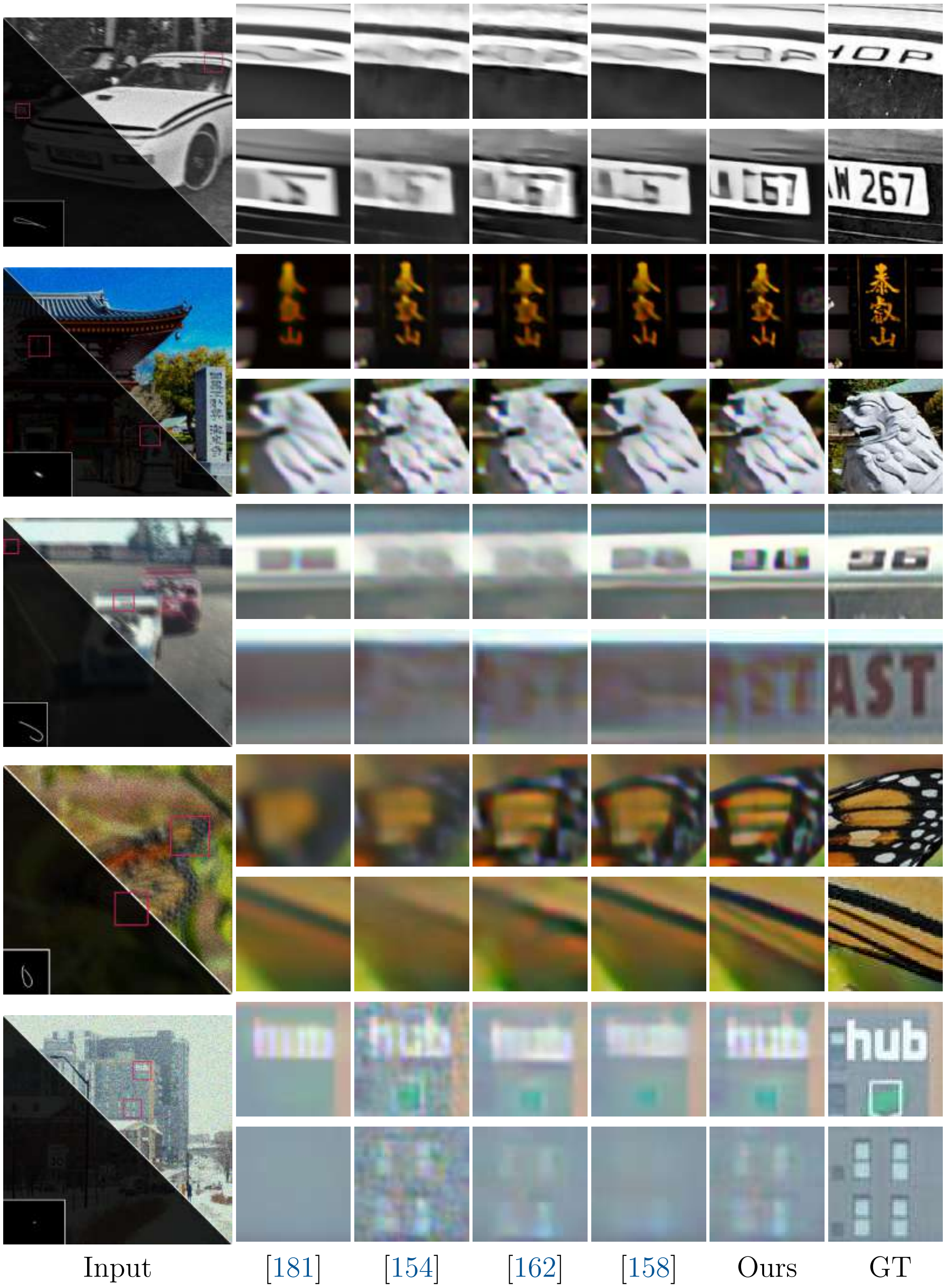


Figure 6.8. Qualitative Evaluation on synthetic images. We compare the performance of the proposed method with competing methods on synthetic grayscale and color images.

and illumination levels. For the dataset by Levin et. al, we outperform the other methods except DPIR at photon level $\alpha = 40$. On both datasets, we observe that the gap between conventional deblurring and our method decreases as the illumination levels increase. This is because as the mean of a Poisson random variable starts increasing, the probability distribution function resembles that of a Gaussian. Therefore, the conventional deblurring methods which are designed for Guassian noise show improved performance.

6.3.4 Comparison between 2-operator and 3-operator splitting

As explained in Section 6.2.2, conventional Plug-and-Play using two-operator splitting is not suitable for algorithm unrolling. The proposed three-operator splitting enables algorithm unrolling because every iterative step is differentiable. It is this end-to-end training that allows us to a better performance. In this experiment, we perform an ablation study to quantify the performance gain through different combinations of unrolling and training.

In Figure **6.9**, we show the reconstruction performance of three schemes on the BSD100 dataset: (a) conventional two-operator splitting PnP using FFDNet denoiser as described in Section 6.2.2 (b) an alternate three-operator splitting formulation using FFDNet as described in Section 6.2.3 and (c) the proposed unrolled version of the scheme described in Section 6.2.3. The results show that the two iterative schemes (a) and (b) perform similarly. However, training the proposed algorithm unrolling achieves a consistent performance gain of more than 1dB across all photon levels.

When implementing the conventional PnP in (a), we use the approach from [166] and solve the $\boldsymbol{x}$-update (6.8a) using a L-BFGS solver [200]. Like the original implementation, we use a surrogate cost function to approximate the near zero entries with a quadratic approximation to avoid the singularities in the original cost function. A pretrained DnCNN [147] for noise level $\sigma = 15/255$ was used for the $\boldsymbol{z}$-update (6.8b). For the three-operator splitting scheme in (b), the same denoiser was used. To ensure a fair comparison, in the proposed fixed iteration unrolled network, we replace the ResUNet denoiser with a DnCNN and train it using the method described in Section 6.3.1. Further details about the experiment are provided in the supplementary document.

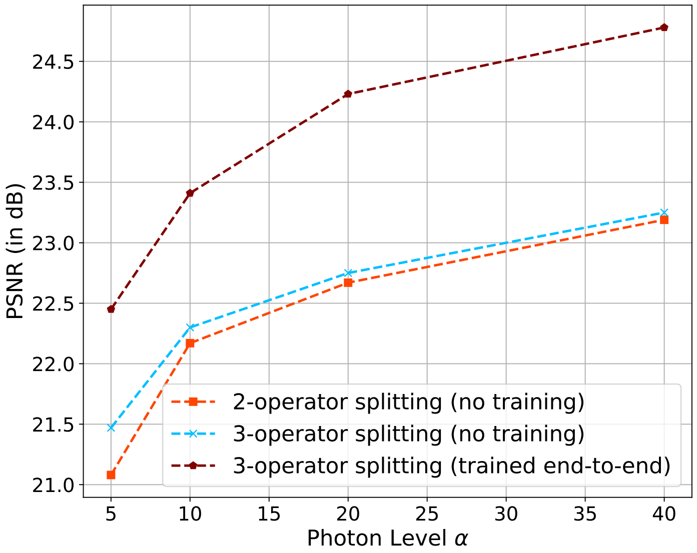
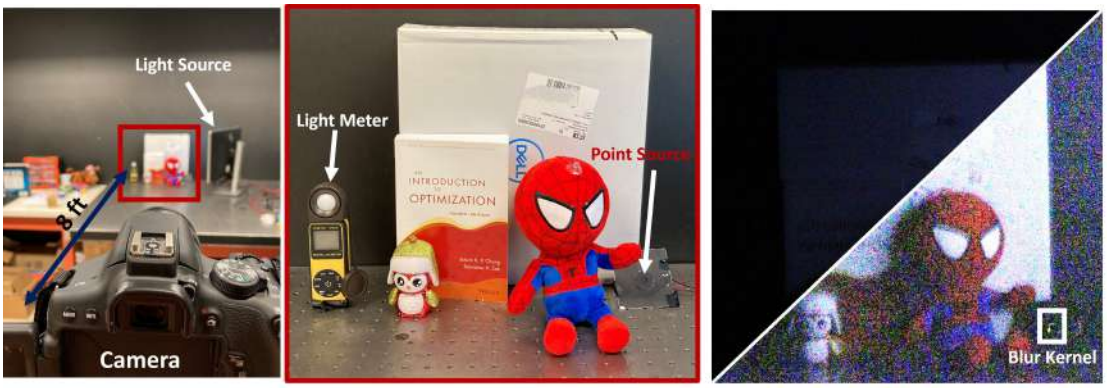


Figure 6.9. Ablation study to quantify significance of algorithm unrolling. We evaluate the following three schemes on the BSD100 dataset **(a)** conventional PnP (two-operator splitting) with a DnCNN denoiser. **(b)** alternate PnP (three-operator splitting) with a DnCNN denoiser. **(c)** proposed fixed iteration unrolled network using a DnCNN denoiser. The results of this experiment show that the significant improvement is achieved due to the network unrolling.



(a) Experimental setup, well illuminated scene (b) Real capture

Figure 6.10. Experimental setup. For evaluation of the proposed method on real images, we collect noisy and blurred images using a DSLR as shown in the setup shown above. To capture a single degraded image, we reduce the illumination to a level that shot noise becomes visible. We blur image using camera shake. For the blur kernel, each scene contains a point source and the corresponding motion kernels can be visualized in Figure 6.11.

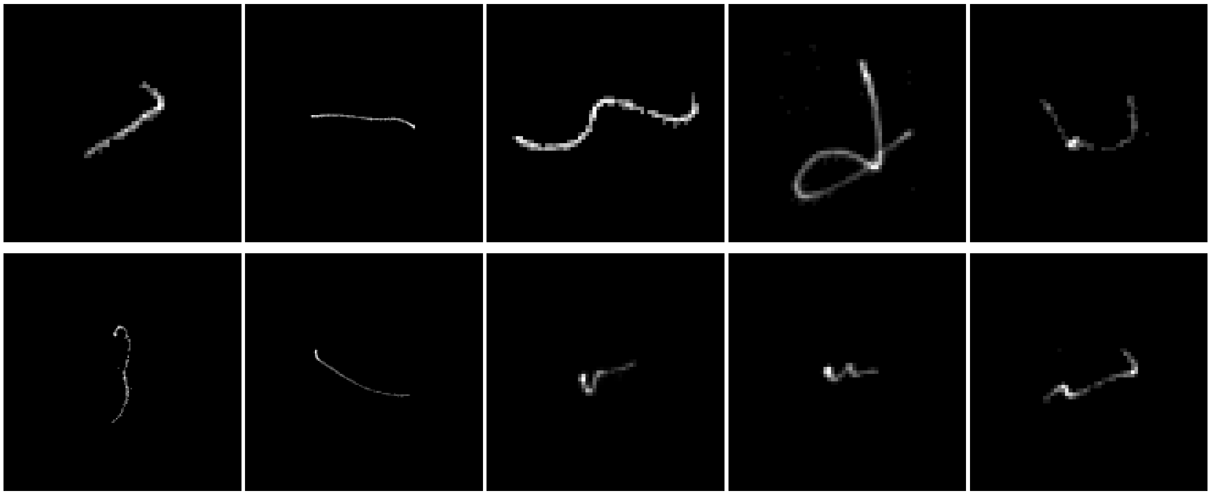


Figure 6.11. **Real kernels** generated by our optical experiment setup.

6.3.5 Color reconstruction

The focus of this work is image deblurring. We acknowledge that most image sensors today acquire color images using the color filter arrays. However, adding the deblurring task with demosaicking is substantially beyond the scope of this work. Even for demosaicking without any blur, the shot noise requires customized design, e.g., [22]. Therefore, color images shown in this work were processed individually for each color channel and then fused using an off-the-shelf demosaicking algorithm. While this approach is sub-optimal, our real image experiments show that the performance is acceptable.

6.4 Experiments using Sensor Data

Unlike conventional deblurring problems where datasets are widely available, photon-limited deblurring data is not easy to collect. In this section we report our efforts in collecting a new dataset for evaluating low-light deblurring algorithms.

6.4.1 Photon-limited deblurring dataset

We collect shot-noise corrupted and blurred images using a digital single lens reflex (DSLR) camera. The DLSR is handheld to generate motion blur. A Dell 24-inch monitor, pointing towards the region of interest, was used as a programmable illumination source to control the photon level α. A light-meter is placed in the scene to measure the photon flux level.

Image Capture: We use an Canon EOS Rebel T6i camera to capture the images with exposure time of 30 ms and aperture $f/5.0$. The ISO was set to the highest possible value of 12800 to maximize the internal gain of the sensor and hence minimize the quantization effects of the analog-to-digital convertor (ADC). The same scene was captured using different illumination levels and correspondingly different motion blur kernels. The raw image files were used for image processing instead of the compressed JPG files.

Generating Blur: To capture the blur kernel along with the image, we place a point source in each scene (see bottom right of middle image in Figure 6.10). The point source is created by placing an LED behind a black screen with a $30\mu m$ pinhole. The strength of the point source is maximized to ensure the kernel is not corrupted with shot noise without saturation of pixel values. Some example kernels collected through this process can be visualized in Figure 6.11.

Photon Level: The illumination of the scenes varies between 1-5 Lux, as measured by the light-meter shown in Figure 6.10. To maximize the amount of photons captured, the aperture is kept as large as possible. However shot noise is still present due to the relatively short exposure time. The estimated average photons-per-pixel (ppp) varied from 5-60.

Generating Ground Truths: For quantitative evaluation, we also provide the ground truth for each noisy blurred image. For each noisy image corrupted by motion and noise - we place the camera on a tripod and capture 10 frames of the scene under the same illumination and camera settings. The frames, captured without any blur due to camera shake, are averaged to reduce the shot noise as much as possible. These images serve as ground truth when evaluating the performance of reconstruction methods using PSNR/SSIM.

6.4.2 Reconstruction from real data

Pre-processing: To reconstruct the images using our network, we first need to convert it into the format representing the number of photons captured from the raw sensor values.

The raw digital data ($\boldsymbol{y}_{\mathrm{raw}}$) from the .RAW file is presented using a 14-bit value. To convert the 14-bit format to the number of photons, we use the following linear transform

$$\boldsymbol{y}_{\mathrm{i}} = \frac{\boldsymbol{y}_{\mathrm{i\ raw}} - b}{G}, \tag{6.17}$$

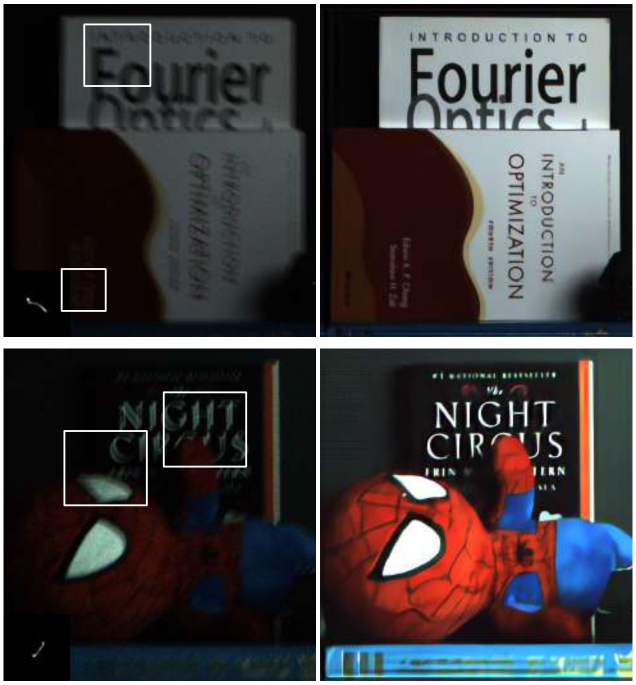
where b represents the zero-level offset of the camera which can be obtained from the meta-data of the image .RAW file and is set equal to $b = 2047$. G represents the gain factor between the digital output of the sensor and the actual electrons collected by the sensor. This gain is calculated from the camera data available at [59]. Specifically, we look at the read noise of the camera in terms of digital numbers and electrons. The ratio of these two data will give the gain G. For Canon EOS Rebel T6i, at ISO 12800, the gain is estimated to be $G \approx 71$.

Our reconstruction results are shown in Figure **6.12**. We also compare reconstructions using proposed method with other contemporary deblurring methods (RGDN, PURE-LET, DPIR and DWDN) in Figure **6.13**. Through a visual inspection, one can conclude that our method is able to reconstruct finer details from the noisy and blurred image while leaving behind fewer artifacts.

Quantitative Evaluation: For evaluation of metrics such as PSNR and SSIM, we register the ground truth to the reconstruction using homography transformation to account for the differences in camera positions. The average PSNR and SSIM on the real datset for the proposed and competing methods are reported in Table 6.4. We outperform the second-best competing methods, i.e. [158], by 0.6dB in terms of PSNR and by 0.005 in terms of SSIM. As shown in Figure **6.13**, when evaluating SSIM on a few patches containing text, the gap between our method and [158] becomes much wider.

Table 6.4. PSNR (in dB) and SSIM evaluated on real dataset of 30 images.

Method	RGDN [154]	PURE-LET [162]	DPIR [181]	DWDN [158]	Ours
PSNR	19.80	20.88	22.09	22.85	**23.48**
SSIM	0.476	0.501	0.548	0.561	**0.566**



(a) Real input (b) Processed

Figure 6.12. Proposed method on real data. For a qualitative comparison of other deblurring approaches on these images, refer to Figure 6.13.

In this work, we formulated the photon-limited deblurring problem as a Poisson inverse problem. We presented an end-to-end trainable solution using a algorithm unrolling technique. We performed extensive numerical experiments to compare our approach with other existing state-of-the-art non-blind deblurring approaches and demonstrated how our method can be applied to real sensor data. Even though the present solution is focused on image deblurring, it can be easily extended to other photon-limited inverse problems such as compressive sensing, lensless imaging, and super-resolution.

The algorithm presented in this work can be used to reconstruct a single clean image from multiple blurred images. This would allow us to take advantage of the temporal redundancy which would be necessary to obtain a meaningful clean signal in much challenging scenarios (e.g. photon level $\alpha \leq 5$). Another interesting but challenging problem which can be attempted using the framework is *low-light blind deconvolution* i.e. recovering the clean image and blur kernel simultaneously from blurred images corrupted with photon shot noise.

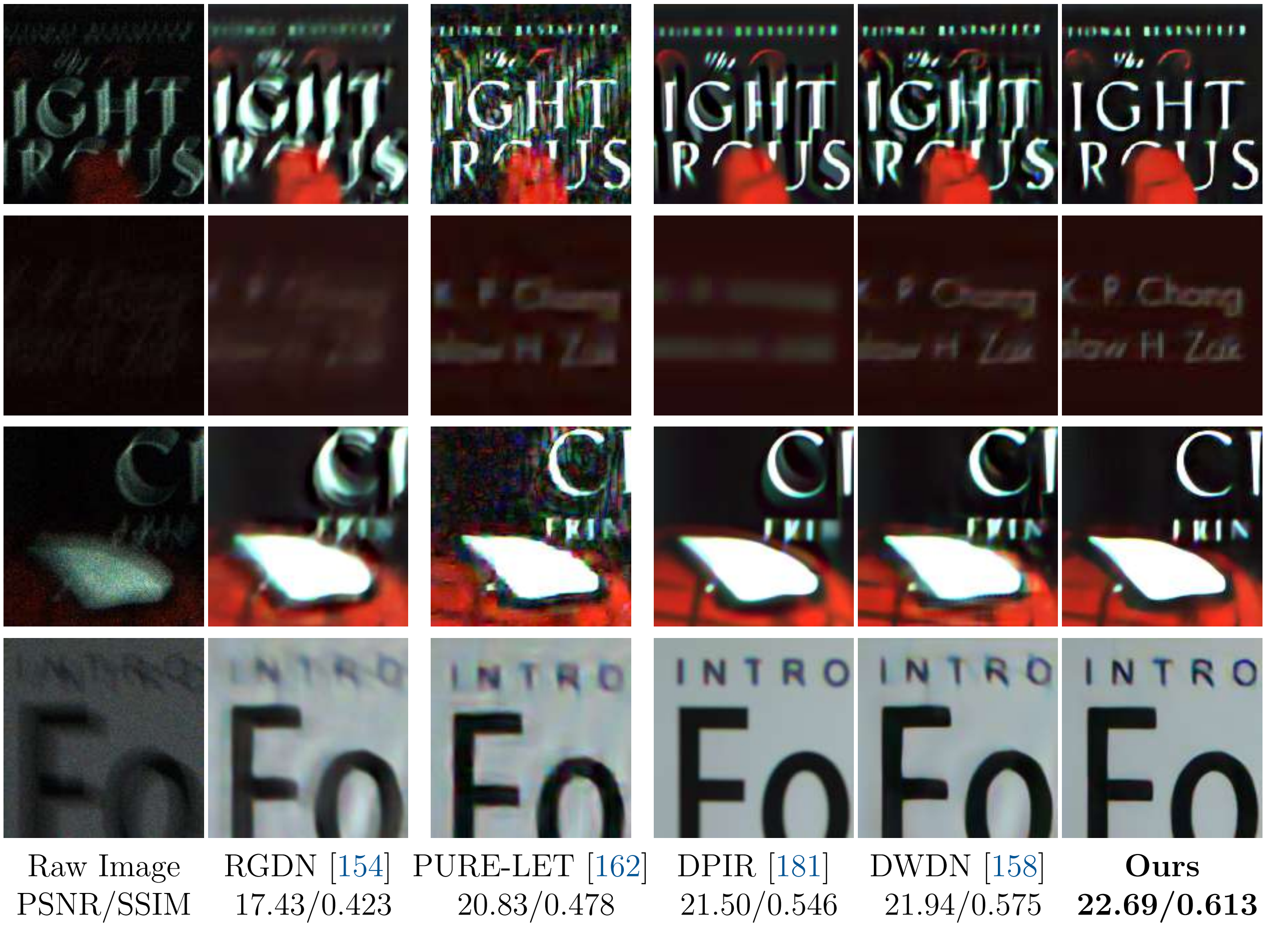


Figure 6.13. Qualitative Comparison We look at zoomed in regions of the reconstructed images from Figure 6.12 using competing methods. The average PSNR and SSIM evaluated on the given patches is provided at the bottom. From visual inspection one can see that our method is able to recover finer details compared to other methods. Note that in the first row, the DPIR output may look qualitatively similar to our result. This is because the former often tends to blur out images for a “cleaner“ looking image as observed in the second row of zoomed in reconstructions.

7. OBJECT DETECTION IN LOW LIGHT

Until now, we have looked at mostly low-level image processing to reconstruct the underlying signal. This chapter looks at downstream and high-level tasks such as image classification and object detection. Robust computer vision that can work in photon-limited scenarios is vital for night vision, surveillance, and microscopy applications. The mainstream "low-light" image enhancement methods have produced promising results that improve the image contrast between the foreground and background through advanced coloring techniques. Nevertheless, the more challenging problem of mitigating the photon shot noise inherited from the random Poisson process remains open. In this chapter, we present photon-limited image classification and object detection frameworks. The primary thread between these two methods is knowledge distillation in the form of student-teacher learning to improve the robustness of the feature extractor against noise. To get the absolute best performance in terms of the light level, we use the more sensitive CIS-QIS (referred to as just QIS in the rest of the chapter) to capture the images. However, the methods discussed are more general and can be used with any image sensors.

7.1 Student-teacher learning

When dealing with photon-limited imaging, the Poisson noise statistics creates a fundamental limit in the performance of image classifiers or object detectors. Therefore, when applying a classification method to the raw data, removing the shot noise becomes necessary. The traditional solution to this problem is to denoise the images as shown at the top of Figure 7.3. This section aims to introduce an alternative approach using the concept of student-teacher learning.

The idea of student-teacher learning can be understood from Figure 7.1. This figure has two networks: A teacher network and a student network. The teacher network is trained using *clean* samples and is pre-trained, i.e., its network parameters are fixed during training of the student network. The student network is trained using *noisy* samples with assistance from the teacher. Because the teacher is trained using clean samples, the features extracted are in principle "good", in contrast to the features of the student which are likely to be

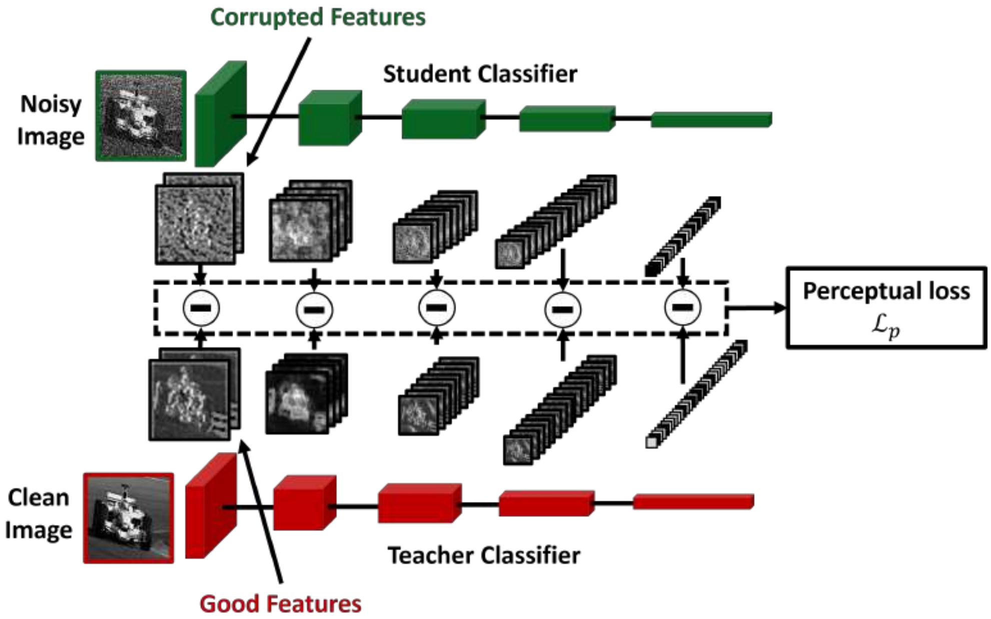


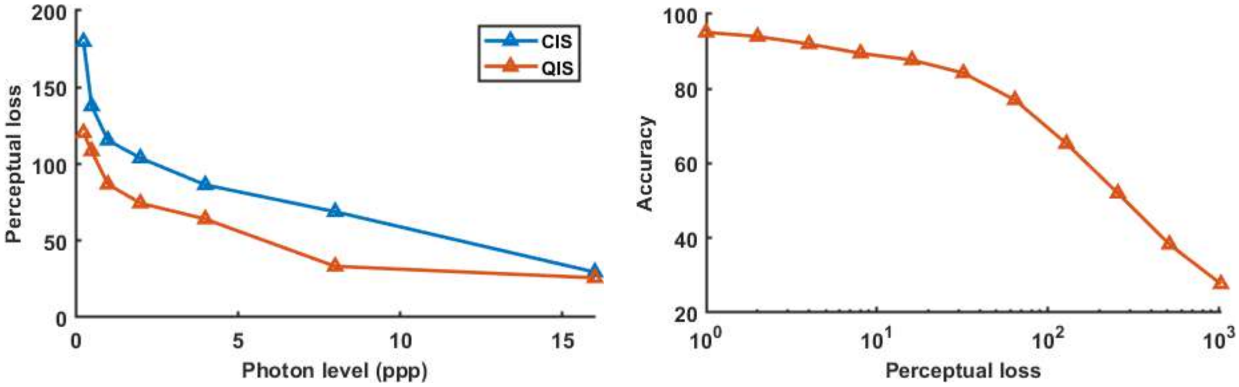
Figure 7.1. Student-teacher learning. Student-teacher learning comprises two networks: A teacher network and a student network. The teacher network is pre-trained using clean samples whereas the student is trained using noisy samples. To transfer knowledge from the teacher to the student, we compare the features extracted by the teacher and the student at different stages of the network. The difference between the features is measured as the perceptual loss.

"corrupted". We propose to transfer the knowledge from the features of the teacher to the features of the student. In order to achieve this, we propose minimizing a *perceptual loss* as defined below. We define the j-th layer's feature of the student network as $\phi^{\mathrm{j}}(\boldsymbol{y}_{\text{noisy}})$, where $\phi^{\mathrm{j}}(\cdot)$ maps the noisy image $\boldsymbol{y}_{\text{noisy}}$ to a feature vector, and we define $\widehat{\phi}^{\mathrm{j}}(\boldsymbol{y}_{\text{clean}})$ as the feature vector extracted by the teacher network from the clean image $\boldsymbol{y}_{\text{clean}}$. The perceptual loss is

$$\mathcal{L}_{\mathrm{p}}(\boldsymbol{y}_{\text{noisy}}, \boldsymbol{y}_{\text{clean}}) = \sum_{\mathrm{j}=1}^{J} \underbrace{\frac{1}{N_{\mathrm{j}}} \left\| \widehat{\phi}^{\mathrm{j}}(\boldsymbol{y}_{\text{clean}}) - \phi^{\mathrm{j}}(\boldsymbol{y}_{\text{noisy}}) \right\|^2}_{\text{j-th layer's perceptual loss}}, \tag{7.1}$$

where N_{j} is the dimension of the j-th feature vector. Since the perceptual loss measures the distance between the student and the teacher, minimizing the perceptual loss forces them to be close, which in turn, forces the network to "denoise" the shot noise and read noise in $\boldsymbol{y}_{\text{noisy}}$ before predicting the label.

We conduct a simple experiment to demonstrate the impact of input noise on perceptual loss and classification accuracy. We first consider a pre-trained teacher network by sending noisy data at different photon levels. As the photon level drops, the quality of the features also drops, and hence the perceptual loss increases. This is illustrated in Figure 7.2(a). Then in Figure 7.2(b), we evaluate the classification accuracy by using the synthetic testing data outlined in the Experiment Section. As the perceptual loss increases, the classification accuracy drops. This result suggests that the classification accuracy can be improved if we minimize the perceptual loss.



(a) Perceptual loss vs Photon level

(b) Accuracy vs Perceptual loss

Figure 7.2. Effectiveness of student-teacher learning. (a) Perceptual loss as a function of photon level. (b) Classification accuracy as a function of the perceptual loss $\mathcal{L}_p(\boldsymbol{y}_{\text{clean}}, \boldsymbol{y}_{\text{clean}})$. The accuracy is measured by repeating the synthetic experiment described in the Experiment Section. The negative correlation suggests that perceptual loss is indeed an influential factor.

Our proposed student-teacher learning is inspired by the knowledge distillation work of Hinton et al. [153] which proposed an effective way to compress networks. Several follow up ideas have been proposed, e.g., [208]–[211], including the MobileNet [212]. The concept of perceptual loss has been used in various computer vision applications such as the texture-synthesis and style-transfer by Johnson et al. [130] and Gatys et al. [213], [214], among many others [215]–[220]. The method we propose here is different because we are not compressing the network. Also, we are not asking the student to mimic the teacher because the teacher and the student are performing two different tasks: The teacher classifies clean data, whereas the student classifies noisy data. In low-light classification and object detection, student-teacher learning has not been applied.

7.2 Image classification

7.2.1 Related works

The majority of the existing work in classification is based on well-illuminated images. The first systematic study of the feasibility of low-light classification was presented by Chen and Perona [221], who observed that low-light classification is achievable by using a few photons. In the same year, Diamond et al. [222] proposed the "Dirty Pixels" method by training a denoiser and a classifier simultaneously. They observed that less aggressive denoisers are better for classification because the features are preserved. Other methods adopt similar strategies, e.g., using discrete cosine transform [223], training a classifier to help denoising [224] or using an ensemble method [225], or training a denoiser that are better suited for pre-trained classifiers [216], [226].

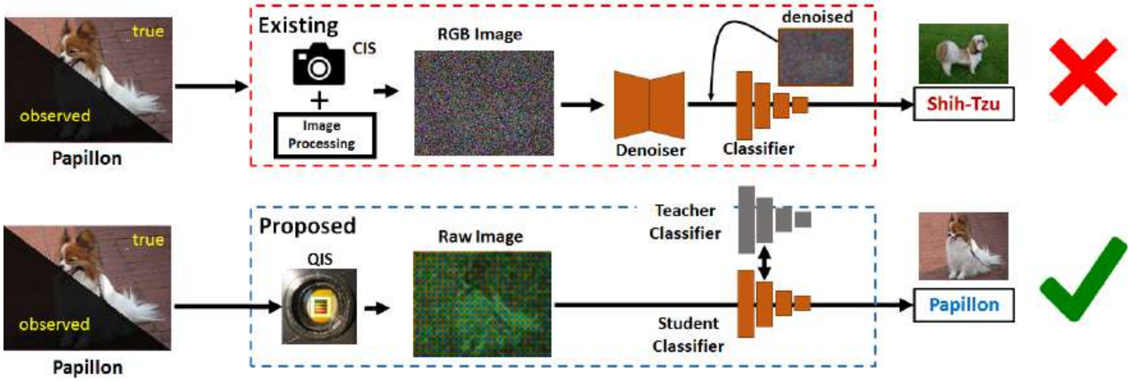


Figure 7.3. Existing classification pipeline vs. Proposed method.[Top] Traditional image classification methods are based on CMOS image sensors (CIS), followed by a denoiser-classifier pipeline. [Bottom] The proposed classification method comprises a QIS and a novel student-teacher learning protocol. QIS generates significantly stronger signals, and student-teacher learning improves the robustness against noise.

7.2.2 Method

In out method, the overall loss function for image classification comprises the perceptual loss and the conventional prediction loss using cross-entropy. The cross-entropy loss $\mathcal{L}_{\text{c}}$, measures the difference between true label y and the predicted label $f_{\boldsymbol{\Theta}}(\boldsymbol{y}_{\text{QIS}})$ generated by the student network, where $f_{\boldsymbol{\Theta}}$ is the student network. The overall loss is mathematically described as

$$\mathcal{L}(\boldsymbol{\Theta}) = \sum_{n=1}^{N} \left\{ \mathcal{L}_{\text{c}}\Big(\ell^n, f_{\boldsymbol{\Theta}}(\boldsymbol{y}_{\text{noisy}}^n)\Big) + \lambda \mathcal{L}_{\text{p}}\Big(\boldsymbol{y}_{\text{clean}}^n, \boldsymbol{y}_{\text{noisy}}^n\Big) \right\}, \tag{7.2}$$

where $\boldsymbol{y}^n$ denotes the n-th training sample with the ground truth label ℓ^n. During the training, we optimize the weights of the student network by solving

$$\widehat{\boldsymbol{\Theta}} = \underset{\boldsymbol{\Theta}}{\text{argmin}}\ \mathcal{L}(\boldsymbol{\Theta}). \tag{7.3}$$

During testing, we feed a testing sample $\boldsymbol{y}_{\text{noisy}}$ to the student network and evaluate the output:

$$\widehat{y} = f_{\widehat{\boldsymbol{\Theta}}}(\boldsymbol{y}_{\text{noisy}}). \tag{7.4}$$

Figure 7.4 illustrates the overall network architecture. In this figure, we emphasize that training is done on the student only. The teacher is fixed and is not trainable. In this particular example, we introduce a very shallow network consisting of 2 convolution layers with 32 and 3 filters respectively. This shallow network is used to perform the necessary demosaicking by converting the raw Bayer pattern to the full RGB before feeding into a standard classification network.

7.2.3 Experiments

Dataset

We consider two datasets. The first dataset (Animal) contains visually distinctive images where the class labels are far apart. The second dataset (Dog) contains visually similar images where the class labels are fine-grained. The two different datasets can help to differ-

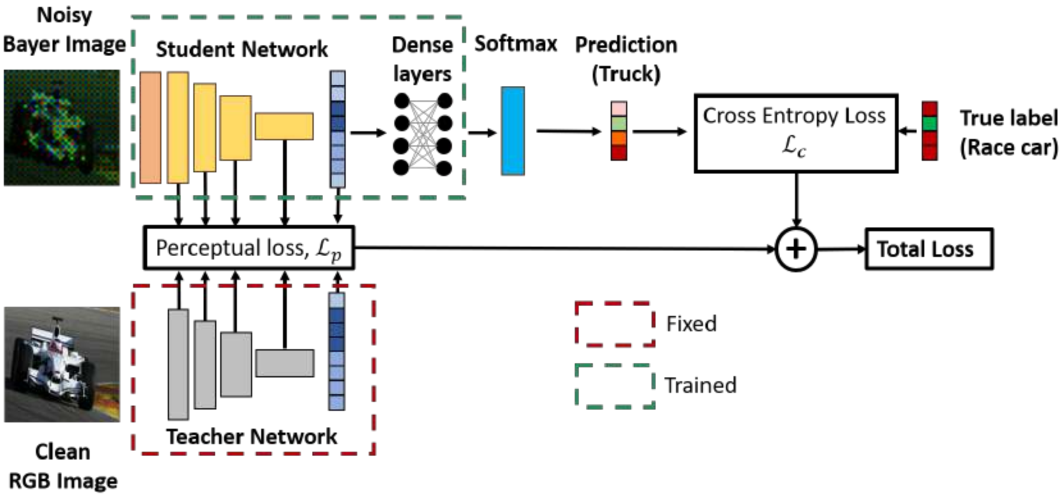
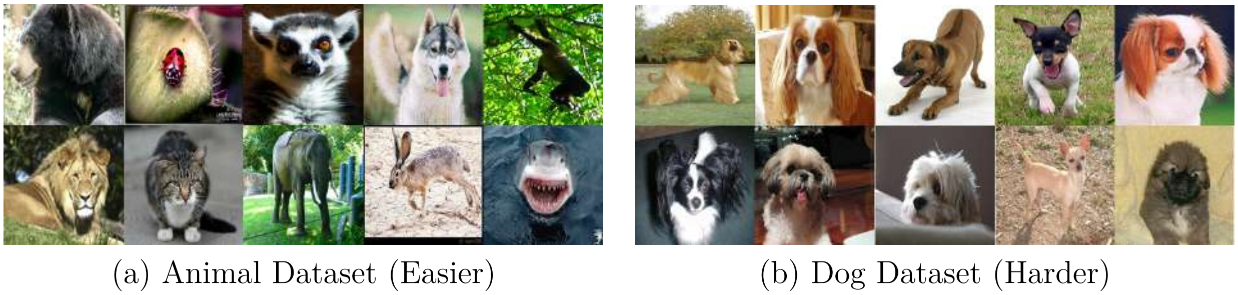


Figure 7.4. Proposed method. The proposed method trains a classification network with two training losses: (1) cross-entropy loss to measure the prediction quality, and (2) perceptual loss to transfer knowledge from teacher to student. During testing, only the student is used. We introduce a 2-layer entrance (colored in orange) for the student network so that the classifier can handle the Bayer image.

entiate the performance regime of the proposed method and its benefits over other state-of-the-art networks.


(a) Animal Dataset (Easier)

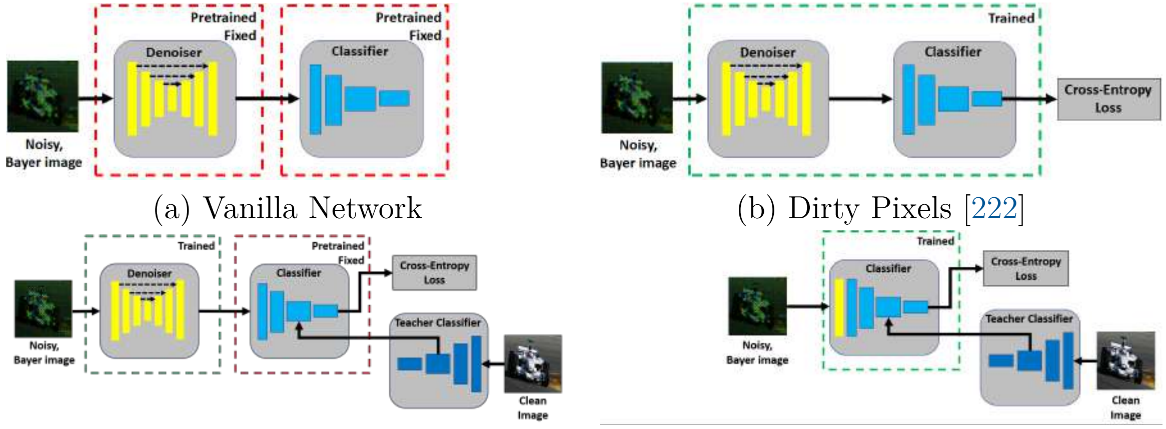
(b) Dog Dataset (Harder)

Figure 7.5. The two datasets for our experiments.

The construction of the two datasets is as follows. For the Animal dataset, we randomly select 10 classes of animals from ImageNet [227], as shown in Figure 7.5(a). Each class contains 1300 images, giving a total of 13K images. Among these, 9K are used for training, 1K for validation, and 3K for testing. For the Dog dataset, we randomly select 10 classes of dogs from the Stanford Dog dataset [228], as shown in Figure 7.5(b). Each class has approximately 150 images, giving a total of 1919 images. We use 1148 for training, 292 for validation, and 479 for testing.

7.2.4 Competing methods and our network

We compare our method with three existing low-light classification methods as shown in Figure 7.6. The three competing methods are (a) Vanilla denoiser + classifier, an “off-the-shelf” solution using pre-trained models. The denoiser is pre-trained on the QIS data, and the classifier is pre-trained on clean images. (b) Dirty Pixels [222], same as Vanilla denoiser + classifier, but trained end-to-end using our noisy data. (c) Restoration Network [216], [226], which trains a denoiser but uses a classifier pre-trained on clean images. Restoration Network can be viewed as a middle-ground solution between Vanilla and Dirty Pixels.



(a) Vanilla Network

(b) Dirty Pixels [222]

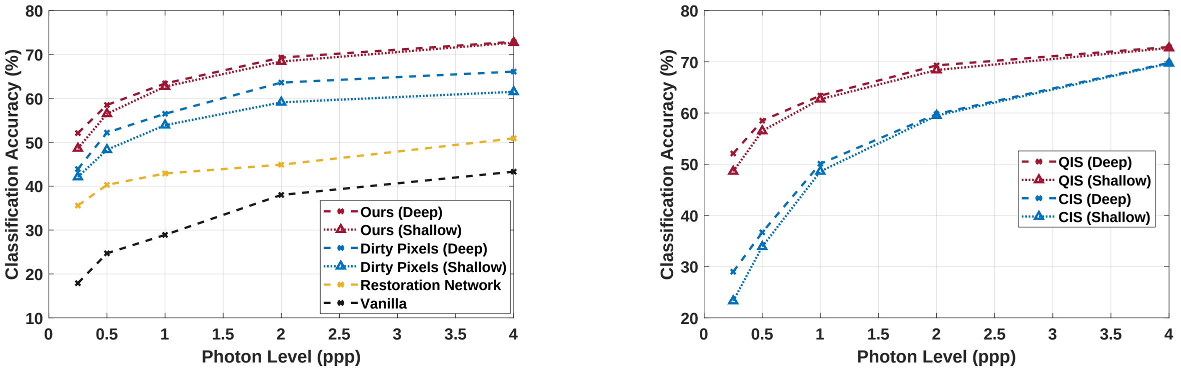
(c) Restoration Network [216]

(d) Proposed Method

Figure 7.6. Competing methods. The major difference between the networks are the trainable modules and the loss functions. For Dirty Pixels and our proposed method, we further split it into two versions: Using a deep denoiser or using a shallow entrance network.

To ensure that the comparison is fair w.r.t. the training protocol and not the architecture, all classifiers in this experiment (including ours) use the same VGG-16 architecture. For methods that use a denoiser, the denoiser is fixed as a UNet. This particular combination of denoiser and classifier will undoubtedly affect the final performance, but the effectiveness of the training protocol can still be observed. Combinations beyond the ones we report here can be found in the ablation study. For Dirty Pixels and our proposed method, we further split them into two versions: (i) Using a deep denoiser as the entrance, i.e., a 20-layer UNet, and (ii) using a shallow two-layer network as the entrance to handle the Bayer pattern, as we described in the proposed method section. We will analyze the influence of this component in the ablation study.

We also conduct experiments comparing how a quanta image sensor (QIS) may help with improving the classification accuracy compared to a CMOS image sensor (CIS).



(a) Different Classifiers

(b) Different Sensors

Figure 7.7. Synthetic data on dog dataset. (a) Comparing different classification methods using QIS as the sensor. (b) Comparing QIS and CIS using our proposed classifier.

7.2.5 Synthetic experiment

The first experiment is based on synthetic data. The training data are created using the QIS model. To simulate the QIS data, we follow the simulation model from chapter 3 by using the Poisson-Gaussian process. the read noise is $\sigma_{\text{read}} = 0.25\text{e}^{-}$ according to [46]. The analog-to-digital converter is set to 5 bits. We use a similar simulation procedure for CIS with the difference being the read noise, which we set to $\sigma_{\text{read}} = 2.0\text{e}^{-}$ [229].

The experiments are conducted for 5 different photon levels corresponding to 0.25, 0.5, 1, 2, and 4 photons per pixel (ppp). The loss function weights λ in Equation Equation (7.2) is tuned for optimal performance.

The results of the synthetic data experiment are shown in Figure 7.7. In Figure 7.7(a), we observe that our proposed classification is consistently better than competing methods the photon levels we tested. Moreover, since all methods reported in Figure 7.7(a) are using QIS as the sensor, the curves in Figure 7.7(a) reveal the effectiveness of just the classification method. In Figure 7.7(b), we compare the difference between using QIS and CIS. As we expect, CIS has worse performance compared to QIS.

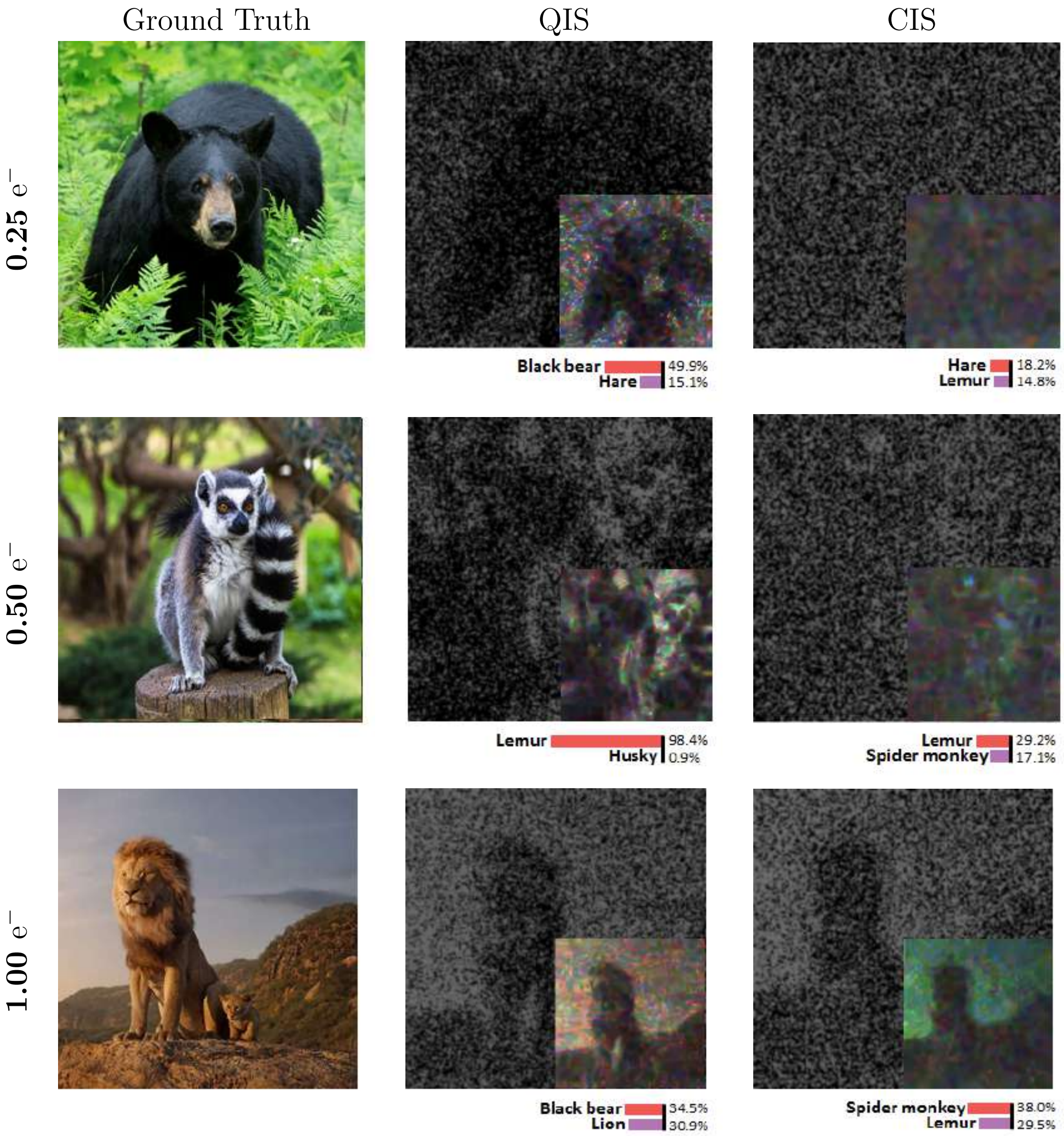
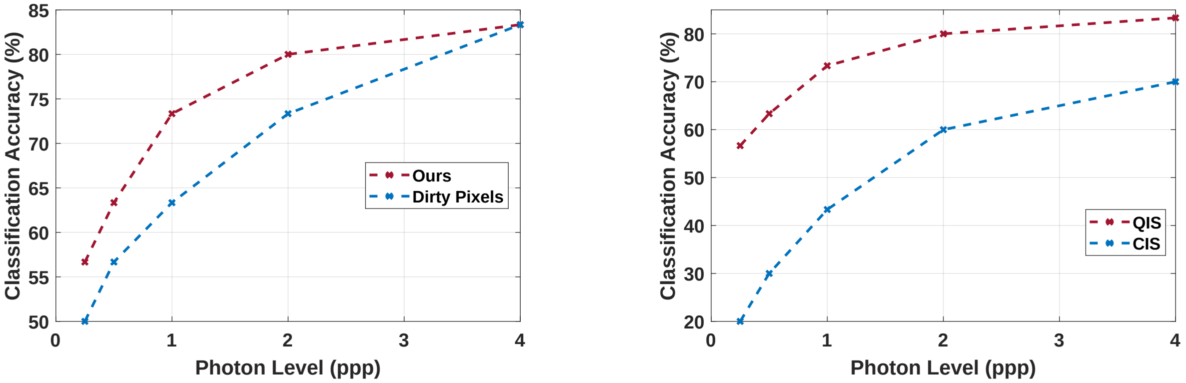


Figure 7.8. Real image results. This figure shows raw Bayer data obtained from a prototype QIS and a commercially available CIS, and how they are classified using our proposed classifier. The inset images show the denoised images (by [230]) for visualization. Notice the heavy noise at 0.25 and 0.5 ppp, only QIS plus our proposed classification method can produce the correct prediction.



(a) Different Classifiers

(b) Different Sensors

Figure 7.9. Real data on animal dataset. (a) Comparing different classification methods using QIS as the sensor. (b) Comparing QIS and CIS using our proposed classifier.

7.2.6 Real experiment

We conduct an experiment using real QIS and CIS data. The real QIS data are collected by a prototype QIS camera Gigajot PathFinder [21], whereas the real CIS data are collected by using a commercially available camera. We display the images on a Dell P2314H LED screen (60Hz). The cameras are positioned 1m from the display so that the field of view covers 256 × 256 pixels of the image. The integration time of the CIS is set to 250μs, and that of QIS is 75μs. Since the CIS and QIS have different lenses, we control their aperture sizes and the screen's brightness such that the average number of photons per pixel is equal for both sensors.

The network training in this real experiment is still done using the synthetic dataset, with the image formation model parameters matched with the actual sensor parameters. However, since the real image sensors have pixel non-uniformity, during the training, we multiply a random PRNU mask to each of the generated images to mimic the process of PRNU. We collect 30 real images at each photon level across 5 different photon levels for testing, which corresponds to 150 real testing images.

When testing, we make two pairs of comparisons: Proposed (shallow) versus Dirty Pixels (shallow), and QIS versus CIS. The result of the first experiment is shown in Figure 7.9(a), where we observe that the proposed method has a consistent improvement over Dirty Pixels. The comparison between QIS and CIS is shown in Figure 7.9(b). QIS has a better performance compared to CIS. Figure 7.8 shows the visualizations. The ground truth images were displayed on the screen, and the background images in QIS and CIS column are actual measurements from the corresponding cameras, cropped to 256 × 256. The thumbnail images in the front are the denoised images for reference. They are not used during the actual classification. The color bars at the bottom report the confidence level of the predicted class. Note the significant visual difference between QIS and CIS and the classification results.

7.2.7 Ablation study

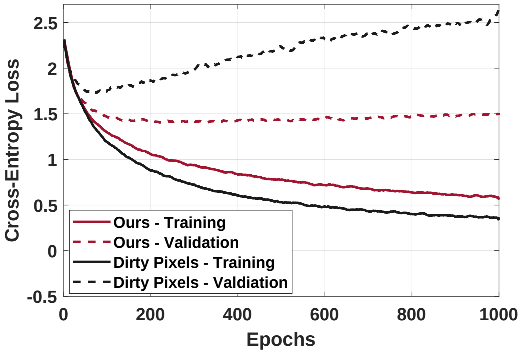
This section reports several ablation study results and highlights the most influencing factors to the design.

Sensor. Our first ablation study is fixing the classifier but changing the sensor from QIS to CIS. This experiment will underline the impact of the sensor on the overall pipeline. The result of this ablation study can be seen from Figure 7.7(b). Specifically, at 4 ppp (high photon level) of the Dogs dataset, QIS + proposed has a classification accuracy of 72.9% while CIS has 69.8%. The difference is 3.1%. As the photon level drops, the gap between QIS and CIS widens to 23.1% at 0.25 ppp. A similar trend is found in the Animals dataset. Thus at low light, QIS has a clear advantage, although CIS can catch up with a sufficient number of photons.

Classification pipeline. The next ablation study is to fix the sensor but change the entire classification pipeline, telling us how important the classifier is and which classifier is more effective. The results in Figure 7.7(a) show that among the competing methods, Dirty Pixels is the most promising one because it is end-to-end trained. However, comparing Dirty Pixels with our proposed method, at 1 ppp, Dirty Pixels (shallow) achieves an accuracy of 53.9% whereas the proposed (shallow) achieves 62.7%. The trend continues as the photon level increases. This ablation analysis shows that a good sensor (QIS) does not automatically translate to better performance.

Student-teacher learning. Let us fix the sensor and the network but change the training protocol, revealing the significance of the proposed student-teacher learning. To conduct this ablation study, we recognize that Dirty Pixels network structure (shallow and deep) is the same as Ours (shallow and deep) since both use the same UNet and VGG-16. The only difference is the training protocol, where ours uses student-teacher learning, and Dirty Pixels is a simple end-to-end. The result of this study is summarized in Figure 7.7(a). Our training protocol offers advantages over Dirty Pixels.

We can further analyze the situation by plotting the training and validation error. Figure 7.10 [Left] shows the comparison between the proposed method (shallow) and Dirty Pixels (shallow). If we look at the validation loss, we can see that it drops and then rises, whereas the training loss keeps dropping, which indicates that the network overfits without student-teacher learning (Dirty Pixels). In contrast, the proposed method appears to mitigate the overfitting issue. One possible reason is that student-teacher learning provides regularization in an implicit form so that the validation loss is maintained at a low level.



	VGG	ResNet	Inception
Baseline (fine-tuning)	42.1%	43.3%	44.3%
Proposed (student-teacher)	48.6%	49.1%	50.0%
Gap	+6.5%	+5.8%	+5.7%

Figure 7.10. Ablation study. [Left] Training and validation loss of Our method and Dirty Pixels. Notice that while our training loss is higher, the validation loss is significantly lower than Dirty Pixels. [Right] Ablation study of different classifiers and different training schemes. Reported numbers are based on QIS synthetic experiments at 0.25 ppp for the Dog Dataset.

Choice of classification network. All experiments reported in this section use VGG-16 as the classifier. In this ablation study, we replace the VGG-16 classifier with other popular classifiers, namely ResNet50 and InceptionV3. These networks are fine-tuned using QIS data. Figure 7.10 [Right] shows the comparisons. Using the baseline training scheme, i.e., simple fine-tuning as in Dirty Pixels, we can observe that there is a minor gap between the different classifiers. However, using the proposed student-teacher training protocol, we observe a substantial improvement for all the classifiers. This ablation study confirms that student-teacher learning is not limited to particular network architecture.

Using a pre-trained classifier. This ablation study analyzes the effect of using a pre-trained classifier (trained on clean images). If we do this, then the overall system is exactly the same as the Restoration network [216] in Figure 7.6(c). Restoration network has three training losses: (i) MSE to measure the image quality, (ii) Perceptual loss to measure feature quality, and (iii) cross-entropy loss. These three losses are used to train the denoiser and not the classifier. Since the classifier is fixed, it becomes necessary for the denoiser to produce high-quality images, or otherwise, the classifier will not work. The results in Figure 7.7(a) suggest that when the photon level is low, the denoiser fails to produce high-quality images, and so the classification fails. For example, at 0.25 ppp, Restoration Network achieves 35.6%, but our proposed method achieves 52.1%. Thus it is imperative that we re-train the classifier for low-light images.

Deep or shallow denoisers? This ablation study analyzes the impact of using a deep denoiser compared to a shallow entrance layer. The result of this study can be found by comparing Ours (deep) and Ours (shallow) in Figure 7.7(a), as well as Dirty (deep) and Dirty (shallow). In both methods, the deep version refers to using a 20-layer UNet, whereas the shallow version refers to using a 2-layer network. The result in Figure 7.7(a) suggests that while the deep denoiser has a significant impact on Dirty Pixels, its influence is quite small compared to the proposed method with the QIS images. One reason is that since we are using student-teacher learning, the features are already properly handled. Therefore, the added benefit from a deep denoiser for QIS is marginal. However, for CIS data at low light, the deep denoiser helps get better classification performance, especially when the signal level is much below the read noise.

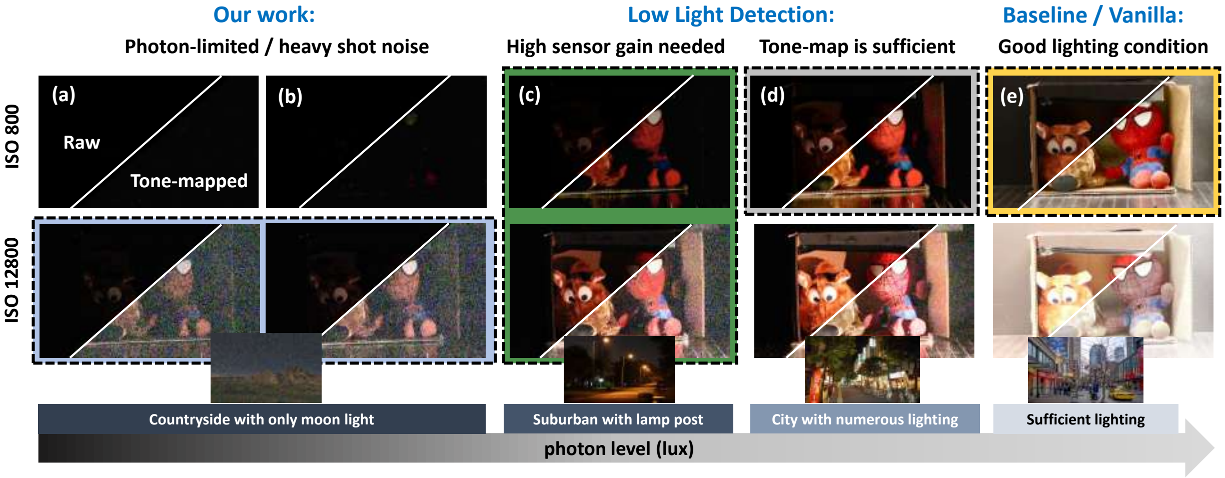


Figure 7.11. Realated works. While baseline/vanilla methods [231]–[247] are designed to handle well-illuminated scenes, this work focuses on the photon-limited regime where signals are very weak. Existing "low-light" methods [230], [248]–[250] typically do not operate in such an extreme condition where the signal is weak even after tone-map and/or adjusting the sensor's ISO.

7.3 Object detection

7.3.1 Related works

The taxonomy of the object detection methods is outlined in Figure 7.11, where we compare different detection tasks/methods against the photon-level (measured in lux) and the sensor gain (measured in ISO).

Baseline / Vanilla Methods

The mainstream object detection methods that are trained using large scale data set such as ILSVRC[227] and COCO[251] typically operate at the right most column of Figure 7.11 where the number of photons is sufficient. Depending on the input data format, the methods can be categorized into the following two groups:

Single-image detection methods that detect objects from a single image. Some of these methods focus on speed and real time processing capability [231]–[234], whereas other methods based on region proposal focus on detection performance [235]–[238]. On top of these methods, various work are proposed by leveraging multi-scale information [252], making

network fully convolutional [236], utilizing multi-task training [238], tackling foreground-background imbalance [232], and improving bounding box prediction quality [253], [254].
Video detection methods that detect objects from multiple frames of a video. The premise of these methods is that the temporal information and the spatial-temporal redundancy provides valuable information for the detection. The aggregation of temporal cues are typically done at two levels: (i) feature level aggregation [239]–[244], and (ii) box level aggregation [244]–[247].

Despite the abundance of baseline methods, the networks and training are not designed for photon-limited conditions. As a result, directly applying these methods to our problem is ineffective (performance is limited even if one augment training data) and inefficient (pre-processing could be computationally expensive but does not necessarily lead to unparalleled performance), as demonstrated in [25], [249] and in our experiment.

Low-light detection methods

Conventional low-light image processing methods can handle darker images than the baselines as shown in Figure **7.11**(c) and (d). The easier case, as shown in Figure **7.11**(d), happens when the lighting condition is not properly adjusted. However, information is mostly intact after tone-mapping and contrast enhancement. Image enhancement for this class of problem has been extensively studied **hasinoff2016_BurstPhotography**, [141], [255]–[267]. For object detection, Loh et al. [248] and Yang et al. [249] created large-scale real low light detection data sets. The state-of-the-art detection systems in this scenario adopt Multi-Scale Retinex with Color Restoration (MSRCR) algorithm [255] for pre-processing and fine tune detectors on pre-processed data [249]. As will be shown in the experiment section, this strategy fails to work on photon-limited images; the strong photon shot noise will void the illumination smoothness assumption held by the Retinex model.

The harder case of the two, as shown in Figure **7.11**(c), happens when the photon level is further reduced. In this operating regime, one needs to switch to a high sensor gain (higher ISO) so that the details can be observed. As far as object detection algorithms are concerned, to the best of our knowledge, no large scale detection dataset is available to date. Instead, Sasagawa et al. [250] treat detection in this scenario as a domain adaptation problem and use knowledge distillation to train a detector with normal lighting detection

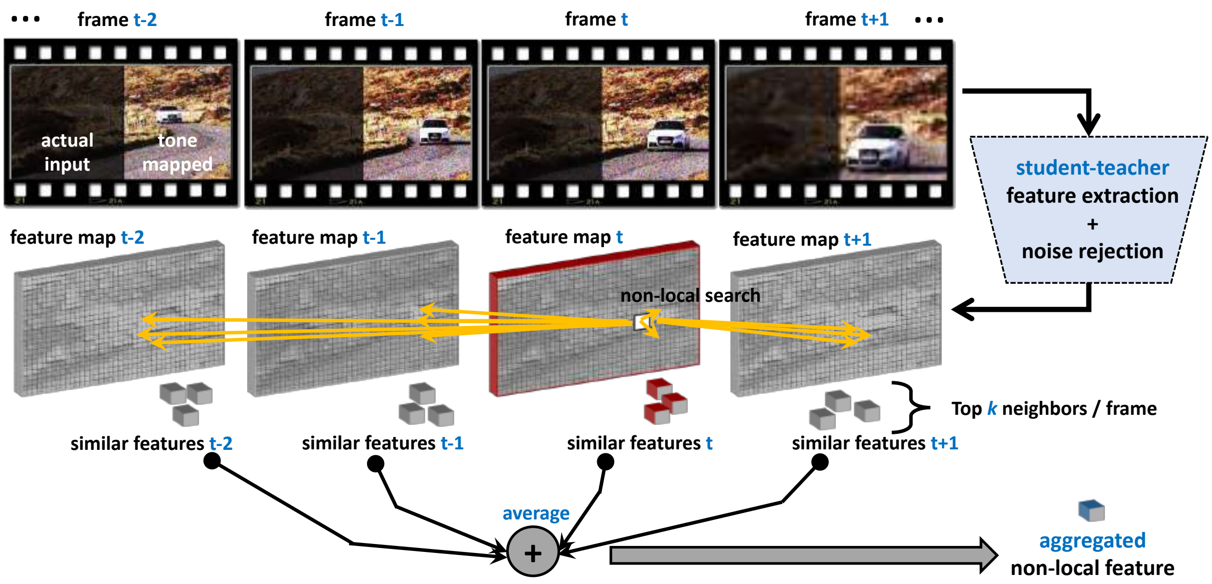


Figure 7.12. Proposed non-local module and student-teacher training scheme. The teacher network is first pre-trained on photon-abundant data and it enforces the student to extract noise-rejected features of each input frame. By applying the non-local search in the feature space, similar spatial-temporal features are aggregated to update the key frame features.

data and SID reconstruction data set [230]. In our study, we simulate the physical process of photon-limited image formation and demonstrate that our simulation enables our model to work on real photon-limited images.

7.3.2 Method

Given a sequence of photon-limited frames, our goal is to localize objects and identify their classes in *all* frames. Our proposed system is trained on data obtained from Chapter 3 and consists of key components: the non-local module (Sec 7.3.2) and the student-teaching learning scheme (Sec 7.1).

Space-time non-local module

The biggest challenge of detecting objects under photon-limited conditions is the presence of intense shot noise. Our solution to extract signals from the noise is to utilize the spatial-temporal redundancy across a burst of frames. Our hypothesis is that if we are able to find similar patches in the space-time volume, we can take a non-local average to boost the signal. To achieve this goal, we design a non-local module as depicted in Figure 7.12.

Given an image sequence, each frame is fed into a feature extractor (the student-teacher module) to obtain the feature maps. For each feature vector at location $(\mathrm{i}, \mathrm{j}, t)$, we conduct a non-local search for similar features by computing the inner-products of this feature and all the candidate features in the adjacent frames. This operation produces a set of scalars representing the similarities between the current feature and the features in the space-time neighborhood. Then for every time t, we select the top-k candidates with the highest inner product values. As shown in appendix Table 7.1, we find that $k = 2$ is an appropriate number for most of the experiments. After picking the top-k features, we take the average to generate the aggregated non-local feature.

Table 7.1. A study of frame numbers and searched similar feature numbers. T is the number of frames input to our model and K is the number of searched features per frame for feature aggregation. We test our model under different photon levels from 0.25 to 5.0. For each column, the best mAP is shown in bold.

	ppp = 0.25		ppp = 0.5		ppp = 1.0		ppp = 2.0		ppp = 5.0	
	$T=3$	$T=8$	$T=3$	$T=8$	$T=3$	$T=8$	$T=3$	$T=8$	$T=3$	$T=8$
$k=1$	32.3	**33.3**	41.5	42.8	49.6	51.9	58.4	59.0	65.1	**66.0**
$k=2$	**32.7**	33.2	**41.6**	**43.0**	**50.0**	51.9	**58.7**	**59.3**	**65.6**	**66.0**
$k=3$	32.4	33.2	41.5	42.8	49.9	**52.1**	58.6	59.2	65.4	65.9
$k=4$	32.5	33.0	41.5	**43.0**	**50.0**	**52.1**	58.6	59.1	65.4	65.9

Our proposed space-time non-local module differs from the traditional non-local neural networks [268] in the following two aspects:

- Before computing the similarity, [268] uses convolutional layers to first project features onto another feature space. This additional feature space is designed to represent high-level semantic meanings of the scene, such as interactions. For photon-limited imaging where the SNR is low, such semantic-level features are generally more corrupted and hence they are less reliable than low-level features. In addition, feature projection could cause confusion to our spatial-temporal feature matching step because the noise is heavy.

- [268] aggregates *all* space-time information via a softmax weighted average. We only average partially the space-time information from the top-k features because irrelevant features in the time-space can distract our model. In the Supplementary Material, we demonstrate that the top-2 features per frame are sufficient for our purpose.

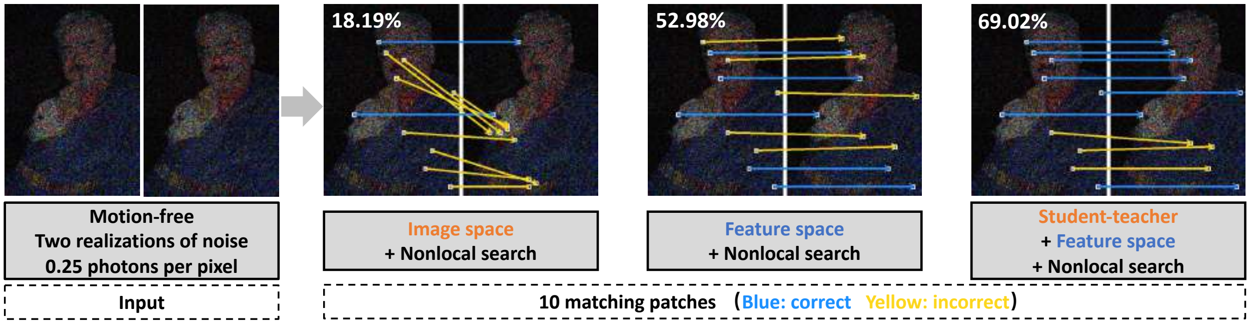


Figure 7.13. Comparison of different non-local patch matching methods. We synthesize two i.i.d. copies of a photon-limited image. For each competing configuration, we visualize 10 matching patch examples. The blue and yellow arrows indicate correct and incorrect matching, respectively. As the image pair is motion-free, the correct matches should be indicated by horizontal arrows. The combination of non-local search and student-teacher learning demonstrates the best performance.

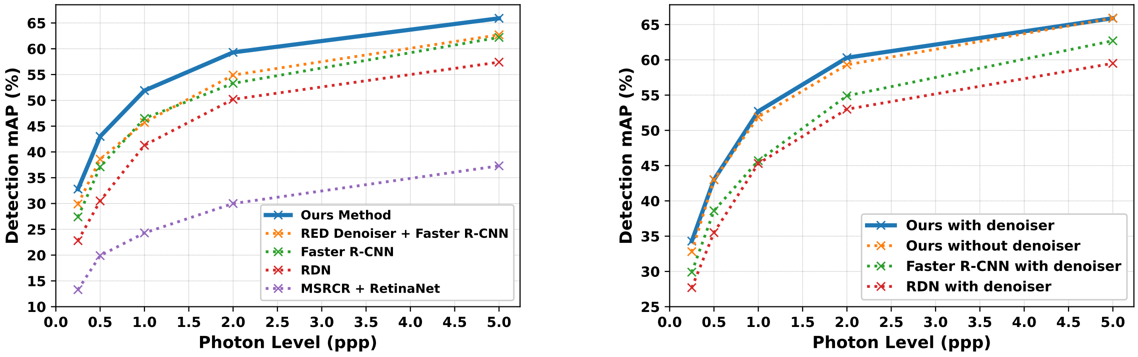
Rationale of our design

To illustrate the benefit of the proposed non-local module and the student-teacher learning scheme, we conduct an experiment in this section.

In Figure 7.12, we synthesize two independent and identically distributed (i.i.d.) copies of a photon-limited image at a photon level of 0.25 photons per pixel (ppp). We use this pair of images to check how the feature matching step performs. Three methods are compared: 1) Non-local search in the image space (i.e., the original non-local search), 2) non-local Search in the feature space, and 3) student-teacher + non-local Search in the feature space. In the image space, for each $h \times w$ patch, we compute its normalized cross-correlation (NCC) with all $h \times w$ patches in the other image and choose the one with the highest NCC as its matching patch. In the feature space, we use features trained with or without student-teacher training and find correspondence for every feature vector. The correspondence is visualized by the center of the receptive field of feature vectors.

The benefit of the proposed method can be seen in two aspects: accuracy and speed. As illustrated in Figure 7.13, the non-local search in the feature space has a much higher success rate of finding correct correspondence than the same method applied to the image space. The student-teacher training further increases the performance by enhancing the robustness of the feature extractor against noise. We performed the experiment for 100 images and we observed that the trend was consistent.

For the speed, non-local search in image space is computationally more expensive than in the feature space. Given an $H \times W$ image with desired patch size $h \times w$, the feature matching process takes approximately $(HW)^2 hw$ floating-point operations (FLOP) in the image space and $(\frac{HW}{S})^2 C$ FLOP's in the feature space, where C is feature vector dimension and S is spatial resolution compression ratio by the feature extractor. Reducing the patch size reduces the computation cost, but the matching quality deteriorates significantly. In our implementation, we use 64 × 64 for the image space search and it takes ∼ 256 times more computation than in the feature space.



(a) Comparison with baselines

(b) Comparison with image denoisers.

Figure 7.14. Experiments on synthetic data. (a) Compare different object detection methods: Faster R-CNN[237], RED[142] + Faster R-CNN[237], RDN[245], and MSRCR[255] + RetinaNet[232]. (b) Compare methods that use image denoising as a pre-processing step.

7.3.3 Experiments

Experimental settings

Dataset. We use the procedure in Chapter 3 to synthesize training data of the photon-limited images from the Pascal VOC 2007 dataset [148]. To synthesize motion across the frames, we introduce a random translation of image patches. The total movement varies from 7 to 35 pixels across 8 frames similar to Chapter 5. For testing, we created a synthetic testing dataset and also collected a dataset of real images. The read noise of our model is assumed to be 0.25e$^-$, based on the sensor reported in [46]. The average photon level we tested ranges from 0.1 to 5.0 photons per pixel (ppp). With an f/1.4 camera, 1.1μm pixel pitch, and 30ms integration, this range of photons roughly translates to 0.02 lux to 5 lux (typical night vision scenarios). For real data, we use the GJ01611 16MP photon counting Quanta Image Sensor developed by GigaJot Technology [46].

Implementation details. Our method is implemented in Pytorch based on [269]. The framework takes a T-frame image sequence as input ($T = 1, 3, 5$ and 8 in the following experiments). We adopt ResNet-101[17] pretrained on ImageNet [227] as the backbone. The perceputual loss is applied to the features from `block_1`, `block_2` and `block_3` of ResNet-101 and the non-local module is processed on the features from `block_3`. We utilize RoIAlign [238] to extract the features from object proposals and `block_4` is further applied to the extracted proposal features before the final classifier. The model is trained for 20 epochs and we use Adam [205] optimizer with default parameters, learning rate 0.001, and weight decay 0.1 every 5 epochs.

Competing methods. We compare our method with four baselines. (a) A generic image object detector: Faster R-CNN [237]; (b) A video object detector: Relation Distillation Network (RDN)[245]; (c) A low-light detection framework: color restoration algorithm (MSRCR) [255] plus a detection RetinaNet [232], which is one of the winning solutions of 2019 UG2$^+$ low-light face detection challenge; (d) A two-stage pre-denoised detection framework: RED-Net [142] plus Faster R-CNN [237]. (a) and (b) are fine-tuned using the synthesized photon-limited data.

7.3.4 Main results

Our first experiment is conducted on synthetic data. We use 8-frame inputs with the number of features for non-local aggregation set to 2 per frame in the following experiments.

Comparison with the baselines. Figure 7.14 (a) shows the detection rate, measured in mean average precision (mAP), as a function of the photon level, measured in photons per pixel (ppp). The proposed method consistently outperforms the competing methods across the tested photon levels from 0.25 ppp to 0.5 ppp. The difference between our method and the second-best method is as large as 6% in terms of mAP when the photon level is 2.0 ppp.

Comparison with image denoisers. When handling noisy images, a natural solution is to first run a denoiser and feed the denoised images into a standard object detector. Figure 7.14 (b) depicts the comparisons with such baseline methods. The denoiser we use is the RED-Net [142] previously used in other photon-limited imaging works such as Chapter 5 and [25]. As the figure indicates, the proposed method outperforms the baselines by a big margin. In addition, adding a denoiser to the proposed method offers almost no additional benefit. Therefore, the proposed method has effectively executed the denoising task without requiring another network for denoising.

Different network designs. Table 7.2 demonstrates the importance of the space-time non-local module and the student-teacher learning module. In this table, we present the relative performance gain compared with Faster R-CNN baseline [237]. The addition of the non-local module and the student-teacher training shows improvement upon the baseline. We observe that the performance gain shrinks when the photon level increases, as detection becomes easier. The combination of both designs shows the best performance across all photon levels, especially in extremely low light, where the relative gain is 20.07%.

Real data. We collected 225 real images in low light and annotate objects from 3 categories: `person`, `sheep`, and `car`. We train our model using the synthetic data and verify the results using the real data. The results are shown in Table 7.3. On average, our proposed method achieves an mAP of 87.9% while the baseline method achieves 66.9%.

Figure 7.15 shows a qualitative comparison between our method and the baseline Faster R-CNN. The result shows that the baseline suffers from either false alarms or missed de-

Table 7.2. Comparison of different network designs. Relative mAP increase are reported with respect to Faster R-CNN baseline. The unit is %. ST: student-teacher learning; NL: non-local module; ST+NL:student-teacher learning + non-local module.

Photon Level (ppp)	0.25	0.5	1.0	2.0	5.0
ST	9.12	6.20	4.52	5.44	2.57
NL	16.06	14.56	9.89	10.13	5.14
ST+NL	**20.07**	**15.90**	**11.61**	**11.26**	**5.95**

Table 7.3. Detection results of real data. Each class column shows the number of correct detections versus ground truth. The last column is the overall mAP.

	person	car	sheep	mAP (%)
Faster R-CNN	54/105	58/60	60/60	66.9
Ours	73/105	60/60	60/60	**87.9**

tection. In contrast, the proposed method is able to detect the static toy car and moving person on the real data when the photon level is 0.52 ppp and 0.19 ppp, respectively. More results are given in Figure 7.17 and Figure 7.18.

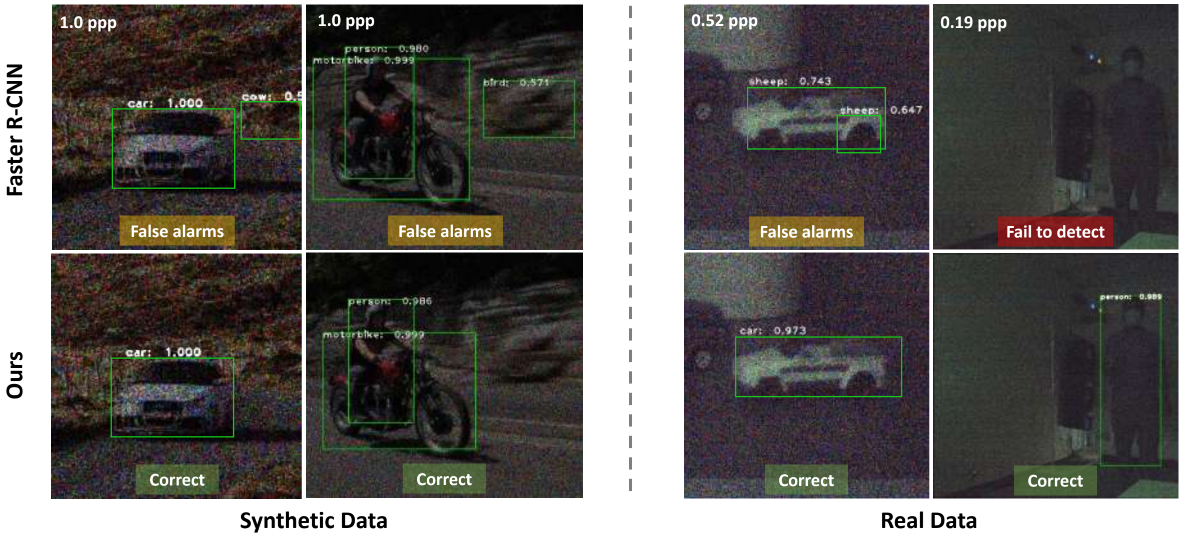


Figure 7.15. Detection results on synthetic and real data. The top row is the Faster R-CNN [237]. The bottom row is our method. The photon level is shown in the top-left corner. The real data is captured by Gigajot Technology 16 MP Photon Counting Quanta Image Sensor (GJ01611).

Performance comparison with CIS and QIS

We evaluate the proposed method with a conventional CMOS image sensor (CIS) from Google Pixel 3XL and a GJ01611 Quanta Image Sensor (QIS) from Gigajot Technology [35] under different illumination levels. By combining the proposed algorithm with the QIS device, we demonstrate the performance of the proposed detection method under extremely photon-limited conditions (0.02 lux and only 0.20 ppp).

To ensure a fair comparison, we note that the CIS has a pixel pitch of 1.4μm and read noise of 2.14e^{-}, while the QIS has 1.1μm pixels and read noise of 0.22e^{-}. In the experiments, the f-number of the lens is adjusted to balance the difference of pixel sizes (f/1.8 for CIS and f/1.4 for QIS) in the two sensors and 30msec exposure time is used for both sensors.

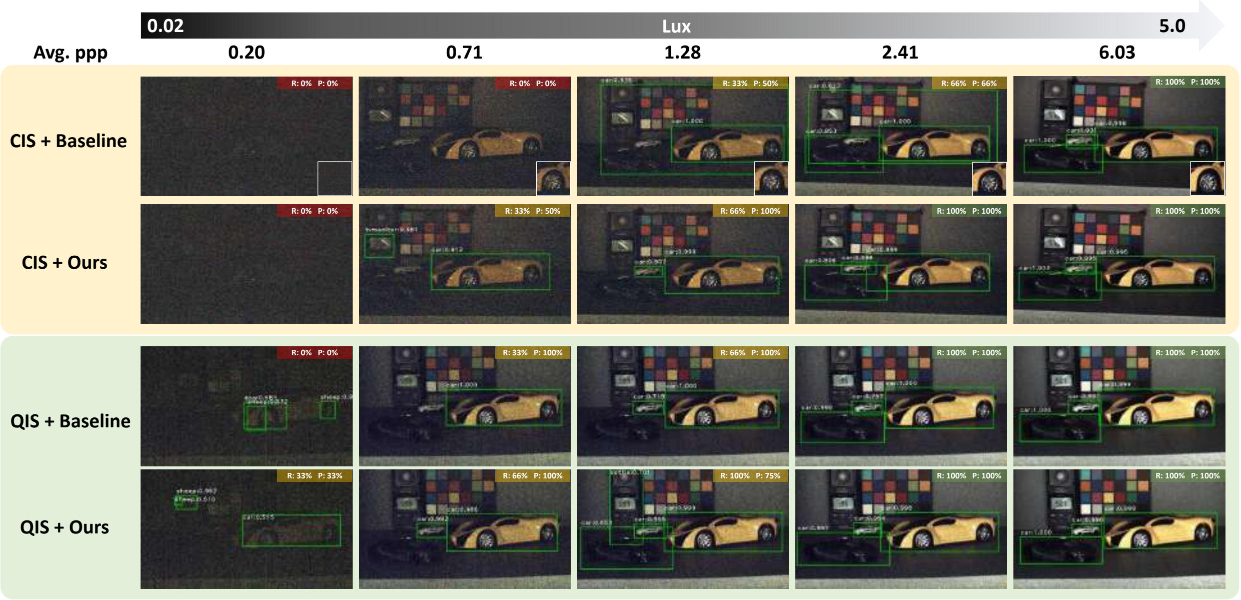


Figure 7.16. Comparison of different sensors and different methods on real data. The visualized figures are tone mapped and the baseline method is Faster R-CNN. We choose 5 different lux levels ranging from 0.02 to 5.0, equivalent to Avg. ppp ranging from 0.20 to 6.03. In the right-top corner of images, the recall (R) and precision (P) are computed, enclosed in frames with different colors. Red/Yellow/Green indicates totally failed/partially correct/totally correct, respectively. In the first row, we zoom into the left-front side of the yellow car and show details in the right-bottom box. We can see that in the extremely low light condition, the images suffer from the high-noise problem.

The comparison results are shown in Figure 7.16. The images were taken under illumination levels from 0.02 lux to 5.0 lux. Under strong illumination conditions such as 5.0 lux, all the compared methods show high detection accuracy without any false alarms. However, as the illumination level decreases, the proposed algorithm shows significant advantages over the baseline methods. This performance improvement is further enhanced with the QIS compared to the CIS because of its ultra-low read noise. For example, under 0.02 lux and an average photon level of 0.20 ppp, only the combination of the proposed algorithm and the QIS device can successfully detect the yellow car in the scene.

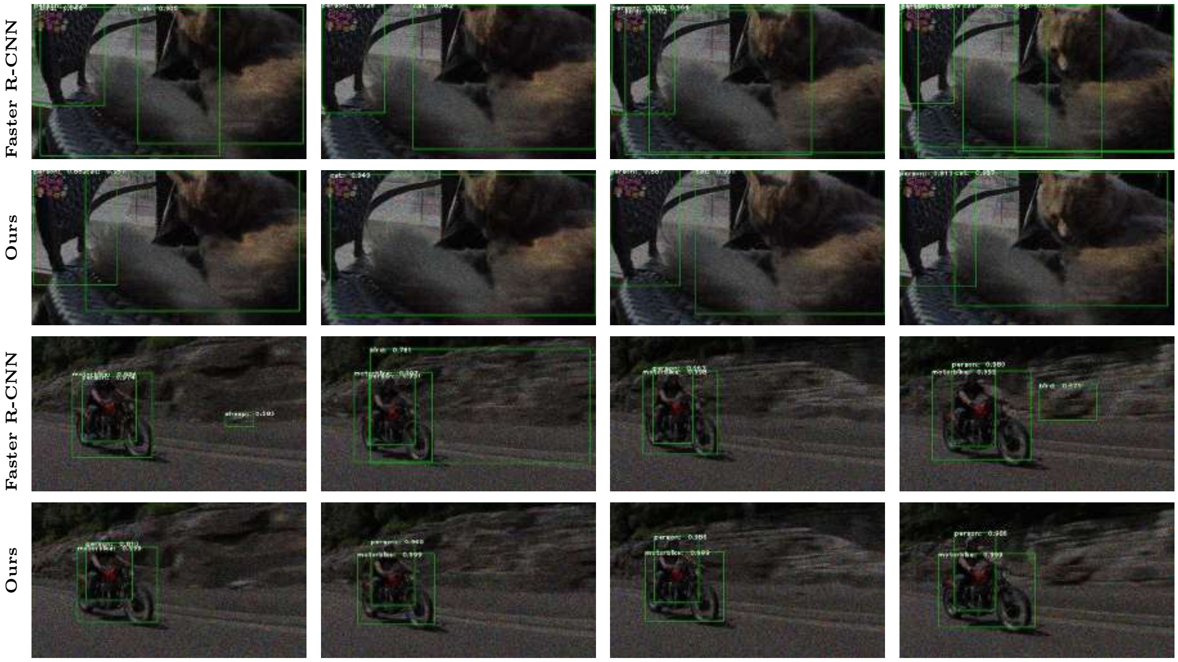


Figure 7.17. **More object detection results - Synthetic**.

7.4 Final thoughts

We have looked at how we can make the computer vision algorithms work in the presence of heavy shot noise. The existing computer vision solutions fail at these light levels as they cannot handle too much noise. We have proposed to use student-teacher learning to deal with this problem. Equipped with this idea, we have proposed a classification and a object

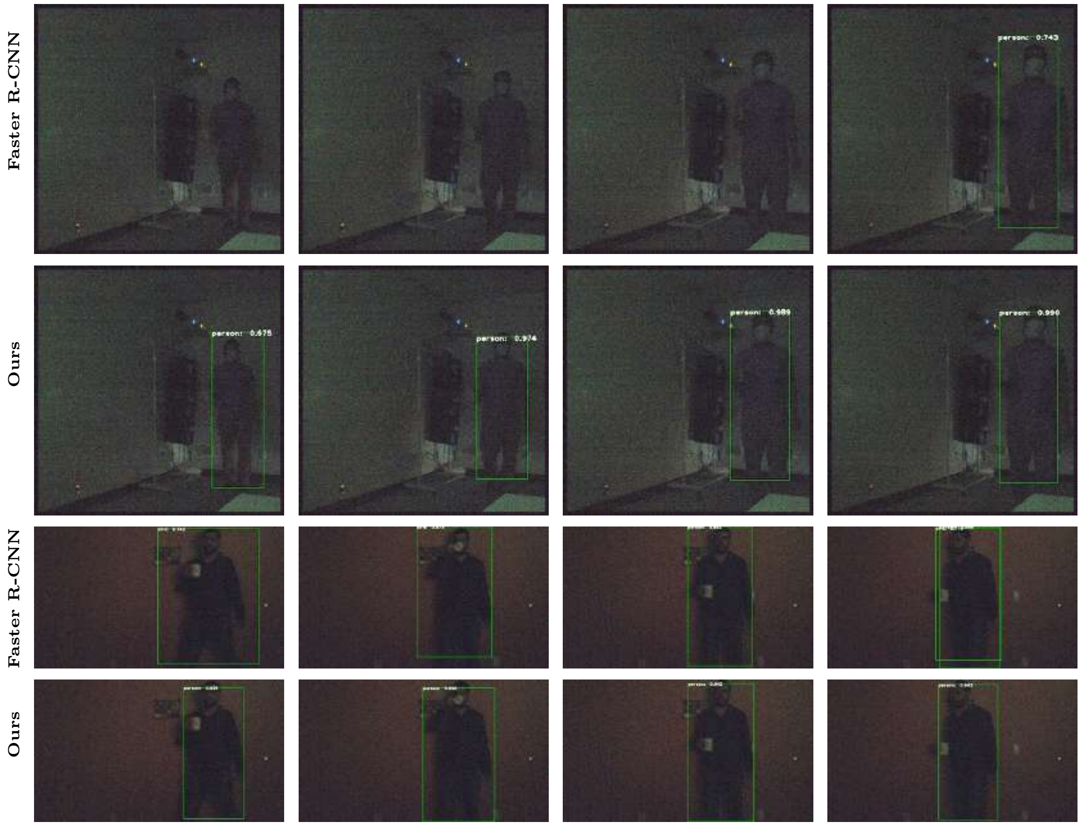


Figure 7.18. **More object detection results - Real**.

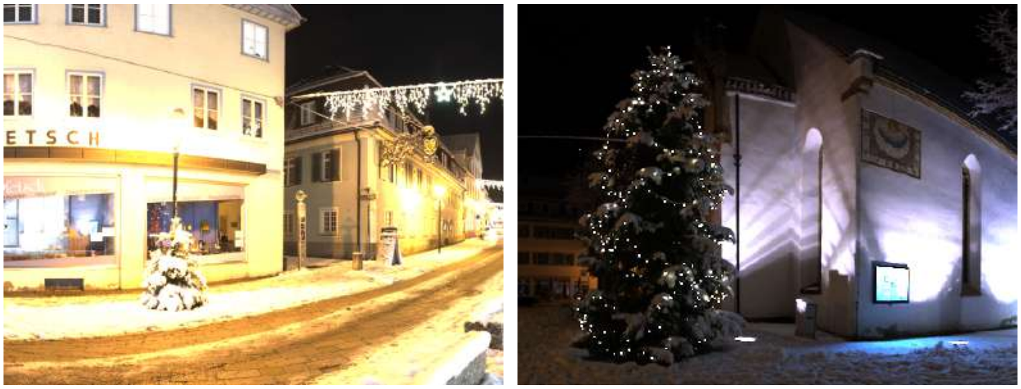
detection algorithm that beats the current state-of-the-art in photon limited computer vision at as low as photon per pixel light level.

7.4.1 Where to, from here?

What we have seen here is how to develop computer vision algorithms for specific camera settings when used at low light. However, these settings need not necessarily be the best configuration to use the sensor in. There may exist other settings such as higher frame rate or even using exposure bracketing and do high dynamic range. It is difficult making this decision before hand without knowing how the algorithm will behave. One way to get around this issue is to let the algorithm itself control the camera settings. If the algorithm can control the sensor based on feedback and also process the data to produce the computer vision results, it may lead to significantly better performance than what we are seeing right now. Given that the image sensors are becoming more programmable, making control algorithms will be a fruitful direction for future research.

8. DYNAMIC RANGE EXTENSION FOR QUANTA IMAGE SENSORS

Dynamic range refers to the range of light levels that a camera can detect. We will get into the formal definition in a bit, but before that, let us get a basic working idea of what dynamic range is. We have all taken photos where certain parts of the image are too bright that the details are washed out, or certain parts are too dark that we cannot see any details. Figure 8.1 shows a couple examples. This means that the light intensity from the pixels too dark or washed out is outside the dynamic range of the camera in whatever setting we are using it in. The signal-to-noise ratio of signals recorded at these pixels is too low to do a good reconstruction.



(a) Washed-out regions.

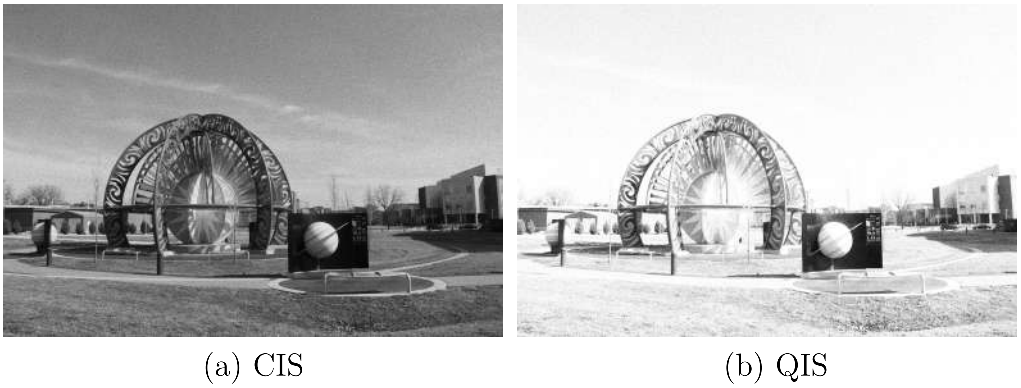
(b) Dark regions.

Figure 8.1. What is dynamic range? In (a), parts of the image around the street lamps are too bright, and the details are washed out. In (b), we cannot see any details behind the house, as it is too dark. This happens because the signals are outside the *dynamic range* of the image sensor. In an ideal world, we would like to recover details from all parts.

Historically, a significant amount of work has gone into increasing the dynamic range of the image sensors. As we just noted, there are two extremes to this problem, where we will lose details in the brighter and the darker regions. The pixels get washed out because the number of photons hitting at these pixels is more than the *full-well capacity*[1] of the image sensor. A typical way of extending the dynamic range on the brighter side is to increase the

[1] ↑Check 2 for a rigorous treatment of what full-well capacity is.

full-well capacity. The signals in the darker regions get lost because there is not enough light arriving at the sensor to cross the image sensor's noise floor to recover the signal properly. A typical way of solving this issue is to make sensors with better read noise and dark current. As we have seen multiple times in this dissertation, quanta image sensors are an excellent tool for extending the dynamic range on the darker side. However, the lower bit-depth severely constraints the full-well capacity of the sensor, which in turn affects the dynamic range in the brighter parts to a great extent. Compared to a single frame captured using a CIS, QIS will have a severely limited dynamic range even when the scene has only a few photons arriving at the sensor. Figure 8.2 shows one such example where only 10 photons on average are coming per pixel. Even at this light level, 3-bit QIS is heavily washed out compared to a CIS.


(a) CIS

(b) QIS

Figure 8.2. Dynamic range of quanta image sensors. The lower bit depth of QIS severely affects the full-well capacity and, in turn, the dynamic range of the image sensor. The QIS image is heavily washed out even at an average of 10 photons per pixel (ppp). However, all hope is not lost. Please keep reading to see how we can use QIS to get a dynamic range even more significant than a CIS.

Does this mean all hope is lost? If it is, then this chapter will end now, but as we can see, it keeps going on for more than a dozen pages. The answer is obviously no. The high-speed capabilities of QIS come to the rescue here, and we will show later that capturing multiple frames help significantly in extending the dynamic range for QIS. We even show that QIS will have a significantly higher dynamic range under certain conditions than the CIS.

8.1 Computational imaging for extending the dynamic range

The ways of extending the dynamic range we have discussed till now, are all hardware-based. Parallelly, there has been a lot of work going on to improve the dynamic range using computational imaging. We refer the readers to the texts by Banterle et al. [18] and Reinhard et al. [270] for an introduction to this subject. In general, these HDR imaging techniques can be categorized into three families: (i) exposure bracketing [75], [79], [271]–[273], (ii) coded exposure [274], [275], and (iii) burst photography **hasinoff2016_BurstPhotography**, [276], [277].

Exposure bracketing. We take multiple exposures of the scene in exposure bracketing, some with long and some with shorter exposures. Then, we use a carefully designed image processing algorithm to merge these differently exposed images to form the final image. Exposure bracketing is popular on hand-held devices because of its simplicity. The downside, however, is that it requires capturing many long and short exposure frames before the fusion step, and the overall acquisition time is thus long.

Despite the variety of exposure bracketing techniques, one thing that remains unchanged is the original linear combination idea. There are multiple ways of achieving linear reconstruction. One can combine the processed images instead of the raw image [79], [272], [278], or directly use the raw data [75], [279]–[281]. The choice of the combination weight also plays a critical role in HDR reconstruction. Mitsunaga et al. [79] choose weights proportional to the SNR to optimize the overall SNR of the combined image. Hasinoff et al. [282] propose two different ways to obtain an HDR image, either by maximizing the minimum SNR in the image or minimizing the overall time taken to obtain an image with a target SNR. In [279], Robertson et al. proved that the maximum likelihood estimate of the HDR estimate is the linear combination with weights inversely proportional to the variance of the signal. Granados et al. [75] extend the work of [279] by including different sources of noise in the model. Mertens et al. [283] combine LDR images without converting them into HDR values. Recently, several neural-network based HDR reconstruction methods that hallucinate HDR images from LDR images [284], [285] or use exposure bracketed images for reconstructing HDR dynamic scenes have also been proposed [286], [287].

CIS-based coded exposure. We can think of coded exposure as a modified version of exposure bracketing. Instead of capturing multiple frames with different exposures, we capture a single frame with different exposures and use carefully developed algorithms to combine these different exposures into a single image. Some representative works include [275], which proposed using spatially varying exposures to obtain HDR imaging in a single frame, and [274] which extended the idea by using convolutional sparse coding.

CIS-based burst photography. Burst photography aims to acquire a burst of short exposure frames so that all the frames are below the saturation limit. Then, by properly aligning the images (for object and camera motion), one can reconstruct an HDR image. Over the past few years, various burst photography algorithms are proposed, ranging from traditional motion alignment methods [288]–[290] to end-to-end deep learning methods [291]–[293]. However, the photon sensitivity of the sensors intrinsically limits the performance of CIS-based burst photography. As the exposure becomes short, the sensor's high read noise and dark current will prohibit the precise measurement of the signals. In addition, the random Poisson statistics of the photon arrivals pose a fundamental limit to CIS-based burst photography, which is difficult to be solved by image processing, including deep learning algorithms.

A lot of these were developed for CIS sensors. It is unclear how well they can port to a QIS. There are very few works like [294] answering this question. If they can be prted, can we use the existing algorithms for the reconstruction? These are a few of the questions we try to answer in this chapter. At the end of this chapter, we will have answers to the following questions:

1. Can we make QIS not too limited in their dynamic range, unlike Figure 8.2?
2. Can we quantify the dynamic range of QIS?
3. Do we get any dynamic range advantage over CIS by using QIS?
4. If we are to use exposure bracketing to increase the dynamic range of QIS, how do we reconstruct the scene from the data?

This chapter builds upon the theoretical work we did on the signal-to-noise ratio (SNR) in chapter 3, based on our work [27].

8.2 Oversampling

The first question is straightforward to answer. Oversampling in QIS helps in extending the dynamic range. Consider the following scenario. Let us take a CIS and QIS. Let the CIS have a full-well capacity of 4000 e^-. Assume QIS can capture 4000 frames when CIS takes a single capture. The sensor response of CIS and QIS under such a scenario, using theorem 3.3.3 is shown in Figure 8.3. We can see that QIS takes much longer to saturate under such a scenario than a CIS. This fact has been utilized in many recent works to capture high dynamic range images [53] using QIS.

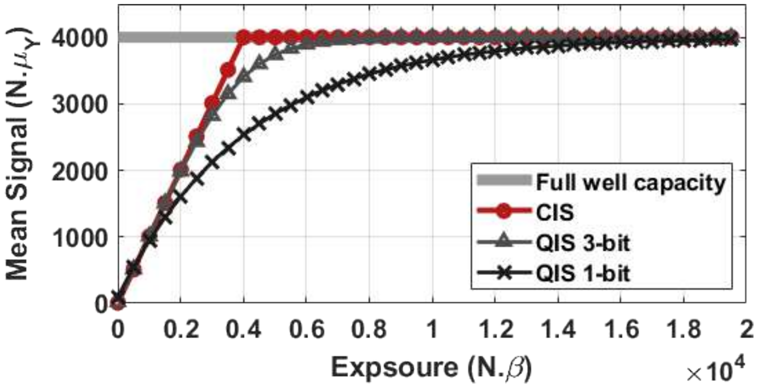


Figure 8.3. Sensor response. The mean signal $\mathrm{E}[Y[n]]$ of a CIS and a QIS, as a function of the exposure β.

8.3 Some HDR theory

8.3.1 Dynamic range

In this section, we theoretically derive the dynamic range of a QIS. Figure 8.5 shows the meaning of the dynamic range. The dynamic range is the range of the exposures where the signal-to-noise ratio (SNR) is above a certain threshold. Therefore, to analyze the dynamic range of QIS, we need to use the SNR defined in Chapter 3.

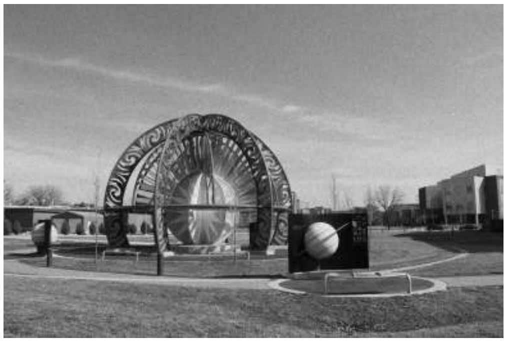


Figure 8.4. Extending dynamic range of QIS with oversampling. This figure repeats the experiment from Figure 8.2, but this time we oversample the scene, i.e., QIS now captures 20 frames instead of a single frame. Oversampling could be a solution for solving the dynamic range problem in QIS.

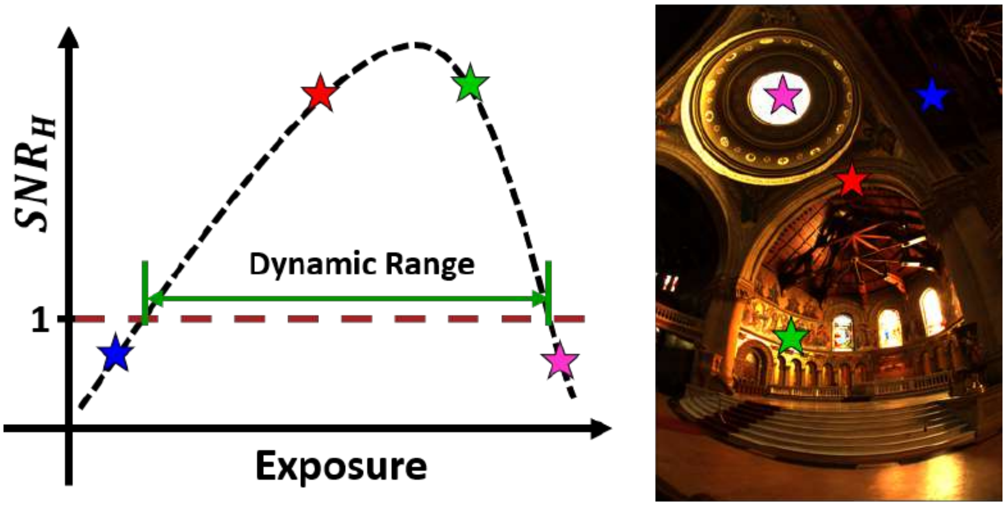


Figure 8.5. What is dynamic range? (Formal definition.) Dynamic range is the range of exposure that the sensor can detect. We define it as the range of exposure for which the SNR is greater than 1. If a particular part of a scene has lower exposure than this range, it will appear black. Similarly, excessive exposure may make a pixel appear saturated.

8.3.2 Exposure bracketing with QIS

Figure 8.6 demonstrates the idea that an additional advantage of oversampling is that we can do exposure bracketing among the frames, where we use a different integration time for each frame. The exposure bracketing helps in extending the dynamic range even further.

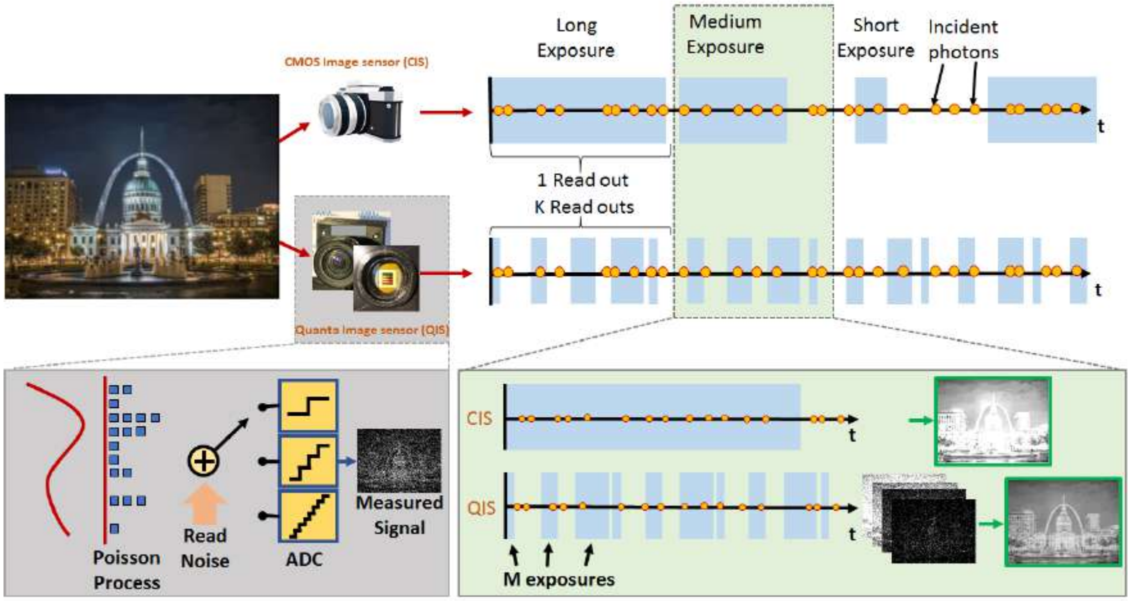


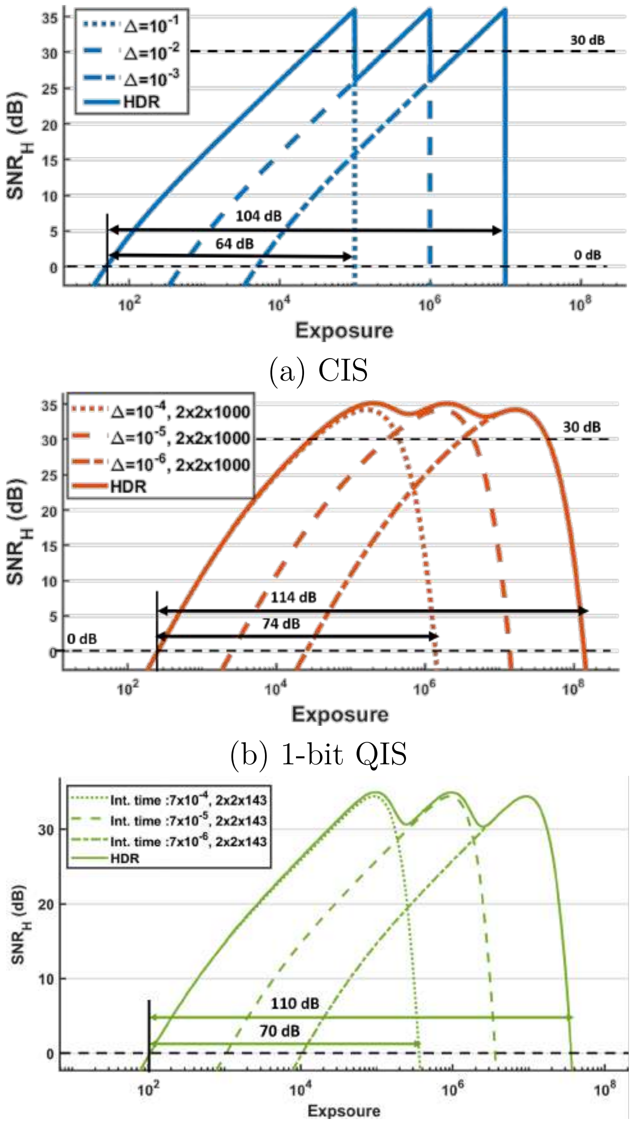
Figure 8.6. HDR imaging with quanta image sensors. Quanta Image Sensors can oversample the scene because of their significantly higher frame rate. In this chapter, we show that when we combine the oversampling ability of Quanta Image Sensors with exposure bracketing, the dynamic range achieved by the system far exceeds the dynamic range of the CMOS Image Sensors.

So, with QIS we are looking at a burst of N frames, divided into M groups of different integration times. Then the captured frames can be represented as

$$\mathcal{Y} = \Bigg\{ \underbrace{\boldsymbol{Y}[1], \ldots, \boldsymbol{Y}[m]}_{\text{Exposure 1}} \;\; \ldots \;\; \underbrace{\boldsymbol{Y}[n], \ldots, \boldsymbol{Y}[N]}_{\text{Exposure M}} \Bigg\}, \tag{8.1}$$

where each $\boldsymbol{Y}[n] = [Y_1[n], \ldots, Y_d[n]]$ is one captured frame. For the rest of this chapter, we assume that we are imaging a static scene, with photon rate of $\boldsymbol{\lambda} = [\lambda_1, \ldots, \lambda_d]$. So, for an integration time of Δ, the mean number of photons arriving at the sensor is

$$\boldsymbol{\beta} = [\beta_1, \ldots, \beta_d] = \boldsymbol{\lambda} \cdot \Delta. \tag{8.2}$$



(a) CIS

(b) 1-bit QIS

(c)3-bit QIS

Figure 8.7. Comparison of signal-to-noise ratio (SNR) for CIS and QIS. CIS is assumed to have a full well capacity of 4000 electrons. QIS is assumed to use a spatial oversampling of 2×2. The number of frames at each integration time is $N = 1000$ for single-bit QIS and $N = 143$ for 3-bit QIS. The oversampling is chosen such that the total signal obtained by both the CIS and QIS is the same. Notice that the QIS has a larger dynamic range than CIS for each exposure and has a more consistent SNR over the entire range when the low dynamic range images are combined to get a single high dynamic range image.

8.4 Comparing CIS and QIS

Using Theorem 3.3.3, we compare the dynamic range of a QIS and a CIS. Recall Figure 8.5. The dynamic range of a sensor is the range of exposures such that the SNR is above unity. Our goal here is to use the theoretical curves to predict how much dynamic range can be offered by each sensor.

Considering a typical setup of a CIS where the full-well capacity is $L = 4000$ e$^-$ and the read-noise is $\sigma_{\text{read}} = 2$e$^-$. We assume that the CIS uses three exposures to capture the image. For QIS, we operate in an oversampling regime by taking multiple short exposures of equal length. We assume that the read-noise of QIS is $\sigma_{\text{read}} = 0.25e^-$. For QIS, we sometimes do spatial oversampling too. The number of frames is configured such that the total duration of the acquisition is the same as a CIS. Afterward, we merge the short-exposure frames to generate an HDR image using the algorithm described in Section 8.5.

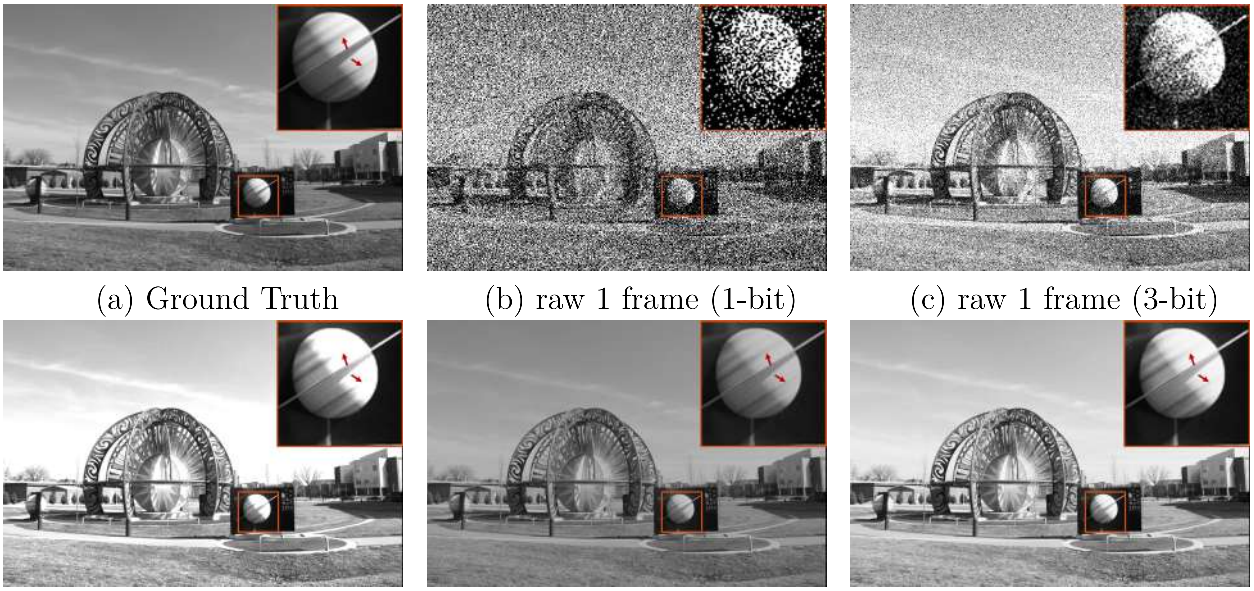

(a) Ground Truth
(b) raw 1 frame (1-bit)
(c) raw 1 frame (3-bit)
(d) CIS
(e) sum of 4000 frames (1-bit)
(f) sum 571 frames (3-bit)

Figure 8.8. Dynamic range of QIS and CIS. The image is simulated so that the maximum illumination of the image is 6×10^6 photons per pixel per second. CIS can count up to 4000 electrons, single bit QIS - 1 electron and 3-bit QIS - 7 electrons. The exposure times are CIS - 1ms, single bit QIS - $0.25\mu s$, and 3-bit QIS - $1.75\mu s$. We use 1 CIS frame, 4000 frames for single bit QIS and 571 frames for 3 bit QIS. We observe that CIS is saturated in the red arrowed regions, whereas QIS still shows the signal.

Figure 8.7 shows the theoretically predicted curves for CIS and 1-bit QIS. For CIS, we show three different integration time $\Delta = 10^{-1}$sec, 10^{-2}sec and 10^{-3}sec. The HDR image formed by a CIS is the sum of the three exposures. QIS with an integration time of 10^{-4} uses an oversampling of $2 \times 2 \times 1000$, meaning a spatial oversampling of a 2×2 bin and 1000 frames of 1-bit measurements. The HDR versions of the QIS data are obtained using the reconstruction method described in section 8.5.

As we can observe from Figure 8.7, the dynamic range of a QIS using just one integration time is 74dB (1-bit), which is already substantially more significant than the 64dB of a CIS. After reconstructing the HDR image by merging multiple integration times, the resulting dynamic range offered by a QIS is also higher than that of a CIS. Also, Figure 8.7 shows that the SNR of a combined QIS image never drops below 30dB, which is a big contrast to CIS, which suddenly drops once the exposure exceeds the full-well capacity. Figure 8.8 illustrates the visual comparison between a CIS and a QIS. Notice that the QIS offers better details for the same amount of photons than a CIS.

If we look at the low-light ends of Figure 8.7, we observe that CIS is performing better than a QIS. This phenomenon is the result of accumulating read noise from adding multiple frames. Since, every readout of a QIS frame has a fixed amount of read noise, the more readouts we do the more read noise we accumulate. In Figure 8.9 we demonstrate this problem. Assuming a read noise level of $\sigma_{\text{read}} = 0.25$, and an integration time of $\Delta = 0.2$sec (or $\Delta = 0.02$sec), we plot the sum of N frames of 3-bit frames. As the number of frames N increases, with the total integration time fixed, the image becomes noisier when $\sigma_{\text{read}} = 0.25$. This is not visible, when $\sigma_{\text{read}} = 0$ or $\sigma_{\text{read}} = 0.15$, because the gaussian read noise is not strong enough to cause any issues with multiple read-outs.

8.4.1 SNR vs dynamic range trade-off

QIS offers a trade-off between the peak SNR and the dynamic range. Figure 8.10 shows four sets of curves: (i) a CIS running in LDR mode. The dynamic range is 64dB. (ii) A QIS operating in LDR mode, with $N = 4000$ frames, which is equivalent to a CIS exposure. The dynamic range is 74dB, but the peak SNR is slightly lower than CIS. (iii) A QIS operating

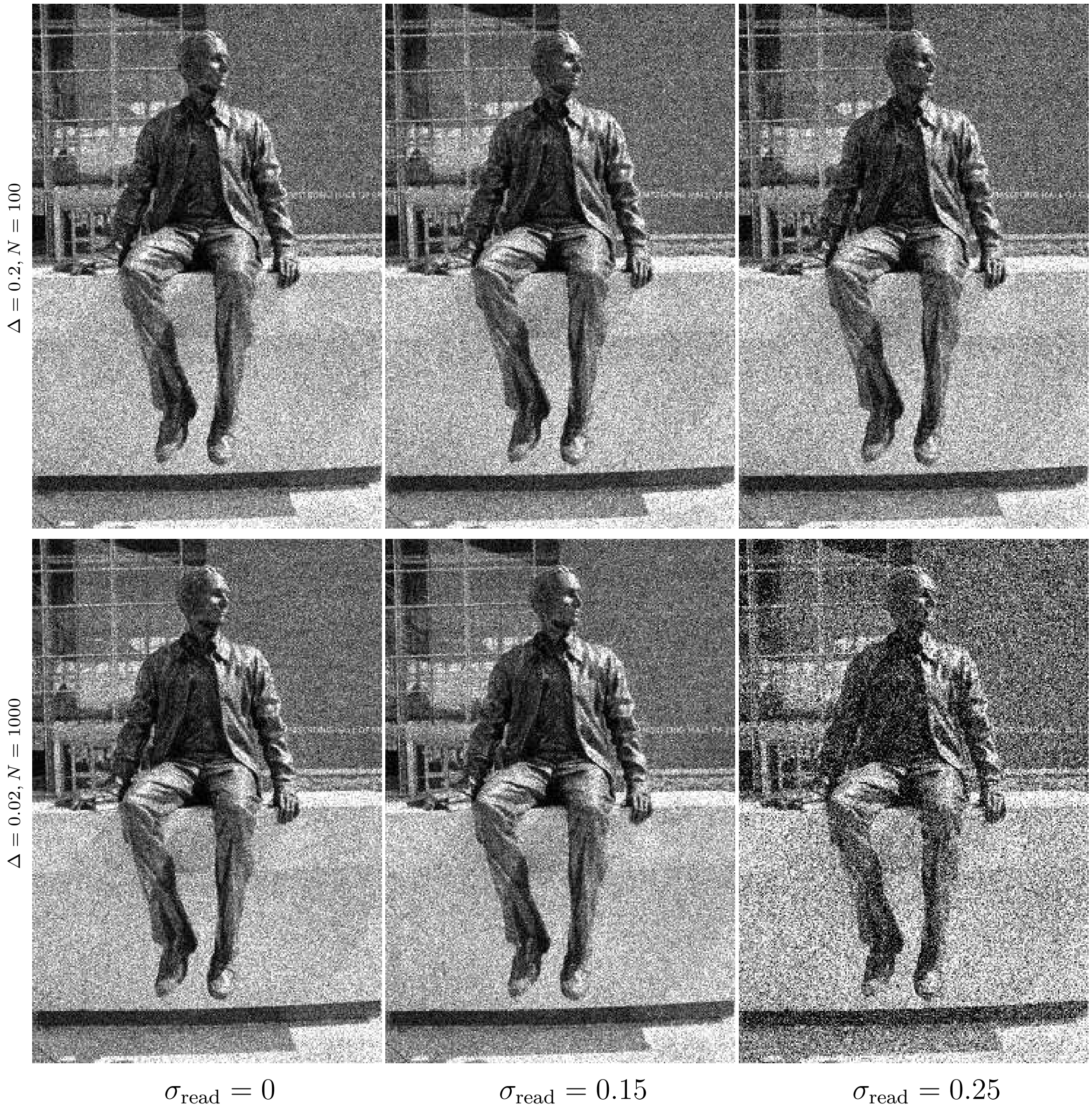


Figure 8.9. Accumulation of noise. The sub-figures show the sum of N simulated QIS frames when using different N and different integration times Δ, such that the total integration time $N\Delta$ is equal for all the cases considered. Because of the finite read noise, by summing more frames, we accumulate error, which in turn leads to a trade-off between the number of frames and the SNR when the total integration time is fixed.

in an LDR mode, with $N = 1000$ frames, which is a much weaker signal than a CIS. (iv) A QIS operating in the way as the previous case, but this time we merge four different LDR images to create an HDR image. We observe that the dynamic range goes up to 127dB

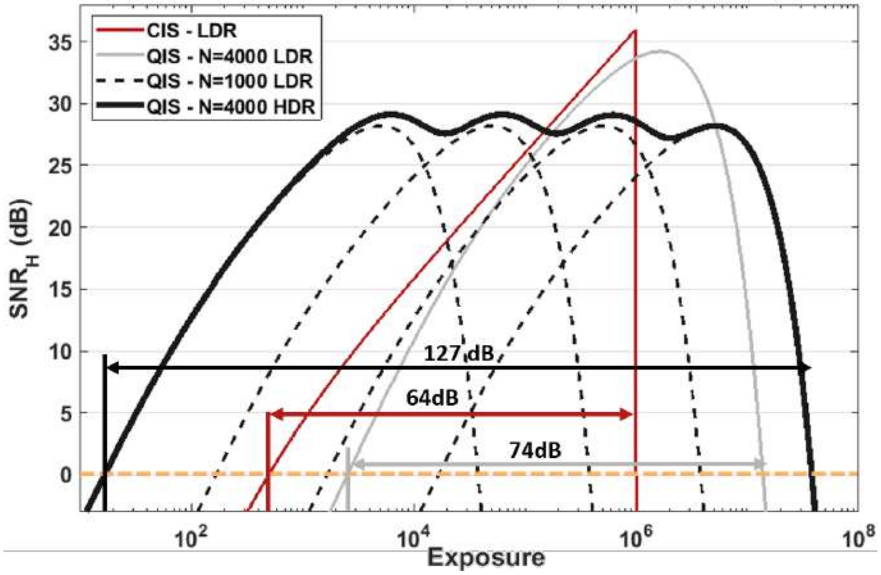


Figure 8.10. SNR vs. dynamicrange tradeoff. QIS offers a unique tradeoff where we can choose a setting based on whether we want an image with a very high SNR or sizeable dynamic range. This figure shows that QIS can operate under an LDR regime with comparable SNR to a CIS or HDR regime, where the dynamic range is significantly higher.

which is 63dB higher than CIS. However, the overall peak of this HDR image is still lower than that of a CIS owing to the lower peak offered by an individual LDR.

This figure shows that we can trade off the peak SNR and the dynamic range of a QIS by controlling the exposure pattern, e.g., using fewer but longer exposures or more but shorter exposures. This flexibility can be of importance for various imaging applications.

8.5 HDR reconstruction for QIS

The significance of Theorem 3.3.3 is two-fold. Till now, we have only seen how it informs us of the SNR and hence the dynamic range of QIS. In this section, we use Theorem 3.3.3 to derive an optimal linear HDR reconstruction algorithm.

8.5.1 Exposure bracketing

Before we discuss the problem formulation, we should first comment on how a conventional CIS performs HDR reconstruction. To a large extent, conventional HDR methods are based on the concept of exposure bracketing [295]. Given a stack of differently exposed images, we construct a *linear combination* of the images to create the final image. Putting this mathematically, if we denote $\boldsymbol{Y}[1], \ldots, \boldsymbol{Y}[N]$ as a sequence of N differently exposed images, then the HDR image $\widehat{\boldsymbol{\lambda}}$ is

$$\widehat{\boldsymbol{\lambda}} = \sum_{n=1}^{N} \boldsymbol{w}[n] \odot \boldsymbol{Y}[n], \tag{8.3}$$

where $\odot$ denotes the element-wise multiplication, and $\{\boldsymbol{w}[n]\}$ is a sequence of weight vectors satisfying the constraint that $\sum_{n=1}^{N} \boldsymbol{w}[n] = \mathbf{1}$. Because the reconstructed image $\widehat{\boldsymbol{\lambda}}$ is the linear combination of the input frames, we call such an exposure bracketing technique a *linear* reconstruction method. We follow the literature by deriving the theoretical results for static scenes.

8.5.2 Optimal weights for QIS

Without loss of generality, let us assume that the QIS has acquired a stack of frames as given by Equation (8.1), where the frames are grouped into M different index sets of exposures $E_1, \ldots, E_M$. For example, the set E_1 contains all the indices of the frames that have used the exposure Δ_1. We further assume that each E_m contains K frames for simplicity. So for M exposures, each having K frames, the total number of frames is $N = KM$.

To make the notation simple we focus only on one pixel. The average number of photons obtained by each exposure Δ_m is

$$\beta[m] = \Delta_m \lambda, \tag{8.4}$$

Intuitively, since the flux λ is constant, the average number of photons is proportional to the exposure time Δ_m.

Following the camera model from Chapter 3, each $\beta[m]$ will generate K observations $Y[n_1], \ldots, Y[n_K]$. Depending on the ADC, each $Y[n]$ can be a one-bit or a multi-bit Poisson random variable. The mean and variance of each $Y[n]$ are respectively defined as

$$\mu_Y[m] \overset{\text{def}}{=} \mathrm{E}\left[Y[n]\right], \quad \text{and} \quad \sigma_Y^2[m] \overset{\text{def}}{=} \mathrm{Var}\left[Y[n]\right], \tag{8.5}$$

for $n \in E_m$, where $m = 1, \ldots, M$. Essentially, this equation says that when we divide the exposures into M groups, we have M different means and variances.

Note that $\mu_Y[m]$ is a function of $\beta[m]$; $Y[n]$ is the truncated Poisson random variable according to the QIS model, and so $\mu_Y[m]$ must be a function of the underlying average photon count $\beta[m]$. Denoting $\mu_Y[m] = f(\beta[m])$ for some function f, it holds that $\beta[m] = f^{-1}(\mu_Y[m])$. For example, if the ADC is 1-bit and $\sigma_{\text{read}} = 0$, then

$$\mu_Y[m] = 1 - \mathrm{e}^{-\beta[m]} \overset{\text{def}}{=} f(\beta[m]),$$

and so $f^{-1}(\mu_Y[m]) = -\log(1 - \mu_Y[m])$. As mentioned in [76], this can be regarded as a tone-mapping.

As far as estimation is concerned, we reconstruct a low dynamic range (LDR) image from a stack of K frames of the same exposure. We thus define the sum as

$$S[m] \overset{\text{def}}{=} \frac{K}{\Delta_m} f^{-1}\left(\frac{1}{K} \sum_{n \in E_m} Y[n]\right). \tag{8.6}$$

Here, the quantity inside f^{-1} is the average of the frames. f^{-1} resolves the tone-mapping. The normalization $1/\Delta_m$ ensures that $S[m]$ is properly scaled with respect to the exposure time.

To construct the HDR image, we consider using a linear combination scheme by defining

$$\widehat{\lambda} = \sum_{m=1}^{M} w[m] S[m], \tag{8.7}$$

where $w[m] \in \mathbf{R}$ is a weight satisfying the property that $\sum_{m=1}^{M} w[m] = 1$. Because of the weighted averaging instead of a simple sum, the exposure referred SNR for this estimator $\widehat{\lambda}$ takes a generalized form of

$$\text{SNR}_{\text{H}}(S) = \sqrt{N} \frac{\beta}{\sigma_Y} \frac{d\mu_Y}{d\beta}. \tag{8.8}$$

Specifically, since each exposure has K frames, the signal in the numerator of Equation (8.8) is

$$\text{signal}^{\text{HDR}} = K\lambda.$$

The denominator of Equation (8.8), which is the exposure-referred noise, becomes

$$\text{noise}^{\text{HDR}} = \sqrt{\sum_{m=1}^{M} \left(\frac{w[m]}{\Delta_m}\right)^2 \sigma_{\text{H}}^2[m]}, \tag{8.9}$$

where $\sigma_{\text{H}}[m]$ is the exposure-referred noise standard deviation of the m-th exposure [2]:

$$\sigma_{\text{H}}[m] = \sqrt{K}\sigma_Y[m] \cdot \frac{d\beta[m]}{d\mu_Y[m]}. \tag{8.10}$$

Here, $\sigma_Y[m]$ and $\frac{d\beta[m]}{d\mu_Y[m]}$ follow from Theorem 3.3.3, where for each exposure m there is a different $\sigma_Y[m]$ and $\frac{d\beta[m]}{d\mu_Y[m]}$. Taking the ratio between signal$^{\text{HDR}}$ and noise$^{\text{HDR}}$ gives us the overall SNR of the HDR image:

$$\text{SNR}_{\text{H}}^{\text{HDR}} = \frac{K\lambda}{\sqrt{\sum_{m=1}^{M} \left(\frac{w[m]}{\Delta_m}\right)^2 \sigma_{\text{H}}^2[m]}}. \tag{8.11}$$

The optimization problem is to find the optimal weights $w[1], \ldots, w[M]$ such that $\text{SNR}_{\text{H}}^{\text{HDR}}$ is maximized. This gives the following constrained problem:

$$\begin{aligned} \underset{w[1],\ldots,w[M]}{\text{maximize}} \quad & \frac{K\lambda}{\sqrt{\sum_{m=1}^{M} \left(\frac{w[m]}{\Delta_m}\right)^2 \sigma_{\text{H}}^2[m]}} \\ \text{subject to} \quad & \sum_{m=1}^{M} w[m] = 1, \text{ and } w[m] \geq 0. \end{aligned} \tag{8.12}$$

To specify the solution of this optimization problem, we define the m-th SNR as

$$\text{SNR}_{\text{H}}[m] \overset{\text{def}}{=} \frac{\beta[m]}{\sigma_{\text{H}}[m]} = \frac{\Delta_m \lambda}{\sigma_{\text{H}}[m]}. \tag{8.13}$$

[2] ↑$\sigma_{\text{H}}[m]$ can be obtained by taking the variance of $\widehat{\lambda}$. The derivative appears as a result of applying the delta method to $f^{-1}(S[m])$.

With this definition, we can determine the solution.

Theorem 8.5.1 (Optimal linear combination)**.** *The optimal weights $w[1], \ldots, w[M]$ which solves the optimization problem Equation (8.12) is given by*

$$w[m] = \frac{SNR_H^2[m]}{\sum_{m=1}^{M} SNR_H^2[m]}, \tag{8.14}$$

where $SNR_H[m]$ is defined by Equation (8.13).

Proof. The optimization problem is

$$\begin{aligned} &\underset{w[1],\ldots,w[M]}{\text{maximize}} && \frac{K\lambda}{\sqrt{\sum_{m=1}^{M}\left(\frac{w[m]}{\Delta_m}\right)^2 \sigma_{\mathrm{H}}^2[m]}} \\ &\text{subject to} && \sum_{m=1}^{M} w[m] = 1, \text{ and } w[m] \geq 0. \end{aligned} \tag{8.15}$$

Using a lagrange multiplier α, we can re-write the optimization problem as

$$\begin{aligned} &\min_{w_{\mathrm{i,j}}} && \sum_{m=1}^{M} (w[m])^2 \left(\frac{\sigma_{\mathrm{H}}^2[m]}{\Delta_m}\right)^2 + \alpha\left(\sum_{m=1}^{M} w[m] - 1\right) \\ &\text{subject to} && \sum_{m=1}^{M} w[m] = 1, \text{ and } w[m] \geq 0. \end{aligned} \tag{8.16}$$

Solving this optimization problem, we get

$$w[m] = \frac{\left(\frac{\Delta_m}{\sigma_H[m]}\right)^2}{\sum\limits_{k=1}^{M}\left(\frac{\Delta_k}{\sigma_H[k]}\right)^2}$$

Comparing this result with the expression for $\mathrm{SNR_H}$, we can obtain the necessary expression. □

8.5.3 Comparison with CIS

It is important to understand why a CIS-based reconstruction such as [75] does not work for QIS. A CIS assumes a linear sensor response until the photon level reaches the full-well capacity, whereas QIS assumes a nonlinear response. The linear response of a CIS implies that *before* saturation we have $\mu_Y[m] = \beta[m]$ so that $d\beta[m]/d\mu_Y[m] = 1$ in Equation (8.10), and *after* saturation, we have that $\mu_Y[m] = L$ where L is the full-well capacity and so $d\beta[m]/d\mu_Y[m] = \infty$. For K frames, each with an exposure Δ_m, the exposure-referred SNR is

$$\begin{aligned}\sigma_{\mathrm{H}}[m] &= \sqrt{K} \cdot \sigma_Y[m] \cdot \frac{d\beta[m]}{d\mu_Y[m]} \\ &= \begin{cases} \sqrt{K}\sqrt{\Delta_m \lambda}, & \text{if } \Delta_m\lambda < L, \\ \infty, & \text{if } \Delta_m\lambda \geq L. \end{cases}\end{aligned} \tag{8.17}$$

Substituting this into $\mathrm{SNR}^{\mathrm{HDR}}$, we show that for CIS,

$$\mathrm{SNR}^{\mathrm{HDR}}_{\mathrm{H}} = \frac{K\lambda}{\sqrt{\sum_{m=1}^{M} \left(\frac{w[m]}{\Delta_m}\right)^2 K\Delta_m\lambda \cdot \mathbb{I}\left\{\Delta_m\lambda < L\right\}}}, \tag{8.18}$$

where $\mathbb{I}\{\cdot\} = 1$ if the argument is true, and is ∞ if the argument is false. Consequently, one can solve a similar optimization as we did to obtain the following weight

$$w[m] = \frac{\Delta_m \cdot \mathbb{I}\left\{\Delta_m\lambda < L\right\}}{\sum_{m=1}^{M} \Delta_m \cdot \mathbb{I}\left\{\Delta_m\lambda < L\right\}}. \tag{8.19}$$

Therefore, as long as the pixels are not saturated for each exposure, the weight is linear with respect to the exposure time Δ_m. This should be intuitive, because when the pixels are not saturated, longer exposure time gives higher SNR and so it should be weighted more. If a pixel becomes saturated, then the SNR will drop abruptly so that the corresponding exposure is invalidated.

The analysis here shows why a CIS-based reconstruction method does *not* apply to QIS. QIS does not have the linear response as CIS does. As a result, the optimal linear reconstruction method for QIS given by Theorem 8.5.1 is not transferable to CIS, and vice versa.

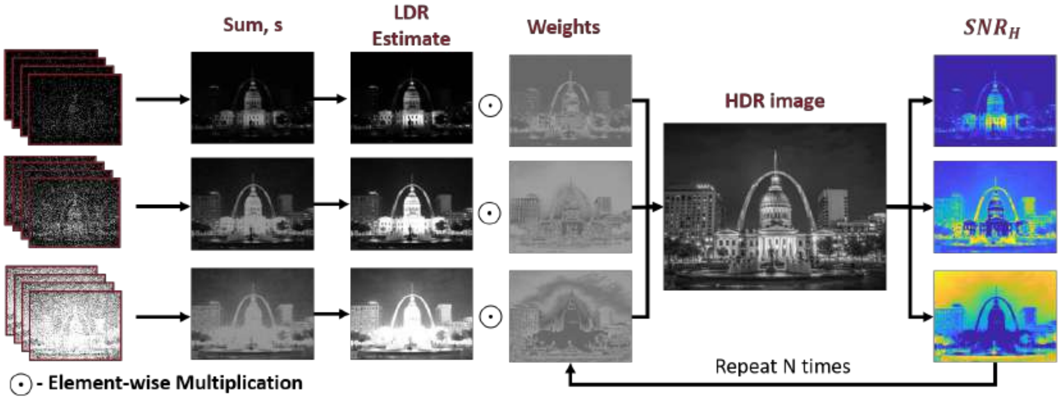


Figure 8.11. HDR reconstruction pipeline. The raw frames from QIS are first summed and denoised. Then the denoised images are linearly combined by giving weights to each image proportional to their SNR_H iteratively.

8.5.4 Reconstruction algorithm

Theorem 8.5.1 suggests a method to construct an HDR image. The idea is that if we knew $\text{SNR}_\text{H}[m]$, then the weight is given according to Equation (8.14). Substituting the weight into Equation (8.7) will give us the estimate.

In practice, however, since we do not know λ, we need to estimate $\text{SNR}_\text{H}[m]$. The estimation is based on an iterative procedure. Denoting $w^k[m]$ as the weight at the k-th iteration, and $\widehat{\lambda}^k$ as the estimated HDR pixel in the k-th iteration, the iterative procedure is given by two steps:

$$\begin{aligned} \widehat{\lambda}^{k+1} &= \sum_{m=1}^{M} w^k[m] S[m], \\ w^{k+1}[m] &= \frac{(\text{SNR}_\text{H}^{k+1}[m])^2}{\sum_{m=1}^{M} (\text{SNR}_\text{H}^{k+1}[m])^2}, \end{aligned}$$

where $\text{SNR}_\text{H}^{k+1}[m]$ is evaluated based on Equation (8.13), and the exposure referred noise $\sigma_\text{H}[m]$, which is a function of $\widehat{\lambda}$, is updated using Theorem 3.3.3. The algorithm is summarized in Algorithm 2.

Algorithm 2 HDR Image Reconstruction

```
1. Acquire QIS frames $Y[n]$, $n = 1, \dots N$.
2. Obtain $M$ LDR images according to Equation (8.6).
3. Initilialize $w^0[m] = 1/M, \forall m = 1, \dots M$.
4. Estimate the HDR image $\widehat{\lambda}^k$ according to Equation (8.7).
5. Update $\mathrm{SNR}_\mathrm{H}^k[m]$ according to Equation (8.13).
6. Update weights $w^k[m]$ according to Equation (8.14).
7. Repeat 4,5,6 till convergence.
```

8.5.5 Practical considerations

Denoising. The proposed HDR reconstruction method does not include any pre-processing of the input LDR images. In practice, it may be desirable to perform some degree of denoising using simple methods such as the one introduced in [76]. The denoising is particularly useful when the number of frames is low. HDR denoising itself is an open problem. We leave the problem on denoising+HDR reconstruction as future work.

Look up tables. The proposed reconstruction method requires calculating the exposure-referred SNR for every pixel at the exposure period, and this is computationally very expensive. However, we notice that the exposure-referred SNR is a function of the mean number of photons collected by the sensor at each frame. It is, therefore, possible to construct a look-up table to store the values by discretizing the mean signal levels. During the computation, one can refer to the look-up table when calculating $\mathrm{SNR}_\mathrm{H}[m]$.

Dynamic Scenes. The optimal reconstruction scheme presented in this chapter is analogous to the optimal linear schemes in the conventional CIS-based HDR problems [75], [279], [280]. Thus, by design, the method is used for static scenes. We acknowledge the importance of HDR imaging for dynamic scenes. However, the reconstruction problem becomes substantially harder in the presence of shot noise and motion. Several methods have demonstrated the feasibility of handling photon-limited data and motion, e.g., [23], [146], [296]. Adding exposure bracketing to these problems is an important future problem.

Number of iterations. The proposed reconstruction algorithm is iterative. In Figure **8.12**, we plot the mean squared error in log scale ($\mathcal{L}$MSE) as used in [284] between the reconstructed image and the ground truth image after each iteration. We use the "aisle" image from the

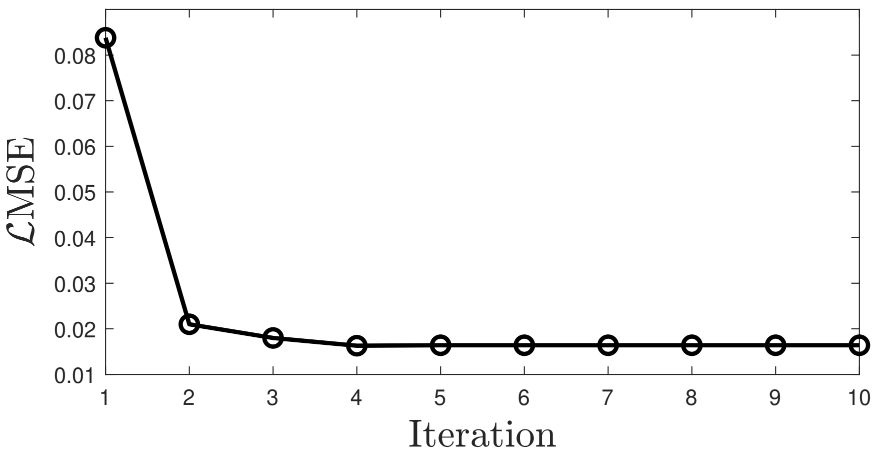


Figure 8.12. Number of iterations for the proposed algorithm to converge. We use three different integration times and 100 frames per integration time and use the proposed HDR reconstruction method. The figure shows that $\mathcal{L}$MSE converges after 5 iterations.

Stanford HDR image dataset [297] for simulating the QIS data for this experiment. We use three different integration times and 100 frames per integration time and use the proposed HDR reconstruction method. We observe that $\mathcal{L}$MSE converges after 5 iterations. We notice similar results with multiple images, different integration times, and a different number of frames.

8.6 Experiments

In this section, we report the experimental results. Our results can be divided into two parts: (i) Comparing CIS with QIS for HDR imaging; (ii) Comparing the optimal HDR reconstruction algorithm and the existing methods.

8.6.1 Comparing CIS with QIS for HDR imaging

The first experiment evaluates the significance of QIS compared to CIS for HDR imaging. Some of the results have already been shown. We summarize them here:

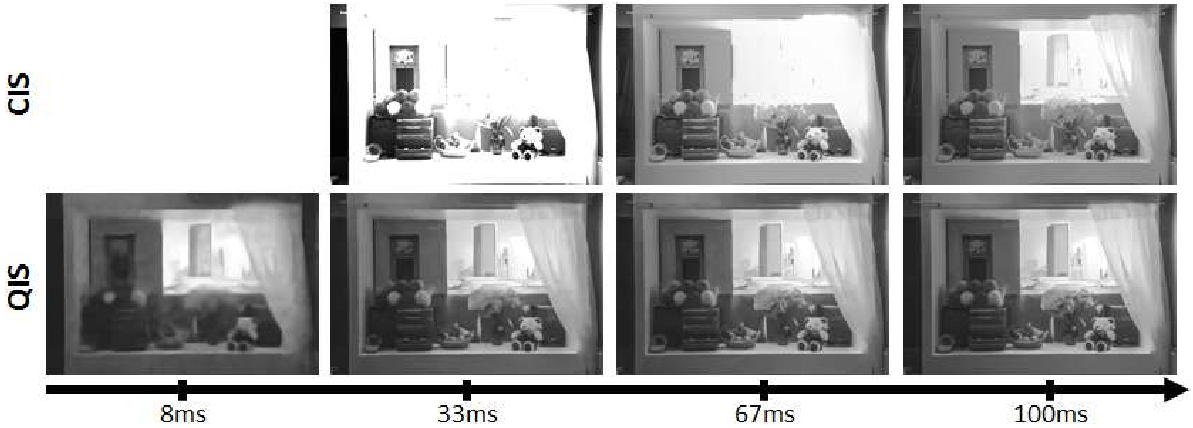
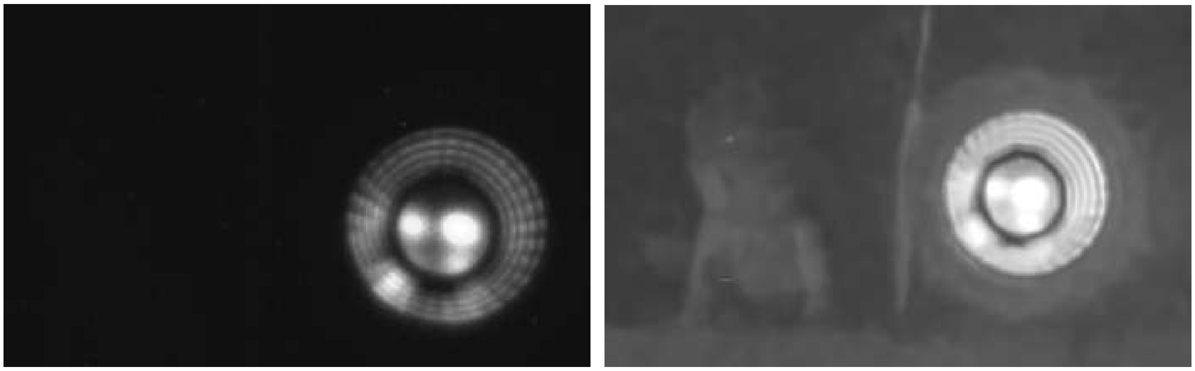


Figure 8.13. Comparing CIS and QIS for HDR imaging. The CIS image is constructed from three frames, each with an exposure of 33 ms, 3.3 ms, and 0.33 ms, respectively. The QIS image is constructed from a set of exposures 1.1 ms, 0.11 ms, and 0.011 ms. The CIS is assumed to have a full well capacity of 4000 electrons. The number of 1-bit QIS frames is 30 times that of CIS so that the overall integration time for CIS and QIS are equal. The timestamps shown at the bottom of the figure are the overall integration time to capture all the exposures. Note that QIS offers better image reconstruction for a short integration, e.g., 33ms or lower.

- Figure 8.8 illustrates the dynamic range that can be offered by one CIS frame (in 1ms) and that offered by multiple QIS frames of different bit-depths (within the same 1ms). Our result shows that CIS saturates, whereas QIS does not.
- Figure 8.10 shows the theoretical dynamic range of CIS and QIS. We observe that a single QIS exposure has a dynamic range of 10dB higher than a CIS, and fusing multiple exposures will widen the gap even further.

In addition to these results, we show in Figure 8.13 a visual comparison between a CIS and a QIS. This experiment considers the practical frame rate limit of a QIS, which was assumed to be 1000 frames per second according to [46], which is approximately 30 times faster than a standard CIS operating at 30 frames per second [298]. While there exists even faster QIS prototypes (e.g., [43]), Figure 8.13 shows that with 1000 fps, QIS already offers an advantage over the CIS.

We simulate 30 QIS frames for every CIS frame to conduct this experiment. The bit-depth of the QIS is 1-bit. Among the QIS exposures, we consider the multi-exposure scheme

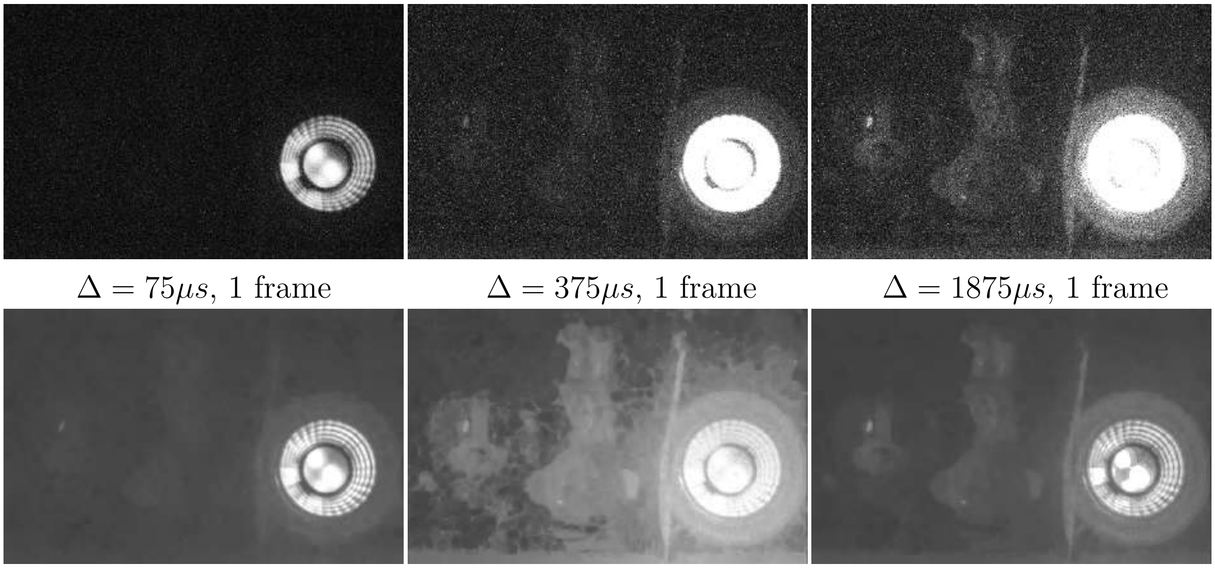


CIS $N = 1$ 1-bit QIS $N = 20$

Figure 8.14. Comparing CIS and QIS. IWe use a commercially available CIS in this real experiment and compare it with a prototype QIS. CIS only captures one frame within a fixed integration time, whereas QIS has captured multiple frames of different exposures.

consisting of integration times 1.1 ms, 0.11 ms, and 0.011 ms. We use integration times 33 ms, 3.3 ms, and 0.33 ms for CIS. The CIS is assumed to have a full well capacity of 4000 electrons. We use the proposed HDR reconstruction method for obtaining the QIS HDR image, and [75] for the CIS HDR image. Notice that CIS initially produces good quality images with limited dynamic range, and the dynamic range improves over time. Compared to this, the QIS can produce images with a larger dynamic range at only a fraction of the time taken by the CIS to produce its first frame. Although the images are noisy initially, the quality gets better over time. At 100 ms, the quality and the dynamic range of the QIS and CIS images are about the same. However, when the total time taken reduces, QIS offers a higher dynamic range than the CIS.

In Figure 8.14, we compare QIS and CIS using real data. We collect a total of $N = 20$ 1-bit QIS frames, with $K = 10$ frames at 2 different integration times of $50\mu s$, $1000\mu s$. We compare this to a CIS image obtained using e-con System's e-CAM40_CUMI4682_MOD camera module, which uses OmniVision's OV4682 image sensor. Figure 8.14 shows a clear distinction between the two sensors.



$\Delta = 75\mu s$, 1 frame $\Delta = 375\mu s$, 1 frame $\Delta = 1875\mu s$, 1 frame

[294] [75] Ours

Figure 8.15. The HDR reconstruction algorithm - Real experiment In this experiment, we collect 10 QIS frames, each at 3 different exposures - $75\mu s$, $375\mu s$, and $1875\mu s$ in 3-bit modes. The result shows the advantage of the proposed HDR reconstruction methods over the other two methods.

8.6.2 Reconstruction algorithm

The second experiment evaluates the optimal reconstruction scheme. While we acknowledge the promising results of deep neural network solutions, in this chapter, we compare with two deterministic schemes [294] and [75] for three reasons:

- The objective of this chapter is not to compete with state-of-the-art HDR image reconstruction algorithms customized for CIS. Moreover, there do not exist QIS datasets for us to conduct a fair comparison.

- Among the deterministic methods, [75] is theoretically optimal for CIS. No other linear method can achieve better results. We compare this method to show that CIS methods cannot be translated to QIS.

- Among the QIS methods, [294] is one of the latest works in the literature. We compare this method to show the effectiveness of our method.

Table 8.1. Comparing the three HDR reconstruction methods.

Metric	Dutton et al. [294]	Granados et al. [75]	Proposed
		1 bit	
$\mathcal{L}$MSE	11.25×10^{-2}	1.23×10^{-2}	$\mathbf{0.61 \times 10^{-2}}$
PU-PSNR	32.53	34.89	**35.92**
PU-SSIM	0.9138	0.9822	**0.9850**
		3 bits	
$\mathcal{L}$MSE	10.02×10^{-2}	0.59×10^{-2}	$\mathbf{0.49 \times 10^{-2}}$
PU-PSNR	33.26	36.42	**36.81**
PU-SSIM	0.9345	0.9901	**0.9912**

We first evaluate the methods using the Stanford-HDR dataset [297] containing 88 HDR images. We normalize the images such that the $0.01 \leq \lambda\Delta \leq 8000,$ if $\lambda \neq 0$ at every pixel. We simulate a total of $N = 3000$ 1-bit and 3-bit frames with with $K = 1000$ frames each at 3 different integration times of Δ, $\Delta/10$, and $\Delta/100$. We use the $\mathcal{L}$MSE, PU-PSNR and PU-SSIM [299] as the metrics for comparison. $\mathcal{L}$MSE measures the mean squared error (MSE) in log-scale. PU-PSNR and PU-SSIM calculate the peak signal-to-noise ratio (PSNR) and structural similarity (SSIM) using a perceptually uniform (PU) encoding. We compare the performance of the proposed HDR reconstruction method with reconstruction methods from [294] and [75] in Table 8.1, using the average $\mathcal{L}$MSE, PU-PSNR, and PU-SSIM compared to the ground-truth for the three methods across the 88 HDR images. We see that the proposed method outperforms the two competing methods in all the three metrics we have considered, in single-bit and three-bit modes.

In Figure **8.16**, we visually compare the three methods. We use 3-bit images. 100 frames are collected at 4 different integration times, thus giving 400 frames. These frames are then used to reconstruct the high dynamic range image. Notice that the proposed method outperforms [294] and [75], both visually and the in the $\mathcal{L}$MSE metric.

Next, we show the comparisons using real QIS data in Figure **8.15**. We collect a total of $N = 30$ frames of 3-bit QIS data, with $K = 10$ frames at 3 different integration times of $75\mu s$, $375\mu s$ and $1875\mu s$. The scene consists of a bright light bulb on the right and two dark objects on the left. The three LDR images show different levels of saturation. We apply [294]

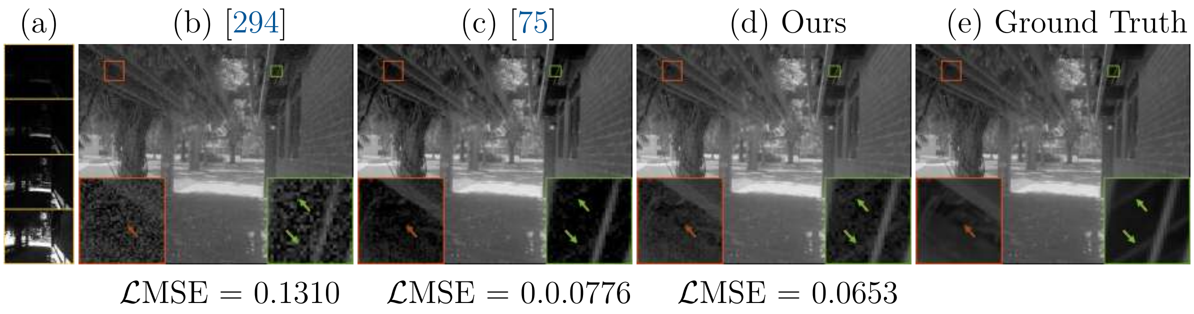

(a)
(b) [294]
(c) [75]
(d) Ours
(e) Ground Truth
$\mathcal{L}$MSE = 0.1310
$\mathcal{L}$MSE = 0.0.0776
$\mathcal{L}$MSE = 0.0653

Figure 8.16. The HDR reconstruction algorithm - Synthetic experiment400 3 bit frames were simulated at 4 different integration times, with 100 frames at each integration time. We can see that the HDR images obtained in (d) using the proposed method are closer to the ground truth than the other two methods. The images are displayed on the log scale. $\mathcal{L}$MSE is the mean squared error measure in log-scale. (Images courtesy : [297])

and [75] to the image stack and reconstruct a HDR image. We observe that the method by Dutton et al. [294] has a weak reconstruction of the darker regions since it provides equal weights to all three integration times. The method by Granados et al. [75] has a better dynamic range, but it also generates artifacts in the brighter regions. The proposed method, which is optimal for QIS, produces an HDR with fewer artifacts.

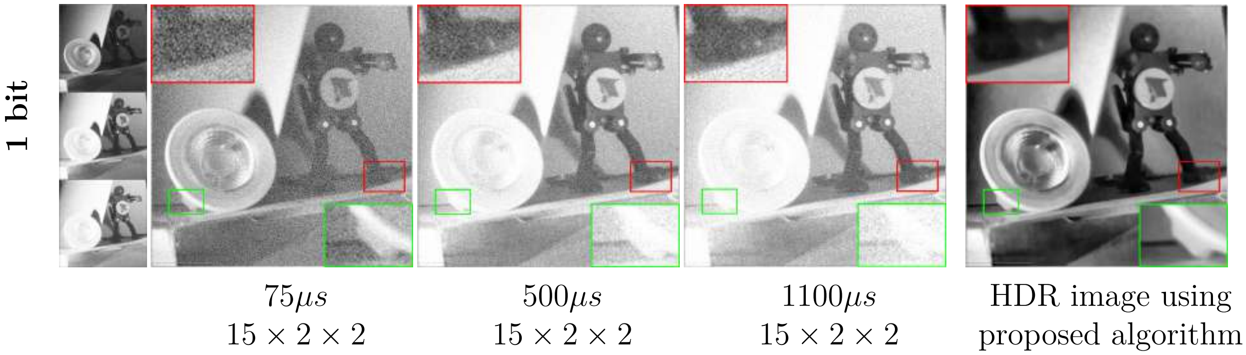


Figure 8.17. Real experiment. In this experiment, we obtain a $K = 15$ frames at 3 different integration times of $\Delta_1 = 75\mu s$, $\Delta_2 = 575\mu s$, and $\Delta_3 = 1175\mu s$. Spatial oversampling of 2×2 is used. The proposed HDR reconstruction algorithm is used to obtain the final HDR image. We use MATLAB's tonemap to re-scale the image intensity for display purposes. The raw (un-normalized) images are shown in the first column's small insets.

Finally, we show the reconstruction results for an image containing more complex content. In Figure 8.17, we collect a total of $N = 45$ frames, with $K = 15$ frames each at 3 different integration times of $75\mu s$, $575\mu s$ and $1175\mu s$. We use 1-bit QIS with a spatial oversampling factor of 2×2. The denoiser from [76] used for denoising the LDR image at each integration time before using the proposed method for HDR reconstruction. As we can observe in the images, the short exposure captures the bright regions well, but the image contains noise, whereas the long exposure has better SNR but is saturated at bright regions. The reconstructed HDR image has recovered the details and maintained the SNR.

8.7 Final thoughts.

We have seen how QIS can revolutionize the HDR imaging scene. The ability of the QIS to oversample combined with lower read noise and dark current is the perfect recipe for high dynamic range imaging. We saw that even with the current prototypes of cameras available, we can still gain the advantage. In the future, the QIS sensors are only going to get better. We have done a theoretical study to understand how much real advantage we will get with these future prototypes, and the results are astounding. We have also contributed a new algorithm for QIS HDR fusion, as the existing methods do not work that well with QIS data.

8.7.1 Where to, from here?

We have made a significant assumption in this chapter that the scene is static. This assumption usually does not hold, and when the scene is dynamic, it is a well-known fact that HDR methods with exposure fusion create ghosting artifacts. Thus it will become imperative to develop methods that deal with motion. This chapter has not explored other computational imaging techniques such as coded exposure for extending the dynamic range. Given that QIS will take some time to get market-ready, it will be interesting to see if the current prototypes can be combined with the CIS image sensors to get a better imaging quality than using only the CIS.

9. SINGLE NEURAL NETWORK FOR MULTIPLE NOISE LEVELS

The following phenomenon could be familiar to those who develop learning-based image denoisers. If a neural network is trained at a noise level σ, then its performance is maximized when the testing noise level is also σ. As soon as the testing noise level deviates from the training noise level, the performance drops [300], [301]. This is a typical mismatch between training and testing, which is arguably universal for all learning-based estimators. This problem becomes even more pronounced when we went a single network to deal with both photon-limited and well illuminated scenes. While this a general phenomenon that shows up for any task, let us deal with denoisers in this chapter.

When such a mismatch problem arises, the most straight-forward solution is to create a suite of networks trained at different noise levels and use the one that matches best with the input noisy image (such as those used in the "Plug-and-Play" priors [165], [181], [186]). However, this ensemble approach is not effective since the model capacity is multiple times larger than necessary.

A more widely adopted solution is to train *one* denoiser and use it for *all* noise levels. The idea is to train the denoiser using a training dataset containing images of different noise levels. The competitiveness of these "one size fits all" denoisers compared to the best individually trained denoisers has been demonstrated in [142], [155], [181], [302]. However, as we will illustrate in this chapter, there is no guarantee for such arbitrarily trained one-size-fits-all denoiser to have a consistent performance over the entire noise range. At some noise levels, usually at the lower tail of the noise range, the performance could be much worse than the best individuals. The cause of this phenomenon is related to how we draw the noisy samples, which is usually *uniform* across the noise range. The question we ask here is that if we allocate more low-noise samples and fewer high-noise samples, will we be able to get a more consistent result?

9.1 One size fits all denoisers

The objective of this chapter is to find a sampling distribution such that for every noise level the performance is consistent. Here, by consistent we meant that the gap between the estimator and the best individuals is balanced. The idea is illustrated in Figure 9.1. The black curve in the figure represents the ensemble of the best individually trained denoisers. It is a virtual curve obtained by training the denoiser at each noise level. A typical "one size fits all" denoiser is trained by using noisy samples from a uniform distribution, which is denoted by the blue curve. This figure illustrates a typical in-consistence where there is a significant gap at low-noise but small gap at high noise. The objective of this chapter is to find a new sampling distribution (denoted by the orange bars) such that we can achieve a consistent performance throughout the entire range. The result returned by our method is a trade-off between the overall performance and the worst cases scenarios.

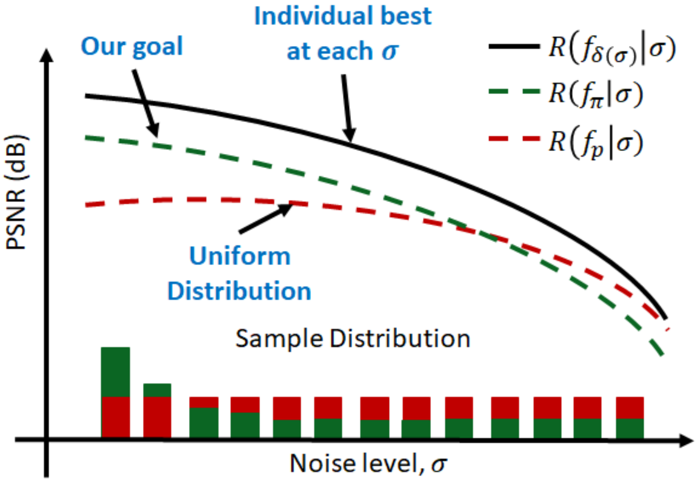


Figure 9.1. Illustration of the objective. The typical uniform sampling (blue bars) will yield a performance curve that is skewed towards one side of the noise range. The objective of this chapter is to find an optimal sampling distribution (orange bars) such that the performance is consistent across the noise range. Notations will be defined in Section 9.3. We plot the risks in terms of the peak signal-to-noise ratio.

The key idea behind the proposed method is a minimax formulation. This minimax optimization minimizes the overall risk of the estimator subject to the constraint that the worst case performance is bounded. We show that under the standard convexity assumptions on the set of all admissible estimators, we can derive a provably convergent algorithm by analyzing the dual. For estimators whose admissible set is not convex, solutions returned by our dual algorithm are the convex-relaxation results. We present the algorithm, and we show that steps of the algorithm can be implemented by iteratively updating the sample distributions.

9.2 Existing solutions

While the above sampling distribution problem may sound familiar, its solution does not seem to be available in the computer vision and machine learning literature.

Image denoising. Recent work in image denoising has been focusing on developing better neural network architectures. When encountering multiple noise levels, [181] presented two approaches: Create a suite of denoisers at different noise levels, or train a denoiser by uniformly sampling noise levels from the range. For the former approach, [300] proposed to combine the estimators by solving a convex optimization problem. [90] proposed an alternative approach by introducing a noise map as an extra channel to the network. Our work shares the same overall goal as [301]. However, they address problem by modifying the network structure whereas we do not change the network. Another related work is [303] which proposed an ad-hoc solution to the sample distribution. Our work offers theoretical justification, convergence guarantee, and optimality. We should also mention [304] which scales the image intensities in order to match with the denoiser trained at a single noise level.

Active learning / Experimental design. Adjusting the distribution of the training samples during the learning procedure is broadly referred to active learning in machine learning [305] or experimental design in statistics [306]. Active learning / experimental design are typically associated with limited training data [307], [308]. The goal is to optimally select the next data point (or batch of data points) so that we can estimate the model parameters, e.g., the mean and variance. The problem we encounter here is not about limited data because we can synthesize as much data as we want since we know the image formation process. The challenge is how to allocate the synthesized data.

Constrained optimization in neural network. Training neural networks under constraints have been considered in classic optimization literature [309] [310]. More recently, there are optimization methods for solving inequality constrained problems in neural networks [311], and equality constrained problems [312]. However, these methods are generic approaches. The convexity of our problem allows us to develop a unique and simple algorithm.

Fairness aware classification. The task of seeking "balanced samples" can be considered as improving the fairness of the estimator. Literature on fairness aware classification is rapidly growing. These methods include modifying the network structure, the data distribution, and loss functions [313]–[317]. Putting the fairness as a constrained optimization has been proposed by [318], but their problem objective and solution are different from ours.

9.3 Problem formulation

9.3.1 Training and testing distributions: $\pi(\sigma)$ and $p(\sigma)$

Consider a clean signal $\boldsymbol{y} \in \mathbb{R}^n$. We assume that this clean signal is corrupted by some random process to produce a corrupted signal $\boldsymbol{x}_\sigma \in \mathbb{R}^n$. The parameter σ can be treated in a broad sense as the level of uncertainty. The support of σ is denoted by the set Ω. We assume that σ is a random variable with a probability density function $p(\sigma)$.

Example 9.1. *In a denoising problem, the image formation model is given by $\boldsymbol{x}_\sigma = \boldsymbol{y} + \sigma\boldsymbol{\eta}$ where $\boldsymbol{\eta}$ is a zero-mean unit-variance i.i.d. Gaussian noise vector. The noise level is measured by σ. For image deblurring, the model becomes $\boldsymbol{x}_\sigma = \boldsymbol{h}_\sigma * \boldsymbol{y} + \boldsymbol{\epsilon}$ where $\boldsymbol{h}_\sigma$ denotes the blur kernel with radius σ, "$*$" denotes convolution, and $\boldsymbol{\epsilon}$ is the noise. In this case, the uncertainty is associated with the blur radius.* □

We focus on *learning-based* estimators. We define an estimator $f : \mathbb{R}^n \to \mathbb{R}^n$ as a mapping that takes a noisy input $\boldsymbol{x}_\sigma$ and maps it to a denoised output $f(\boldsymbol{x}_\sigma)$. We assume that f

is parametrized by $\theta \in \Theta$, but for notation simplicity we omit the parameter θ when the context is clear. The set of all admissible f's is denoted as $\mathcal{F} = \{f(\cdot, \theta) \mid \theta \in \Theta\}$.

To train the estimator f, we draw training samples from the set $\mathcal{S} \overset{\text{def}}{=} \{\boldsymbol{x}_\sigma^{(\ell)} \mid \ell = 1, \ldots, N,\ \sigma \overset{\text{i.i.d.}}{\sim} \boldsymbol{\pi}(\sigma)\}$, where ℓ refers to the ℓ-th training sample, and $\boldsymbol{\pi}(\sigma)$ is the distribution of the noise levels in the training samples. Note that $\boldsymbol{\pi}$ is not necessarily the same as p. The distribution $\boldsymbol{\pi}$ is the distribution of the *training* samples, and the distribution p is the distribution of the *testing* samples. In most learning scenarios, we want $\boldsymbol{\pi}$ to match with p so that the generalization error is minimized. However, in this work, we are purposely designing a $\boldsymbol{\pi}$ which is different from p because the goal is to seek an optimal trade-off. To emphasize the dependency of f on $\boldsymbol{\pi}$, we denote f as $f_{\boldsymbol{\pi}}$.

9.3.2 Risk and conditional risk: $R(f)$ and $R(f \mid \sigma)$

Training an estimator $f_{\boldsymbol{\pi}}$ requires a loss function. We denote the loss between a predicted signal $f_{\boldsymbol{\pi}}(\boldsymbol{x}_\sigma)$ and the truth $\boldsymbol{y}$ as $\mathcal{L}(f_{\boldsymbol{\pi}}(\boldsymbol{x}_\sigma), \boldsymbol{y})$. An example of the loss function is the Euclidean distance:

$$\mathcal{L}(f_{\boldsymbol{\pi}}(\boldsymbol{x}), \boldsymbol{y}) = mid f_{\boldsymbol{\pi}}(\boldsymbol{x}_\sigma) - \boldsymbol{y} mid^2. \tag{9.1}$$

Other types of loss functions can also be used as long as they are convex in $f_{\boldsymbol{\pi}}$. To quantify the performance of the estimator $f_{\boldsymbol{\pi}}$, we define the notion of *conditional risk/*

Definition 9.3.1 (Conditional risk)**.** *We define conditional risk as*

$$R(f_{\boldsymbol{\pi}} \mid \sigma) \overset{\text{def}}{=} \mathrm{E}_{(\boldsymbol{x}_\sigma, \boldsymbol{y})|\sigma}\left[\mathcal{L}(f_{\boldsymbol{\pi}}(\boldsymbol{x}_\sigma), \boldsymbol{y}) \mid \sigma\right]. \tag{9.2}$$

The conditional risk can be interpreted as the risk of the estimator $f_{\boldsymbol{\pi}}$ evaluated at a particular noise level σ. The overall risk is defined through iterated expectation.

Definition 9.3.2 (Overall risk)**.** *We defined overall risk as*

$$
\begin{aligned}
R(f_\pi) &\overset{\text{def}}{=} \mathrm{E}_{\sigma\sim p(\sigma)}\left\{R(f_\pi \mid \sigma)\right\} \\
&= \int \underbrace{\mathrm{E}_{(\boldsymbol{x}_\sigma,\boldsymbol{y})|\sigma}\left[\mathcal{L}(f_\pi(\boldsymbol{x}_\sigma),\boldsymbol{y}) \mid \sigma\right]}_{=R(f_\pi \,|\, \sigma)} \; p(\sigma)d\sigma. \qquad (9.3)
\end{aligned}
$$

Note that the expectation of σ is taken with respect to the true distribution p since we are evaluating the estimator f_π.

9.3.3 Three estimators: f_π, f_p and $f_{\delta(\sigma)}$

The estimator f_π is determined by minimizing the training loss. In our problem, since the training set follows a distribution $\pi(\sigma)$, f_π is obtained by minimizing over this distribution.

Definition 9.3.3. *The function f_π is defined as*

$$
f_\pi \overset{\text{def}}{=} \underset{f}{argmin} \int R(f \mid \sigma)\pi(\sigma)\, d\sigma. \qquad (9.4)
$$

This definition can be understood by noting that $R(f \mid \sigma)$ is the conditional risk evaluated at σ. Since $\pi(\sigma)$ specifies the probability of obtaining a noisy samples with noise level σ, the integration in Equation (9.4) defines the training loss when the noisy samples are proportional to $\pi(\sigma)$. Therefore, by minimizing this training loss, we will obtain f_π.

Example 9.2. *Suppose that we are training a denoiser over the range of $\sigma \in [a, b]$. If the training set is uniform, i.e., $\pi(\sigma) = 1/(b-a)$ for $\sigma \in [a, b]$ and is 0 otherwise, then f_π is obtained by minimizing the sum of the individual losses $f_\pi = argmin_f \sum_{\ell=1}^{N} \mathcal{L}(f(\boldsymbol{x}_\sigma^{(\ell)}), \boldsymbol{y}^{(\ell)})$ where the N training samples have equally likely noise levels.* □

If we replace the training distribution $\boldsymbol{\pi}$ by the testing distribution p, then we obtain the following estimator:

$$f_p = \underset{f}{\operatorname{argmin}} \int R(f \mid \sigma) p(\sigma)\, d\sigma = \underset{f}{\operatorname{argmin}} \;\; R(f). \tag{9.5}$$

Since f_p minimizes the overall risk, we expect $R(f_p) \leq R(f_{\boldsymbol{\pi}})$ for all $\boldsymbol{\pi}$. This is summarized in the lemma below.

Lemma 9.1. *The risk of f_p is a lower bound of the risk of all other $f_{\boldsymbol{\pi}}$:*

$$R(f_p) \leq R(f_{\boldsymbol{\pi}}), \qquad \forall \boldsymbol{\pi}. \tag{9.6}$$

Proof. By construction, f_p is the minimizer of the risk according to Equation (9.5), it holds that $R(f_p) = \inf_f R(f)$. Therefore, for any $\boldsymbol{\pi}$ we have $R(f_p) \leq R(f_{\boldsymbol{\pi}})$. □

The consequence of Lemma 9.1 is that if we minimize $R(f)$ without any constraint, we will reach a trivial solution of $\boldsymbol{\pi} = p$. This explains why this work is uninteresting if the goal is to purely minimize the generalization error without considering any constraint.

Before we proceed, let us define one more distribution δ which has a point mass at a particular σ, i.e., $p(\sigma)$ is a delta function such that $p(\sigma') = \delta(\sigma' - \sigma)$.

Definition 9.3.4. *The estimator $f_{\delta(\alpha)}$ is defined as*

$$f_{\delta(\sigma)} = \underset{f}{\mathit{argmin}} \int R(f \mid \sigma')\delta(\sigma' - \sigma)\, d\sigma', \tag{9.7}$$

which is equivalent to minimizing the conditional risk $f_{\delta(\sigma)} = \underset{f}{\mathit{argmin}} \; R(f \mid \sigma)$.

Because we are minimizing the conditional risk at a particular σ, $f_{\delta(\sigma)}$ gives the best individual estimate at σ. However, having the best estimate at σ does not mean that $f_{\delta(\sigma)}$ can generalize. It is possible that $f_{\delta(\sigma)}$ performs well for one σ but poorly for other σ's. However, the ensemble of all these point-wise estimates $\{f_{\delta(\sigma)}\}$ will form the lower bound of the conditional risks such that $R(f_{\delta(\sigma)} \mid \sigma) \leq R(f_p \mid \sigma)$ at every σ.

9.3.4 Main problem (P1)

We now state the main problem. The problem we want to solve is the following constrained optimization.

Definition 9.3.5 (Main Problem)**.**

$$\begin{aligned} f^* \quad \overset{\text{def}}{=} \quad & \underset{f \in \mathcal{F}}{\mathit{argmin}} \qquad R(f) \\ & \text{subject to} \qquad \sup_{\sigma \in \Omega} \left\{ R(f \mid \sigma) - R(f_{\delta(\sigma)} \mid \sigma) \right\} \leq \epsilon. \end{aligned} \tag{P1}$$

The objective function reflects our original goal of minimizing the overall risk. However, instead of doing it without any constraint (which has a trivial solution of $f^* = f_p$), we introduce a constraint that the gap between the current estimator f and the best individual $f_{\delta(\sigma)}$ is no worse than ϵ, where ϵ is some threshold. The intuition here is that we are willing to sacrifice some of the overall risk by limiting the gap between f and $f_{\delta(\sigma)}$ so that we have a consistent performance over the entire range of noise levels.

Referring back to Figure **9.1**, we note that the black curve is $R(f_{\delta(\sigma)} \mid \sigma)$. The blue curve is $R(f_p \mid \sigma)$ for the case where $p(\sigma)$ is a uniform distribution. The orange curve is $R(f^* \mid \sigma)$. We show in Section 9.4.2 that f^* is equivalent to $f_{\boldsymbol{\pi}}$ for some $\boldsymbol{\pi}(\sigma)$. Note that all curves are the conditional risks.

9.4 Dual ascent

In this section we discuss how to solve Equation (P1). Solving Equation (P1) is challenging because minimizing over f involves updating the estimator f which could be nonlinear w.r.t. the loss. To address this issue, we first show that as long as the admissible set $\mathcal{F}$ is convex, Equation (P1) is convex even if the estimators f themselves are non-convex. We then derive an algorithm to solve the dual problem.

9.4.1 Convexity of Equation (P1)

We start by showing that under mild conditions, Equation (P1) is convex.

Lemma 9.2. *Let $\mathcal{F}$ be a closed and convex set. Then, for any convex loss function $\mathcal{L}$, the risk $R(f)$ and the conditional risk $R(f \mid \sigma)$ are convex in f, for any $\sigma \in \Omega$.*

Proof. Let f_1 and f_2 be two estimators in $\mathcal{F}$ and let $\lambda \in [0,1]$ be a constant. Then, by the convexity of $\mathcal{L}$, the conditional risk $R(\cdot \mid \sigma)$ satisfies

$$\begin{aligned} R(\lambda f_1 + (1-\lambda) f_2 \mid \sigma) &= \mathrm{E}\left\{\mathcal{L}(\lambda f_1 + (1-\lambda) f_2) \mid \sigma\right\} \\ &\le \mathrm{E}\left\{\lambda \mathcal{L}(f_1) + (1-\lambda)\mathcal{L}(f_2) \mid \sigma\right\} \\ &= \lambda R(f_1 \mid \sigma) + (1-\lambda) R(f_2 \mid \sigma), \end{aligned}$$

which is convex. The overall risk $R(f)$ is found by taking the expectation of the conditional risk over σ. Since taking expectation is equivalent to integrating the conditional risk times the distribution $p(\sigma)$ (which is positive), convexity preserves and so $R(f)$ is also convex. □

We emphasize that the convexity of $R(\cdot)$ is defined w.r.t. f and not the underlying parameters (e.g., the network weights). For any convex combination of the parameters θ's, we have that $R(f(\cdot \mid \lambda\theta_1 + (1-\lambda)\theta_2)) \not\le \lambda R(f(\cdot \mid \theta_1)) + (1-\lambda) R(f(\cdot \mid \theta_2))$ because f is not necessarily convex.

The following corollary shows that the optimization problem Equation (P1) is convex.

Corollary 9.1. *Let $\mathcal{F}$ be a closed and convex set. Then, for any convex loss function $\mathcal{L}$, Equation (P1) is convex in f.*

Proof. Since the objective function R is convex (by Lemma 9.2), we only need to show that the constraint set is also convex. Note that the "sup" operation is equivalent to requiring $R(f \mid \sigma) - R(f_{\delta(\sigma)|\sigma}) \le \epsilon$ for *all* $\sigma \in \Omega$. Since $R(f_{\delta(\sigma)} \mid \sigma)$ is constant w.r.t. f, we can define

$\epsilon(\sigma) \overset{\text{def}}{=} \epsilon + R(f_{\delta(\sigma)} \mid \sigma)$ so that the constraint becomes $R(f \mid \sigma) - \epsilon(\sigma) \leq 0$. Consequently the constraint set is convex because the conditional risk $R(f \mid \sigma)$ is convex. □

The convexity of $\mathcal{F}$ is subtle but essential for Lemma 9.2 and Corollary 9.1. In a standard optimization over $\mathbb{R}^n$, the convexity is granted if the admissible set is an interval in $\mathbb{R}^n$. In our problem, $\mathcal{F}$ denotes the set of all admissible estimators, which by construction are parametrized by θ. Thus, the convexity of $\mathcal{F}$ requires that a convex combination of two admissible f's remains admissible. All estimators based on generalized linear models satisfy this property. However, for deep neural networks it is generally unclear how the topology looks like although some recent studies are suggesting negative results [319]. Nevertheless, even if $\mathcal{F}$ is non-convex, we can solve the dual problem which is always convex. The dual solution provides the convex-relaxation of the primal problem. The duality gap is zero when the Slater's condition holds, i.e., when $\mathcal{F}$ is convex and ϵ is chosen such that the constraint set is strictly feasible.

9.4.2 Dual of Equation (P1)

Let us develop the dual formulation of Equation (P1). The dual problem is defined through the Lagrangian:

$$\begin{aligned} L(f,\lambda) &\overset{\text{def}}{=} R(f) + \int \left\{ R(f \mid \sigma) - \underbrace{\Big(R(f_{\delta(\sigma)} \mid \sigma) + \epsilon\Big)}_{\overset{\text{def}}{=}\epsilon(\sigma)} \right\} \lambda(\sigma) d\sigma \\ &= \int R(f \mid \sigma) \Big\{ p(\sigma) + \lambda(\sigma) \Big\} d\sigma - \int \epsilon(\sigma)\lambda(\sigma) d\sigma, \end{aligned} \tag{9.8}$$

by which we can determine the Lagrange dual function as

$$g(\lambda) = \inf_f L(f,\lambda), \tag{9.9}$$

and the dual solution:

$$\begin{aligned}\lambda^* &= \underset{\lambda \geq 0}{\text{argmax}} \;\; g(\lambda) \\ &= \underset{\lambda \geq 0}{\text{argmax}} \;\; \left\{ \inf_f \left\{ \int R(f \mid \sigma)\Big[p(\sigma) + \lambda(\sigma)\Big] d\sigma \right\} \right. \\ &\qquad\qquad\qquad \left. - \int \epsilon(\sigma)\lambda(\sigma) d\sigma \right\}. \end{aligned} \tag{9.10}$$

Given the dual solution λ^*, we can translate it back to the primal solution $\widehat{f}$ by minimizing the inner problem in Equation (9.10), which is

$$\widehat{f} = \underset{f}{\text{argmin}} \;\; \int R(f \mid \sigma) \underbrace{\left\{ p(\sigma) + \lambda^*(\sigma) \right\}}_{\overset{\text{def}}{=} \boldsymbol{\pi}^*(\sigma)} d\sigma. \tag{9.11}$$

This minimization is nothing but training the estimator f using samples who noise levels are distributed according to $p(\sigma) + \lambda^*(\sigma)$. [1] Therefore, by solving the dual problem we have simultaneously obtained the distribution $\boldsymbol{\pi}^*(\sigma)$, which is $\boldsymbol{\pi}^*(\sigma) = p(\sigma) + \lambda^*(\sigma)$, and the estimator $\widehat{f}$ trained using the distribution $\boldsymbol{\pi}^*$.

As we have discussed, if the admissible set $\mathcal{F}$ is convex then Equation (P1) is convex and so $\widehat{f}$ is exactly the primal solution f^*. If $\mathcal{F}$ is not convex, then $\widehat{f}$ is the solution of the convex relaxation of Equation (P1). The duality gap is $R(f^*) - g(\lambda^*)$.

9.4.3 Dual ascent algorithm

The algorithm for solving the dual is based on the fact that the point-wise $\inf_f \; L(f, \lambda)$ is concave in λ. As such, one can use the standard dual ascent method to find the solution. The idea is to sequentially update λ's and f's via

[1] ↑For $p(\sigma) + \lambda(\sigma)$ to be a legitimate distribution, we need to normalize it by the constant $Z = \int\{p(\sigma) + \lambda(\sigma)\}d\sigma$. But as far as the minimization in Equation (9.11) is concerned, the constant is unimportant.

$$f^{t+1} = \underset{f}{\operatorname{argmin}} \int R(f \mid \sigma)\Big\{p(\sigma) + \lambda^t(\sigma)\Big\}d\sigma \tag{9.12}$$

$$\lambda^{t+1}(\sigma) = \Big[\lambda^t(\sigma) + \alpha^t(\sigma)\Big\{R(f^{t+1} \mid \sigma) - \epsilon(\sigma)\Big\}\Big]_+ \tag{9.13}$$

Here, α^t is the step size of the gradient ascent step, and $[\ \cdot\ ]_+ = \max(\cdot, 0)$ returns the positive part of the argument. At each iteration, Equation (9.12) is solved by training an estimator using noise samples drawn from the distribution $\boldsymbol{\pi}(\sigma)^t = p(\sigma) + \lambda^t(\sigma)$. The λ-step in Equation (9.13) computes the conditional risk $R(f^{t+1} \mid \sigma)$ and updates λ.

Since the dual is convex, the dual ascent algorithm is guaranteed to converge to the dual solution using an appropriate step size. We refer readers to standard texts, e.g., [320].

9.5 Uniform gap

The solution of Equation (P1) depends on the tolerance ϵ. This tolerance ϵ cannot be arbitrarily small, or otherwise the constraint set will become empty. The smallest ϵ which still ensures a non-empty constraint set is defined as ϵ_{min}. The goal of this section is to determine ϵ_{min} and discuss its implications.

9.5.1 The uniform gap problem (P2)

The motivation of studying the so-called Uniform Gap problem is the inadequacy of Equation (P1) when the tolerance ϵ is larger than ϵ_{min} (i.e., we tolerate more than needed). The situation can be understood from Figure 9.2. For any allowable ϵ, the solution returned by Equation (P1) can only ensure that the largest gap is no more than ϵ. It is possible that the high-ends have a significantly smaller gap than the low-ends. The gap will become uniform only when $\epsilon = \epsilon_{\text{min}}$ which is typically not known a-priori.

If we want to maintain a constant gap throughout the entire range of σ, then the optimization goal will become minimizing the maximum risk gap and not worry about the overall risk. In other words, we solve the following problem:

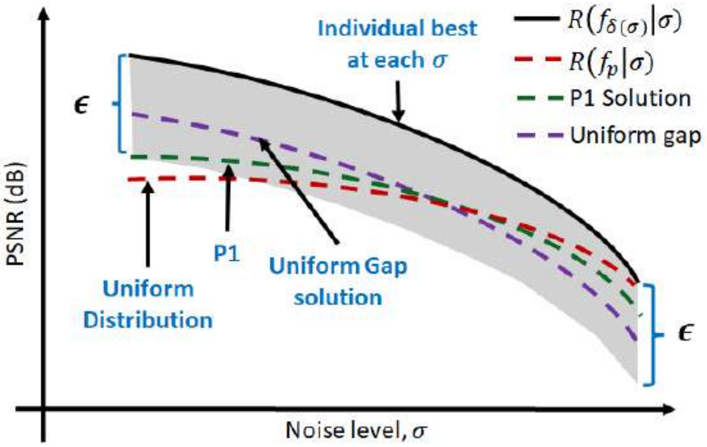


Figure 9.2. Difference between Equation (P1) and Equation (P2). In Equation (P1), the solution only needs to make sure that the worst case gap is upper bounded by ϵ. There is no control over places where the gap is intrinsically less than ϵ. Uniform Gap problem Equation (P2) addresses this issue by forcing the gap to be uniform. Note that neither Equation (P1) nor Equation (P2) is absolutely more superior. It is a trade-off between the noise levels, and how much we know about the testing distribution p.

Definition 9.5.1 (Uniform gapproblem)**.**

$$f^* = \mathop{argmin}_{f} \sup_{\sigma\in\Omega} \left\{ R(f \mid \sigma) - R(f_{\delta(\sigma)} \mid \sigma) \right\}. \tag{P2}$$

When Equation (P2) is solved, the corresponding risk gap is exactly $\epsilon_{\min}$, defined as

$$\epsilon_{\min} \overset{\text{def}}{=} \sup_{\sigma\in\Omega} \left\{ R(f^* \mid \sigma) - R(f_{\delta(\sigma)} \mid \sigma) \right\}. \tag{9.14}$$

The supremum in the above equation can be lifted because by construction, Equation (P2) guarantees a constant gap for all σ.

The difference between Equation (P2) and Equation (P1) is the switched roles of the objective function and the constraint. In Equation (P1), the tolerance ϵ defines a user-controlled upper bound on the risk gap, whereas in Equation (P2) the ϵ is eliminated. Note that the omission of ϵ in Equation (P2) does not imply better or worse since Equation (P1) and

Equation (P2) are serving two different goals. Equation (P1) utilizes the underlying testing distribution $p(\sigma)$ whereas Equation (P2) does not. It is possible that $p(\sigma)$ is skewed towards high noise scenarios so that a constant risk gap will suffer from insufficient performance at high-noise and over-perform at low-noise which does not matter because of $p(\sigma)$.

In practice (i.e., in the absence of any knowledge about an appropriate ϵ), one can solve Equation (P2) first to obtain the tightest gap ϵ_{min}. Once ϵ_{min} is determined, we can choose an $\epsilon > \epsilon_{\text{min}}$ to minimize the overall risk using Equation (P1).

9.5.2 Algorithm for solving Equation (P2)

The algorithm to solve Equation (P2) is slightly different from that of Equation (P1) because of the omission of the constraint.

We first rewrite problem Equation (P2) as

$$\begin{aligned} &\underset{f,t}{\text{minimize}} && t \\ &\text{subject to} && R(f \mid \sigma) - \underbrace{R(f_{\delta(\sigma)} \mid \sigma)}_{\overset{\text{def}}{=} r(\sigma)} \leq t, \;\; \forall \sigma. \end{aligned} \tag{9.15}$$

Then the Lagrangian is defined as

$$\begin{aligned} L(f,t,\lambda) &\overset{\text{def}}{=} t + \int \Big\{ R(f \mid \sigma) - r(\sigma) - t \Big\} \lambda(\sigma)\, d\sigma \\ &= t\Big(1 - \int \lambda(\sigma) d\sigma\Big) + \int \Big\{ R(f \mid \sigma) - r(\sigma) \Big\} \lambda(\sigma) d\sigma. \end{aligned} \tag{9.16}$$

Minimizing over f and t yields the dual function:

$$\begin{aligned} g(\lambda) &\overset{\text{def}}{=} \inf_{f,t} L(f,t,\lambda) \\ &= \begin{cases} \inf_f \int \Big[R(f \mid \sigma) - r(\sigma) \Big] \lambda(\sigma) d\sigma, & \text{if } \int \lambda(\sigma) d\sigma = 1, \\ -\infty, & \text{otherwise.} \end{cases} \end{aligned} \tag{9.17}$$

Consequently, the dual problem is defined as

$$\begin{aligned}\lambda^* = \underset{\lambda \geq 0}{\operatorname{argmax}} \; \inf_{f} \left\{ \int \Big[R(f \mid \sigma) - r(\sigma) \Big] \lambda(\sigma) d\sigma \right\} \\ \text{subject to} \int \lambda(\sigma) d\sigma = 1. \end{aligned} \tag{9.18}$$

Again, if $\mathcal{F}$ is convex then solving the dual problem Equation (9.18) is necessary and sufficient to determine the primal problem Equation (9.15) which is equivalent to Equation (P2). The dual problem is solvable using the dual ascent algorithm, where we update λ and f according to the following sequence:

$$f^{t+1} = \underset{f}{\operatorname{argmin}} \; \left\{ \int \Big[R(f \mid \sigma) - r(\sigma) \Big] \lambda^t(\sigma) d\sigma \right\} \tag{9.19}$$

$$\lambda^{t+\frac{1}{2}} = \left[\lambda^t + \alpha^t \Big(R(f^{t+1} \mid \sigma) - r(\sigma) \Big) \right]_+ \tag{9.20}$$

$$\lambda^{t+1} = \lambda^{t+\frac{1}{2}} \Big/ \int \lambda^{t+\frac{1}{2}}(\sigma) d\sigma. \tag{9.21}$$

Here, Equation (9.19) solves the inner optimization in Equation (9.18) by fixing a λ, and Equation (9.20) is a gradient ascent step for the dual variable. The normalization in Equation (9.21) ensures that the constraint of Equation (9.18) is satisfied. The non-negativity operation $[\,\cdot\,]_+$ in Equation (9.20) can be lifted because by definition $r(\sigma) \overset{\text{def}}{=} R(f_{\delta(\sigma)} \mid \sigma) \geq R(f \mid \sigma)$ for all σ. The final sampling distribution is $\boldsymbol{\pi}^*(\sigma) = \lambda^*(\sigma)$.

Like Equation (P1), the dual ascent algorithm for Equation (P2) has guaranteed convergence as long as the loss function $\mathcal{L}$ is convex.

9.6 Practical considerations

The actual implementation of the dual ascent algorithms for Equation (P1) and Equation (P2) require additional modifications. We list a few of them here.

9.6.1 Finite epochs

In principle, the f-subproblems in Equation (9.12) and Equation (9.19) are determined by training a network completely using the sample distributions at the t-th iteration $\boldsymbol{\pi}^t(\sigma) = p(\sigma) + \lambda^t(\sigma)$ and $\boldsymbol{\pi}^t(\sigma) = \lambda^t(\sigma)$, respectively. However, in practice, we can reduce the training time by training the network inexactly. Depending on the specific network architecture and problem type, the number of epochs varies between 10 - 50 epochs per dual ascent iteration.

9.6.2 Discrete noise levels

The theoretical results presented in this chapter are based on continuous distributions $\lambda(\sigma)$ and $p(\sigma)$. In practice, a continuum is not necessary since nearby noise levels are usually indistinguishable visually. As such, we discretize the noise levels in a finite number of bins. And we use the average-value of each bin as the representative noise level for the bin, so that the integration can be simplified to summation.

9.6.3 Interpolate best individuals

The theory above require knowledge of the best individuals $R(f_{\delta(\sigma)} \mid \sigma)$ at all σ's which is computationally infeasible. We approximate this by first obtaining a set of values $R(f_{\delta(\sigma)} \mid \sigma)$ at several specific σ's. This involves training the network separately for a few noise levels. Afterwards, a simple linear interpolation can be used to predict $R(f_{\delta(\sigma)} \mid \sigma)$ at σ's that are not trained. Since the function $R(f_{\delta(\sigma)} \mid \sigma)$ is typically smooth, linear interpolation is reasonably accurate.

9.6.4 log-Scale constraints

Most image restoration applications measure the restoration quality in the log-scale, e.g., the peak signal-to-noise ratio (PSNR) which is defined as PSNR $= -10 \log_{10}$ MSE where MSE is the mean squared error. Learning in the log-scale can be achieved by enforcing constraint in the log-space.

We define the the log-scale risk function as:

$$R_{\log}(f \mid \sigma) \overset{\text{def}}{=} \mathrm{E}\left[\log \mathcal{L}(f(\boldsymbol{x}_\sigma), \boldsymbol{y}) \ \mid \ \sigma\right]. \tag{9.22}$$

With this definition, it follows the the constraints in the log-scale are represented as $\sup_{\sigma\in\Omega}\{R_{\log}(f \mid \sigma) - R_{\log}(f_{\delta(\sigma)} \mid \sigma)\} \leq \epsilon$. To turn this log-scale constraint into a linear form, we use the follow lemma by exploiting the fact that the risk gap is typically small.

Lemma 9.3. *The log-scale constraint*

$$\sup_{\sigma\in\Omega}\left\{R_{\log}(f \mid \sigma) - R_{\log}(f_{\delta(\sigma)} \mid \sigma)\right\} \leq \epsilon \tag{9.23}$$

can be approximated by

$$\sup_{\sigma\in\Omega}\left\{\frac{\mathrm{E}\left[\mathcal{L}(f(\boldsymbol{x}_\sigma), \boldsymbol{y})\right]}{L_\delta(\sigma)}\right\} \leq 1 + \epsilon, \tag{9.24}$$

where $L_\delta(\sigma)$ is a constant (w.r.t. f) such that the log of $L_\delta(\sigma)$ equals $R_{\log}(f_{\delta(\sigma)} \mid \sigma)$:

$$\log L_\delta(\sigma) \overset{\text{def}}{=} \mathrm{E}\left[\log \mathcal{L}(f_\delta(\boldsymbol{x}_\sigma), \boldsymbol{y}) \mid \sigma\right]. \tag{9.25}$$

Proof. First, we observe that $R_{\log}(f_{\delta(\sigma)} \mid \sigma)$ is a deterministic quantity and is independent of f. Using the fact that $L_\delta(\sigma)$ is a deterministic constant, we can show that

$$\begin{aligned}
R_{\log}(f \mid \sigma) - R_{\log}(f_{\delta(\sigma)} \mid \sigma) &= \mathrm{E}\left[\log \mathcal{L}(f(\boldsymbol{x}_\sigma), \boldsymbol{y}) \mid \sigma\right] - \log L_\delta(\sigma) \\
&= \mathrm{E}\left[\log\left(\frac{\mathcal{L}(f(\boldsymbol{x}_\sigma), \boldsymbol{y})}{L_\delta(\sigma)}\right) \ \mid \ \sigma\right] \\
&= \mathrm{E}\left[\log\left(1 + \frac{\mathcal{L}(f(\boldsymbol{x}_\sigma), \boldsymbol{y}) - L_\delta(\sigma)}{L_\delta(\sigma)}\right) \ \mid \ \sigma\right] \\
&\approx \mathrm{E}\left[\frac{\mathcal{L}(f(\boldsymbol{x}_\sigma), \boldsymbol{y}) - L_\delta(\sigma)}{L_\delta(\sigma)} \ \mid \ \sigma\right],
\end{aligned}$$

where we used the fact that $\mathcal{L}(f(\boldsymbol{x}_\sigma), \boldsymbol{y}) - L_\delta(\sigma) \ll L_\delta(\sigma)$ so that $\log(1+x) \approx x$. Putting these into the constraint $R_{\log}(f \mid \sigma) - R_{\log}(f_{\delta(\sigma)} \mid \sigma) \leq \epsilon$ and rearranging the terms completes the proof. □

The consequence of the above analysis leads to the following approximate problem for training in the log-scale:

$$\begin{aligned} f^* &\overset{\text{def}}{=} \underset{f}{\text{argmin}} \quad R(f), \\ &\text{s.t.} \quad \sup_{\sigma \in \Omega} \left\{ \frac{R(f \mid \sigma)}{L_\delta(\sigma)} \right\} \leq 1 + \epsilon. \end{aligned} \tag{P1-log}$$

The implication of Equation (P1-log) is that the optimization problem with log-scale constraints can be solved using the linear-scale approaches. Notice that the new distribution is now $\boldsymbol{\pi}(\sigma) = p(\sigma) + \frac{\lambda(\sigma)}{L_\delta(\sigma)}$. The other change is that we replace $R(f_{\delta(\sigma)} \mid \sigma)$ with $L_\delta(\sigma)$, which are determined offline.

9.7 Experiments

We evaluate the proposed framework through two experiments. The first experiment is based on a linear estimator where analytic solutions are available to verify the dual ascent algorithm. The second experiment is based on training a real deep neural network.

9.7.1 Linear estimator

We consider a linear (scalar) estimator so that we can access the analytic solutions. We define the clean signal as $y \sim \mathcal{N}(0, \sigma_y^2)$ and the noisy signal as $x = y + \sigma\eta$, where $\eta \sim \mathcal{N}(0, 1)$. The estimator we choose here is $f_{\boldsymbol{\pi}}(x) = a_{\boldsymbol{\pi}} x$ for some parameter $a_{\boldsymbol{\pi}}$ depending on the underlying sampling distribution $\boldsymbol{\pi}$.

Because of the linear model formulation, we can train the estimator $\widehat{a}_{\boldsymbol{\pi}}$ using closed-form equation as

$$\widehat{a}_{\boldsymbol{\pi}} = \underset{a}{\text{argmin}} \int \mathrm{E}\left[(ax - y)^2 \mid \sigma\right] \boldsymbol{\pi}(\sigma) d\sigma = \frac{\sigma_y^2}{\sigma_y^2 + \overline{\sigma}_{\boldsymbol{\pi}}^2},$$

where $\overline{\sigma}_{\pi}^2 \overset{\text{def}}{=} \int \sigma^2 \boldsymbol{\pi}(\sigma) d\sigma$. Substituting $\widehat{a}_{\pi}$ into the loss we can show that the conditional risk is

$$\begin{aligned} R(f_{\pi} \mid \sigma) &= \mathrm{E}\left[(\widehat{a}_{\pi}x - y)^2 \mid \sigma\right] \\ &= \frac{\sigma_y^4 \left[\sigma_y^2(\sigma_y^2 + \sigma^2) - 2\sigma_y^2(\sigma_y^2 + \overline{\sigma}_{\pi}^2) + (\sigma_y^2 + \overline{\sigma}_{\pi}^2)^2\right]}{(\sigma_y^2 + \overline{\sigma}_{\pi}^2)^2}. \end{aligned}$$

Based on this condition risks, we can run the dual ascent algorithm to alternatingly estimate $\boldsymbol{\pi}$ and $\widehat{a}_{\pi}$ according to Equation (P1). Figure **9.3** shows the conditional risks returned by different iterations of the dual ascent algorithm. In this numerical example, we let $\sigma_y = 10$ and $\epsilon = 9$. Observe that as the dual ascent algorithm proceeds, the worst case gap is reducing [2]. When the algorithm converges, it matches exactly with the theoretical solution.

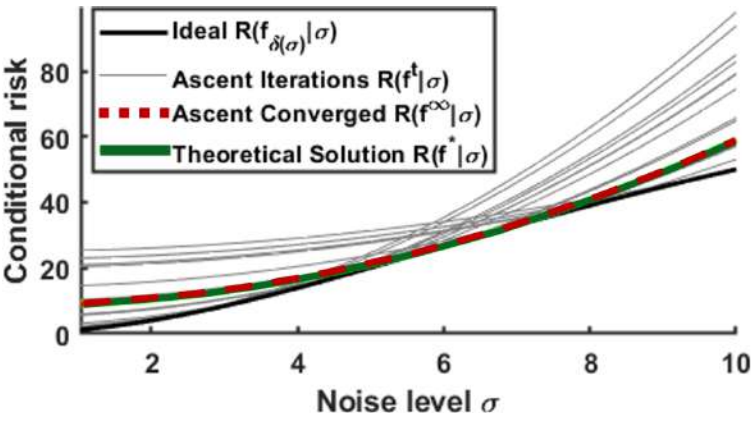


Figure 9.3. Conditional risks of the linear problem. As the dual ascent algorithm proceeds, the risk approaches the optimal solution.

9.7.2 Deep neural networks

The second experiment evaluates the effectiveness of the proposed framework on real deep neural networks for the task of denoising. We shall focus on the MSE loss with PSNR constraints, although our theory applies to other loss functions such as SSIM [134] and MS-SSIM [321] also as long as they are convex. The noise model we assume is that $\boldsymbol{x}_{\sigma} = \boldsymbol{y} + \sigma\boldsymbol{\eta}$,

[2] ↑The small gap in the middle of the plot is intrinsic to this problem, since for any $\overline{\sigma}_{\pi}^2$ there always exists a σ such that $\overline{\sigma}_{\pi}^2 = \sigma$. At this σ, the conditional risk will always touch the ideal curve.

Table 9.1. Results of Section 9.7.2. This table shows the PSNR values returned by one-size-fits-all DnCNN denoisers whose sample distributions are defined according to (i) uniform distribution, (ii) solution of Equation (P1), and (iii) solution of Equation (P2).

Noise level (σ)	0-10	10-20	20-30	30-40	40-50	50-60	60-70	70-80	80-90	90-100
Ideal (Best Individually Trained Denoisers)										
PSNR	38.04	31.73	29.23	27.72	26.66	25.86	25.24	24.70	24.25	23.84
Uniform Distribution										
Distribution	10.0%	10.0%	10.0%	10.0%	10.0%	10.0%	10.0%	10.0%	10.0%	10.0%
PSNR	37.24	31.41	29.04	27.60	26.58	25.81	25.19	24.67	24.23	23.84
Solution to Equation (P1) with 0.4dB gap										
Distribution	32.7%	12.0%	9.4%	7.9%	6.8%	6.3%	6.4%	6.2%	6.2%	6.1%
PSNR	37.64	31.46	29.03	27.58	26.56	25.78	25.15	24.63	24.19	23.80
Solution to Equation (P2)										
Distribution	81.3%	7.6%	3.4%	2.0%	1.3%	1.0%	0.9%	0.9%	0.8%	0.8%
PSNR	37.86	31.54	29.06	27.57	26.53	25.74	25.10	24.57	24.12	23.70

where $\boldsymbol{\eta} \sim \mathcal{N}(0, \boldsymbol{I})$ with $\sigma \in [0, 100]$ (w.r.t. an 8-bit signal of 256 levels). The network we consider is a 20-layer DnCNN [181]. We choose DnCNN just for demonstration. Since our framework does not depend on a specific network architecture, the theoretical results hold regardless the choice of the networks.

The training procedure is as follows. The training set consists of 400 images from the dataset in [207]. Each image has a size of 180×180. We randomly crop 50×50 patches from these images to construct the training set. The total number of patches we used is determined by the mini-batch size of the training algorithm. Specifically, for each dual ascent iteration we use 3000 mini-batches where each batch consists of 128 patches. This gives us 384k training patches per epoch. To create the noisy training samples, for each patch we add additive i.i.d. Gaussian noise where the noise level is randomly drawn from the distribution $\boldsymbol{\pi}(\sigma)$. The noise generation process is done online. We run our proposed algorithm for 25 dual ascent iterations, where each iteration consists of 10 epochs. For computational efficiency, we break the noise range $[0, 100]$ into 10 equally sized bins. For example, a uniform distribution corresponds to allocating 10% of the total number of training samples per bin. The validation set consists of 12 "standard images" (e.g., Lena). The testing set is the BSD68 dataset [322], tested individually for every noise bin. The testing distribution $p(\sigma)$ for Equation (P1) is assumed to be uniform in Figure 9.4. Two other distributions are illustrated in Figure 9.5. Notice that Equation (P2) does not require the testing distribution to be known.

The average PSNR values (conditional on σ) are reported in Table 9.1 and the performance gaps are illustrated in Figure 9.4. Specifically, the first two rows of the Table show the PSNR of the best individually trained denosiers and the uniform distributions. The proposed sampling distributions and the corresponding PSNR values are shown in the third row for Equation (P1) and the fourth row for Equation (P2). For Equation (P1), we set the tolerance level as 0.4dB. Table 9.1 and Figure 9.4 confirm the validity of our method. A more interesting observation is the percentages of the training samples. For Equation (P1), we need to allocate 32.7% of the data to low-noise, and this percentage goes up to 81.3% for Equation (P2). This suggests that the optimal sampling distribution could be substantially different from the uniform distribution we use today.

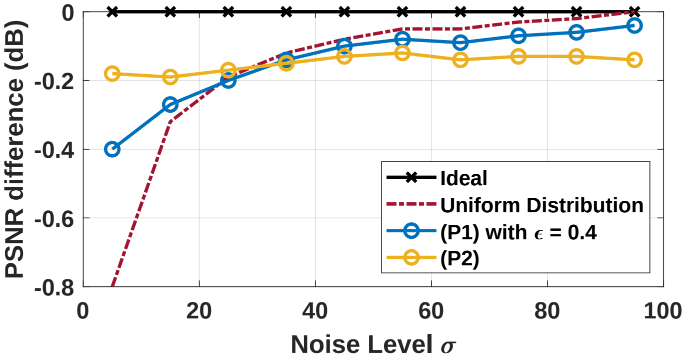


Figure 9.4. This figure shows the PSNR difference between the one-size-fits-all denoisers and the ideal denoiser. Observe that the uniform distribution favors high-noise cases and performs poorly on low-noise cases. By using the proposed algorithm we are able to allocate training samples such that the gap is consistent across the range. Equation (P1) ensures that the gap will not exceed 0.4dB, whereas Equation (P2) ensures that the gap is constant.

9.8 Discussions

9.8.1 Consistent gap = better?

It is important to note that one-size-fits-all denosiers are about the trade-off between high-noise and low-noise cases; we offer more degrees of freedom for the low-noise cases because the high-noise cases can be learned well using fewer samples. However, achieving

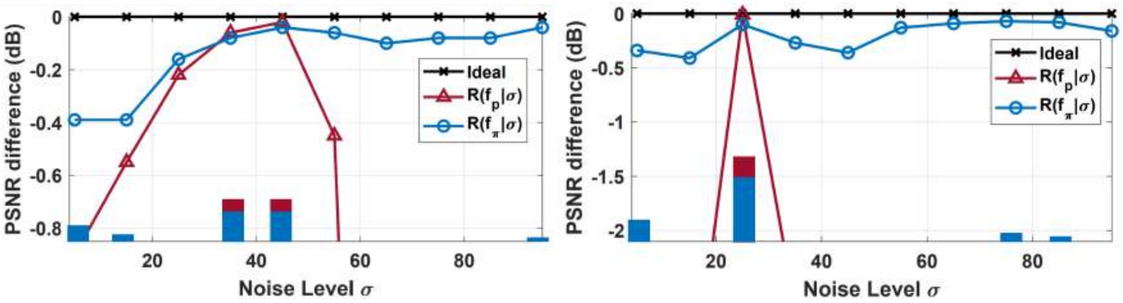


Figure 9.5. Usage of Equation (P1) when we are not certain about the true distribution. (a) The true distribution is unknown but we hypothesize that it is Uniform[30,50]. If we train the network using this distribution, we obtain the red curve. Equation (P1) starts with this hypothesis, and returns the blue curve. (b) Same experiment by hypothesizing that the distribution is Uniform[20,30]. Observe the robust performance of our method in both cases. The experimental setup is the same as Section 9.7.2.

a consistent gap does not mean that we are doing "better". The solution of Equation (P2) is not necessarily "better" than the solution of Equation (P1). The ultimate decision is application specific. If we care more about the heavy noise cases such as imaging in the dark (e.g., [25], [106]) and we are willing to compromise some performance for the weak noise cases, then Equation (P1) could be a better than the uniform gap solution returned by Equation (P2). Vice versa, if we know nothing about the testing distribution and we want to be conservative, then Equation (P2) is more useful.

Another consideration is how much we know about $p(\sigma)$. If we are absolutely certain that the noise is concentrated at a single value, then we should just allocate all the samples at that noise level. However, if we know something about $p(\sigma)$ but we are not absolutely sure, then Equation (P1) can provide the worst case performance guarantee. This is illustrated in Figure 9.5, where we solved Equation (P1) using two hypothesized distributions. It can be observed that if we train the network using the hypothesized distributions, the performance could be bad for extreme situations. In contrast, Equation (P1) compromises the peak performance by offering more robust performance in other situations.

9.8.2 Rule-of-thumb distribution — the “80-20” rule

Suppose that we are only looking at image denoisers with $p(\sigma)$ being uniform, and our goal is to achieve a consistent gap, then we can construct some “rule-of-thumb” distributions that are applicable to a few network architectures. Figure 9.6 shows three deep networks: REDNet [142], DnCNN [181], FFDNet [155], all trained using a so-called “80-20” rule. In this rule, we allocate the majority of the samples to the weak cases and a few to the strong cases. The exact percentage of the “80-20” rule is network dependent but the trend is usually similar. For example, in Figure 9.6 we allocate 70% to [0,10], 15% to [10,20], 8% to [20,30], 5% to [30,40], and 2% to [40,50] to all the three networks. While there are some fluctuations of the PSNR differences, in general the resulting curves are quite uniform.

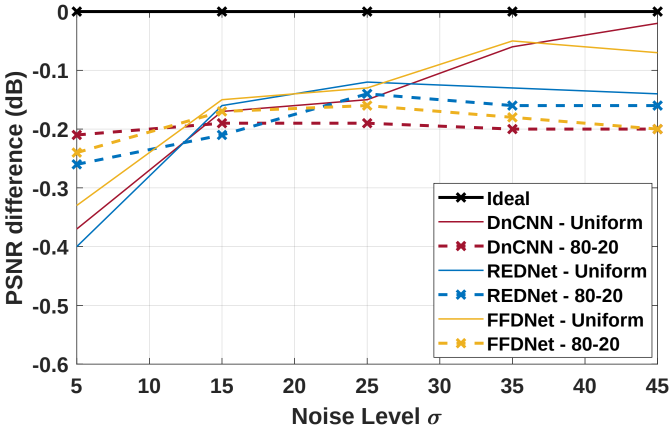


Figure 9.6. “80-20” Rule. We train the network with training samples drawn from different different noise levels according to the following distribution. $[0, 10] - 70\%$, $[10, 20] - 15\%$, $[20, 30] - 8\%$, $[30, 40] - 5\%$, $[40 - 50] - 2\%$. We observe that this distribution gives reasonably consistent performance at all the noise levels over all the denoisers considered here.

9.9 Some final thoughts

Imbalanced sampling of the training set is arguably very common in image restoration and related tasks. This chapter presents a framework which allows us to allocate training samples so that the overall performance of the one-size-fits-all denoiser is consistent across all noise levels. The convexity of the problem, the minimax formulation, and the dual ascent algorithm appear to be general for all learning-based estimators. The idea is likely to be applicable to adversarial training in classification tasks.

10. CONCLUSION AND FUTURE DIRECTIONS

Over the course of this dissertation, we have looked at ways to deal with imaging in challenging conditions, especially in low-light situations. To extend the imaging capabilities of the cameras to extremely low-light conditions, where the number of photons is scarce, it becomes necessary to use sensors sensitive to each photon. This is where image sensors like quanta image sensors (QIS) come in. In Chapter 2, we looked at how image sensors work and how different technologies differ. We built upon chapter 2 to understand the imaging model of the image sensors in chapter 3. We developed imaging models and tools such as signal-to-noise-ratio (SNR). The imaging model can be used to simulate different image sensors, and the SNR can be used to evaluate the performance of different types of sensors.

Based on the mathematical basis built in chapter 3, we solve some of the low light imaging problems in chapters 4 to 7. Chapter 4 deals with the issue of color imaging in low-light. Assuming that we use a standard Bayer pattern color filter array (CFA), we propose a traditional demosaicing algorithm based on plug-and-play ADMM and a neural network based solution. The proposed solutions could be used for QIS and standard CMOS image sensors. In chapter 5, we deal with the problem of reconstructing a moving scene utilizing a burst of frames captured using QIS. The proposed solution uses student-teacher learning to deal with noise and motion at the same time. In chapter 6, a non-blind deblurring method for low light is presented. The solution is not limited to any particular type of image sensor and could be used with any image sensor. We also collect a Canon camera dataset to evaluate how different methods work on real data in low light. Chapter 7 proposes two high-level computer vision tasks - image classification and object detection at low light. Both the methods are based on student-teacher learning.

In chapters 8 and 9, we start looking if our solutions are limited to low light alone or if we can extend them to other scenarios too. In chapter 8, we look at how quanta image sensors could be used to solve the problem of high dynamic range imaging. Chapter 9 examines how the same neural network can be trained to deal with low and high-light images.

Throughout this dissertation, we have looked at some solutions that require specific image sensors to achieve the goal. We have also looked at other generic solutions, which could be easily extended to any type of data. Student-teacher learning is one such example. It could be used for any noisy data, as long as we can access the corresponding clean data. Ultimately, the solution to achieve good low light imaging performance does not depend on a single technology or a solution. It can be achieved only by carefully designing image sensors and algorithms that complement each other.

The complementary design of algorithms and image sensors has already started redefining how we image. Until now, we usually have image sensors that generate data and make algorithms to handle this data. But for many purposes, the data captured by the image sensor may not be optimal. By co-designing the algorithm and the sensor, we can capture optimal data for the algorithms. Tools such as the SNR we developed in chapter 3 are the first step toward achieving this goal. As the image sensors become more and more flexible, we are moving toward what one can call *software-defined cameras.* In this dissertation, we have looked at changing integration times in the HDR chapter. Other than that, we haven't played around with the camera settings that much. The ability to control the camera settings such as frame rate, bit-depth, exposure time, and sensor gain on the fly based which gets input from the downstream algorithms will be a game-changer. The new sensor technologies such as quanta image sensors will play a significant role in achieving this goal because of the flexibility they offer.

REFERENCES

[1] M. Aubert, P. Setiawan, A. A. Oktaviana, A. Brumm, P. H. Sulistyarto, *et al.*, "Palaeolithic cave art in Borneo," *Nature*, vol. 564, no. 7735, pp. 254–257, 2018.

[2] *The Creation of the Heavens (detail) 1508-12*, https://www.michelangelo-gallery.org/The-Creation-Of-The-Heavens-Detail-1508-12.html, Accessed: Nov-11-2021.

[3] *The first photograph*, https://petapixel.com/2013/10/02/first-photo/, Accessed: Nov-11-2021.

[4] *The first digital photograph*, https://www.nist.gov/news-events/news/2007/05/fiftieth-anniversary-first-digital-image-marked, Accessed: Nov-11-2021.

[5] *Illustration of the camera obscura principle from James Ayscough's A short account of the eye and nature of vision (1755 fourth edition)*, https://en.wikipedia.org/wiki/Camera_obscura#/media/File:1755_james_ayscough.jpg, Accessed: Apr-15-2022.

[6] *Kodak Brownie*, https://commons.wikimedia.org/wiki/File:2014-365-233_The_Basic_Brownie_Camera_(14809795240).jpg, Accessed: Mar-29-2022.

[7] *Polaroid Land Camera 95*, https://www.psnwa.org/ws/12293-autosave-v1/, Accessed: Apr-15-2022.

[8] *First color photograph*, https://en.wikipedia.org/wiki/File:Tartan_Ribbon.jpg, Accessed: Apr-15-2022.

[9] *Casio QV-10 LCD Digital Camera Registered as an Essential Historical Material for Science and Technology by Japan's National Museum of Nature and Science*, https://arch.casio.com/news/2012/0913_qv10/, Accessed: Mar-29-2022.

[10] D. A. Spak, J. Plaxco, L. Santiago, M. Dryden, and B. Dogan, "BI-RADS fifth edition: A summary of changes," *Diagnostic and Interventional Imaging*, vol. 98, no. 3, pp. 179–190, 2017.

[11] *Pollen grains*, https://en.wikipedia.org/wiki/Scanning_electron_microscope#/media/File:Misc_pollen.jpg, Accessed: Mar-29-2022.

[12] *Fluoroscopy*, https://www.iowaradiology.com/services/fluoroscopy/, Accessed: Mar-29-2022.

[13] *Gamma camera*, https://commons.wikimedia.org/wiki/File:Gamma_Camera_Bone_Scan_Head_Cest_Knees_Pelvis.jpg, Accessed: Mar-29-2022.

[14] *How Does MRI Help Diagnose MS?* https://www.radiology.ca/article/how-does-mri-help-diagnose-ms, Accessed: Mar-29-2022.

[15] *Seeing light echoes*, https://www.nasa.gov/content/discoveries-highlights-seeing-light-echoes, Accessed: Mar-29-2022.

[16] A. Soloman, "Beitrage zur pathologie und klinik des mammakarzinoms," *Arch F Kun Chir*, vol. 101, p. 573, 1913.

[17] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE, 2016, pp. 770–778.

[18] F. Banterle, A. Artusi, K. Debattista, and A. Chalmers, *Advanced High Dynamic Range Imaging: Theory and Practice.* CRC Press, 2011.

[19] X. Guo, Y. Li, and H. Ling, "LIME: Low-light image enhancement via illumination map estimation," *IEEE Transactions on Image Processing*, vol. 26, no. 2, pp. 982–993, 2016.

[20] A. Gnanasambandam and S. H. Chan, "Exposure-referred signal-to-noise ratio for digital image sensors," *arXiv preprint arXiv:2112.05817*, 2021.

[21] A. Gnanasambandam, O. Elgendy, J. Ma, and S. H. Chan, "Megapixel photon-counting color imaging using quanta image sensor," *Optics Express*, vol. 27, no. 12, pp. 17 298–17 310, Jun. 2019.

[22] O. A. Elgendy, A. Gnanasambandam, S. H. Chan, and J. Ma, "Low-light demosaicking and denoising for small pixels using learned frequency selection," *IEEE Transactions on Computational Imaging*, vol. 7, pp. 137–150, 2021.

[23] Y. Chi, A. Gnanasambandam, V. Koltun, and S. H. Chan, "Dynamic low-light imaging with quanta image sensors," in *Proc. European Conference on Computer Vision (ECCV)*, Springer, 2020.

[24] Y. Sanghvi, A. Gnanasambandam, and S. H. Chan, "Photon limited non-blind deblurring using algorithm unrolling," *arXiv preprint arXiv:2110.15314*, 2021.

[25] A. Gnanasambandam and S. H. Chan, "Image classification in the dark using quanta image sensors," in *Proc. European Conference on Computer Vision (ECCV)*, Springer, 2020, pp. 484–501.

[26] C. Li, X. Qu, A. Gnanasambandam, O. A. Elgendy, J. Ma, and S. H. Chan, “Photon-limited object detection using non-local feature matching and knowledge distillation,” in *Proc. IEEE/CVF International Conference on Computer Vision (ICCV)*, IEEE, 2021, pp. 3976–3987.

[27] A. Gnanasambandam and S. H. Chan, “HDR imaging with quanta image sensors: Theoretical limits and optimal reconstruction,” *IEEE Transactions on Computational Imaging*, vol. 6, pp. 1571–1585, 2020.

[28] A. Gnanasambandam and S. H. Chan, “One size fits all: Can we train one denoiser for all noise levels?” In *Proc. International Conference on Machine Learning (ICML)*, PMLR, 2020, pp. 3576–3586.

[29] J. Nakamura, *Image sensors and signal processing for digital still cameras*. CRC press, 2005.

[30] *History of the camera*, https://en.wikipedia.org/wiki/History_of_the_camera, Accessed: Feb-08-2022.

[31] G. Eastman, *Photographic film*, US Patent 306,470, 1884.

[32] W. A. Adcock, *Electronic photography system*, US Patent 4,057,830, Nov. 1977.

[33] E. R. Fossum, “Some thoughts on future digital still cameras,” *Image sensors and signal processing for digital still cameras*, 2006.

[34] J. Ma and E. R. Fossum, “A pump-gate jot device with high conversion gain for a quanta image sensor,” *IEEE Journal of the Electron Devices Society*, vol. 3, no. 2, pp. 73–77, 2015.

[35] J. Ma, D. Zhang, O. A. Elgendy, and S. Masoodian, “A 0.19 e-rms read noise 16.7 Mpixel stacked quanta image sensor with 1.1 μm-pitch backside illuminated pixels,” *IEEE Electron Device Letters*, vol. 42, no. 6, pp. 891–894, 2021.

[36] A. Rochas, M. Gosch, A. Serov, *et al.*, “First fully integrated 2-D array of single-photon detectors in standard CMOS technology,” *IEEE Photonics technology letters*, vol. 15, no. 7, pp. 963–965, 2003.

[37] H. Shiraki, N. Teranishi, and Y. Ishihara, *Solid-state imaging device having a reduced image lag*, US Patent 4,484,210, Nov. 1984.

[38] *Image sensors world - Digicam market research*, http://image-sensors-world.blogspot.com/2006/12/digicam-market-research.html, Accessed: Feb-11-2022.

[39] N. Teranishi, A. Kohono, Y. Ishihara, E. Oda, and K. Arai, “No image lag photodiode structure in the interline CCD image sensor,” in *Proc. International Electron Devices Meeting*, IEEE, 1982, pp. 324–327.

[40] A. Tiwari and R. Talwekar, “Journey of visual prosthesis with progressive development of electrode design techniques and experience with CMOS image sensors: A review,” *IETE Journal of Research*, vol. 65, no. 2, pp. 172–200, 2019.

[41] J. Ma and E. R. Fossum, “Quanta image sensor jot with sub 0.3 e- rms read noise and photon counting capability,” *IEEE Electron Device Letters*, vol. 36, no. 9, pp. 926–928, 2015.

[42] D. Bronzi, F. Villa, S. Tisa, A. Tosi, and F. Zappa, “SPAD figures of merit for photon-counting, photon-timing, and imaging applications: A review,” *IEEE Sensors Journal*, vol. 16, no. 1, pp. 3–12, 2015.

[43] C. Bruschini, S. Burri, S. Lindner, *et al.*, “Monolithic SPAD arrays for high-performance, time-resolved single-photon imaging,” in *International Conference on Optical MEMS and Nanophotonics (OMN)*, IEEE, 2018, pp. 1–5.

[44] *Canon develops SPAD sensor*, https://global.canon/en/news/2021/20211215.html, Accessed: Feb-14-2022.

[45] *iXon Ultra*, https://andor.oxinst.com/assets/uploads/products/andor/documents/andor-ixon-ultra-emccd-specifications.pdf, Accessed: Feb-14-2022.

[46] J. Ma, S. Masoodian, D. A. Starkey, and E. R. Fossum, “Photon-number-resolving megapixel image sensor at room temperature without avalanche gain,” *OSA Optica*, vol. 4, no. 12, pp. 1474–1481, Dec. 2017.

[47] I. M. Antolovic, S. Burri, C. Bruschini, R. Hoebe, and E. Charbon, “Nonuniformity analysis of a 65-Kpixel CMOS SPAD imager,” *IEEE Transactions on Electron Devices*, vol. 63, no. 1, pp. 57–64, 2015.

[48] J. W. Goodman, *Statistical optics.* John Wiley & Sons, 2015.

[49] M. Delbracio, D. Kelly, M. S. Brown, and P. Milanfar, “Mobile computational photography: A tour,” *Annual Review of Vision Science*, vol. 7, pp. 571–604, 2021.

[50] L. Mandel, “Fluctuations of photon beams: The distribution of the photo-electrons,” *Proc. Physical Society*, vol. 74, no. 3, p. 233, 1959.

[51] K. G. Binmore and K. G. Binmore, *Mathematical Analysis: A straightforward approach.* Cambridge University Press, 1982.

[52] M. Hansard, S. Lee, O. Choi, and R. P. Horaud, *Time-of-flight cameras: Principles, methods and applications*. Springer Science & Business Media, 2012.

[53] A. Ingle, T. Seets, M. Buttafava, *et al.*, "Passive inter-photon imaging," in *Proc. IEEE/CVF Conference on Computer Vision and Pattern Recognition*, IEEE, 2021, pp. 8585–8595.

[54] P. R. Norton, "Infrared image sensors," *Optical Engineering*, vol. 30, no. 11, pp. 1649–1663, 1991.

[55] J. Kowalczyk, J. J. Perkins, A. DeAngelis, *et al.*, "Bulk measurement of copper and sodium content in CuIn(0.7)Ga(0.3)Se(2)(cigs) solar cells with nanosecond pulse length laser induced breakdown spectroscopy (LIBS)," *arXiv preprint arXiv:1301.1313*, 2013.

[56] D. P. Bertsekas and J. N. Tsitsiklis, *Introduction to probability*. Athena Scientific, 2000.

[57] *Solar spectrum*, https://commons.wikimedia.org/wiki/File:Solar_spectrum_en.svg, Accessed: Jan-31-2022.

[58] *onsemi - KAF-8300*, https://www.onsemi.com/pdf/datasheet/kaf-8300-d.pdf, Accessed: Dec-08-2021.

[59] *Photons to photos - Independent sensor data*, https://www.photonstophotos.net/, Accessed: Jan-27-2022.

[60] *EVMA - Standard for characterization of image sensors and cameras*, https://www.emva.org/wp-content/uploads/EMVA1288-3.1rc1.pdf, Accessed: Jan-27-2022.

[61] *PRNU and DSNU*, https://www.photometrics.com/learn/advanced-imaging/pattern-noise-dsnu-and-prnu, Accessed: Jan-27-2022.

[62] M. H. White, D. R. Lampe, F. C. Blaha, and I. A. Mack, "Characterization of surface channel CCD image arrays at low light levels," *IEEE Journal of Solid-State Circuits*, vol. 9, no. 1, pp. 1–12, 1974.

[63] *EE392B: Introduction to image sensors and digital cameras - Abbas El Gamal, Stanford University*, www.stanford.edu/class/ee392b/, Accessed: Jan-27-2022.

[64] H. Farid, *Photo forensics*. MIT press, 2016.

[65] E. R. Fossum, "Modeling the performance of single-bit and multi-bit quanta image sensors," *IEEE Journal of the Electron Devices Society*, vol. 1, no. 9, pp. 166–174, 2013.

[66] *Clarkvision*, https://clarkvision.com/, Accessed: Jan-28-2022.

[67] *Laser interference*, https://commons.wikimedia.org/wiki/File:Laser_Interference.JPG, Accessed: Jan-31-2022.

[68] L. Anzagira, *Imaging performance in advanced small pixel and low light image sensors.* Dartmouth College, 2016.

[69] K. Mabuchi, *Back-illuminated type solid-state imaging device*, US Patent 7,795,676, Sep. 2010.

[70] Z. F. Li, *Microlens array*, US Patent 6,587,147, Jul. 2003.

[71] L. Anzagira and E. R. Fossum, "Color filter array patterns designed to mitigate crosstalk effects in small pixel image sensors," *Journal of the Optical Society of America A*, vol. 32, no. 1, pp. 28–34, 2015.

[72] O. A. Elgendy and S. H. Chan, "Color filter arrays for quanta image sensors," *IEEE Transactions on Computational Imaging*, vol. 6, pp. 652–665, 2020.

[73] C. E. Shannon, "Communication in the presence of noise," *Proc. IRE*, vol. 37, no. 1, pp. 10–21, 1949.

[74] S. Lim, "Characterization of noise in digital photographs for image processing," in *Digital Photography II*, SPIE, vol. 6069, 2006, pp. 219–228.

[75] M. Granados, B. Ajdin, M. Wand, C. Theobalt, H.-P. Seidel, and H. P. Lensch, "Optimal HDR reconstruction with linear digital cameras," in *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE, 2010, pp. 215–222.

[76] S. H. Chan, O. A. Elgendy, and X. Wang, "Images from bits: Non-iterative image reconstruction for quanta image sensors," *MDPI Sensors*, vol. 16, no. 11, p. 1961, 2016.

[77] F. Yang, Y. M. Lu, L. Sbaiz, and M. Vetterli, "Bits from photons: Oversampled image acquisition using binary poisson statistics," *IEEE Transactions on Image Processing*, vol. 21, no. 4, pp. 1421–1436, 2011.

[78] H. Tian, B. Fowler, and A. E. Gamal, "Analysis of temporal noise in CMOS photodiode active pixel sensor," *IEEE Journal of Solid-State Circuits*, vol. 36, no. 1, pp. 92–101, 2001.

[79] T. Mitsunaga and S. K. Nayar, "Radiometric self calibration," in *Proc. Conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE, vol. 1, 1999, pp. 374–380.

[80] O. A. Elgendy and S. H. Chan, "Optimal threshold design for quanta image sensor," *IEEE Transactions on Computational Imaging*, vol. 4, no. 1, pp. 99–111, 2017.

[81] J. R. Whittlesey, “Incomplete gamma functions for evaluating erlang process probabilities,” *Mathematics of Computation*, pp. 11–17, 1963.

[82] H. S. Malvar, L. W. He, and R. Cutler, “High-quality linear interpolation for demosaicing of bayer-patterned color images,” in *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, IEEE, vol. 3, 2004, pp. iii–485.

[83] K. Cui, Z. Jin, and E. Steinbach, “Color image demosaicking using a 3-stage convolutional neural network structure,” in *IProc. IEEE International Conference on Image Processing (ICIP)*, IEEE, 2018, pp. 2177–2181.

[84] N. Yan and J. Ouyang, “Cross-channel correlation preserved three-stream lightweight CNNs for demosaicking,” 2019.

[85] D. Menon and G. Calvagno, “Color image demosaicking: An overview,” *Signal Processing: Image Communication*, vol. 26, no. 8-9, pp. 518–533, 2011.

[86] S. H. Park, H. S. Kim, S. Lansel, M. Parmar, and B. A. Wandell, “A case for denoising before demosaicking color filter array data,” in *Asilomar Conference on Signals, Systems and Computers*, IEEE, 2009, pp. 860–864.

[87] P. Chatterjee, N. Joshi, S. B. Kang, and Y. Matsushita, “Noise suppression in low-light images through joint denoising and demosaicing,” in *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE, 2011, pp. 321–328.

[88] Q. Jin, G. Facciolo, and J.-M. Morel, “A review of an old dilemma: Demosaicking first, or denoising first?” In *Proc. IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, 2020, pp. 514–515.

[89] T. Seybold, C. Keimel, M. Knopp, and W. Stechele, “Towards an evaluation of denoising algorithms with respect to realistic camera noise,” in *International Symposium on Multimedia*, IEEE, 2013, pp. 203–210.

[90] M. Gharbi, G. Chaurasia, S. Paris, and F. Durand, “Deep joint demosaicking and denoising,” *ACM Transactions on Graphics (ToG)*, vol. 35, no. 6, pp. 1–12, 2016.

[91] W. Dong, M. Yuan, X. Li, and G. Shi, “Joint demosaicing and denoising with perceptual optimization on a generative adversarial network,” *arXiv preprint arXiv:1802.04723*, 2018.

[92] T. Huang, F. F. Wu, W. Dong, G. Shi, and X. Li, “Lightweight deep residue learning for joint color image demosaicking and denoising,” in *International Conference on Pattern Recognition (ICPR)*, IEEE, 2018, pp. 127–132.

[93] T. Ehret, A. Davy, P. Arias, and G. Facciolo, “Joint demosaicking and denoising by fine-tuning of bursts of raw images,” in *Proc. IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019, pp. 8868–8877.

[94] L. Condat and S. Mosaddegh, “Joint demosaicking and denoising by total variation minimization,” in *Proc. IEEE International Conference on Image Processing (ICIP)*, IEEE, 2012, pp. 2781–2784.

[95] F. Heide, M. Steinberger, Y.-T. Tsai, *et al.*, “FlexISP: A flexible camera image processing framework,” *ACM Transactions on Graphics (ToG)*, vol. 33, no. 6, pp. 1–13, 2014.

[96] D. Khashabi, S. Nowozin, J. Jancsary, and A. W. Fitzgibbon, “Joint demosaicing and denoising via learned nonparametric random fields,” *IEEE Transactions on Image Processing*, vol. 23, no. 12, pp. 4968–4981, 2014.

[97] T. Klatzer, K. Hammernik, P. Knobelreiter, and T. Pock, “Learning joint demosaicing and denoising based on sequential energy minimization,” in *Proc. IEEE International Conference on Computational Photography (ICCP)*, 2016, pp. 1–11.

[98] H. Tan, X. Zeng, S. Lai, Y. Liu, and M. Zhang, “Joint demosaicing and denoising of noisy bayer images with ADMM,” in *Proc. IEEE International Conference on Image Processing (ICIP)*, IEEE, 2017, pp. 2951–2955.

[99] F. Kokkinos and S. Lefkimmiatis, “Iterative joint image demosaicking and denoising using a residual denoising network,” *IEEE Transactions on Image Processing*, vol. 28, no. 8, pp. 4177–4188, 2019.

[100] J. Wu, R. Timofte, and L. Van Gool, “Demosaicing based on directional difference regression and efficient regression priors,” *IEEE Transactions on Image Processing*, vol. 25, no. 8, pp. 3862–3874, 2016.

[101] D. Kiku, Y. Monno, M. Tanaka, and M. Okutomi, “Beyond color difference: Residual interpolation for color image demosaicking,” *IEEE Transactions on Image Processing*, vol. 25, no. 3, pp. 1288–1300, 2016.

[102] F. Yang, Y. M. Lu, L. Sbaiz, and M. Vetterli, “An optimal algorithm for reconstructing images from binary measurements,” in *Computational Imaging VIII*, International Society for Optics and Photonics, vol. 7533, 2010, pp. 158–169.

[103] F. Yang, Y. M. Lu, L. Sbaiz, and M. Vetterli, “Bits from photons: Oversampled image acquisition using binary Poisson statistics,” *IEEE Transactions on Image Processing*, vol. 21, no. 4, 2012.

[104] S. H. Chan and Y. M. Lu, "Efficient image reconstruction for gigapixel quantum image sensors," in *Global Conference on Signal and Information Processing*, IEEE, 2014, pp. 312–316.

[105] T. Remez, O. Litany, and A. Bronstein, "A picture is worth a billion bits: Real-time image reconstruction from dense binary threshold pixels," in *Proc. IEEE International Conference on Computational Photography (ICCP)*, IEEE, 2016, pp. 1–9.

[106] J. H. Choi, O. A. Elgendy, and S. H. Chan, "Image reconstruction for quanta image sensors using deep neural networks," in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, IEEE, 2018, pp. 6543–6547.

[107] P. Chandramouli, S. Burri, C. Bruschini, E. Charbon, and A. Kolb, "A little bit too much? High speed imaging from sparse photon counts," *arXiv preprint arXiv:1811.02396*, 2018.

[108] R. A. Rojas, W. Luo, V. Murray, and Y. M. Lu, "Learning optimal parameters for binary sensing image reconstruction algorithms," in *Proc. of IEEE International Conference on Image Processing (ICIP)*, IEEE, 2017, pp. 2791–2795.

[109] L. Azzari and A. Foi, "Variance stabilization in Poisson image deblurring," in *International Symposium on Biomedical Imaging*, IEEE, 2017, pp. 728–731.

[110] L. Azzari and A. Foi, "Variance stabilization for noisy+ estimate combination in iterative Poisson denoising.," *IEEE Signal Processing Letters*, vol. 23, no. 8, 2016.

[111] A. Foi, "Noise estimation and removal in MR imaging: The variance-stabilization approach," in *Proc. International symposium on biomedical imaging: from nano to macro*, IEEE, 2011, pp. 1809–1814.

[112] F. J. Anscombe, "The transformation of Poisson, binomial and negative-binomial data," *Biometrika*, vol. 35, no. 3-4, 1948.

[113] A. Foi, "Optimization of variance-stabilizing transformations," *Preprint*, vol. 94, pp. 1809–1814, 2009.

[114] M. Wang, B. Lyu, and G. Yu, "ConvexVST: A convex optimization approach to variance-stabilizing transformation," in *Proc. International Conference on Machine Learning (ICML)*, PMLR, 2021, pp. 10 839–10 848.

[115] S. H. Chan, "Performance analysis of plug-and-play ADMM: A graph signal processing perspective," *IEEE Transactions on Computational Imaging*, 2019.

[116] S. H. Chan, X. Wang, and O. A. Elgendy, “Plug-and-play ADMM for image restoration: Fixed-point convergence and applications,” *IEEE Transactions on Computational Imaging*, vol. 3, no. 1, 2017.

[117] G. Jeon and E. Dubois, “Demosaicking of noisy bayer-sampled color images with least-squares luma-chroma demultiplexing and noise level estimation,” *IEEE Transactions on Image Processing*, vol. 22, no. 1, pp. 146–156, 2012.

[118] J. T. Korneliussen and K. Hirakawa, “Camera processing with chromatic aberration,” *IEEE Transactions on Image Processing*, vol. 23, no. 10, pp. 4539–4552, 2014.

[119] K. Hirakawa, X.-L. Meng, and P. J. Wolfe, “A framework for wavelet-based analysis and processing of color filter array images with applications to denoising and demosaicing,” in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, IEEE, vol. 1, 2007, pp. I–597.

[120] E. Dubois, “Frequency-domain methods for demosaicking of bayer-sampled color images,” *IEEE Signal Processing Letters*, vol. 12, no. 12, pp. 847–850, 2005.

[121] D. Alleysson, S. Susstrunk, and J. Hérault, “Linear demosaicing inspired by the human visual system,” *IEEE Transactions on Image Processing*, vol. 14, no. 4, pp. 439–449, 2005.

[122] A. V. Oppenheim, A. S. Willsky, S. H. Nawab, G. M. Hernández, *et al.*, *Signals & systems*. Pearson Educación, 1997.

[123] L. Condat, “A new color filter array with optimal properties for noiseless and noisy color image acquisition,” *IEEE Transactions on Image Processing*, vol. 20, no. 8, pp. 2200–2210, 2011.

[124] N.-X. Lian, V. Zagorodnov, and Y.-P. Tan, “Edge-preserving image denoising via optimal color space projection,” *IEEE Transactions on Image Processing*, vol. 15, no. 9, pp. 2575–2587, 2006.

[125] K. Dabov, A. Foi, V. Katkovnik, and K. Egiazarian, “Color image denoising via sparse 3D collaborative filtering with grouping constraint in luminance-chrominance space,” in *Proc. IEEE International Conference on Image Processing (ICIP)*, IEEE, vol. 1, 2007, pp. I–313.

[126] O. Ronneberger, P. Fischer, and T. Brox, “U-net: Convolutional networks for biomedical image segmentation,” in *International Conference on Medical image computing and computer-assisted intervention*, Springer, 2015, pp. 234–241.

[127] S. Laine, T. Karras, J. Lehtinen, and T. Aila, “High-quality self-supervised deep image denoising,” *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 32, 2019.

[128] K. Imai and T. Miyata, “Gated texture CNN for efficient and configurable image denoising,” in *Proc. European Conference on Computer Vision (ECCV)*, Springer, 2020, pp. 665–681.

[129] T. Marinč, V. Srinivasan, S. Gül, C. Hellge, and W. Samek, “Multi-kernel prediction networks for denoising of burst images,” in *Proc. IEEE International Conference on Image Processing (ICIP)*, IEEE, 2019, pp. 2404–2408.

[130] J. Johnson, A. Alahi, and L. Fei-Fei, “Perceptual losses for real-time style transfer and super-resolution,” in *Proc. European Conference on Computer Vision (ECCV)*, Springer, 2016.

[131] K. Ma, Z. Duanmu, Q. Wu, *et al.*, “Waterloo exploration database: New challenges for image quality assessment models,” *IEEE Transactions on Image Processing*, vol. 26, no. 2, pp. 1004–1016, 2016.

[132] L. Zhang, X. Wu, A. Buades, and X. Li, “Color demosaicking by local directional interpolation and nonlocal adaptive thresholding,” *Journal of Electronic imaging*, vol. 20, no. 2, p. 023 016, 2011.

[133] E. Agustsson and R. Timofte, “NTIRE 2017 challenge on single image super-resolution: Dataset and study,” in *Proc. IEEE Conference on Computer Vision and Pattern Recognition(CVPR) Workshops*, 2017, pp. 126–135.

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

[135] G. Sharma, W. Wu, and E. N. Dalal, “The CIEDE2000 color-difference formula: Implementation notes, supplementary test data, and mathematical observations,” *Color Research & Application*, vol. 30, no. 1, pp. 21–30, 2005.

[136] S. Van der Walt, J. L. Schönberger, J. Nunez-Iglesias, *et al.*, “Scikit-image: Image processing in python,” *PeerJ*, vol. 2, e453, 2014.

[137] R. Zhang, P. Isola, A. A. Efros, E. Shechtman, and O. Wang, “The unreasonable effectiveness of deep features as a perceptual metric,” in *Proc. IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018, pp. 586–595.

[138] L. Condat, “A simple, fast and efficient approach to denoisaicking: Joint demosaicking and denoising,” in *Proc. IEEE International Conference on Image Processing (ICIP)*, IEEE, 2010, pp. 905–908.

[139] *FLOPS counter*, https://github.com/sovrasov/flops-counter.pytorch/, Accessed: Feb-24-2021.

[140] G. Boracchi and A. Foi, “Uniform motion blur in poissonian noise: Blur/noise tradeoff,” *IEEE Transactions on Image Processing*, vol. 20, no. 2, pp. 592–598, 2010.

[141] B. Mildenhall, J. T. Barron, J. Chen, D. Sharlet, R. Ng, and R. Carroll, “Burst denoising with kernel prediction networks,” in *Proc. IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE, 2018, pp. 2502–2510.

[142] X. Mao, C. Shen, and Y.-B. Yang, “Image restoration using very deep convolutional encoder-decoder networks with symmetric skip connections,” *Advances in neural information processing systems (NeurIPS)*, vol. 29, 2016.

[143] B. K. Horn and B. G. Schunck, “Determining optical flow,” in *Techniques and Applications of Image Understanding*, Intl. Society Optics and Photonics, vol. 281, 1981.

[144] C. Sutour, C.-A. Deledalle, and J.-F. Aujol, “Adaptive regularization of the NL-means: Application to image and video denoising,” *IEEE Transactions on Image Processing*, vol. 23, no. 8, pp. 3506–3521, 2014.

[145] A. Buades, B. Coll, and J.-M. Morel, “Denoising image sequences does not require motion estimation,” in *Proc. IEEE Conference on Advanced Video and Signal Based Surveillance*, IEEE, 2005, pp. 70–74.

[146] S. Ma, S. Gupta, A. C. Ulku, C. Brushini, E. Charbon, and M. Gupta, “Quanta burst photography,” *ACM Transactions on Graphics*, vol. 39, no. 4, pp. 79–94, 2020.

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

[148] M. Everingham, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman, “The Pascal visual object classes (VOC) challenge,” *International Journal of Computer Vision*, vol. 88, no. 2, pp. 303–338, 2010.

[149] S. Gould, R. Fulton, and D. Koller, “Decomposing a scene into geometric and semantically consistent regions,” in *Proc. IEEE International Conference on Computer Vision ICCV*, IEEE, 2009, pp. 1–8.

[150] M. Maggioni, V. Katkovnik, K. Egiazarian, and A. Foi, “Nonlocal transform-domain filter for volumetric data denoising and reconstruction,” *IEEE Transactions on Image Processing*, vol. 22, no. 1, pp. 119–133, 2012.

[151] F. Chollet *et al.*, *Keras*, https://keras.io, 2015.

[152] Martín Abadi, Ashish Agarwal, Paul Barham, *et al.*, *TensorFlow: Large-scale machine learning on heterogeneous systems*, 2015. [Online]. Available: https://www.tensorflow.org/.

[153] G. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," *NeurIPS Deep Learning and Representation Learning Workshop*, 2015.

[154] D. Gong, Z. Zhang, Q. Shi, A. van den Hengel, C. Shen, and Y. Zhang, "Learning deep gradient descent optimization for image deconvolution," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 31, no. 12, pp. 5468–5482, 2020.

[155] K. Zhang, W. Zuo, and L. Zhang, "FFDNet: Toward a fast and flexible solution for cnn-based image denoising," *IEEE Transactions on Image Processing*, vol. 27, no. 9, pp. 4608–4622, 2018.

[156] T. Eboli, J. Sun, and J. Ponce, "End-to-end interpretable learning of non-blind image deblurring," in *Proc. European Conference on Computer Vision (ECCV)*, Springer, 2020, pp. 314–331.

[157] Y. Nan, Y. Quan, and H. Ji, "Variational-EM-based deep learning for noise-blind image deblurring," in *Proc. IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020, pp. 3626–3635.

[158] J. Dong, S. Roth, and B. Schiele, "Deep Wiener deconvolution: Wiener meets deep learning for image deblurring," *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 33, pp. 1048–1059, 2020.

[159] W. Dong, P. Wang, W. Yin, G. Shi, F. Wu, and X. Lu, "Denoising prior driven deep neural network for image restoration," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 41, no. 10, pp. 2305–2318, 2018.

[160] H. Wang and P. C. Miller, "Scaled heavy-ball acceleration of the Richardson-Lucy algorithm for 3d microscopy image restoration," *IEEE Transactions on Image Processing*, vol. 23, no. 2, pp. 848–854, 2013.

[161] J.-L. Starck and F. Murtagh, *Astronomical image and data analysis.* Springer Science & Business Media, 2007.

[162] J. Li, F. Luisier, and T. Blu, "PURE-LET image deconvolution," *IEEE Transactions on Image Processing*, vol. 27, no. 1, pp. 92–105, 2017.

[163] W. H. Richardson, "Bayesian-based iterative method of image restoration," *J. Opt. Soc. Am.*, vol. 62, no. 1, pp. 55–59, 1972.

[164] L. B. Lucy, “An iterative technique for the rectification of observed distributions,” *The Astronomical Journal*, vol. 79, p. 745, 1974.

[165] S. H. Chan, X. Wang, and O. A. Elgendy, “Plug-and-play ADMM for image restoration: Fixed-point convergence and applications,” *IEEE Transactions on Computational Imaging*, vol. 3, no. 1, pp. 84–98, 2016.

[166] A. Rond, R. Giryes, and M. Elad, “Poisson inverse problems by the plug-and-play scheme,” *Journal of Visual Communication and Image Representation*, vol. 41, pp. 96–108, 2016.

[167] M. Bertero, P. Boccacci, G. Desiderà, and G. Vicidomini, “Image deblurring with Poisson data: From cells to galaxies,” *Inverse Problems*, vol. 25, no. 12, p. 123 006, 2009.

[168] L. A. Shepp and Y. Vardi, “Maximum likelihood reconstruction for emission tomography,” *IEEE Transactions on Medical Imaging*, vol. 1, no. 2, pp. 113–122, 1982.

[169] N. Dey, L. Blanc-Feraud, C. Zimmer, *et al.*, “Richardson-Lucy algorithm with total variation regularization for 3d confocal microscope deconvolution,” *Microscopy Research and Technique*, vol. 69, no. 4, pp. 260–266, 2006.

[170] M. Laasmaa, M. Vendelin, and P. Peterson, “Application of regularized Richardson-Lucy algorithm for deconvolution of confocal microscopy images,” *Journal of Microscopy*, vol. 243, no. 2, pp. 124–140, 2011.

[171] M. A. Figueiredo and J. M. Bioucas-Dias, “Restoration of Poissonian images using alternating direction optimization,” *IEEE transactions on Image Processing*, vol. 19, no. 12, pp. 3133–3145, 2010.

[172] Z. T. Harmany, R. F. Marcia, and R. M. Willett, “This is SPIRAL-TAP: Sparse Poisson intensity reconstruction algorithms—theory and practice,” *IEEE Transactions on Image Processing*, vol. 21, no. 3, pp. 1084–1096, 2011.

[173] R. D. Nowak and E. D. Kolaczyk, “A statistical multiscale framework for Poisson inverse problems,” *IEEE Transactions on Information Theory*, vol. 46, no. 5, pp. 1811–1825, 2000.

[174] T. Blu and F. Luisier, “The SURE-LET approach to image denoising,” *IEEE Transactions on Image Processing*, vol. 16, no. 11, pp. 2778–2786, 2007.

[175] F. Xue, F. Luisier, and T. Blu, “Multi-Wiener SURE-LET deconvolution,” *IEEE Transactions on Image Processing*, vol. 22, no. 5, pp. 1954–1968, 2013.

[176] M. Makitalo and A. Foi, “Optimal inversion of the Anscombe transformation in low-count Poisson image denoising,” *IEEE Transactions on Image Processing*, vol. 20, no. 1, pp. 99–109, 2010.

[177] F. Luisier, C. Vonesch, T. Blu, and M. Unser, “Fast interscale wavelet denoising of Poisson-corrupted images,” *Signal processing*, vol. 90, no. 2, pp. 415–427, 2010.

[178] B. Zhang, J. M. Fadili, and J.-L. Starck, “Wavelets, ridgelets, and curvelets for Poisson noise removal,” *IEEE Transactions on Image Processing*, vol. 17, no. 7, pp. 1093–1108, 2008.

[179] S. Sreehari, S. V. Venkatakrishnan, B. Wohlberg, *et al.*, “Plug-and-play priors for bright field electron tomography and sparse interpolation,” *IEEE Transactions on Computational Imaging*, vol. 2, no. 4, pp. 408–423, 2016.

[180] R. Ahmad, C. A. Bouman, G. T. Buzzard, *et al.*, “Plug-and-play methods for magnetic resonance imaging: Using denoisers for image recovery,” *IEEE Signal Processing Magazine*, vol. 37, no. 1, pp. 105–116, 2020.

[181] K. Zhang, W. Zuo, S. Gu, and L. Zhang, “Learning deep CNN denoiser prior for image restoration,” in *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE, 2017, pp. 3929–3938.

[182] T. He, Y. Sun, B. Chen, J. Qi, W. Liu, and J. Hu, “Plug-and-play inertial forward–backward algorithm for Poisson image deconvolution,” *Journal of Electronic Imaging*, vol. 28, no. 4, p. 043 020, 2019.

[183] U. S. Kamilov, H. Mansour, and B. Wohlberg, “A plug-and-play priors approach for solving nonlinear imaging inverse problems,” *IEEE Signal Processing Letters*, vol. 24, no. 12, pp. 1872–1876, 2017.

[184] Y. Sun, B. Wohlberg, and U. S. Kamilov, “An online plug-and-play algorithm for regularized image reconstruction,” *IEEE Transactions on Computational Imaging*, vol. 5, no. 3, pp. 395–408, 2019.

[185] G. T. Buzzard, S. H. Chan, S. Sreehari, and C. A. Bouman, “Plug-and-play unplugged: Optimization-free reconstruction using consensus equilibrium,” *SIAM Journal on Imaging Sciences*, vol. 11, no. 3, pp. 2001–2020, 2018.

[186] S. H. Chan, “Performance analysis of plug-and-play ADMM: A graph signal processing perspective,” *IEEE Transactions on Computational Imaging*, vol. 5, no. 2, pp. 274–286, 2019.

[187] E. Ryu, J. Liu, S. Wang, X. Chen, Z. Wang, and W. Yin, “Plug-and-play methods provably converge with properly trained denoisers,” in *Proc. International Conference on Machine Learning (ICML)*, PMLR, 2019, pp. 5546–5557.

[188] Y. Romano, M. Elad, and P. Milanfar, “The little engine that could: Regularization by denoising (RED),” *SIAM Journal on Imaging Sciences*, vol. 10, no. 4, pp. 1804–1844, 2017.

[189] R. Cohen, M. Elad, and P. Milanfar, “Regularization by denoising via fixed-point projection (red-pro),” *SIAM Journal on Imaging Sciences*, vol. 14, no. 3, pp. 1374–1406, 2021.

[190] E. T. Reehorst and P. Schniter, “Regularization by denoising: Clarifications and new interpretations,” *IEEE Transactions on Computational Imaging*, vol. 5, no. 1, pp. 52–67, 2018.

[191] K. Gregor and Y. LeCun, “Learning fast approximations of sparse coding,” in *Proc. International Conference on Machine Learning (ICML)*, PMLR, 2010, pp. 399–406.

[192] K. Zhang, L. V. Gool, and R. Timofte, “Deep unfolding network for image super-resolution,” in *Proc. IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE, 2020, pp. 3217–3226.

[193] Y. Li, M. Tofighi, V. Monga, and Y. C. Eldar, “An algorithm unrolling approach to deep image deblurring,” in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, IEEE, 2019, pp. 7675–7679.

[194] Y. Li, M. Tofighi, J. Geng, V. Monga, and Y. C. Eldar, “Efficient and interpretable deep blind image deblurring via algorithm unrolling,” *IEEE Transactions on Computational Imaging*, vol. 6, pp. 666–681, 2020.

[195] Y. Yang, J. Sun, H. Li, and Z. Xu, “Deep ADMM-Net for compressive sensing MRI,” in *Proc. International Conference on Neural Information Processing Systems (NeurIPS)*, 2016, pp. 10–18.

[196] R. Liu, S. Cheng, L. Ma, X. Fan, and Z. Luo, “Deep proximal unrolling: Algorithmic framework, convergence analysis and applications,” *IEEE Transactions on Image Processing*, vol. 28, no. 10, pp. 5013–5026, 2019.

[197] V. Monga, Y. Li, and Y. C. Eldar, “Algorithm unrolling: Interpretable, efficient deep learning for signal and image processing,” *IEEE Signal Processing Magazine*, vol. 38, no. 2, pp. 18–44, 2021.

[198] D. Gilton, G. Ongie, and R. Willett, “Deep equilibrium architectures for inverse problems in imaging,” *IEEE Transactions on Computational Imaging*, vol. 7, pp. 1123–1133, 2021.

[199] S. Boyd, N. Parikh, and E. Chu, *Distributed optimization and statistical learning via the alternating direction method of multipliers*. Now Publishers Inc., 2011.

[200] R. H. Byrd, P. Lu, J. Nocedal, and C. Zhu, “A limited memory algorithm for bound constrained optimization,” *SIAM Journal on scientific computing*, vol. 16, no. 5, pp. 1190–1208, 1995.

[201] M. A. Figueiredo and J. M. Bioucas-Dias, “Deconvolution of Poissonian images using variable splitting and augmented lagrangian optimization,” in *Proc. IEEE/SP Workshop on Statistical Signal Processing*, IEEE, 2009, pp. 733–736.

[202] B. Lim, S. Son, H. Kim, S. Nah, and K. Mu Lee, “Enhanced deep residual networks for single image super-resolution,” in *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, IEEE, 2017, pp. 136–144.

[203] G. Boracchi and A. Foi, “Modeling the performance of image restoration from motion blur,” *IEEE Transactions on Image Processing*, vol. 21, no. 8, pp. 3502–3517, 2012.

[204] A. Levin, Y. Weiss, F. Durand, and W. T. Freeman, “Understanding and evaluating blind deconvolution algorithms,” in *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE, 2009, pp. 1964–1971.

[205] D. P. Kingma and J. Ba, “Adam: A Method for Stochastic Optimization,” in *International Conference on Learning Representations (ICLR)*, 2014.

[206] X. Glorot and Y. Bengio, “Understanding the difficulty of training deep feedforward neural networks,” in *Proc. International Conference on Artificial Intelligence and Statistics*, JMLR, 2010, pp. 249–256.

[207] D. Martin, C. Fowlkes, D. Tal, and J. Malik, “A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics,” in *Proc. IEEE International Conference on Computer Vision (ICCV)*, vol. 2, 2001, pp. 416–423.

[208] J. Ba and R. Caruana, “Do deep nets really need to be deep?” In *Advances in Neural Information Processing Systems (NeurIPS)*, 2014.

[209] Y. Zhang, T. Xiang, T. M. Hospedales, and H. Lu, “Deep mutual learning,” in *Proc. IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018, pp. 4320–4328.

[210] C. Yang, L. Xie, S. Qiao, and A. Yuille, “Training deep neural networks in generations: A more tolerant teacher educates better students,” in *Proc. AAAI Conf. Artificial Intelligence*, 2019.

[211] T. Guo, C. Xu, S. He, B. Shi, C. Xu, and D. Tao, "Robust student network learning," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 31, no. 7, pp. 2455–2468, 2019.

[212] A. G. Howard, M. Zhu, B. Chen, *et al.*, "Mobilenets: Efficient convolutional neural networks for mobile vision applications," *arXiv preprint arXiv:1704.04861*, 2017.

[213] L. A. Gatys, A. S. Ecker, and M. Bethge, "Texture synthesis using convolutional neural networks," in *Proc. Advances in neural information processing systems (NeurIPS)*, 2015.

[214] L. A. Gatys, A. S. Ecker, and M. Bethge, "A neural algorithm of artistic style," *arXiv preprint arXiv:1508.06576*, 2015.

[215] G. Chen, Y. Li, and S. N. Srihari, "Joint visual denoising and classification using deep learning," in *Proc. IEEE International Conference on Image Processing (ICIP)*, IEEE, 2016, pp. 3673–3677.

[216] D. Liu, B. Wen, J. Jiao, X. Liu, Z. Wang, and T. S. Huang, "Connecting image denoising and high-level vision tasks via deep learning," *IEEE Transactions on Image Processing*, vol. 29, pp. 3695–3706, 2020.

[217] H. Talebi and P. Milanfar, "Learned perceptual image enhancement," in *Proc. IEEE International Conference on Computational Photography (ICCP)*, IEEE, 2018, pp. 1–13.

[218] A. Mahendran and A. Vedaldi, "Understanding deep image representations by inverting them," in *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR*, IEEE, 2015, pp. 5188–5196.

[219] K. Simonyan, A. Vedaldi, and A. Zisserman, "Deep inside convolutional networks: Visualising image classification models and saliency maps," *International Conference on Learning Representations (ICLR)*, 2014.

[220] J. Yosinski, J. Clune, A. Nguyen, T. Fuchs, and H. Lipson, "Understanding neural networks through deep visualization," *International Conference of Machine Learning (ICML) Deep Learning Workshop*, 2015.

[221] B. Chen and P. Perona, "Seeing into darkness: Scotopic visual recognition," in *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE, 2017.

[222] S. Diamond, V. Sitzmann, F. Julca-Aguilar, S. Boyd, G. Wetzstein, and F. Heide, "Dirty pixels: Towards end-to-end image processing and perception," *ACM Transactions on Graphics (TOG)*, vol. 40, no. 3, pp. 1–15, 2021.

[223] M. T. Hossain, S. W. Teng, D. Zhang, S. Lim, and G. Lu, “Distortion robust image classification using deep convolutional neural network with discrete cosine transform,” in *Proc. IEEE International Conference on Image Processing (ICIP)*, IEEE, 2019.

[224] J. Wu, R. Timofte, Z. Huang, and L. Van Gool, “On the relation between color image denoising and classification,” *arXiv preprint arXiv:1704.01372*, 2017.

[225] S. Dodge and L. Karam, “Quality resilient deep neural networks,” *arXiv preprint arXiv: 1703.08119*, 2017.

[226] Z. Liu, T. Zhou, Z. Shen, B. Kang, and T. Darrell, “Transferable recognition-aware image processing,” *arXiv preprint arXiv:1910.09185*, 2019.

[227] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, “Imagenet: A large-scale hierarchical image database,” in *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE, 2009.

[228] A. Khosla, N. Jayadevaprakash, B. Yao, and F.-F. Li, “Novel dataset for fine-grained image categorization: Stanford dogs,” in *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, IEEE, 2011.

[229] *FLIR Sensor Review: Mono Camera*, https://www.flir.com/globalassets/industrial/iis/guidebooks/2019-machine-vision-emva1288-sensor-review.pdf, Accessed: Feb-16-2022.

[230] C. Chen, Q. Chen, J. Xu, and V. Koltun, “Learning to see in the dark,” in *Proc. IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE, 2018.

[231] H. Law and J. Deng, “Cornernet: Detecting objects as paired keypoints,” in *Proc. European Conference on Computer Vision (ECCV)*, Springer, 2018, pp. 734–750.

[232] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, “Focal loss for dense object detection,” in *Proc. IEEE International Conference on Computer Vision (ICCV)*, IEEE, 2017, pp. 2980–2988.

[233] W. Liu, D. Anguelov, D. Erhan, *et al.*, “SSD: Single shot multibox detector,” in *Proc. European Conference on Computer Vision (ECCV)*, Springer, 2016, pp. 21–37.

[234] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, “You only look once: Unified, real-time object detection,” in *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE, 2016, pp. 779–788.

[235] K. He, X. Zhang, S. Ren, and J. Sun, “Spatial pyramid pooling in deep convolutional networks for visual recognition,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 37, no. 9, pp. 1904–1916, 2015.

[236] J. Dai, Y. Li, K. He, and J. Sun, “R-FCN: Object detection via region-based fully convolutional networks,” *Proc. Advances in Neural Information Processing Systems (NeurIPS)*, vol. 29, 2016.

[237] S. Ren, K. He, R. Girshick, and J. Sun, “Faster R-CNN: Towards real-time object detection with region proposal networks,” *Proc. Advances in Neural Information Processing Systems (NeurIPS)*, vol. 28, 2015.

[238] K. He, G. Gkioxari, P. Dollár, and R. Girshick, “Mask R-CNN,” in *Proc. IEEE International Conference on Computer Vision (ICCV)*, IEEE, 2017, pp. 2961–2969.

[239] X. Zhu, Y. Wang, J. Dai, L. Yuan, and Y. Wei, “Flow-guided feature aggregation for video object detection,” in *Proc. IEEE International Conference on Computer Vision (ICCV)*, IEEE, 2017, pp. 408–417.

[240] X. Zhu, J. Dai, L. Yuan, and Y. Wei, “Towards high performance video object detection,” in *Proc. IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018, pp. 7210–7218.

[241] F. Xiao and Y. J. Lee, “Video object detection with an aligned spatial-temporal memory,” in *Proc. European Conference on Computer Vision (ECCV)*, Springer, 2018, pp. 485–501.

[242] M. Liu and M. Zhu, “Mobile video object detection with temporally-aware feature maps,” in *Proc. IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE, 2018, pp. 5686–5695.

[243] G. Bertasius, L. Torresani, and J. Shi, “Object detection in video with spatiotemporal sampling networks,” in *Proc. European Conference on Computer Vision (ECCV)*, Springer, 2018, pp. 331–346.

[244] S. Wang, Y. Zhou, J. Yan, and Z. Deng, “Fully motion-aware network for video object detection,” in *Proc. European conference on computer vision (ECCV)*, Springer, 2018, pp. 542–557.

[245] J. Deng, Y. Pan, T. Yao, W. Zhou, H. Li, and T. Mei, “Relation distillation networks for video object detection,” in *Proc. IEEE/CVF International Conference on Computer Vision*, IEEE, 2019, pp. 7023–7032.

[246] C. Feichtenhofer, A. Pinz, and A. Zisserman, “Detect to track and track to detect,” in *Proc. IEEE International Conference on Computer Vision (ICCV)*, ICCV, 2017, pp. 3038–3046.

[247] Y. Chen, Y. Cao, H. Hu, and L. Wang, "Memory enhanced global-local aggregation for video object detection," in *Proc. IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE, 2020, pp. 10 337–10 346.

[248] Y. P. Loh and C. S. Chan, "Getting to know low-light images with the exclusively dark dataset," *Elsevier Computer Vision and Image Understanding*, vol. 178, pp. 30–42, 2019.

[249] W. Yang, Y. Yuan, W. Ren, *et al.*, "Advancing image understanding in poor visibility environments: A collective benchmark study," *IEEE Transactions on Image Processing*, vol. 29, pp. 5737–5752, 2020.

[250] Y. Sasagawa and H. Nagahara, "Yolo in the dark-domain adaptation method for merging multiple models," in *Proc. European Conference on Computer Vision (ECCV)*, Springer, 2020, pp. 345–359.

[251] T.-Y. Lin, M. Maire, S. Belongie, *et al.*, "Microsoft Coco: Common objects in context," in *Proc. European Conference on Computer Vision (ECCV*, Springer, 2014, pp. 740–755.

[252] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature pyramid networks for object detection," in *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVR)*, IEEE, 2017, pp. 2117–2125.

[253] J. Yu, Y. Jiang, Z. Wang, Z. Cao, and T. Huang, "Unitbox: An advanced object detection network," in *Proc. ACM International Conference on Multimedia*, ACM, 2016, pp. 516–520.

[254] Y. He, C. Zhu, J. Wang, M. Savvides, and X. Zhang, "Bounding box regression with uncertainty for accurate object detection," in *Proc. IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE, 2019, pp. 2888–2897.

[255] D. J. Jobson, Z.-u. Rahman, and G. A. Woodell, "A multiscale retinex for bridging the gap between color images and the human observation of scenes," *IEEE Transactions on Image processing*, vol. 6, no. 7, pp. 965–976, 1997.

[256] D. Coltuc, P. Bolon, and J.-M. Chassery, "Exact histogram specification," *IEEE Transactions on Image processing*, vol. 15, no. 5, pp. 1143–1152, 2006.

[257] H. Ibrahim and N. S. P. Kong, "Brightness preserving dynamic histogram equalization for image contrast enhancement," *IEEE Transactions on Consumer Electronics*, vol. 53, no. 4, pp. 1752–1758, 2007.

[258] Y.-T. Kim, "Contrast enhancement using brightness preserving bi-histogram equalization," *IEEE Transactions on Consumer Electronics*, vol. 43, no. 1, pp. 1–8, 1997.

[259] X. Fu, D. Zeng, Y. Huang, Y. Liao, X. Ding, and J. Paisley, "A fusion-based enhancing method for weakly illuminated images," *Elsevier Signal Processing*, vol. 129, pp. 82–96, 2016.

[260] C. Guo, C. Li, J. Guo, *et al.*, "Zero-reference deep curve estimation for low-light image enhancement," in *Proc. IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE, 2020, pp. 1780–1789.

[261] W. Ren, S. Liu, L. Ma, *et al.*, "Low-light image enhancement via a deep hybrid network," *IEEE Transactions on Image Processing*, vol. 28, no. 9, pp. 4364–4375, 2019.

[262] R. Wang, Q. Zhang, C.-W. Fu, X. Shen, W.-S. Zheng, and J. Jia, "Underexposed photo enhancement using deep illumination estimation," in *Proc. IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE, 2019, pp. 6849–6857.

[263] W. Yang, S. Wang, Y. Fang, Y. Wang, and J. Liu, "From fidelity to perceptual quality: A semi-supervised approach for low-light image enhancement," in *Proc. IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE, 2020, pp. 3063–3072.

[264] Y. Atoum, M. Ye, L. Ren, Y. Tai, and X. Liu, "Color-wise attention network for low-light image enhancement," in *Proc. IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, IEEE, 2020, pp. 2130–2139.

[265] S. Gu, Y. Li, L. V. Gool, and R. Timofte, "Self-guided network for fast image denoising," in *Proc. IEEE/CVF International Conference on Computer Vision (ICCV)*, IEEE, 2019, pp. 2511–2520.

[266] K. Xu, X. Yang, B. Yin, and R. W. Lau, "Learning to restore low-light images via decomposition-and-enhancement," in *Proc. IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE, 2020, pp. 2281–2290.

[267] F. Lv, Y. Li, and F. Lu, "Attention guided low-light image enhancement with a large scale low-light simulation dataset," *International Journal of Computer Vision*, vol. 129, pp. 2175–2193, Jul. 2021.

[268] X. Wang, R. Girshick, A. Gupta, and K. He, "Non-local neural networks," in *Proc. IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE, 2018.

[269] J. Yang, J. Lu, D. Batra, and D. Parikh, "A faster Pytorch implementation of faster R-CNN," *https://github.com/jwyang/faster-rcnn.pytorch*, 2017.

[270] E. Reinhard, W. Heidrich, P. Debevec, S. Pattanaik, G. Ward, and K. Myszkowski, *High dynamic range imaging: Acquisition, display, and image-based lighting.* Morgan Kaufmann, 2010.

[271] O. Gallo and P. Sen, “Stack-based algorithms for HDR capture and reconstruction,” in *High Dynamic Range Video*, Elsevier, 2016, pp. 85–119.

[272] P. E. Debevec and J. Malik, “Recovering high dynamic range radiance maps from photographs,” in *Proceesings of ACM SIGGRAPH Conference*, 1997.

[273] P. Sen and C. Aguerrebere, “Practical high dynamic range imaging of everyday scenes: Photographing the world as we see it with our own eyes,” *IEEE Signal Processing Magazine*, vol. 33, no. 5, pp. 36–44, 2016.

[274] A. Serrano, F. Heide, D. Gutierrez, G. Wetzstein, and B. Masia, “Convolutional sparse coding for high dynamic range imaging,” *Computer Graphics Forum*, vol. 35, no. 2, pp. 153–163, 2016.

[275] S. K. Nayar and T. Mitsunaga, “High dynamic range imaging: Spatially varying pixel exposures,” in *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE, vol. 1, 2000, pp. 472–479.

[276] T. Buades, Y. Lou, J.-M. Morel, and Z. Tang, “A note on multi-image denoising,” in *IEEE International Workshop on Local and Non-Local Approximation in Image Processing*, 2009, pp. 1–15.

[277] N. Joshi and M. F. Cohen, “Seeing Mt. Rainier: Lucky imaging for multi-image denoising, sharpening, and haze removal,” in *Proc. IEEE International Conference on Computational Photography (ICCP)*, IEEE, 2010, pp. 1–8.

[278] Y. Tsin, V. Ramesh, and T. Kanade, “Statistical calibration of CCD imaging process,” in *Proc. IEEE International Conference on Computer Vision (ICCV)*, IEEE, vol. 1, 2001, pp. 480–487.

[279] M. A. Robertson, S. Borman, and R. L. Stevenson, “Estimation-theoretic approach to dynamic range enhancement using multiple exposures,” *SPIE Journal of Electronic Imaging*, vol. 12, no. 2, pp. 219–228, 2003.

[280] K. Kirk and H. J. Andersen, “Noise characterization of weighting schemes for combination of multiple exposures,” in *Proc. British Machine Vision Conference (BMVC)*, Citeseer, vol. 3, 2006, pp. 1129–1138.

[281] J. Kronander, S. Gustavson, G. Bonnet, and J. Unger, “Unified HDR reconstruction from raw CFA data,” in *Proc. IEEE International Conference on Computational Photography (ICCP)*, IEEE, 2013, pp. 1–9.

[282] S. W. Hasinoff, F. Durand, and W. T. Freeman, “Noise-optimal capture for high dynamic range photography,” in *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE, 2010, pp. 553–560.

[283] T. Mertens, J. Kautz, and F. Van Reeth, “Exposure fusion: A simple and practical alternative to high dynamic range photography,” in *Computer Graphics Forum*, Wiley Online Library, vol. 28, 2009, pp. 161–171.

[284] G. Eilertsen, J. Kronander, G. Denes, R. K. Mantiuk, and J. Unger, “HDR image reconstruction from a single exposure using deep CNNs,” *ACM Transactions on Graphics*, vol. 36, no. 6, pp. 1–15, 2017.

[285] D. Marnerides, T. Bashford-Rogers, J. Hatchett, and K. Debattista, “ExpandNet: A deep convolutional neural network for high dynamic range expansion from low dynamic range content,” *Computer Graphics Forum*, vol. 37, no. 2, pp. 37–49, 2018.

[286] N. K. Kalantari and R. Ramamoorthi, “Deep high dynamic range imaging of dynamic scenes.,” *ACM Transactions on Graphics*, vol. 36, no. 4, pp. 144–1, 2017.

[287] S. Wu, J. Xu, Y.-W. Tai, and C.-K. Tang, “Deep high dynamic range imaging with large foreground motions,” in *Proc. European Conference on Computer Vision (ECCV)*, IEEE, 2018, pp. 117–132.

[288] J.-F. Cai, H. Ji, C. Liu, and Z. Shen, “Blind motion deblurring using multiple images,” *Elsevier Journal of Computational Physics*, vol. 228, no. 14, pp. 5057–5071, 2009.

[289] H. Zhang, D. Wipf, and Y. Zhang, “Multi-observation blind deconvolution with an adaptive sparse prior,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 36, no. 8, pp. 1628–1643, 2014.

[290] M. Delbracio and G. Sapiro, “Burst deblurring: Removing camera shake through Fourier burst accumulation,” in *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE, 2015, pp. 2385–2393.

[291] P. Wieschollek, M. Hirsch, B. Scholkopf, and H. Lensch, “Learning blind motion deblurring,” in *Proc. IEEE International Conference on Computer Vision (ICCV)*, IEEE, 2017, pp. 231–240.

[292] M. Aittala and F. Durand, “Burst image deblurring using permutation invariant convolutional neural networks,” in *Proc. European Conference on Computer Vision*, Springer, 2018, pp. 731–747.

[293] X. Tao, H. Gao, X. Shen, J. Wang, and J. Jia, “Scale-recurrent network for deep image deblurring,” in *Proc. IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE, 2018, pp. 8174–8182.

[294] N. Dutton, T. Al Abbas, I. Gyongy, F. Mattioli Della Rocca, and R. Henderson, “High dynamic range imaging at the quantum limit with Single Photon Avalanche Diode based image sensors,” *MDPI Sensors*, vol. 18, no. 4, p. 1166, 2018.

[295] S. Mann and R. Picard, “On being undigital with digital cameras: Extending dynamic range by combining exposed pictures,” in *IS&T 48th Annual Conference*, 1994, pp. 422–428.

[296] I. Gyongy, N. Dutton, and R. Henderson, “Single-photon tracking for high-speed vision,” *MDPI Sensors*, vol. 18, no. 2, 2018.

[297] F. Xiao, J. M. DiCarlo, P. B. Catrysse, and B. A. Wandell, “High Dynamic Range imaging of natural scenes,” in *Color and Imaging Conference*, Society for Imaging Science and Technology, 2002, pp. 337–342.

[298] B. McLernon, *Canon EOS 5D Mark II Digital Field Guide.* John Wiley & Sons, 2012, vol. 204.

[299] T. O. Aydın, R. Mantiuk, and H.-P. Seidel, “Extending quality metrics to full luminance range images,” in *Human Vision and Electronic Imaging XIII*, International Society for Optics and Photonics, vol. 6806, 2008, 68060B.

[300] J. H. Choi, O. A. Elgendy, and S. H. Chan, “Optimal combination of image denoisers,” *IEEE Transactions on Image Processing*, vol. 28, no. 8, pp. 4016–4031, 2019.

[301] Y. Kim, J. W. Soh, and N. I. Cho, “Adaptively tuning a convolutional neural network by gate process for image denoising,” *IEEE Access*, vol. 7, pp. 63 447–63 456, 2019.

[302] T. Remez, O. Litany, R. Giryes, and A. M. Bronstein, “Deep class-aware image denoising,” in *Proc. IEEE International Conference on Sampling Theory and Applications (SampTA)*, IEEE, 2017, pp. 138–142.

[303] R. Gao and K. Grauman, “On-demand learning for deep image restoration,” in *Proc. IEEE International Conference on Computer Vision (ICCV)*, IEEE, 2017, pp. 1086–1095.

[304] Y.-Q. Wang and J.-M. Morel, “Can a single image denoising neural network handle all levels of gaussian noise?” *IEEE Signal Processing Letters*, vol. 21, no. 9, pp. 1150–1153, 2014.

[305] B. Settles, *Active learning literature survey.* University of Wisconsin-Madison Department of Computer Sciences, 2009.

[306] K. Chaloner and I. Verdinelli, “Bayesian experimental design: A review,” *Statistical Science*, pp. 273–304, Aug. 1995.

[307] Y. Gal, R. Islam, and Z. Ghahramani, “Deep Bayesian active learning with image data,” in *Proc. International Conference on Machine Learning (ICML)*, vol. 70, 2017, pp. 1183–1192.

[308] O. Sener and S. Savarese, “Active learning for convolutional neural networks: A core-set approach,” in *International Conference on Learning Representations (ICLR)*, 2018.

[309] J. C. Platt and A. H. Barr, “Constrained differential optimization,” in *International Conference on Neural Information Processing Systems*, 1987, pp. 612–621.

[310] S. H. Zak, V. Upatising, and S. Hui, “Solving Linear Programming problems with Neural Networks: A comparative study,” *IEEE Transactions on Neural Networks*, vol. 6, no. 1, pp. 94–104, Jan. 1995.

[311] D. Pathak, P. Krahenbuhl, and T. Darrell, “Constrained convolutional neural networks for weakly supervised segmentation,” in *Proc. IEEE International Conference on Computer Vision (ICCV)*, IEEE, Dec. 2015, pp. 1796–1804.

[312] P. Márquez-Neila, M. Salzmann, and P. Fua, “Imposing hard constraints on deep networks: Promises and limitations,” *arXiv:1706.02025*, 2017.

[313] M. B. Zafar, I. Valera, M. G. Rodriguez, and K. P. Gummadi, “Fairness constraints: Mechanisms for fair classification,” *arXiv:1507.05259*, 2015.

[314] D. Pedreshi, S. Ruggieri, and F. Turini, “Discrimination-aware data mining,” in *International Conference on Knowledge Discovery and Data Mining*, Aug. 2008, pp. 560–568.

[315] T. Calders and S. Verwer, “Three naive Bayes approaches for discrimination-free classification,” *Data Mining and Knowledge Discovery*, vol. 21, no. 2, pp. 277–292, Sep. 2010.

[316] M. Hardt, E. Price, and N. Srebro, “Equality of opportunity in supervised learning,” in *Advances in Neural Information Processing Systems (NeurIPS)*, Dec. 2016, pp. 3315–3323.

[317] T. Kamishima, S. Akaho, H. Asoh, and J. Sakuma, “Fairness-aware classifier with prejudice remover regularizer,” in *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, Sep. 2012, pp. 35–50.

[318] M. B. Zafar, I. Valera, M. Gomez Rodriguez, and K. P. Gummadi, “Fairness beyond disparate treatment & disparate impact: Learning classification without disparate mistreatment,” in *International World Wide Web Conference*, Apr. 2017, pp. 1171–1180.

[319] P. Petersen, M. Raslan, and F. Voigtlaender, “Topological properties of the set of functions generated by neural networks of fixed size,” *Foundations of computational mathematics*, vol. 21, no. 2, pp. 375–444, 2021.

[320] *Machine Learning 10-725, CMU*, https://www.stat.cmu.edu/~ryantibs/convexopt-F18/lectures/dual-ascent.pdf, Accessed: Feb-10-2022.

[321] Z. Wang, E. P. Simoncelli, and A. C. Bovik, “Multiscale structural similarity for image quality assessment,” in *Asilomar Conference on Signals, Systems & Computers*, IEEE, vol. 2, 2003, pp. 1398–1402.

[322] S. Roth and M. J. Black, “Fields of experts: A framework for learning image priors,” in *Proc. IEEE Computer Vision and Pattern Recognition (CVPR)*, vol. 2, Jun. 2005, pp. 860–867.

A. GAUSSIAN APPROXIMATION TO POISSON

When deriving the first-order derivative of the incomplete Gamma function, it was mentioned that the Poisson distribution can be approximated by a Gaussian. The formal statement is as follows.

Lemma A.1 (Gaussian approximation of Poisson)**.** *For large θ (i.e., $\theta \gg 1$), it holds that*

$$p_X(x) \overset{\text{def}}{=} \frac{\theta^x \mathrm{e}^{-\theta}}{x!} \approx \frac{1}{\sqrt{2\pi\theta}} \mathrm{e}^{-\frac{(x-\theta)^2}{2\theta}}. \tag{A.1}$$

Note that this is *not* the Central Limit theorem because it does not involve any sample average. The approximation compares the two functions.

Proof. First of all, take the log on the Poisson equation:

$$\log p_X(x) = \log\left\{\frac{\theta^x \mathrm{e}^{-\theta}}{x!}\right\} = x\log\theta - \theta - \log x!$$

Stirling's formula states that for $x \to \infty$, we have $x! \approx x^x \mathrm{e}^{-x}\sqrt{2\pi x}$. Substitute into the previous equation yields

$$\begin{aligned} \log p_X(x) &\approx x\log\theta - \theta - \log\left(x^x \mathrm{e}^{-x}\sqrt{2\pi x}\right) \\ &= x\log\theta - \theta - x\log x + x - \log\sqrt{2\pi x}. \end{aligned}$$

The Gaussian has to fit the Poisson well around the mean, which is θ. Thus define $x = \theta + \epsilon$ with $\theta \gg \epsilon$. Then,

$$\begin{aligned}
\log p_X(x) &= x \log\theta - \theta - x\log x + x - \log\sqrt{2\pi x} \\
&= (\theta+\epsilon)\log\theta - \theta - (\theta+\epsilon)\log(\theta+\epsilon) \\
&\qquad + (\theta+\epsilon) - \log\sqrt{2\pi(\theta+\epsilon)} \\
&= \epsilon + (\theta+\epsilon)\log\frac{\theta}{\theta+\epsilon} - \log\sqrt{2\pi(\theta+\epsilon)} \\
&= \epsilon - (\theta+\epsilon)\log\left(1+\frac{\epsilon}{\theta}\right) - \log\sqrt{2\pi\theta} - \frac{1}{2}\log\left(1+\frac{\epsilon}{\theta}\right) \\
&= \epsilon - \log\sqrt{2\pi\theta} - \left(\theta+\epsilon+\frac{1}{2}\right)\log\left(1+\frac{\epsilon}{\theta}\right).
\end{aligned}$$

For $\frac{\epsilon}{\theta} \ll 1$, it holds that $\log(1+\frac{\epsilon}{\theta}) \approx \frac{\epsilon}{\theta} - \frac{\epsilon^2}{2\theta^2} + \ldots$. Therefore,

$$\begin{aligned}
\log p_X(x) &\approx \epsilon - \log\sqrt{2\pi\theta} - \left(\theta+\epsilon+\frac{1}{2}\right)\left(\frac{\epsilon}{\theta} - \frac{\epsilon^2}{2\theta^2} + \ldots\right) \\
&= \epsilon - \log\sqrt{2\pi\theta} - \epsilon - \frac{\epsilon^2}{\theta} - \frac{\epsilon}{2\theta} + \frac{\epsilon^2}{2\theta} + \frac{\epsilon^2}{4\theta^2} + \ldots.
\end{aligned}$$

By canceling terms, and removing $\frac{\epsilon^2}{4\theta^2}$ and $\frac{\epsilon}{2\theta}$ (because $\frac{\epsilon}{\theta} \ll 1$), it follows that

$$\log p_X(x) \approx -\frac{\epsilon^2}{2\theta} - \log\sqrt{2\pi\theta}.$$

This implies that

$$p_X(x) \approx \frac{1}{\sqrt{2\pi\theta}} \mathrm{e}^{-\frac{\epsilon^2}{2\theta}}.$$

Substituting $x = \theta + \epsilon$ completes the proof. □

B. EXPOSURE REFERRED SNR FOR TRUNCATED POISSON

Theorem B.0.1 (SNR$_{\text{exp}}(\beta)$ for truncated Poisson Gaussian)**.** *Consider the truncated Poisson Gaussian statistics defined in Equation (3.81). Let $\widehat{\beta}(\cdot)$ be an estimator satisfying the mean invariance property, i.e., $\widehat{\beta}(\mathrm{E}[Z]) = \beta$. Then the exposure-referred SNR is*

$$SNR_{exp}(\beta) = \frac{\beta}{\sqrt{\mathrm{Var}[Z]}} \cdot \frac{d\mu}{d\beta}, \tag{B.1}$$

where

$$\begin{aligned} \mathrm{E}[Z] = \mu &= \beta\Psi_{L-1}(\beta) + L(1-\Psi_L(\beta)) + \Delta_\mu(\beta), \\ \mathrm{Var}[Z] &= \beta^2\Psi_{L-2}(\beta) + \beta\Psi_{L-1}(\beta) + L^2(1-\Psi_L(\beta)) - \mu^2 + \Delta_\sigma^2(\beta), \end{aligned}$$

and the quantities $\Delta_\mu(\beta)$ and $\Delta_{\sigma^2}(\beta)$ are respectively

$$\begin{aligned} \Delta_\mu(\beta) = \sum_{k=-\infty}^{\infty} p_k \Bigg(&\sum_{q=[k]_+}^{L-1} \left(\frac{\mathrm{e}^{-\beta}\beta^{q-k}}{(q-k)!} - \frac{\mathrm{e}^{-\beta}\beta^q}{q!} \right) q \\ &+ L(\Psi_L(\beta) - \Psi_{[L-k]_+}(\beta)) \Bigg) \end{aligned} \tag{B.2}$$

$$\begin{aligned} \Delta_{\sigma^2}(\beta) = \sum_{k=-\infty}^{\infty} p_k \Bigg(&\sum_{q=[k]_+}^{L-1} \left(\frac{\mathrm{e}^{-\beta}\beta^{q-k}}{(q-k)!} - \frac{\mathrm{e}^{-\beta}\beta^q}{q!} \right) q^2 \\ &+ L^2(\Psi_L(\beta) - \Psi_{[L-k]_+}(\beta)) \Bigg), \end{aligned} \tag{B.3}$$

where $[\,\cdot\,]_+ = \max(\cdot, 0)$ returns the positive value, and

$$p_k = \int_{k-0.5}^{k+0.5} \frac{1}{\sqrt{2\pi\sigma_{read}^2}} \mathrm{e}^{-\frac{x^2}{2\sigma_{read}^2}} dx \tag{B.4}$$

is the error probability due to read noise. The derivative $d\mu_Z/d\beta$ is

$$\begin{aligned} \frac{\partial \mu_Z}{\partial \beta} &= \Psi_{L-1}(\beta) - \beta \frac{\mathrm{e}^{-\beta}\beta^{[L-2]_+}}{[L-2]_+!} + L\frac{\mathrm{e}^{-\beta}\beta^{[L-1]_+}}{[L-1]_+!} \\ &+ \sum_{k=-\infty}^{\infty} p_k \Bigg(\sum_{q=[k]_+}^{L-1} \bigg(-\frac{\mathrm{e}^{-\beta}\beta^{q-k}}{(q-k)!} + \frac{\mathrm{e}^{-\beta}\beta^{q}}{q!} \bigg) q \\ &+ \sum_{q=[k]_+}^{L-2} \bigg(\frac{\mathrm{e}^{-\beta}\beta^{q-k}}{(q-k)!} - \frac{\mathrm{e}^{-\beta}\beta^{q}}{q!} \bigg) (q+1) \\ &+ L\Big(-\frac{\beta^{(L-1)}\mathrm{e}^{-\beta}}{(L-1)!} + \frac{\beta^{([L-k-1]_+)}\mathrm{e}^{-\beta}}{[L-k-1]_+!} \Big) \Bigg). \end{aligned} \tag{B.5}$$

Proof. We first make the observation that

$$Z = \mathrm{ADC}\big(Y + \eta\big) = \mathrm{ADC}\big(\lceil Y + \eta \rfloor\big).$$

We introduce a new random variable variable $R = \lceil Y + \eta \rfloor$. Now,

$$\begin{aligned} R &= \lceil Y + \eta \rfloor \\ &= Y + \lceil \eta \rfloor . \end{aligned}$$

The final step is possible because Y is an integer. We introduce another new random variable $\gamma = \lceil \eta \rfloor$. Now the pmf of γ is

$$p_k = \mathbb{P}(\gamma = k) = \int_{k-0.5}^{k+0.5} \frac{1}{\sqrt{2\pi\sigma_{\text{read}}^2}} \mathrm{e}^{-\frac{x^2}{2\sigma_{\text{read}}^2}} dx. \tag{B.6}$$

So, $R = Y + \gamma$, where

$$\mathbb{P}(Y = \mathrm{j}) = \frac{\mathrm{e}^{-\beta}\beta^{\mathrm{j}}}{\mathrm{j}!} \text{ if } \mathrm{j} \geq 0$$

and γ is simulated according to Equation (B.6). So,

$$\mathbb{P}(R = \mathrm{j}) = \sum_{k=-\infty}^{\infty} p_k \cdot \mathbb{P}(Y = \mathrm{j} - k).$$

Now, the probability mass function of Z is

$$\mathbb{P}(Z=\mathrm{i}) = \begin{cases} \sum\limits_{\mathrm{j}=-\infty}^{0} \mathbb{P}(R=\mathrm{j}) & \text{if } \mathrm{i}=0, \\ \mathbb{P}(R=\mathrm{i}) & \text{if } 1 \leq \mathrm{i} \leq L-1, \\ \sum\limits_{\mathrm{j}=L}^{\infty} \mathbb{P}(R=\mathrm{j}) & \text{if } \mathrm{i}=L, \\ 0 & \text{otherwise.} \end{cases}$$

Now,

$$\begin{aligned}
\mathbb{E}(Z) &= \sum_{q=0}^{L} \mathrm{i} \cdot \mathbb{P}(Z=q) \\
&= \sum_{q=1}^{L-1} q \cdot \mathbb{P}(Z=q) + L.\mathbb{P}(Z=L) \\
&= \sum_{q=1}^{L-1} q \cdot \left(\sum_{k=-\infty}^{\infty} p_k \cdot \mathbb{P}(Y=q-k) \right) + L \cdot \left(\sum_{q=L}^{\infty} \sum_{k=-\infty}^{\infty} p_k \cdot \mathbb{P}(Y=q-k) \right) \\
&= \sum_{k=-\infty}^{\infty} p_k \cdot \left(\sum_{q=1}^{L-1} q \cdot \mathbb{P}(Y=q-k) + \sum_{q=L}^{\infty} L \cdot \mathbb{P}(Y=q-k) \right) \\
&= \sum_{q=1}^{L-1} q \cdot \mathbb{P}(Y=q) + L \cdot \sum_{q=L}^{\infty} \mathbb{P}(Y=q) + \sum_{k=-\infty}^{\infty} p_k \cdot \Big(\sum_{q=1}^{L-1} q\{\mathbb{P}(Y=q-k) - \mathbb{P}(Y=q)\} \\
&+ L \cdot \sum_{q=L}^{\infty} \{\mathbb{P}(Y=q-k) - \mathbb{P}(Y=q)\} \Big)
\end{aligned} \tag{B.7}$$

In Equation (B.7), $\mathbb{E}(Y) = \sum\limits_{q=1}^{L-1} q \cdot \mathbb{P}(Y=q) + L \cdot \sum\limits_{q=L}^{\infty} \mathbb{P}(Y=q) = \mathbb{E}(Y)$, when the read noise $\sigma_{\text{read}} = 0$. The expression corresponding to this was derived in Thm. 3.3.2 as $\beta(\Psi_{L-1}(\beta)) + L(1-\Psi_L(\beta))$. By re-arranging the rest of the terms and utilizing the fact that $\Psi_q(\beta) = \sum\limits_{k=0}^{q-1} \frac{\beta^k \mathrm{e}^{-\beta}}{k!}$ and $\mathbb{P}(Y=\mathrm{j}) = \frac{\mathrm{e}^{-\beta}\beta^{\mathrm{j}}}{\mathrm{j}!}$, we can obtain the expression for $\mu_Z = \mathbb{E}(Z)$. We can clearly see that all the terms in Equation (B.7) is differentiable. Thus, taking the derivative of Equation (B.7) w.r.t. β gives us the expression for $\frac{d\mu_Z}{d\beta}$

The expression for σ_Z^2 can also be calculated by following similar steps as above. □

VITA

Abhiram Gnanasambandam

EDUCATION

Purdue University *August 2017 - May 2022*

PhD, Electrical and Computer Engineering (3.95 / 4.0)

Recipient of "Outstanding Graduate Student Research" Award.

Research Interests : Computational Imaging, Machine Learning

Advisor : Stanley H. Chan

Indian Institute of Technology, Madras *August 2012 - May 2017*

Bachelor of Technology, Master of Technology, Electrical Engineering

RESEARCH EXPERIENCE

Low Light Computer Vision, *Purdue University* *August 2017 - May 2022*

Intelligent Imaging Lab, Prof. Stanley Chan

- Developed an unrolled optimization solution for linear inverse problems in low light.
- Came up with a sample distribution to use while training a neural network when the noise level is uncertain. (*ICML, 2020*).
- Developed a novel student-teacher training scheme for image classification in the dark. (*ECCV, 2020*).
- Developed an algorithm for HDR image reconstruction using Quanta Image Sensors. (*TCI, 2020*)
- Developed an algorithm for color image reconstruction at low light using Quanta Image Sensors. (*Optics Express, 2019*)

Physical Adversarial Attacks, *Purdue University* *May 2020 - Apr 2021*

Intelligent Imaging Lab, Prof. Stanley Chan

- Proposed a new optical adversarial attack framework to attack physical 3D objects using a projector.
- Conducted a theoretical study to understand the feasibility of the attacks on different kinds of surfaces and objects.

Gaussian Interference Channels, *IIT Madras* *May 2016 - May 2017*

Prof. Srikrishna Bhashyam

- Found a general achievable region for the discrete memoryless many-to-one and one-to-many interference channels using superposition coding. (*Asilomar 2017*)
- Derived achievable regions for Gaussian channels using different strategies that do not involve time-sharing or power splitting. Found new regions where these strategies achieve sum capacity.

INDUSTRY EXPERIENCE

Gigajot Technology Inc. *Summer 2019*

Engineering Intern

- Implemented and improved the color reconstruction on the Quanta Image Sensors.
- Improved the speed of the reconstruction algorithm for real time video processing.
- Implemented the HDR reconstruction algorithm for Quanta Image Sensors.
- Characterized the dynamic range of the sensor.

Gigajot Technology Inc. *Summer 2018*

Engineering Intern

- Worked on the image processing for the Quanta Image Sensors being developed at Gigajot.
- Processed the data from the sensor to help debug the hardware.
- Worked on integrating the denoising algorithm to the hardware by implementing on the FPGA board.

PUBLICATIONS

Abhiram Gnanasambandam, and Stanley H. Chan, "Exposure-Referred Signal-to-Noise Ratio for Digital Image Sensors" IEEE Transactions on Computational Imaging (Under Review). (Manuscript)

Xue Zhang, Gene Cheung, Jiahao Pang, Yash Sanghvi, **Abhiram Gnanasambandam**, Stanley H Chan, "Graph-Based Depth Denoising & Dequantization for Point Cloud Enhancement" IEEE Transactions on Image Processing (Under Review). (Manuscript)

Yash Sanghvi, **Abhiram Gnanasambandam**, and Stanley H. Chan, "Photon Limited Non-Blind Deblurring Using Algorithm Unrolling" IEEE Transactions on Computational Imaging (Under Review). (Manuscript)

Omar Elgendy, **Abhiram Gnanasambandam**, Stanley H. Chan, and Jiaju Ma, "Low-Light Demosaicking and Denoising for Small Pixels Using Learned Frequency Selection,", IEEE Transactions on Computational Imaging, 2021. (Manuscript)

Abhiram Gnanasambandam, and Stanley H. Chan, "Optical Adversarial Attack," International Conference on Computer Vision (ICCV) Workshop, 2021. (Manuscript)

Chengxi Li, Xiangyu Qu, **Abhiram Gnanasambandam**, Omar Elgendy, Jiaju Ma, and Stanley H. Chan, "Photon-limited object detection using non-local feature matching and knowledge distillation," International Conference on Computer Vision (ICCV) Workshop, 2021. (Manuscript)

Abhiram Gnanasambandam , and Stanley H. Chan, "HDR Imaging With Quanta Image Sensors: Theoretical Limits and Optimal Reconstruction," IEEE Transactions on Computational Imaging, 2020. (Manuscript)

Abhiram Gnanasambandam, and Stanley H. Chan, "One Size Fits All: Can We Train One Denoiser for All Noise Levels?," International Conference on Machine Learning (ICML), 2020. (Manuscript)

Abhiram Gnanasambandam, and Stanley H. Chan, "Image Classification in the dark using Quanta Image Sensors," European Conference on Computer Vision (ECCV), 2020. (Manuscript)

Yiheng Chi, **Abhiram Gnanasambandam**, Vladlen Koltun, and Stanley H. Chan, "Dynamic Low-light Imaging with Quanta Image Sensors," European Conference on Computer Vision (ECCV), 2020. (Manuscript)

Abhiram Gnanasambandam, Omar Elgendy, Jiaju Ma and Stanley H. Chan, "Megapixel photon-counting color imaging using Quanta Image Sensor," OSA Optics Express, June 2019. (Manuscript)

Jiaju Ma, Yu-Wing Chung, **Abhiram Gnanasambandam**, Stanley H. Chan, Saleh Masoodian "Photon-Counting Imaging with Multi-Bit Quanta Image Sensor," International Image Sensor Workshop (IISW), June 2019. (Manuscript)

Abhiram Gnanasambandam, Ragini Chaluvadi, Srikrishna Bhashyam, "On the sum capacity of many-to-one and one-to-many Gaussian interference channels," Asilomar Conference on Signals, Systems, and Computers, 2017. (Manuscript)

PROFESSIONAL EXPERIENCE

Teaching Assistant

- *Advanced C Programming (ECE 264)* - Purdue University, Fall '17
- *Signals and Systems (ECE 301)* - Purdue University, Spring '18, Fall '18, Spring '19

Reviewer for *IEEE Transactions on Computational Imaging* (TCI), *IEEE Transactions on Pattern Analysis and Machine Intelligence* (TPAMI), *ICASSP* (2020, 2021, 2022), *CVPR* (2022) and *ECCV* (2022).

TECHNICAL PROFICIENCY

Programming Languages C, C++, Matlab, Python.

Libraries Proficient in OpenCV, NumPy, PyTorch.

COURSES

Computer Vision

Model Based Image Processing

Real Analysis

Random Variables

Statistical Machine Learning

Numerical Linear Algebra

Digital Signal Processing

Neural Networks