LOW POWER APPROACHES FOR IMAGE ACQUISITION SYSTEMS

A Dissertation

by

PRAVIR SINGH GUPTA

Submitted to the Office of Graduate and Professional Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

Chair of Committee,	Gwan Seong Choi
Committee Members,	Krishna Narayanan
	Eun Jung Kim
	Jiang Hu
Head of Department,	Miroslav M. Begovic

August 2020

Major Subject: Electrical Engineering

ABSTRACT

This work presents novel image acquisition methodology to improve power and performance metrics of image acquisition system. Given the slowing Moore's law, ubiquitous mobile devices like smartphones and focus on multimedia content in today's world, it is the need of hour to adopt an algorithmic approach to achieve system efficiency in imaging systems. Towards this end, this work employs Compressed Sensing and Deep Learning techniques and tries to find a balance between performance and practicality of implementation. It makes necessary modifications of the algorithms to reduce the entire system redesign efforts which happen to be both expensive and time consuming process. By following the methodology and trade-offs suggested in this work, one can improve power and performance metrics by 50% while maintaining good quality of final images.

DEDICATION

To my family, academic advisors and friends.

ACKNOWLEDGMENTS

I would like to thank the Texas A&M University for providing me an opportunity to perform research as a part of their their PhD Program. I would also like to thank ECEN Department and Faculty for guiding me and providing me support at every moment. I would also like to thank my committee members for their insightful inputs and guidance. Finally I would like to thank my advisor Dr. Gwan Seong Choi for constant guidance and motivation throughout my PhD program.

CONTRIBUTORS AND FUNDING SOURCES

Contributors

This work was partially supported by TAMU Supercomputing Center.

NOMENCLATURE

ASIC	Application Specific Integrated Circuits
BW	Bit-Width
CS	Compressed Sensing
DL	Deep Learning
HDR	High Dynamic Range
ISET	Image Systems Evaluation Toolbox
ISP	Image Sensor Processor
JPEG	Joint Photographic Experts Group
LDR	Low Dynamic Range
NoC	Network on Chip
PPA	Power, Performance, Area
SNR	Signal to Noise Ratio
SoC	System on Chip
SR	Super-Resolution
SSIM	Structural Similarity Index Metric

TABLE OF CONTENTS

Page

LIST OF FIGURES

LIST OF TABLES

TABLE Page

1. INTRODUCTION

In the present multimedia world, with the advent of Internet and mobile devices, the amount of multimedia content generated by users is increasing at a tremendous rate. In addition, with the improvement in VLSI (Very Large Scale Integration) technology, resolution of image sensors are also increasing. Smartphones with image sensor resolution greater than 20 Megapixel are commonly used and image sensor vendors are also offering sensors with more than 40 Megapixel range. Cisco Visual Networking Index Forecast predicts an increase in mobile data traffic to 49 Exabytes per month by 2021 [6]. The video traffic would be 82 percent of all consumer traffic by 2021 [6]. Apart from consumer based consumption of video, many tasks are getting automated using DL (Deep Learning) based computer vision techniques like autonomous navigation, surveillance etc. These tasks require and generate huge amounts of images and videos adding to the pressure on storage and transmission networks (on-chip, off-chip, wireless etc.). It also presents a challenge in terms of power consumption and performance/latency requirements in image acquisition devices such as mobile devices. The increased amount of data processing requires newer VLSI technology nodes because they are faster and more power efficient.

However, lately Moore's law has started to saturate but the demand for high quality multimedia content is only increasing. To add to slowing down of device scaling problem, the newer technology nodes are becoming almost exponentially expensive [7]. Hence relying on device scaling is way too expensive and may be even impossible. Thus algorithmic and system design innovations need to be employed to meet the requirements. It might lead to more specific ASIC designs as opposed to relying on increasing clock speeds of processors to run specific algorithms [8].

To address these issues, we propose a novel image acquisition scheme which has following features -

- It performs compression on entire imaging pipeline, i.e. raw as well as finished data coming out of image sensor and ISP respectively, to achieve power and performance improvement.

- Use of simplistic and hardware friendly/implementable techniques to achieve compression.
- Novel applications of algorithm to achieve good quality reconstruction.

Thus the core idea is to compress image along entire path from source to destination and reconstruct image only when it is being viewed or operated upon. Where as, traditionally the focus of compression is on finished data only leading to no compression of raw data coming out from image sensor. This results in a huge amount of raw data moving from image sensor to ISP (Image Sensor Processor) over a communication bus, where one gets a finished image incurring both latency and power consumption. In modern day SoCs communication using NoCs can consume between 20-30% of total power [9]. Hence it justifies to reduce the dataflow from source to destination resulting in reduced downstream processing effort therby improve both power and performance metrics. However, using complicated compression algorithms at image sensor is not feasible because it is space constrained and there is little space for additional circuitry. Thus one needs to adopt more pragmatic and simplistic approach for raw data compression. Another thing to keep in mind is that if compression at the source results in data that does not have image-like distribution then popular image compression algorithms on finished images like JPEG wont work. Thus it leaves one with limited option and makes the task hard. Hence we rely on simplistic downsampling methods as they can be easily implemented.

While simplistic compression methods can help one achieve significant power savings, there is definitely a power burden because of image reconstruction algorithm. However, the image needs to be reconstructed only when it is being viewed or processed by a computer program. These days mobile devices like smartphones use cloud storage to store images and other important data. Thus compression will also help in achieving reduced network bandwidth as well as storage and plenty of power is available in cloud storage to recover back original information. Additionally, for a system like drones transmitting images to a base station, power consumption at base station is not an issue. Thus in this work we do not focus on power spent on image reconstruction.

Another important thing to keep in mind is that, while the choice of image reconstruction algorithm can be independent of compression technique, it is the performance of reconstruction tech-

nique that determines how much compression can be achieved while maintain good reconstruction quality. Towards this end, we have studied two main algorithms/techniques - CS (Compressed Sensing) and DL (Deep Learning) for reconstruction. These methods/algorithms perform the task of both artifact removal and superresolution. While there are many existing works in literature which have studied these algorithms, we have used these algorithm for images compressed using our proposed techniques which are system friendly. On the contrary, most of the work in literature has been studied from the perspective of compression techniques which have significance from algorithmic point of view but impractical from system's perspective. Algorithmic focused studies are important first step to establish the efficacy of algorithm. However one generally need to make some simplifying assumptions to make it suitable for an actual system. Such simplifications from compression point of view have been discussed in our work which make it suitable for images compressed in an actual system.

Additionally from the knowledge gained using the reconstruction algorithm from superresolution perspective, we extend our superresolution to HDR (High Dynamic Range) imaging potentially improving power metrics and frame rate of the cameras.

In the next section we will describe our system pipeline in more detail.

1.1 Novel Image Acquisition Scheme

The previous section discussed the novel features of proposed imaging pipeline and the motivation behind it. In this section, we will discuss the same pipeline in more detail to achieve our objective. We refer to image acquisition portion of this methodology as HCAS (Hardware based Compressed Acquisition Scheme). A simple schematic of proposed imaging pipeline is shown in Fig. 1.1.

Entire imaging pipeline consists of 6 stages of which HCAS constitutes first 3 stages. In the stage 1-3, image gets compressed using downsampling by averaging, bit truncation and JPEG. Stage 1-2 can be performed on the image sensor itself, since the ADC is located inside the image sensor chip. This work uses simplistic compression schemes in stage 1-2 because as will see in subsequent sections, they are very easy to implement at hardware level. Any sophisticated

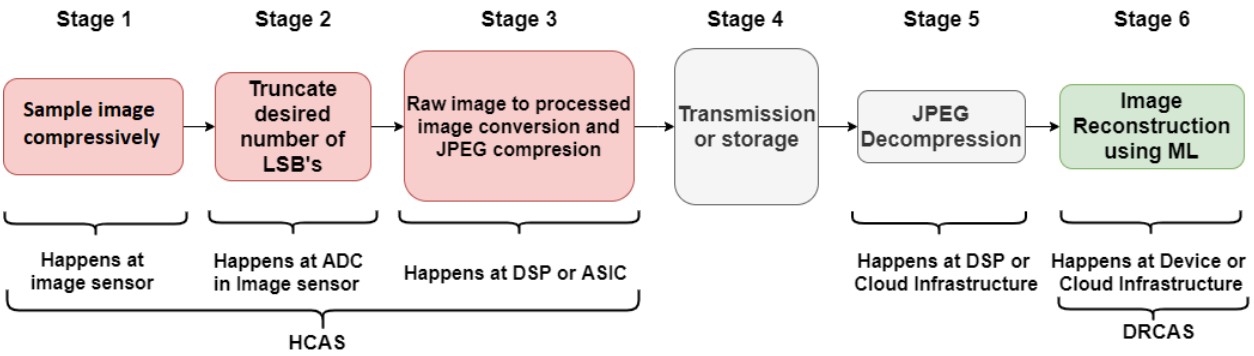


Figure 1.1: Proposed Image Acquisition Methodology: It consists of 6 stages. In the stages 1-3 (HCAS), an image gets compressed using downsampling, bit truncation and JPEG. Stage 4 represents transmission of image which can be a wireless medium or an on-chip bus or even storage. Stage 5 performs JPEG decompression and stage 6 consists of the proposed DRCAS to restore the desired image.

compression algorithm will only run in ISP (Image Sensor Processor) or DSP (Digital Signal Processor) unit. Thus there will be no compression of raw data between image sensor and ISP. Also any sensor based compression should not alter the distribution of image if one wants to utilize the JPEG compression. Thus it justifies use of simplistic and hardware friendly compression schemes. Stage 3 happens on a dedicated JPEG chip or an ISP or DSP. Stage 4 represents transmission of image which can be a wireless medium or an on-chip bus or even storage. Stage 5 can happen on a ISP/DSP processor in the device itself or it can be clubbed together with Stage 6 and can happen in the cloud or on the specialized processor on the image acquisition device itself. The idea of this work is to save energy during acquisition i.e. , from stage 1 to stage 5 as these processes often consume a significant amount of power in edge devices and stage 6 is not required unless a user is viewing image (e.g. smart phones, surveillance cameras etc.) or a computer program is operating on images e.g. object recognition. For stage 6, we use compressed reconstruction and deep learning based techniques and we make appropriate modifications to them to suit our image compression techniques. While the proposed schemes lead to degradation of image quality, but so does JPEG which is widely used for image compression in commercial systems. Thus our overall goal should be to meet the required standard of final output image using a mix of all compression schemes from stage 1 - 3 in a way such that power and performance benefits are maximized. Towards this

end we perform several experiments as follows -

- We propose a novel pixel design to perform CS at pixel level.
- We propose the use of pixel binning technique to achieve programmable implementation of CS at pixel level.
- We perform system simulation of entire proposed imaging pipeline.
- We propose a fixed point implementation of CS reconstruction algorithm.
- We propose use of Deep Learning based reconstruction algorithm for our proposed system
- We extend our Deep Learning Network for HDR Imaging.

In the next few chapters we will first cover some background concepts and then discuss each of the above experiments in detail in separate chapters. Specifically speaking, in Chapter 2 we will discuss key background concepts needed to understand our proposed methodology. In Chapter 3 we will discuss the novel pixel design for performing CS at pixel level. Going further, in Chapter 4 we discuss techniques to make CS programmable. In Chapter 5 and 6 we discuss full system simulation and fixed point implementation of CS respectively. Subsequently in Chapter 7 and 8 we perform experiments using Deep Learning for our proposed system pipeline and HDR imaging respectively. Lastly in Chapter 9, we conclude our work and briefly discuss few potential research directions.

2. BACKGROUND CONCEPTS [1]

The previous chapter introduced the motivation behind the proposed image acquisition scheme. To better understand the same, this chapter focuses on introduction of some key background concepts related to our proposed imaging pipeline. It has 4 sections -

- PPA: PPA stands for 'Power Performance Area' and as name suggests, this chapter focuses on power, performance and area of digital systems.
- Imaging System and Techniques: This chapter describes some key components of imaging system which are most important to our work. It also discusses image compression techniques and HDR Imaging.
- Compressed Sensing and Reconstruction: This section describes some theory behind compressed sensing and reconstruction.
- Deep Learning: This chapter focus on the deep learning techniques used in our work.

2.1 PPA: Power Performance Area

Since our goal is to improve power and performance metrics of an imaging system, it is necessary to understand the factors affecting them as well as interaction between them. The interaction between them are not straightforward making the job of meeting these constrains hard. In general for any digital circuit/chip, power, performance and area are the most important design constraint. These are generally required to be fixed at earliest phase of design cycle of a digital circuit as

[1] Parts of this chapter are reprinted with permission from "Image acquisition system using on sensor compressed sampling technique," by P.S. Gupta and G. S. Choi, 2018, Journal of Electronic Imaging, vol. 27, no. 1, p. 013019, © 2018, International Society for Optics and Photonics; "Programmable compressed sensing using simple deterministic sensing matrices," by P. S. Gupta and G. S. Choi, 2018, Optoelectronic Imaging and Multimedia Technology V, vol. 10817,p. 108170C, © 2018, International Society for Optics and Photonics; "Accurate simulation of on-sensor compressed sensing using iset," by P. S. Gupta and G. S. Choi, Image Sensing Technologies: Materials, Devices, Systems, and Applications VI, vol. 10980, p. 1098012, © 2019, International Society for Optics and Photonics; "Fixed point simulation of compressed sensing and reconstruction," by P. S. Gupta and G. S. Choi, Computational Imaging IV, vol. 10990, p. 109900I, © 2019, International Society for Optics and Photonics.

these are guided by the specifications of system or requirements of the customer. For example, a smartphone manufacturer will fix the power budget of camera i.e. the maximum power it can consume, performance such as frames per second or speed of the image sensor processor for a given resolution of image etc. in the initial phase of design which needs to be met by the camera/imaging system designers. These constraints are not independent and changing one of them affects others. The next few subsections will discuss how these quantities behave and one will be able to understand the interaction between them.

2.1.1 Power

Power consumption is one of the major concerns in digital circuits especially in mobile devices. The are two main independent components of power consumption/dissipation in a digital circuit - *Static power and Dynamic power*. Hence total power consumption can be written as

$$P_{total} = P_{static} + P_{dynamic}. \tag{2.1}$$

Static power is the power consumed when there is no activity in a digital circuit i.e. all inputs are static and none of the signals change. In this work we are not much concerned about static power consumption. This is because we are focusing on image acquisition which means the system is doing work of acquiring images. Whereas, *Dynamic power* is the power consumed due to switching signals in a digital circuit. Dynamic power can be further broken down as

$$P_{dynamic} = P_{switching} + P_{short\ circuit}, \tag{2.2}$$

where $P_{switching}$ refers to the power required to charge or discharge the nodes in digital circuit. These nodes are capacitances (mostly parasitic) present in the design. $P_{short\ circuit}$ refers to transient power consumption. The transient power consumption happens when there is a conductive path between the power supply and ground. This usually happens for a very short duration when gates switch from one state to another. $P_{switching}$ can be calculated as follows -

$$P_{switching} = \alpha C V^2 F, \tag{2.3}$$

where α is switching activity factor i.e. , number of times a signal switches from 0 to 1 per cycle, F denotes the frequency, V represents the voltage and C denotes switched capacitance at the node. $P_{switching}$ is one of the major concerns when comes to power consumption in a digital circuit. One can see from Eq. (2.3) that power consumption is linearly proportional to frequency F and quadratically proportional to voltage V. In general frequency of digital system is determined by the data processing requirements i.e. , how much data must be processed per second. Thus if one wants to process half the amount of data in the same time, one can halve the operating frequency. Voltage and Frequency in Eq. (2.3) are not independent quantities but follow a proportional relationship. If the operating frequency is increased, the operating voltage must be increased and vice-versa due to device physics and noise margin requirements. Therefore, if there is a reduction in the data to be processed one can decrease voltage and frequency to achieve quadratic reduction in energy consumption. The energy reduction is quadratic not cubic as energy is a product of power and time required for the task($1/F$). Since a task will finish faster in a high frequency system than a low frequency system, hence energy scales quadratically. When this reduction in voltage and frequency is performed on the fly depending upon the data processing requirement it is called DVFS (Dynamic Voltage Frequency Scaling). However DVFS modules are feasible for large SoC designs because they require significant area in the chip and their power consumption should be a small percentage of total power consumption.

2.1.2 Bitwidth

Bitwidth of a system has effect on both power and performance of system. These are generally determined by input signal bitwidth and output signal bitwidth. For example, in case of images the input and output bitwidth of images will determine the bitwidth of the entire system. Its effect on power and performance depends on the architecture of the system. For example, if we have to add two 8 bit numbers and have an 8 bit adder, we can do it in one cycle. However, if we want to

add two 16 bit numbers then, we require 2 cycles and almost double energy consumption. Thus bitwidth has significant effect on power consumption and minimizing redundant and insignificant bits is a high priority as it not only effects the processing elements but memory, storage and transmission as well. Similarly, floating point computation tend to consume more energy than fixed point computations. Fixed point computations are much faster too. Thus system designers must use only what is required and avoid use extra bits and complicated units like floating point units.

2.1.3 Performance

Performance refers to how fast the system operates i.e. its operating frequency. The target operating frequency of a system is determined by how much data must be processed per unit time and parallelization degree. Thus, more the data, higher should be the operating frequency for a given degree of parallelization. For example the frame rate of a camera for a given resolution and architecture will determine its operating frequency. The higher the frame rate, the higher will be the operating frequency. However, the frequency at which the system operates is determined by the critical path of the system. The critical path of a system is the slowest path (generally the longest path is slowest path) in the system. Bitwidht of a system also effects performance of system. If the datapath is serial then it will imply longer time to transfer data and if it is parallel it will imply a wider data bus.

Other important parameter related performance of a system are latency and throughput of the system. Latency of a system is the time needed for the input change to reflect the output change of the system. Throughput of a system refers to the rate at which data can be processed.

While there are many other factors impacting power and performance, a detailed analysis of those are beyond the scope of this work and only relevant issues are discussed here.

2.2 Imaging system and techniques

An imaging system is a fairly complex system composed of many parts. At broad level, it is composed of components such as - optical lenses, filters, sensors, processors and/or displays. Each component has a significant effect on the performance of system. From imaging system point of

view, we will only focus on image sensors and its components. Regarding ISP, the general PPA concepts discussed in Section 2.1 is applicable to them. Discussion of other components mentioned above is beyond the scope of this work. This section also focuses briefly on simulation of entire imaging system which we have used in our work. It also discusses JPEG compression technique since this is a key part of our proposed pipeline and ends with discussion on HDR imaging, which we use in our work as discussed in previous chapter.

2.2.1 Image Sensors

Image Sensor is one of the most important components of imaging system. An image sensor is a rectangular grid of pixels. Pixels are responsible for sensing the light and transforming it into electrical signal. Pixels are broadly composed of two parts - photo-detector element (mostly a photodiode) and sensing circuitry. The pixels are addressed row-wise using row address lines. Each row collectively outputs the signal in column line which is connected to Analog to Digital Converter (ADC) in the other end. ADC transforms the analog signal to a digitized output and sends it to output interface. A simple schematic for an image sensor is shown in Fig. 2.1. Thus it is at ADC where the signal becomes digital. ADC and output interface are responsible for major chunk of power consumption. Thus it is not the pixel circuitry that consumes power but the data transactions because of those pixels which consume most power. The next few subsections will briefly explain the components and features of an image sensor like - pixels, photodetectors, non-idealities in image sensor and binning.

2.2.1.1 Pixels

This section discuss some important aspects related to image pixels which are necessary to understand our proposed pixel design. Simply speaking an image pixel can be broadly divided into two parts, photo-detector element, and sensing circuit. Depending on sensing circuit there are two main families of image pixels, active pixel sensor, and passive pixel sensor. Passive pixel sensor carries out the charge of the photodetector and amplifies them later. Active pixel sensor has a photodetector and an active amplifier. Passive pixel sensors have mostly been implemented

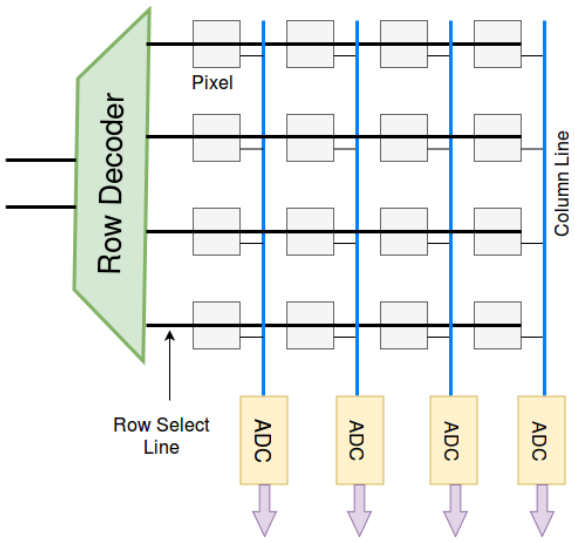


Figure 2.1: A simple schematic for an image sensor.

with Charge Coupled Device (CCD) technology while active pixel sensors are implemented using CMOS technology. Decreasing size and cost of CMOS elements has made CMOS image sensors viable and technology of choice (Ref. [10]). Ever decreasing size of transistors has made high-resolution image sensors possible. The most popular active pixel sensors design are 3T, 4T and CTIA (Capacitive Trans-Impedance Amplifier) pixels.

CTIA is mostly used in scientific applications while 3T and 4T are mostly used in commercial systems. We will not be discussing CTIA but the results presented can be applied in CTIA pixel as well. The schematic diagram for 3T and 4T pixel is shown in Fig. 2.2.

3T pixel is very compact but has less sensitivity and unstable bias voltage across photodiode. This pixel architecture consists of a photodiode and three transistors- Reset (M_R), Source Follower (M_SF) and a Row Select Transistor (M_RS). In 3T pixel operation, first the photodiode is reset using Reset transistor. Now, the charge gets collected on the photodiode proportional to light signal and exposure time. After a set "integration" time, the row select transistor is turned on to read out the signal using external readout circuitry.

The 4T (four transistor) pixel architecture is shown in Fig. 2.2 (Ref. [11]). Its architecture has

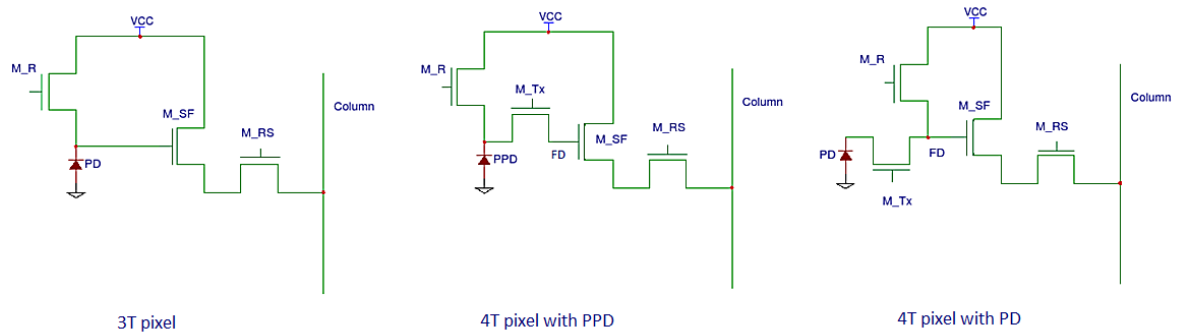


Figure 2.2: 3T and 4T Pixel Schematic diagram. M_R stands for reset transistor, M_Tx stands for transmission gate, M_SF stands for source follower, M_RS stands for row select transistor, PD stands for photodiode, PPD stands for pinned photodiode and FD stands for floating diffusion node.

two additional elements compared to the 3T architecture namely, the transfer gate (TX) and the floating diffusion node (FD). It uses either a Pinned Photo-Diode (PPD) or a normal Photo-Diode (PD) depending upon the design shown in Fig. 2.2. As long as TX is off, charge is accumulated in PPD or PD. When TX is on for set 'Integration' time period, charge is transferred to the diffusion node. We have used 4T pixel design with PD as our choice for implementation as we did not have pinned photodiode (PPD) model to perform the simulation. It is expected that result will be similar with PPD as explained earlier subsection.

Because charge collection area and readout area are separated in the 4T pixel via M_Tx transistor, it offers some key advantages. While the 3T design can only implement rolling shutter, the 4T design can implement both rolling as well as global shutter. Global shutter is very important for the high speed imaging application. The 4T pixel also allows low noise operation through the use of the Correlated Double Sampling (CDS) technique. The reset noise or kTC noise is the main source of noise resulting from the resetting operation of floating diffusion node through the resistive channel of the reset transistor. Thus, CDS technique can be employed to sample the floating diffusion node before and after M_Tx is turned on within a short time interval, thereby eliminating kTC noise. This operation is shown in Fig. 2.3.

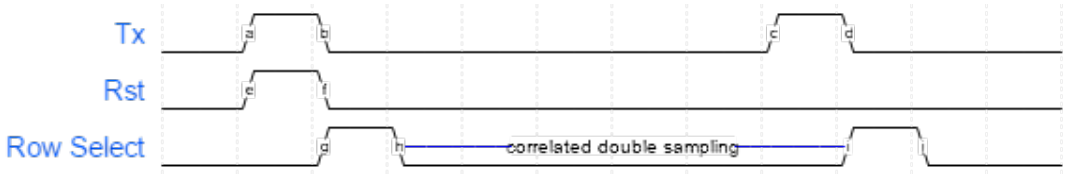


Figure 2.3: Correlated Double Sampling (CDS) for a single image.

Transfer transistor or M_Tx makes the bias voltage across photodiode very stable. It also helps us to increase sensitivity because the integration capacitor can be kept small. CTIA has around 8 transistors but has highest sensitivity among all of them and stable photodiode voltage. Because of large pixel size, it is not much used in commercial systems. It is mostly used in scientific applications.

2.2.1.2 Photodetectors

Photodetectors as the name suggests are the actual light sensors responsible for converting light to electrical signal. There are mainly 3 types of photo sensing elements - photogates, phototransistors and photodiodes. In this work, we have used photodiodes. There are different types of photodiodes too. We have used simple p-n junction, although we can use more sophisticated p-i-n junction to improve the efficiency of an image sensor. As the name implies, p-i-n junction consists of intrinsic region between p and n region. The p-i-n junction device reduces dark current and charge-transfer noise (Ref. [12]). Hence using p-n junction over p-i-n junction does not affect the demonstration of main functionality of our system design methodology.

There are various types of p-n junction photodiodes also. They are - n+/p-sub, n-well/p-sub, p+/n-well/p-sub. Murari el al. (Ref. [13]) list the parameters and advantages of various photodiodes. We are using n+/p-sub because of the large fill factor, low dark current per unit area values and ease of implementation to demonstrate our concept. Its schematic diagram is shown in Fig. 2.4.

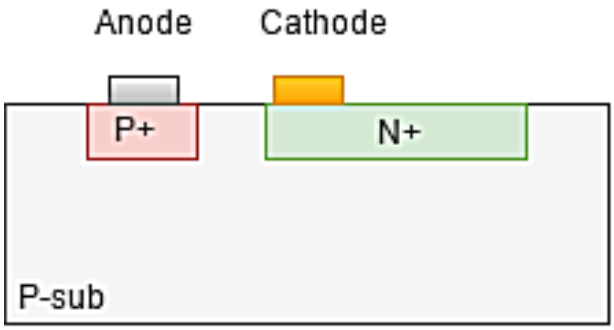


Figure 2.4: n+/p-sub photodiode (Ref. [1]).

2.2.1.3 Nonidealities in Image Sensors

Non-idealities can be broadly classified into two major groups - pixel level non-idealities and readout-level non-idealities (Ref. [14]). Both of them present challenges to the image sensor designers. Major pixel level non-idealities are - Dark Signal Non-Uniformity, Offset Fixed Pattern Noise, Photo-response non-uniformity, Pixel response non-linearity and pixel temporal noise. Major readout level non-linearities are - offset column fixed pattern noise, gain error column fixed pattern noise, readout non-linearity, readout temporal noise, readout output voltage range and quantization.

In this work, we will not deal with temporal noise but we will consider the fixed pattern noise and application of CS to overcome the challenges posed by fixed pattern noise. We are also not dealing with readout output voltage range and quantization noise as it is a research problem by itself and has been included in future course of our work. Offset fixed pattern noise can be easily dealt with by using correlated double sampling technique. But photoresponse non-uniformity, pixel response non-linearity and gain error fixed pattern noise require sophisticated circuitry to deal with. A simple way to deal with this problem is discussed in the Sec. 3.3 of the paper. An example image with column and pixel level fixed pattern noise is shown in Fig. 2.5.

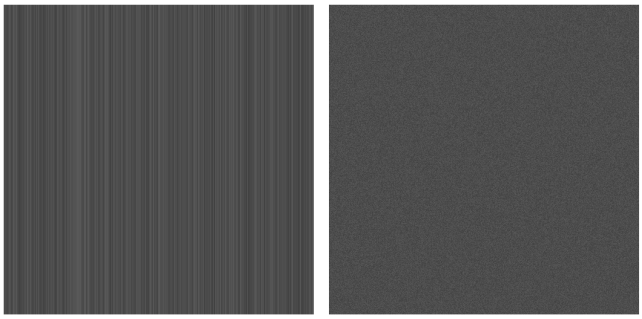


Figure 2.5: Column and Pixel fixed pattern noise example.

2.2.1.4 Binning

While previous sections were dedicated more to circuitry, this section is dedicated to an import feature of image sensors called binning, which is extensively used in our proposed pipeline. Though it is implemented at pixel level, we will mostly treat it as a feature or function since the implementation is not important. Binning means combination neighboring pixel signals together to form an output signal. This reduces resolution of image sensor by a factor equal to the number of pixels binned together. Pixels can be binned together either by adding or averaging them together. Addition of pixels has advantage during low light conditions as it increases the low light sensitivity but might result in saturation during common lighting conditions. Averaging of pixels prevent saturation at common lighting conditions but results in a poor low light sensitivity (Ref. [15]). Pixels can be combined in various fashion like 1×2 (column binning), 2×1 (row binning), 2×2 (full binning) etc. Many implementations of binning circuit are available in literature (Ref. [16, 17, 18, 19, 15]). Binning has been used for various applications like low light imaging (Ref. [16]), background noise suppression (Ref. [17]), power reduction (Ref. [18]), multi-resolution (Ref. [19, 15]) etc. A detailed discussion about these is beyond the scope of this paper.

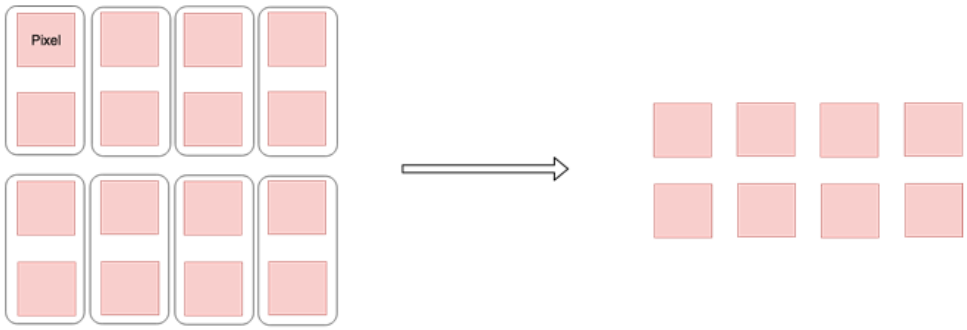


Figure 2.6: Figure showing how pixels are combined in different binning schemes. The enclosed pixels are binned together.

2.2.2 Imaging System Simulation

Designing an entire imaging system is a very massive engineering task and involves significant manpower and resources. Hence proposing a change or modification to an existing system can be expensive and time consuming to test using actual prototypes. However, one can create a simulation of entire imaging system to see the effect of proposed modification. Fortunately, simulators with reasonable accuracy are available. One such simulator is ISET and this simulator has been used in our work to study the proposed pipeline using CS. ISET (Ref. [2]) is very popular toolbox for simulating imaging systems. It takes into account the effects of each component of the imaging system to perform accurate simulations. It does this by defining a software object for each of these components such as - scene, optics, sensor, processor and display. A scene is defined as a radiometric description of the input data. The optical component includes the properties of lenses which are responsible for converting the scene into irradiance image at sensor surface. The sensor component includes the properties of pixels, sensor array, filters etc. The processor component consists of algorithms that define how sensor data gets digitized. The display component is a radiometric description of the final image for any calibrated display. Thus ISET allows to experiment new designs and algorithms without incurring the cost of building an actual system. A simple schematic diagram of simulation environment is shown in Fig. 2.7.

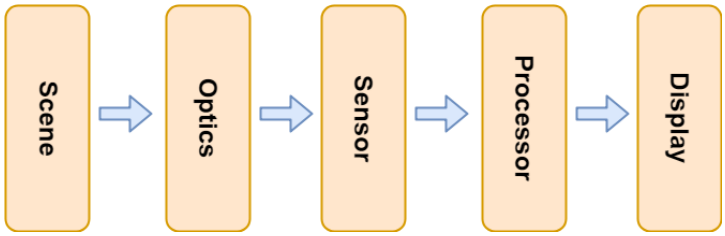


Figure 2.7: ISET Simulation environment (Ref. [2]).

2.2.3 JPEG Theory

JPEG stands for Joint Photographics Expert Group. It is a very widely used lossy image compression technique. However, it only works on finished images. Thus it does not help in compression of raw data. In our work we use JPEG along with simple compression on raw data to achieve compression throughout the imaging pipeline. JPEG is mostly used for lossy compression, however it can perform lossless compression too. Lossy compression relies on the fact that most of the image information is contained in very few coefficients in the Discrete Cosine Transform (DCT) domain. So a vast majority of insignificant coefficients can be discarded without much loss in perceptual quality resulting in large compression ratios.

JPEG first divides the image into 8×8 pixel blocks and then calculates DCT of each block. A quantizer rounds off the resulting DCT coefficients according to the quantization matrix, which controls the amount of compression one wants to do. This step represents "lossy" part of JPEG but allows for large compression ratios. We can also control the amount of compression by appropriately setting the quantization matrix. After quantization, data is compressed further by the use of variable length encoding of these coefficients.

While JPEG has been applied previously to CS sampled images (Ref. [20]) but compression performance has not been mentioned. Li et. al. (Ref. [20]) also use the Gaussian random matrix to compressively sample the image. When we sample an image with the Gaussian random matrix, the sampled image has Gaussian distribution and the image-like properties are lost. This results in a very poor JPEG compression performance which will significantly increase the effort/energy

required to store the image.

2.2.4 HDR Imaging: High Dynamic Range Imaging

Until now we were mostly looking at conventional imaging systems/techniques. These imaging systems have limited dynamic range while the natural world around us has a very high dynamic range. Most commonly these devices operate at 8-bits per color channel i.e. the pixel values are in the range 0-255 for each color channel. It can get really bright on a sunny day and pitch black inside a room with no lights in the night. Thus the devices do not have enough range to capture scene in all scenarios. To handle scenes of such different dynamic ranges, the camera or the operator tries to set exposure so that it can capture the interesting parts of scene. Sometimes automatic or manual exposure setting can handle the scenarios effectively. However many times within a single scene the dynamic range is much higher than that of capture device. In such scenarios exposure setting alone will not help and a different approach needs to be taken. This is where HDR imaging comes in. HDR is a computational imaging technique which uses multiple LDR (Low Dynamic Range) images of the same scene from same device. Each image in the set of LDR images is captured under different exposure conditions such that the shortest exposure time captures the brightest portions of the image and longest exposure times capture the darkest region of the image effectively. The HDR imaging algorithm uses this set of images to calculate the irradiance of the scene. The images with shorter exposure time gives a more accurate irradiance of the the bright part of the scene than the images with longer duration. Similarly the images with longer exposure time gives a more accurate irradiance of the dark parts of the scene. The more accurate representations of irradiance are given higher weight than less accurate ones to calculate the irradiance image of the scene. This represents the HDR image. This image is then further processed using tone mapping to get a finished image in 8 bit per channel format such that all parts of the scene are properly exposed to enhance the visibility. While the concept of HDR imaging is very sound, one of the problems associated with it is that it requires a set of LDR images which increases total exposure time and also generates huge amount of data. During this period, motion blur and handshaking blur should be avoided as it is detrimental to HDR performance. To address these issues, we employ the super-resolution

networks we developed for our imaging pipeline. Until now we mostly discussed concepts related to digital circuits and imaging systems. In the next sections we will discuss the reconstruction algorithms and techniques used in our work i.e. CS and DL.

2.3 Compressed Sensing and Reconstruction

In this chapter, we discuss CS theory and reconstruction algorithm. CS is one of the algorithm that we will be using for reconstruction, the other being DL. While traditional implementations of CS use Gaussian random sampling matrices, we use simplistic averaging sampling matrices. The reasoning behind this has been explained in Section 3.3.1. In this section we will only discuss Cs theory in basic form which will enable one to understand modifications in later chapters.

2.3.1 Compressed Sensing (CS) Theory

Suppose we have signal X, having N samples such that, $X \in R^{N\times 1}$. And we want to recover X from Y, where

$$Y = \Phi X, \tag{2.4}$$

such that Φ is a $M \times N$ matrix and $M << N$. Because number of unknowns is significantly larger than observations, it is difficult to recover X from Y because Eq. (2.4) has infinitely many possible solutions. But if X is sufficiently sparse, exact recovery is possible. This is compressed sensing (Ref. [21]). A popular choice for Φ i.e. measurement basis, is randomly generated matrix. In this work we also assume Φ is orthonormal i.e.

$$\Phi\Phi^T = I. \tag{2.5}$$

Lets say that the signal X is sparse in some domain Ψ. Then the signal can be represented in sparse domain as follows -

$$X = \Psi T, \tag{2.6}$$

where T represents the signal X in the transform domain Ψ^{-1}. Using Eq. (2.6) and Eq. (2.4) we get,

$$Y = \Phi\Psi T. \tag{2.7}$$

Lets assume following,

$$A = \Phi\Psi. \tag{2.8}$$

Then using Eq. (2.7) and Eq. (2.8) we get,

$$Y = AT. \tag{2.9}$$

Since A is $M \times N$ and $M << N$ recovery of the original signal is difficult because the system of equation represented by Eq. (2.9) has infinitely many solution. This is where CS comes to rescue. If the sensing matrix A satisfies the Restricted Isometric Property stated (RIP) (Ref. [22]) below

$$1 - \epsilon \leq \frac{||AT||_2}{||T||_2} \leq 1 + \epsilon, \tag{2.10}$$

for some $\epsilon > 0$ then perfect reconstruction is guaranteed with very high probability. To reconstruct signal we can solve the following equation using linear programming techniques.

$$\min_T ||T||_{l1} \; such \; that \; Y = AT. \tag{2.11}$$

Another condition related to RIP is that sparsity basis should be incoherent with the sampling basis (Ref. [23]). The coherence between the two can be calculated as follows -

$$\mu(\Phi, \Psi) = \sqrt{N} \times max_{1 \leq k,j \leq N} |\phi_k, \psi_j|, \tag{2.12}$$

where ϕ and ψ are the basis vectors in sampling basis Φ and sparsity basis Ψ respectively. The coherence ranges from 1 to $\sqrt{N}$. If μ is close to 1 then matrices are incoherent and vice versa. While the requirement of incoherence is implicit in Eq. (2.10), it is explicit in another sufficient

condition for recovery of compressively sampled signals. Select M measurements uniformly at random in Φ domain. Then if,

$$M > C\mu^2(\Phi, \Psi)S \log N, \tag{2.13}$$

for some positive constant C and S-sparse signal (i.e. only S coefficients of signal are non-zero), solution to Eq. (2.11) is guaranteed with very high probability (Ref. [24]). Eq. (2.13) also indicates that if incoherence is less we need more samples to reconstruct the original signal with high probability (Ref. [23]).

The above discussion was applicable to strictly sparse signal which means the signal has a lot of perfect zero values when represented in sparse domain. But such signals are rarely found in nature. Images represented in the matrix form are no exception. Many natural signals are only approximately sparse, which means most of the coefficients are very small in magnitude. In such case, small coefficients can be discarded without much loss of perceptual quality. Lets say the signal X is approximately sparse. Lets set all but S largest elements of our approximately sparse signal X as zero and denote the resulting signal by X_S. Lets denote the corresponding transform by T_S. Because Ψ is orthonormal basis,

$$||T - T_S||_2 = ||X - X_S||_2. \tag{2.14}$$

So if T can be classified as sparse or compressible, meaning sorted magnitudes of the T decay quickly, then X can be approximated by X_S and, therefore, the error $||X - X_S||_2$ is small (Ref. [24]). This means we can discard a significant fraction of coefficients without much loss of quality. This is why CS works well with natural images.

For images popular sparsity basis' are Wavelet, Fourier or Gradient. The measurement matrices which satisfy incoherence requirements broadly fall in 4 categories - random or Gaussian random matrices (Ref. [25]), scrambled Fourier matrices (Ref. [26]), Partial Noiselets (Ref. [23]) and scrambled block Hadamard matrices (Ref. [27, 28]). Unfortunately, these matrices have very

expensive and challenging hardware implementation. Any attempt to implement these matrices negates the advantage gained by CS in terms of sampling effort per bit. To make matter worse, storage of the sampled image becomes even more challenging.

For images, the sampling matrix can be quite huge i.e. of the order of 1 Million. Storing or generating a matrix of such size is not feasible in a camera or a portable device. To solve this problem Block-Based CS is used which is explained in next subsection.

2.3.2 Block-Based CS

In block-based CS sampling, the image is divided into $B \times B$ blocks. The sampling is done using $\frac{M}{N} \times B^2$ sampling matrix where compression ratio = $\frac{M}{N}$. Hence we need to store only $\frac{M}{N} B \times B$ numbers rather than the full ensemble which results in huge savings in circuitry and power (Ref. [29]). This is a very useful result which we will be utilizing in our work too.

$$\Phi = \begin{pmatrix} \phi_B & & & & & \\ & \phi_B & & & & \\ & & \phi_B & & & \\ & & & . & & \\ & & & & . & \\ & & & & & \phi_B \end{pmatrix} \tag{2.15}$$

where, off-diagonal elements are all zeros.

There is a trade-off involved between memory and reconstruction performance in the selection of block dimension. Small B means less memory but poor reconstruction performance while large B means more memory but superior reconstruction performance.

For block-based CS image has to be vectorized in one dimension either by using raster scan or by just reshaping the matrix. In our work, we have used an even simplified version of block CS. We have not vectorized the image in one dimension. Instead, we keep the image as such and use $(M/N \times B) \times B$ sampling matrix. This leads to even more simplified implementation. For our case, the block size does not have any affect on reconstruction performance in our simulation. An

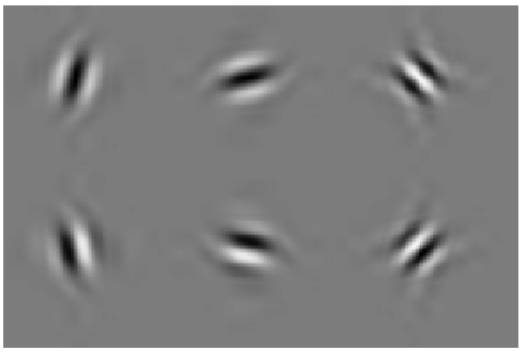


Figure 2.8: The six wavelets.

explanation about this has been provided in Sec. 3.3. Hence we choose smallest possible block size (i.e. 2×4) for simplicity.

The next subsection introduces the transform domain in which natural images are sparse, a key requirement for compressed sensing/reconstruction.

2.3.3 Directional Transforms for Sparse Representation

There are many transforms which can be used to represent an image as a sparse or approximately sparse signal. A popular one is Discrete Wavelet Transform (DWT). DWT lacks important properties such as shift invariance or directional selectivity. There are many modifications to DWT which have been extensively studied to preserve a much higher degree of directional representation than DWTs. One of them is DDWT (Dual Tree Discrete Wavelet Transform) (Ref. [30]). DDWT has an advantage over DWT as it provides efficient representation of directional features such as edges and contours. It has a redundancy of $2^m : 1$ for m-dimensional signals. Hence for 2-dimensional image, redundancy will be 4:1. It consists of both real and imaginary part but only real or imaginary part of DDWT guarantees perfect reconstruction and hence can be used as a standalone transform (Ref. [31]). While DWT is ambiguous in directionality property, mixing $+45$ and -45 together, DDWT has unique wavelet in each direction. They are oriented at $+/-75, +/-15, +/-45$. The wavelets are shown in Fig. 2.8.

The next subsection introduces the reconstruction algorithms for images sampled using CS technique.

2.3.4 Reconstruction Algorithm

A major problem associated with Block based CS is blocking-artifacts. A solution to this problem was presented by Gan et al. (Ref. [29]) by incorporating Weiner filtering into the basic PL (Projected Landweber) framework. This filtering helps to impose smoothness as well as sparsity inherent in PL algorithm. The algorithm (Ref. [32]) is given below

$function\ X^{i+1} = SPL(X^i, y, \phi_B, \Psi, \lambda)$

$Step\ 1:$ $\hat{X}^i = Wiener(X^i)$

$for\ each\ block\ j$

$Step\ 2:$ $\hat{\hat{X}}_j^i = \hat{X}_j^i + \phi_B^T(y - \phi_B \hat{X}_j^i)$

$Step\ 3:$ $\check{\check{T}}^i = \Psi^{-1}\hat{\hat{X}}$

$Step\ 4:$ $\check{T}^i = Threshold(\check{\check{T}}^i, \lambda)$

$Step\ 5:$ $\bar{X}^i = \Psi\check{T}^i$

$for\ each\ block\ j$

$Step\ 6:$ $X^{i+1} = \bar{X}_j^i + \phi_B^T(y - \phi_B \bar{X}_j^i).$

In the above algorithm Weiner() represents pixel-wise adaptive weiner filtering using a neighborhood of 3×3. The initial value is given below:

$$x^0 = \Phi^T y, \tag{2.16}$$

and the termination criteria is as follows -

$$|D^{(i+1)} - D^{(i)}| < 10^{-4}, \tag{2.17}$$

$$where,\ D^{(i)} = \frac{1}{\sqrt{N}}||x^i - \hat{\hat{x}}^{(i-1)}||_2. \tag{2.18}$$

2.4 Deep Learning

The previous sections were devoted to imaging system and CS. In this section we will briefly go through some Machine Learning concepts used in our work particularly - CNNs (Convolution Neural Networks) and Residual Neural Networks. This is a very brief introduction and a detailed explanation of these concepts is beyond the scope of this work. Interested readers can refer to [33].

2.4.1 Convolution Neural Networks (CNNs)

CNN (Convolution Neural Network) is a special type of Neural Network which is inspired by the human visual cortex system and is thus designed to operate on images/videos. As the name suggests CNN are composed of multiple blocks of convolution layers. Each convolution layers make use of a set of filters to apply convolution operation on input image (also known as input feature maps) to generate output image (also known as output feature maps). It is this filter in the convolution layer which is learnable. This output feature map is then passed through a non-linear function, generally ReLU activation function. This transformation of input feature map by a linear convolution operation and a non-linear activation function makes a single convolution layer. The presence non-linear activation function makers the convolution layer a non-linear function. Multiple such convolution layers are concatenated to make deep convolution neural networks. The combination of such multiple convolution based learnable non-linear functions makes CNN very powerful functional approximators for computer vision tasks.

The main building block of CNN is the convolutional layer. Convolution is a mathematical operation to merge two sets of information. In our case the convolution is applied on the input data using a convolution filter to produce a feature map. There are a lot of terms being used so let's visualize them one by one.

2.5 Deep residual networks

With the recent advances of ML, deep learning based algorithms have demonstrated superior performance than conventional methods for image super-resolution. Most of them are based on convolution neural networks (CNNs), which apply convolution operation to the input data followed

by an activation function to produce the output. To improve training of CNN, Residual Neural Network (ResNet) were first introduced by He et al. . in [3]. To understand ResNets, let us denote the underlying mapping between input (x) and output of network as $H(x)$. Then residual mapping can be defined as,

$$F(x) = H(x) - x. \tag{2.19}$$

Simply speaking, residual mapping is the difference between input and expected output of network. The original mapping can now be defined in terms of residual mapping as

$$H(x) = F(x) + x. \tag{2.20}$$

A simple graph of residual network is shown in Fig. 2.9. ResNets performs superior because it is easier to optimize the residual mapping than the original [3]. There is ample evidence in the literature indicating that network depth is of crucial importance and deeper networks in general achieve better results [34, 35]. With ResNets it becomes easier to train big networks.

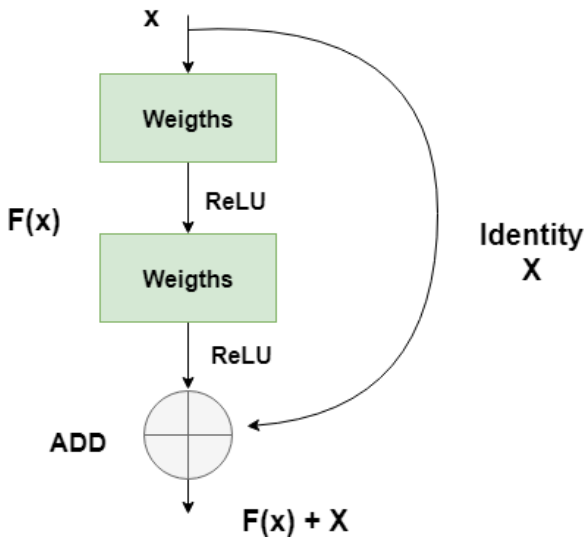


Figure 2.9: Residual Network [3]

With this we come to the end of this chapter related to background concepts. In the coming

chapters we will discuss about the experiments related to our novel imaging system.

3. CS BASED RECONSTRUCTION [1]

The previous chapter discussed about the background concepts related to our proposed imaging pipeline. This chapter focuses on implementation of such pipeline using a specialized pixel design and CS reconstruction algorithm. Towards this end, this chapter will also discuss the rational behind the proposed averaging based sampling from CS perspective.

3.1 Related Work

A lot of research has been conducted for compressed sampling of natural images, however the traditional methods of CS makes matter worse when comes to acquisition effort per bit and storage effort per bit. Single pixel camera proposed by Duarte et al. suffers from the same problem. The biggest issue with that approach is the sampled image looses image-like properties and hence image compression techniques like JPEG do not work well resulting in increase of storage effort per bit. Another issue with single pixel camera is that it is entirely different imaging system concept as it uses micro-mirror array to generate samples. This micro mirror array has a limit effect on resolution of image sensors. Oike et al. (Ref. [36]) applied CS at Analog-to-Digital conversion level in traditional imaging system, however the image distribution becomes Gaussian and JPEG cannot be applied resulting in increase of storage effort per bit. It also uses a pseudo-random generator which consumes additional energy. The design presented by Dadkhah et al. (Ref. [37]) does CS at the pixel level. But it wires the output of pseudo-random generator to each block. In addition to the problems associated with design presented by Oike et al. (Ref. [36]), it also consumes significant wiring area in the pixel and decreases the active area in the pixel. This will result in poor Peak-Signal-to-Noise-Ratio (PSNR) performance of pixel. Katic et al. (Ref. [14]) also present design on similar lines. Their design also contains random number generator which needs to be routed to pixels consuming wiring area and power.

[1] Parts of this chapter are reprinted with permission from "Image acquisition system using on sensor compressed sampling technique," by P.S. Gupta and G. S. Choi, 2018, Journal of Electronic Imaging, vol. 27, no. 1, p. 013019, © 2018, International Society for Optics and Photonics

3.2 Deterministic CS and Super-Resolution (SR)

Traditionally, the projection or sampling matrix Φ for CS is chosen as Gaussian Random matrix as it possess good RIP and is highly incoherent with most sparsifying basis. However, hardware implementation of Gaussian random matrix is infeasible. A deterministic construction of sampling matrix can result in considerable simplification of hardware implementation. A method for deterministic construction of matrices were first introduced in detail in Ref. [38]. The author used finite fields to construct cyclic matrices which satisfy RIP. This is popularly known as deterministic CS. Other methods for deterministic construction have also been proposed such as one in Ref. [39] where authors used Euler Square based binary CS matrices which outperformed their Gaussian counterparts.

Super-resolution (SR) implies construction of high-resolution images from one or more low resolution images. Traditionally SR had been done using a set of low-resolution images. The idea is to enforce the constraint of sparsity in the transform domain such as wavelet to reconstruct the image. But using CS for SR means, that sampling matrix is no longer random but deterministic. The sampling or projection matrix for SR is guided by imaging model. SR sampling matrix L can be viewed as product of two matrices as follows (see Ref. [40]) -

$$L \;=\; R \times L_p, \tag{3.1}$$

where R is decimation operator or downsampler and L_p is low pass filter. Since there is a low pass filter involved in construction of L, it will have frequency discriminative nature. It will filter out high frequency components but preserve low frequency components. Where as, a Gaussian random matrix will preserve all frequencies. This means L exhibits good RIP characteristics for a class of signals that contain low frequency information only, but Gaussian random matrix has good characteristics for any class of signals (see Ref. [40]). However, in case of natural images, most of the energy in concentrated in low frequency signals only. Hence if cutoff frequency for L_p is appropriately set, the loss might not be too much resulting in reasonable reconstruction. Lossy

image compression algorithm too weed out or reduce the high frequency component during the process of compression. Sen et al. performed SR CS reconstruction (Ref. [41]) using filtered and point down sampled image. In our work, we present a novel image sensor design (see Sec. 3.4) for filtering and downsampling the image in the CMOS image sensor itself without additional hardware and resulting in significant power savings. An advantage is that because we are using filtering and downsampling, we do not need randomization of sampling matrix. This also results in significant savings in terms of hardware and power consumption as there is no need of random generator and associated wiring.

3.3 Simulation using CS

3.3.1 Motivation behind sensing matrices

In this section we first discuss the construction of matrices. Subsequently we will discuss the rationale behind this matrices and compare the performance of this matrix with Gaussian random sampling matrices.

We use the binary block diagonal (Φ_B) and non-binary block diagonal (Φ_{NB}) sampling matrix as mentioned below -

$$\Phi_B = \begin{pmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{pmatrix} \tag{3.2}$$

$$\Phi_{NB} = \begin{pmatrix} 9 & 7 & 0 & 0 \\ 0 & 0 & 9 & 7 \end{pmatrix} \tag{3.3}$$

Because of the way our sampling matrix is constructed, block size will not have any affect on reconstruction performance. We can see this from two matrices of different block sizes presented below.

$$\Phi_{B,2\times4} = \begin{pmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{pmatrix} \tag{3.4}$$

$$\Phi_{B,4\times8} = \begin{pmatrix} 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \end{pmatrix} \tag{3.5}$$

We can see that both the above block matrix, when used as sampling matrix actually perform the same function of adding two rows. 4×8 matrix is actually two 2×4 matrix along the main diagonal of the sampling matrix presented in Eq. 2.15. Thus 4×8 can be expressed in terms of 2×4 matrix as follows -

$$\Phi_{B,4\times8} = \begin{pmatrix} \Phi_{B,2\times4} & \\ & \Phi_{B,2\times4} \end{pmatrix}. \tag{3.6}$$

Thus both of them lead to same sampling matrix presented in Eq. 2.15. Hence there will not be any affect in performance. The same reasoning applies to non-binary sampling matrix also.

Since matrix Φ_B adds two neighboring pixels, it does not significantly alter the statistical distribution of image and hence preserves image-like properties. But matrix Φ_{NB} performs weighted addition, so it does alter the distribution but still preserves some image-like properties. It does not alter the image as significantly as random Gaussian sampling matrix which makes the distribution of resulting sampled image as Gaussian. We arrived at Φ_{NB} empirically and it was found to be most optimal. One pixel is weighed approximately 1.3 times relative to the other in Φ_{NB}. One can try higher relative weights also but it will be difficult to implement in hardware due to large capacitor requirement (as per our design presented later in the section). Since we have fixed bitwidth ADC's, we can only use integer weights to sample image otherwise we will loose the information contained in the decimal part. The image sampled using Φ_{NB} requires 12 bits to store each pixel of resulting image. For Φ_B 9 bits are required for the same.

According to Eq. (3.1), Eq. (3.4) can be viewed as product of downsampler (R) and a circulant

averaging filter (L_p). These matrices are as follows -

$$L_p = \begin{pmatrix} 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 \end{pmatrix} \tag{3.7}$$

$$R = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} \tag{3.8}$$

Similar lines of reasoning hold for Eq. (3.3). However these sampling matrix in Eq. (3.2) and Eq. (3.3) are sparse leading to less incoherency with sparse transforms such as wavelet transform which is used as sparse basis (Ref. [28]). But it still works well for CS because according to Eq. (2.13) and Ref. [28] if incoherence is less, we need more samples to reconstruct signal with high probability. A comparison between Gaussian Random sampling matrix and simple binary averaging matrix is provided in Table 3.1 for different compression ratios. A compression ratio (C.R.) of 4 for binary sampling matrix means we add 4 neighboring pixels together to form a measurement and there is no overlap between different measurements as shown below -

$$\Phi_{B,CR\ 4} = \begin{pmatrix} 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{pmatrix} \tag{3.9}$$

We can see in Table 3.1 that as we go for higher C.R. (Compression Ratio) the reconstruction performance of Gaussian Random sampling matrix becomes better than averaging sampling matrices. However for C.R. = 2, the performance of our proposed sampling matrix is better than Gaussian. This makes it a good choice as JPEG can be applied on sampled image and hardware implementation is simplified.

Our sampling matrix in Eq. (3.2) and Eq. (3.3) do not satisfy the orthogonality requirement stated by Eq. (2.5). We call our sampling matrix in Eq. (3.2) and Eq. (3.3) as front-end sampling

Table 3.1: Comparison between Gaussian Sampling and Averaging. C.R. refers to Compression Ratio.

$C.R.$	$Gaussian\ Random\ (dB)$	$Averaging\ (dB)$
2	30.85	32.45
4	27.14	27.20
8	24.84	24.32
16	22.86	22.18

matrix. We have to perform a transformation on front-end sampling matrix so that it satisfies Eq. (2.5). The matrix resulting from transformation is known as back-end sampling matrix. We use front-end sampling matrix because it is very easy to implement on sensor level. The transformation from front-end to back-end is very simple. We have to just multiply the front-end sampling matrix by a normalization constant. The normalization constant is simply the square root of the sum of squares of all the elements in a row of the matrix.

$$\Phi_B = \frac{1}{\sqrt{N}} \times \Phi_F \tag{3.10}$$

where, N=$Sum\ of\ squares\ of\ row\ elements\ of\ matrix.$

The back end sampling matrix generated from Eq. (3.10) will satisfy Eq. (2.5) and this transformation can be implemented in the reconstruction algorithm itself. Multiplication of this transformation constant with the compressively sampled image (using front end sampling matrix) is equivalent to sampling the image using back-end sampling matrix which is what is desired. Thus we use back-end sampling matrix as the sampling matrix in the reconstruction algorithm. Using the transformation we calculated our back-end sampling matrix as follows -

$$\Phi_{B,backend} = \begin{pmatrix} 1/\sqrt{2} & 1/\sqrt{2} & 0 & 0 \\ 0 & 0 & 1/\sqrt{2} & 1/\sqrt{2} \end{pmatrix} \tag{3.11}$$

Figure 3.1: The set of 30 images.

$$\Phi_{NB,backend} = \begin{pmatrix} .7894 & .6139 & 0 & 0 \\ 0 & 0 & .7894 & .6139 \end{pmatrix} \tag{3.12}$$

The level of lossy compression in JPEG is controlled using the $quality$ parameter of MATLAB function. We have measured the size of this compressed image. The baseline for image size is taken as the size of JPEG image ($quality = 75$, $bits = 8$), i.e. raw image stored as JPEG with $quality = 75$ and $bitdepth = 8$. The size of a compressively sampled image is reported as the relative percentage of this baseline. The baseline image for PSNR measurement is the raw image. The metrics for baseline is shown in Table 3.2.

We have used a set of 30 images to perform the above simulation. These images are shown in Fig. 3.1. All the images are in Grayscale 512×512 format. In Fig. 3.2 we have shown how the average of the normalized size of the raw image stored in JPEG format scales with the Quality factor. Similarly in Fig. 3.3 we have shown how reconstruction performance of JPEG (measured as the average of PSNR of 30 images) varies with the Quality factor of JPEG.

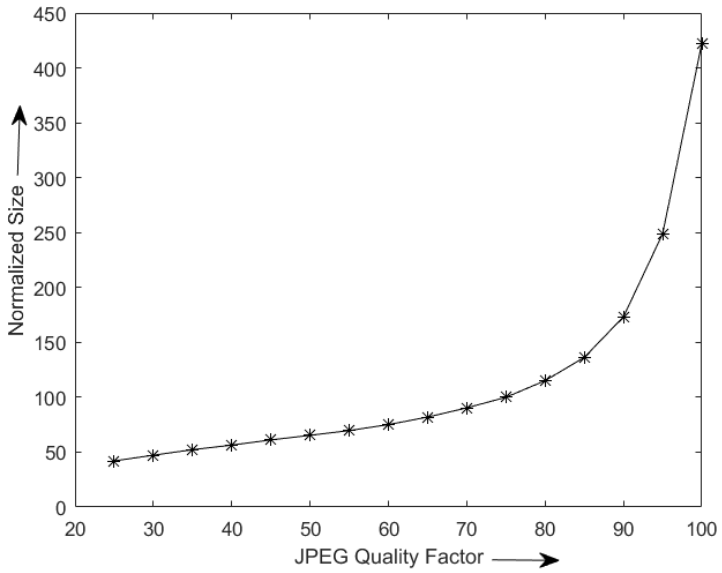


Figure 3.2: Variation of Normalized Image Size with JPEG Quality factor.

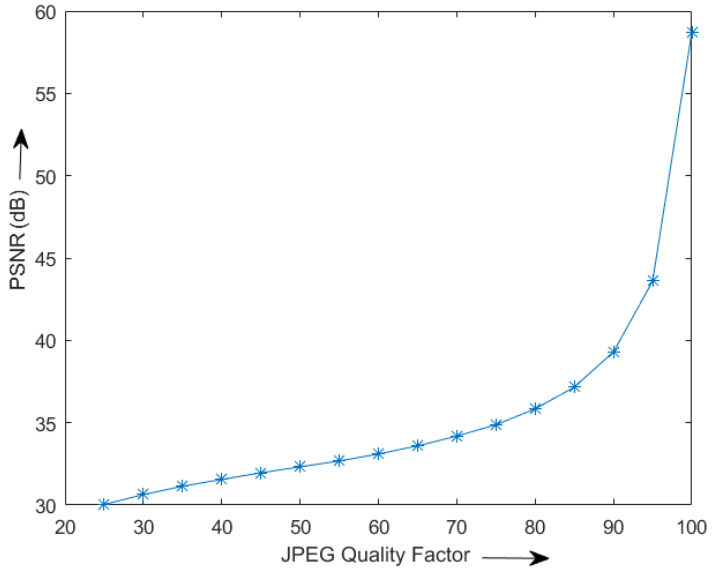


Figure 3.3: Variation of PSNR with JPEG Quality factor.

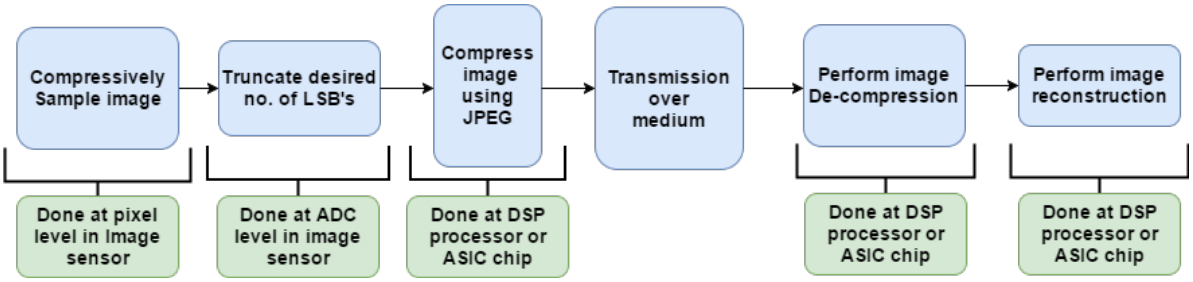


Figure 3.4: System methodology

A block diagram of system similar to HCAS mentioned in Chapter 1.1 is shown in Fig. 3.4. The Table 3.3 lists the results for system depicted in Fig. 3.4 for binary sampling matrix with input parameters such as Quality factor of JPEG and Bitdepth of the image. The output values are Normalized Size, PSNR for reconstruction and On-chip compression.

$$On - chip\ Compression = \frac{(16 - Bitdepth)}{16}, \tag{3.13}$$

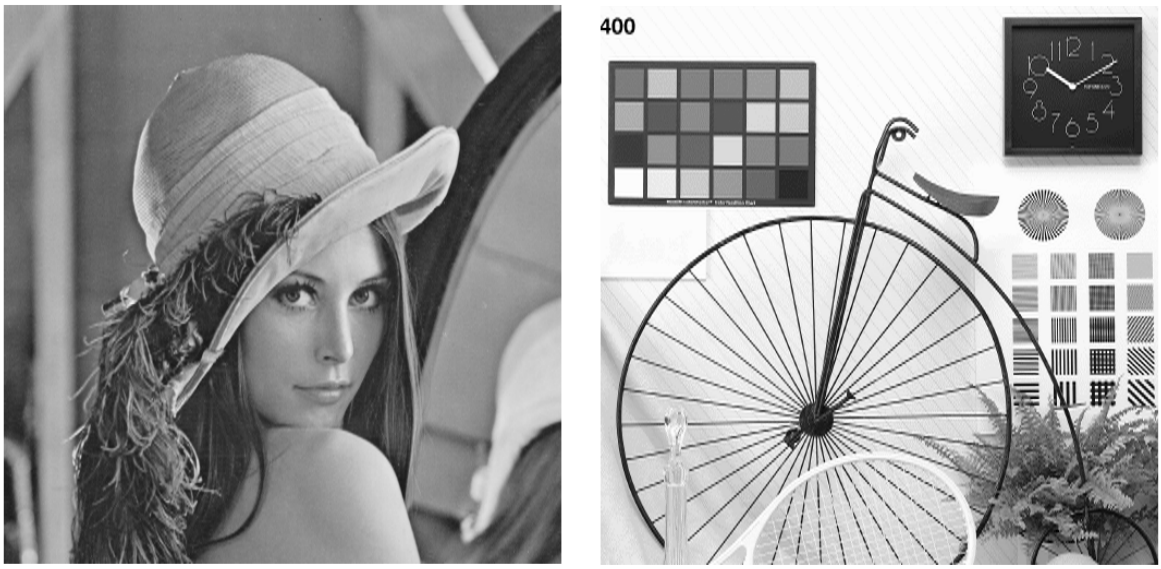
where, $Bitdepth$ is the number of bits required to represent each pixel of compressively sampled image. Since each raw-pixel is 8 bits, and we are adding 2 pixels while doing CS, we are calculating on-chip compression relative to 16 bits in Eq. (3.13).

Similarly, Table 3.4 shows the same for the non-binary matrix. The best case and worst case image reconstruction for the non-binary CS followed by lossless JPEG is shown in Fig. 3.5.

Table 3.2: Result for Baseline Image.

$ImageType$	$Quality$	$Bitdepth$	$Normalized\ Size$	(dB)
$JPEG$	75	8	100	34.87

Since we are adding two 8 bit pixels for the binary sampling matrix, we need 9 bits to represent



(a) (b)

Figure 3.5: (a) Best Case Reconstruction. PSNR = 39.71. (b) Worst Case Reconstruction. PSNR = 24.45.

the addition perfectly. For the non-binary matrix, we are using weights of 9 and 7 for each pixel. So the max value of weighted pixels can be 16×255. Hence we need 12 bits to represent the weighted addition perfectly.

We can see from the Table 3.3 and Table 3.4 that the performance of the binary and non-binary matrix for CS with lossless JPEG and without bit-truncation is almost the same. This is in agreement with the results stated in Ref. [28]. We can also see that the storage size is very high for CS with lossless JPEG. This has the potential to degrade the performance of imaging system when comes to storage and we will need a much more complicated JPEG decoder. To decrease the size, we can either decrease $quality$ or truncate LSB's or both. By truncating LSB's we not only decrease the size of the image but also significantly simplify ADC design as well as JPEG encoder and decoder design. This simplified decoder will also consume less energy because of reduced switching activity resulting from reduced bitwidth. Similarly we can also decrease the quality factor to decrease the size. For example, if we use default Quality factor i.e. 75 we can see

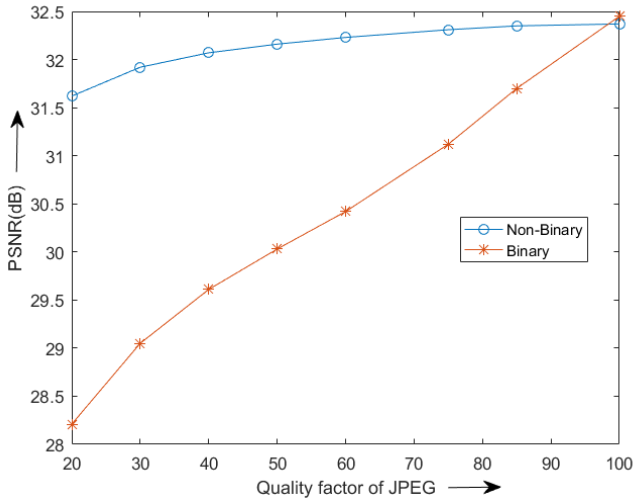


Figure 3.6: Graph showing PSNR of image reconstruction for binary and non-binary matrix cases vs JPEG Quality. LSB's have not been truncated.

that performance loss is not much but size is much smaller.

In general, for a given quality factor, the non-binary matrix performs quite better than the binary matrix. This is because it can preserve much more information than binary matrix because of larger bitwidth. This makes it more resilient to degradation during JPEG quantization step. This is also evident in the graph shown in Fig. 3.6 where none of the LSB's have been truncated. The better PSNR for non-binary sampling matrix comes at the cost of increased image size. A comparison between the normalized image-size resulting binary and non-binary sampling matrix for $bitdepth = 9$ and $bitdepth = 12$ respectively is shown in Fig. 3.7. By pruning some LSB's we can decrease image size at the cost of the PSNR of reconstructed image. Thus the non-binary sampling matrix offers more control over image quality than the binary sampling matrix.

We can also see from Table 3.3 and Table 3.4 that for a given quality factor as we truncate the LSB's of CS sampled image in the non-binary sampling method, the result approaches that of the binary sampling method i.e. the performance of the non-binary matrix almost equals that of the binary matrix for same bitdepth. For the maximum performance case i.e. CS with lossless JPEG, the performance of both sampling matrix is same for full bitdepth for each case respectively. While

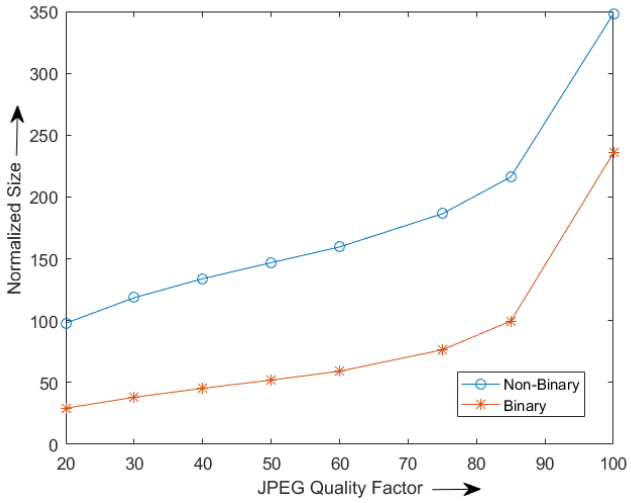


Figure 3.7: Graph showing normalized image-size binary and non-binary matrix cases vs JPEG Quality. LSB's have not been truncated.

the result for maximum performance case for CS is roughly 2dB less than the basline JPEG case of Table 3.2, the former provides roughly 43% raw data compression but the latter provides none. Reduction in raw data rate will significantly simplify our system design. This is discussed in our next section.

These were the simulations for gray-scale images. For colored images the procedure is straightforward. In the case of RGB image, the three different color planes can be thought of as three different images and CS can be applied to each of the 3 images. The reconstruction performance for colored Lenna image is mentioned in Table 3.5.

The next subsection will discuss the novel implementation of front-end sampling matrix at image sensor level.

3.4 Design

This subsection discusses the novel sensor level design to implement front-end sampling matrix presented in the previous section. It also discusses briefly about the ADC and JPEG encoder.

When comes to hardware implementation, binary block diagonal matrix means an addition of

the row or column pixels. The number of pixels to be added is the number of ones in the row of sampling matrix. For our binary sampling matrix, we can simply implement this by using double sized pixels. We can choose any pixel design i.e. 3T or 4T. Large pixels have better SNR values because dark current decreases much faster than sensitivity as area increases (Ref. [1]). Even if noise is larger in smaller pixels, it is taken care of by using Correlated Double Sampling technique. So the higher noise level of smaller pixel is not much of an issue. If we use a large photodiode to implement binary sampling matrix, it means increase in the fill-factor of pixel. If fill factor for a given pixel design is f then using a double sized photodiode will roughly give $2f/(1+f)$ fill factor. For $f = 0.7$ we get a rough approximation for new factor as $f = 0.82$. This increased fill factor can compensate the loss due to reconstruction algorithm.

The non-binary block diagonal matrix has to perform weighted addition. This can be done by using our novel design shown in Fig. 3.8. This is inspired by the 4T design. We have used a very simple technique to perform weighted addition. We have used a small capacitance (gate capacitance of MOS) to decrease response of one of the photodiode by placing it before the shutter or Tx Transistor. This MOS is labeled as cap in Fig. 3.8. This effectively decreases the sensitivity of the photodiode and it generates less output (output of a photodiode is actually a decrease in the output voltage w.r.t. reset voltage level of photodiode because photocurrent flows to discharge the junction capacitance of photodiode) as compared to the other photodiode without additional capacitance. So if the same amount of light falls in both photodiode then one photodiode will generate less output voltage than the other. When the shutter MOS (i.e. Tx_1 and Tx_2) opens, then current drains from the floating diffusion node to the photodiode. Since one photodiode has less voltage than other so one will draw less current than other. This is because our circuit is operated in transient state rather than steady state. The shutter open time is set such that the circuit remains in transient state. Since both the currents are unequal, the resulting voltage at the floating diffusion node i.e. FD is like weighted addition of two equal signals. For the non-binary sampling matrix, we used weights of 9 and 7. So, the relative weight of one pixel w.r.t. to another is approximately $1.3(9/7)$. The circuit depicted in Fig. 3.8 also achieves approximately the same

weight. Since even after truncating some LSB's we can get good images, the weighted addition does not have to be very exact as the errors will get truncated too. The Spectre simulation results for the circuit are stated in Table 3.6. The weight has been calculated in the table keeping CDS technique in mind. The weight has been calculated by curve fitting for 100 different points. For generating these points, the photocurrent in each photodiode was varied from $100fA$ to $1000fA$ in steps of $100fA$. This generates 10 points for each photodiode. Then all possible permutations of these 2 sets (one set for each photodiode) of 10 points were taken to generate 100 different points. Fig. 3.9 shows the sweep analysis performed for these 100 points (offset voltage has been removed). Fig. 3.10 shows a plot to demonstrate weighted addition of photodiode outputs. The curves in the plot represents the output voltage values for the proposed pixel circuit for 2 different cases. In each case, the photocurrent of one of the photodiode is fixed at $100fA$ and other one is varied form 100fA to 100fA in steps of 100fA. Thus, for a given current value in x-axis of the plot, total charge generated in the pixel will be same. But, the output of pixel is different for both cases because of weighted addition of photodiode output.

Addition of capacitor in one of the photodiode results in a decrease on sensitivity. In traditional designs, a decrease of sensitivity implies a loss of resolution, but in our design reconstruction algorithms help us recover this information.

If we are truncating the bits, we are significantly simplifying ADC design too. Bit truncation in the simulation can be implemented in hardware by decreasing the ADC resolution. This will result in a simpler and power efficient ADC. Since at lower resolutions noise and linearity requirements are relaxed, voltage scaling can help us achieve an exponential reduction in power consumption (Ref. [42]). Since ADC is responsible for a major chunk of power consumption during the process of raw image acquisition (Ref. [36, 43]), our technique will have a significant impact in reducing the power consumption.

We have designed our pixel for both Front Side Illumination and Backside Illumination (BSI) (Ref. [44]). The FSI layout for the Fig. 3.8 circuit is shown in Fig. 3.11. In FSI layout light enters from the frontside of the sensor where as in BSI it enters from the backside. This means in BSI we

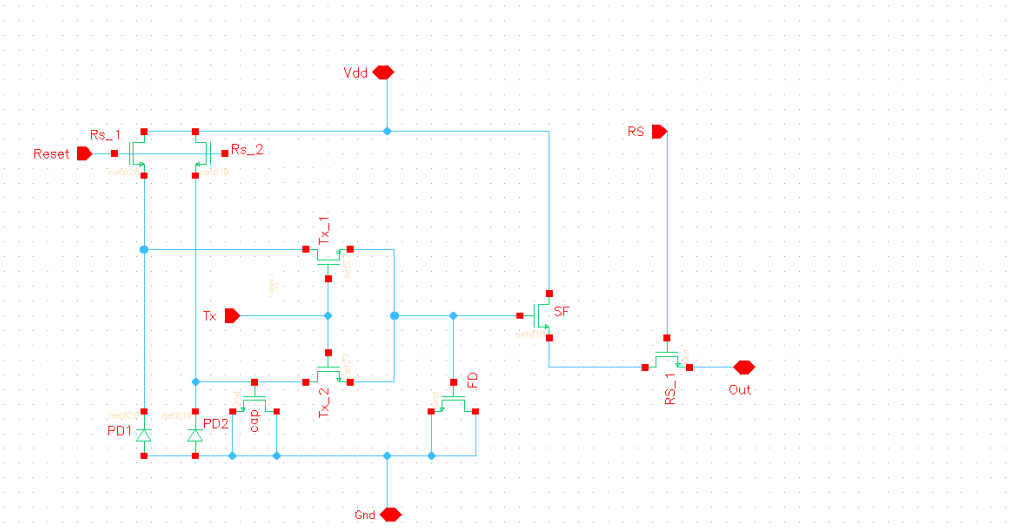


Figure 3.8: Schematic design for on-pixel compressed sensing.

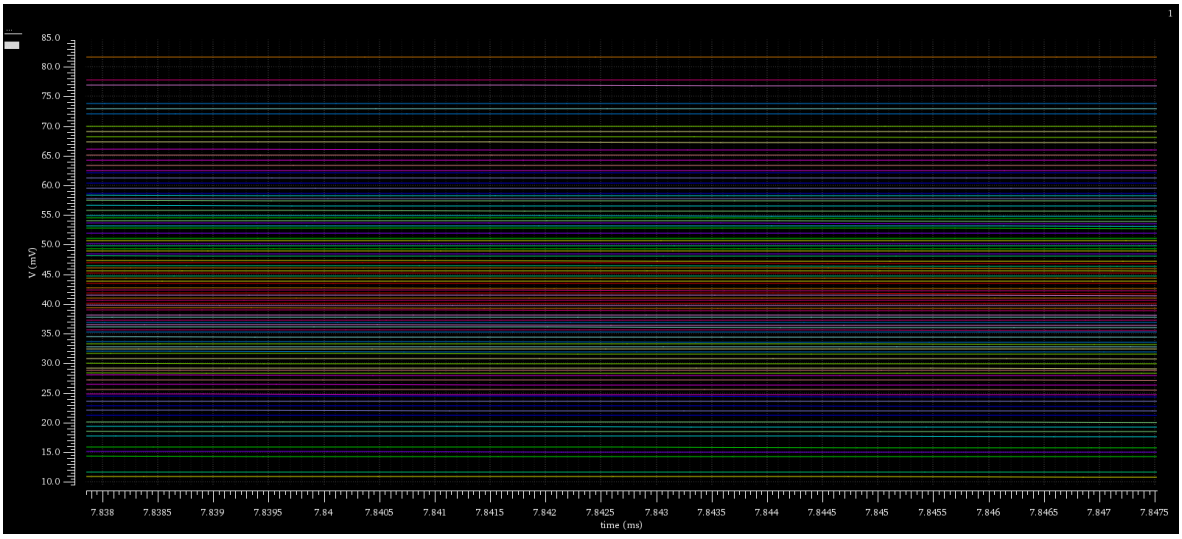


Figure 3.9: Sweep analysis for our proposed pixel circuit.

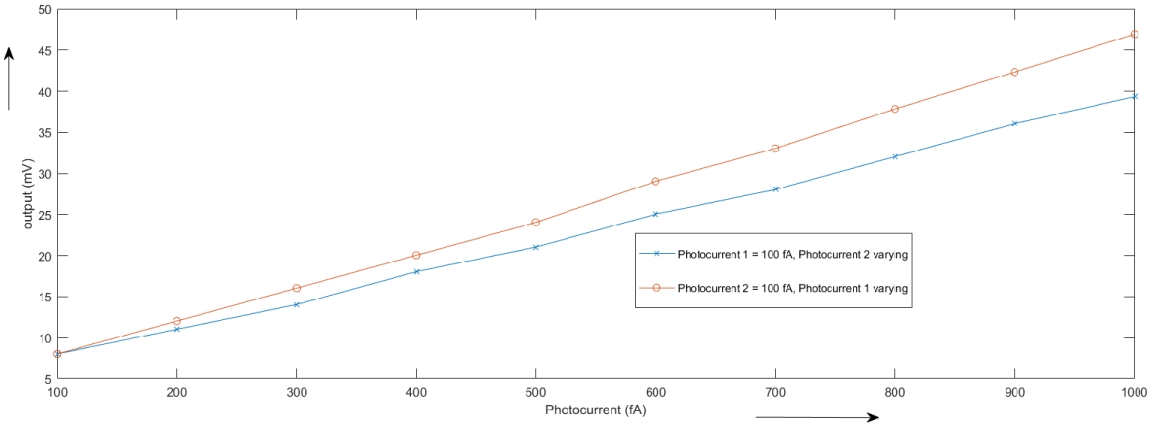


Figure 3.10: Plot showing weighted addition of photodiode outputs. For each curve, photocurrent in one of the photodiode is fixed at 100fA while the other one is varied from 100fA to 1000fA in steps of 100fA. Each point in x-axis represents same amount of charge generated in the pixel but output is different due to weighted addition of photodiode outputs.

can draw metal lines over photodiode and increase fill factor. There are two different technologies in BSI which are shown in Fig. 3.12. They are conventional BSI and stacked BSI (Ref. [44]). In conventional BSI, the logic circuit and the pixel circuit are in the same plane. Metal wiring can be drawn over pixel circuit as light enters from backside. This results in an increase of the fill factor. In stacked BSI, the logic circuit and pixels are in different planes. This means the fill factor is almost 100% for stacked BSI. The layout for conventional BSI and stacked BSI for our novel pixel circuit is given in Fig. 3.13 and Fig. 3.14. We used TSMC 200nm technology library and Cadence Design tools to implement our design. The advantages associated with on-chip implementation of CS does not depend on the technology of choice. It works equally well in any technology.

The junction capacitance, responsivity and dark current for the photodiode used in our pixel was estimated using the data and graphs presented in Ref. [1] and Ref. [13]. The formula for junction capacitance is given below,

$$C_{jdep} = \frac{C_{J0}A_D}{1-(\frac{V_d}{v_j})^m} + \frac{C_{J0sw}P_D}{1-(\frac{V_d}{v_{jsw}})^{m_{jsw}}} \tag{3.14}$$

where C_{J0} and C_{J0sw} represent zero-bias capacitance at the bottom and sidewall components,

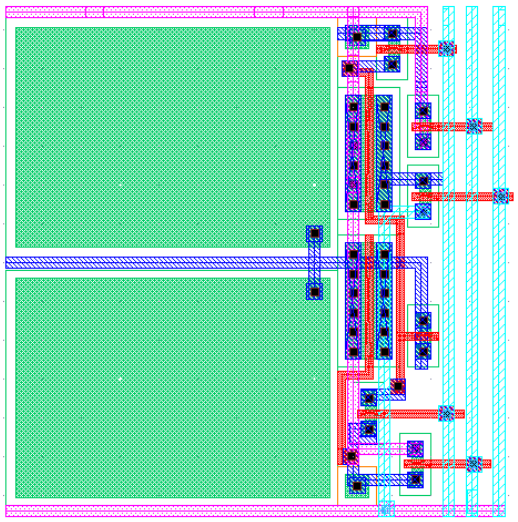


Figure 3.11: Proposed FSI Pixel Layout to implement on-chip compressed sampling.

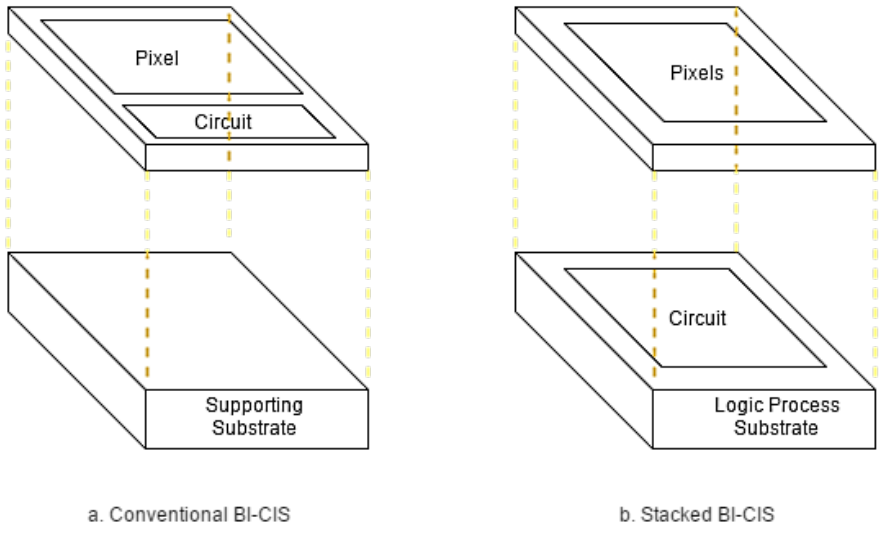


Figure 3.12: A schematic diagram explaining different Back-illuminated CMOS Image Sensor (BI-CIS).

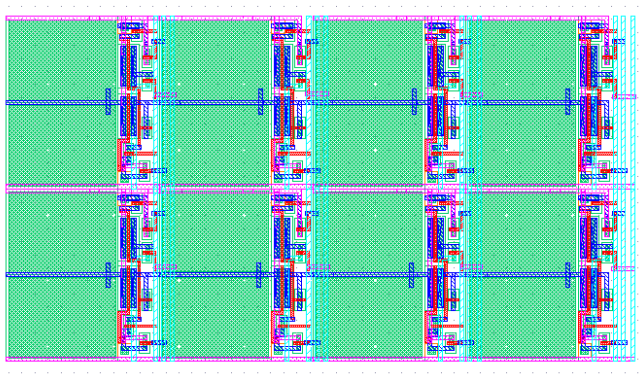


Figure 3.13: Proposed Conventional BSI Layout for pixels.

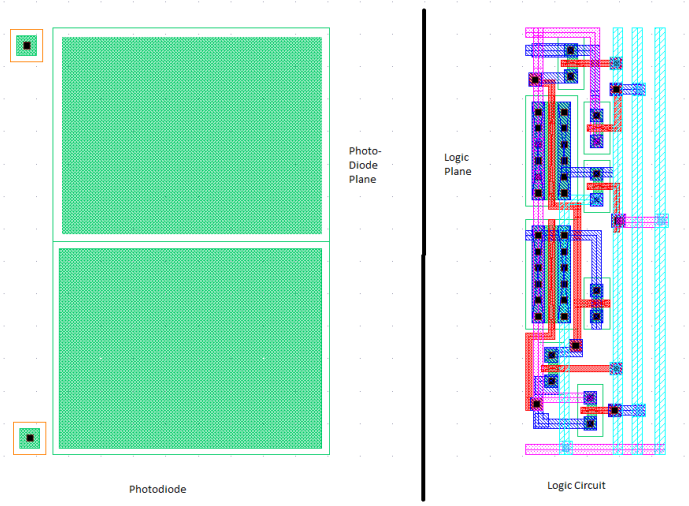


Figure 3.14: Proposed Stacked BSI Layout for pixels.

V_d is the voltage applied to the photodiode, v_j and v_{jsw} stand for the built-in potential of the bottom and the sidewall respectively, m and m_{jsw} are the grading coefficients of the bottom and the sidewalls, A_D is the photodiode area in m^2 and P_D represents the photodiode perimeter in m. These parameters are given in the Table 3.7 for our design. The table also lists the fill factor for our layout.

A problem with our pixel design is that it is non-linear. Switching transistor, source follower, active capacitances, all contribute to non-linearity. This non-linearity can be removed by curve fitting. In an actual system a lookup table can be used to simplify implementation. The equation obtained after curve fitting is reproduced below -

$$\begin{aligned} output = 1.037 - (3.324 \times 10^{-5})p2 - (4.065 \times 10^{-5})p1 - (1.678 \times 10^{-9})p2^2 \\ -(3.88 \times 10^{-9})p1p2 - (2.564 \times 10^{-9})p1^2, \end{aligned} \tag{3.15}$$

where $p1$ and $p2$ refers to photocurrent in photodiode 1 and photodiode 2 respectively in fA and output refers to output voltage in V of the pixel shown in Fig. 3.8. The weight for wighted addition has been calculated by taking the ratio of coefficients of $p1$ and $p2$.

Our system design methodology simplifies JPEG encoder and decoder design as well. JPEG generally takes DCT of image blocks of size 8×8. Since we are combining two pixels to one we are effectively reducing the number of blocks by half. This will cut energy spent during encoding by half. Since the encoder design is mostly pipelined, it will also reduce encoding latency by half. Since workload is reduced one can reduce the voltage and frequency of operation of JPEG encoder to maintain same latency. This will result in an exponential decrease in energy consumption during encoding and decoding process. If we truncate the LSB's of image this will lead to additional simplification of encoder and decoder design and power savings. It leads to a proportional decrease in switching activity and hence dynamic power. It reduces register as well as arithmetic unit bitwidth. Reduction in bitwidth of arithmetic unit can lead to a direct reduction in latency of such units and the chip floor-area.

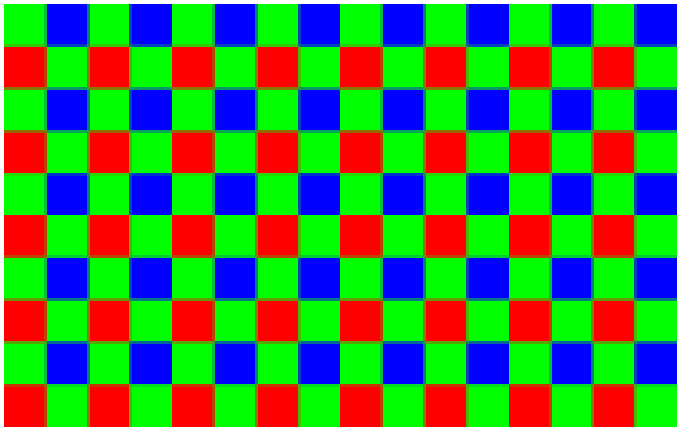


Figure 3.15: Bayer Pattern

Other implementations for both the binary and non-binary sampling matrix is also possible. For implementing the binary sampling matrix we can also use the design presented in Fig. 3.8. We can remove the cap MOS from the circuit to do so. This design will be especially useful when we have two photodiodes separated apart, as in colored image sensor implemented using popular Bayer Pattern (Ref. [45]). This is shown in Fig. 3.15. One can see that each photodiode representing a color have to be separated apart. It might not be possible to make large single color photodiode because of resolution reasons. Hence we can use design presented in Fig. 3.8 for both the non-binary matrix as well as binary matrix (i.e. without cap).

While the theoretical assumption is that all source followers or photodiode provide same gain, this is hardly the case practically. Because of manufacturing inconsistencies, the two photodiodes will have different responses and parasitics. So the binary matrix will become non-binary in actual implementation. This will not pose any problem because non-binary matrix works equally well. By incorporating such inconsistencies further into the sampling matrix, we can solve problems posed by fixed pattern noise during the reconstruction. There are multiple sources for fixed pattern noise. Photoresponse non-uniformity, source follower mismatch etc. are sources of mismatch (Ref. [46]). We can handle this mismatch by incorporating the mismatch in the sampling matrix. If all source followers or photodiodes provide different gain/response then we can use different weights for our sampling matrix for each of the pixels. If we truncate some LSB's, then we do not need

incorporate mismatch also as long as noise is less than or equal to the LSB's truncated. This is because noise will be truncated with the LSB's. For CMOS image sensor design presented in Ref. [43], the FPN noise is less than 1 LSB. So if we implement CS with bit truncation in such system, we can get rid of the noise by truncating just 1 LSB.

Yet another way to implement binary or non-binary matrix is to sum the pixels at Analog to Digital Converter (ADC) level, similar to what was presented in Ref. [36]. This has an advantage of having an option to choose between CS mode and non-CS mode of operation. But one has to pass address for each pixel. This will slow down frame rate and increase power consumption. There are certain additional disadvantages associated with it which are discussed later in the paragraph below.

The advantage of CS implemented at the sensor level is not just limited to reduction of data rate only. Our implementation shown in Fig. 3.8 for the non-binary matrix requires only 6 transistors (excluding floating diffusion nodes and capacitance) per 2 pixels i.e. 3 transistors per pixel with global shutter. This means improvement in fill factor and reduction in the size of pixels. This also means less power consumption. A simple analysis of power consumption for image acquisition can be performed by looking at the data mentioned in Ref. [36] and Ref. [43]. Both the papers use different design, technology and specification. Hence power consumption is different for both of them. But the relative breakdown of power spent in I/O, ADC, Pixel and Other operations are approximately the same. Roughly 90% power is spent in I/O and ADC in the image acquisition. Our proposed CS implementation cuts the I/O and ADC operations exactly by on-chip compression ratio i.e. by $25\% - 68.75\%$. We have used the data for the normal mode of operation at 120 frames/sec in our work. The data has been reproduced in Table 3.8. The table also lists the estimation of power if our system design methodology is implemented in the normal image sensor described in the paper. We have also incorporated the power spent during JPEG compression in the same table using the design presented in Ref. [47] as reference. Please note that we have only performed rough approximation of JPEG power consumption based switching activity. We can see from Table 3.8 that one can achieve approximately $23.5\% - 65\%$ power savings for compression

ratio of $25\% - 68.75\%$ respectively. In this work, we only consider the energy spent during image acquisition and compression process.

Our proposed CS technique also reduces the wiring area in the die as we combine two rows or columns in one. We cut the amount of wiring required by half. We need half the number of rows or column select, Reset and Transmit for Global Shutter. We also need half size address decoder leading to a reduction in power and chip area. Because of the reduced size of the pixel and reduced wiring area, we can fit more pixels in the same die area using existing technology. It is not possible to exploit these advantages if we implement CS at ADC level as mentioned previously and in Ref. [36].

3.5 Discussion

This chapter established the use and implementation of simple deterministic matrices. Use of these matrices helps one to use CS in conjunction with JPEG to achieve compression throughout the imaging pipeline. Sampling using the matrices presented in this paper is achieved through the use of specialized pixel designs resulting in on-chip compression ranging from $25\% - 68.75\%$. Since most of the power spent is spent in I/O and ADC, this leads to significant reduction in power. It also leads to reduction in power spent in JPEG as the input image resolution to JPEG is reduced. Doing a specialized imaging system design can lead to significant savings in power and area and improvement in performance. However specialized design comes with disadvantage of being inflexible i.e. it will operate only for the configuration for which it is designed so it can be only used for the applications for which it is suitable. By using specialized pixels, we reduce the number of row/column wiring, address decoder, power supply, reset and global shutter control wiring in image sensor by half. We also improve fill factor and reduce transistor count per pixel by 1 for global shutter. Looking at the other way round, this savings in space can be used to increase resolution of image sensors which has potential to compensate for loss due to reconstruction algorithm. This aspect can be explored by testing prototypes which is beyond the scope of this work. In the next chapter we will look at flexible implementation of CS rather than specialized pixel designs. Flexible implementation have the advantage of being programmable and

user can shift from CS mode to non-CS mode with push of a button.

Table 3.3: Results for Binary Block Diagonal matrix.

$ImageType$	$Quality$	$Bitdepth$	$Normalized\ Size$	$PSNR$ (dB)	$On-Chip$ $Compression(\%)$
$JPEG + CS_B$	$lossless$	9	220.42	32.45	43.75
$JPEG + CS_B$	$lossless$	8	188.05	32.43	50
$JPEG + CS_B$	$lossless$	7	154.99	32.37	56.25
$JPEG + CS_B$	$lossless$	6	121.97	32.19	62.5
$JPEG + CS_B$	$lossless$	5	92.58	31.58	68.75
$JPEG + CS_B$	100	9	235.63	32.45	43.75
$JPEG + CS_B$	100	8	198.19	32.44	50
$JPEG + CS_B$	100	7	161.13	32.30	56.25
$JPEG + CS_B$	100	6	126.23	31.93	62.5
$JPEG + CS_B$	85	9	99.59	31.70	43.75
$JPEG + CS_B$	85	8	69.55	30.87	50
$JPEG + CS_B$	75	9	76.37	31.12	43.75
$JPEG + CS_B$	75	8	51.68	30.02	50
$JPEG + CS_B$	75	7	33.75	28.64	56.25
$JPEG + CS_B$	75	6	21.08	27.11	62.5
$JPEG + CS_B$	75	5	12.33	25.45	68.75
$JPEG + CS_B$	60	9	59.11	30.42	43.75
$JPEG + CS_B$	50	9	51.97	30.03	43.75
$JPEG + CS_B$	40	9	45.31	29.61	43.75
$JPEG + CS_B$	30	9	38.08	29.05	43.75
$JPEG + CS_B$	20	9	29.23	28.21	43.75

Table 3.4: Results for Non-Binary Block Diagonal matrix.

$Image\ Type$	$Quality$	$Bitdepth$	$Normalized\ Size$	$PSNR$ (dB)	$On-Chip$ $Compression(\%)$
$JPEG + CS_{NB}$	$lossless$	12	317	32.37	25
$JPEG + CS_{NB}$	$lossless$	11	285.57	32.37	31.25
$JPEG + CS_{NB}$	$lossless$	10	253.55	32.37	37.5
$JPEG + CS_{NB}$	$lossless$	9	221.88	32.37	43.75
$JPEG + CS_{NB}$	$lossless$	8	189.23	32.36	50
$JPEG + CS_{NB}$	$lossless$	6	122.27	32.13	62.5
$JPEG + CS_{NB}$	$lossless$	5	92.82	31.52	68.75
$JPEG + CS_{NB}$	100	12	348.21	32.37	25
$JPEG + CS_{NB}$	100	11	310.88	32.37	31.25
$JPEG + CS_{NB}$	85	12	216.14	32.35	25
$JPEG + CS_{NB}$	85	11	176.32	32.29	31.25
$JPEG + CS_{NB}$	75	12	186.45	32.31	25
$JPEG + CS_{NB}$	75	11	146.28	32.16	31.25
$JPEG + CS_{NB}$	75	10	108.39	31.78	37.5
$JPEG + CS_{NB}$	75	9	76.48	31.09	43.75
$JPEG + CS_{NB}$	75	8	51.78	29.99	50
$JPEG + CS_{NB}$	75	7	33.79	28.63	56.25
$JPEG + CS_{NB}$	75	6	21.11	27.11	62.5
$JPEG + CS_{NB}$	60	12	159.67	32.23	25
$JPEG + CS_{NB}$	50	12	146.85	32.16	25
$JPEG + CS_{NB}$	50	10	76.86	31.09	37.5
$JPEG + CS_{NB}$	50	9	52.05	30.01	43.75
$JPEG + CS_{NB}$	40	12	133.90	32.07	25
$JPEG + CS_{NB}$	40	10	67.86	30.78	37.5
$JPEG + CS_{NB}$	30	12	118.58	31.92	25
$JPEG + CS_{NB}$	20	12	97.93	31.62	25

Table 3.5: Results for Non-Binary and binary block diagonal matrix for colored Lenna image.

$CS\ Type$	$PSNR(dB)$ Red	$PSNR(dB)$ $Green$	$PSNR(dB)$ $Blue$
$Lossless\ JPEG + CS_B$	41.27	37.52	35.94
$Lossless\ JPEG + CS_{NB}$	41.22	37.48	35.91

Table 3.6: Results of Spectre simulation for weight calculation.

$Photodiode\ 1\ (fA)$	$Photodiode\ 2\ (fA)$	$Output(Pixel\ Circuit)\ (mV)$
100	100	8
200	100	11
100	200	12
500	500	38.9
1000	1000	81.7
$Calculated\ weight\ using\ curve\ fitting(100\ points)\ =\ 1.22$		

Table 3.7: Table for Photodiode and Pixel Parameters

C_{jdep} (fF)	C_{j0} (mF/m^2)	C_{j0sw} (F/m) $\times 10^{-10}$	v_j (V)	v_{jsw} (V)	m	m_{jsw}	V_d (V)	A_D (μm^2)	P_D μm	FSI $F.F.$	BSI $F.F.$
32.8	1.067	1.6	.8	.65	.41	.35	1.8	45.76	27.5	54.16%	65.54%

Note. F.F. stands for Fill-Factor of pixel. In case of BSI, Fill Factor is mentioned for conventional BSI only.

Table 3.8: Table for Power Estimation

Operation	**Design 1 (mW) (Ref. [36])**	**Design 2 (mW) (Ref. [43])**	**CS for Design 1 (mW)** $C.R.: 68.75\% - 25\%$	**CS for Design 2 (mW)** $C.R.: 68.75\% - 25\%$
I/O	27	70	$8.34 - 20.25$	$21.87 - 52.5$
ADC	60	209	$18.75 - 45$	$65.31 - 156.75$
Pixel	1.8	23	1.8	23
Other	4.2	20	4.2	20
JPEG (Ref. [47])	13.18	386.3	$4.11 - 9.88$	$120.71 - 289.72$
Total	106.18	708.3	$37.2 - 81.13$	$250.89 - 541.97$
Power Savings	0%	0%	$64.96\% - 23.59\%$	$64.57\% - 23.48\%$

Note. CS stands for our proposed Compressed Sensing and C.R. stands for on-chip Compression Ratio.

4. CS RECONSTRUCTION USING BINNING [1]

The previous chapter focused on specialized pixel design for CS. While specialized design has the advantage of simplified hardware implementation, it makes the design constrained in the sense that it will only operate on CS mode. In this chapter we will discuss how images sampled using binning operation already available in modern image sensors can be easily used in CS reconstruction algorithm without requiring any pixel redesign. Binning can be implemented easily by manipulating the control signals of pixel circuits which are generally exposed to the system programmer. Thus it makes CS programmable i.e. users can shift between compressed and non-compressed sensing modes.

4.1 Related Work

Implementations of CS has been around for a while. However as discussed in previous chapter, most of the earlier work like the ones by Oike et al. (Ref. [36]), Dadkhah et al. (Ref. [37]) and Katic et al. (Ref. [14]) focused on traditional methods of CS implementation i.e. using blocks of Gaussian random matrices for sampling. These designs increased the complexity of image sensors and made matter worse when comes to acquisition and storage effort per bit. It significantly modified the image sensor design which is an expensive task. Recent work by Leitner et al. (Ref. [48, 49]) used sparse deterministic measurement matrices. However they operate by summing column lines together and need additional circuitry for that inside image sensor. Whereas our method can be implemented just by controlling the timing of the pixel control signals as well as by summing the column lines. This is because there is none or little overlap between our CS measurements. At block level we have zero overlap and our block size is only 4 pixels, whereas for Leitner et al. the block size is entire image. This makes their implementation more complex. Also we use DDWT as sparsity basis, which is superior, as opposed to DCT basis in the work by

[1] Parts of this chapter are reprinted with permission from "Programmable compressed sensing using simple deterministic sensing matrices," by P. S. Gupta and G. S. Choi, 2018, Optoelectronic Imaging and Multimedia Technology V, vol. 10817, p. 108170C, © 2018, International Society for Optics and Photonics

Leitner et al. .

4.2 Simulation and Results

A block diagram of our proposed methodology is shown in Fig. 4.1. Step 1 and Step 2 of the methodology shown in Fig. 4.1 correspond to Step 1 and Step 2 of HCAS methodology (Fig. 1.1). The Step 3 of Fig. 4.1, which corresponds to reconstruction is actually Step 6 of 1.1). The intermediate steps are omitted in this chapter for the sake of simplicity. Compressed Sampling occurs in Step 1 (see Fig. 4.1) using binning technique which corresponds to Step 1 of HCAS (Fig. 1.1).

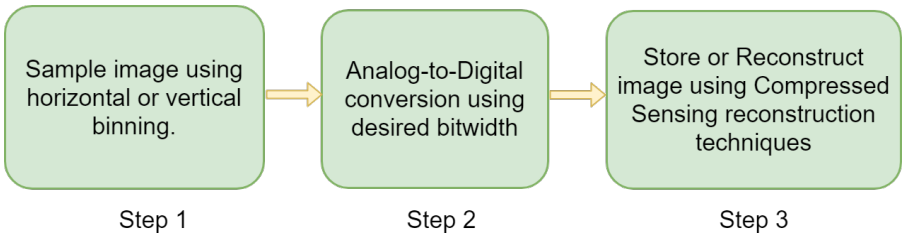


Figure 4.1: Block diagram of our proposed methodology. Step 1,Step 2 and Step 3 of this methodology correspond to Step 1, Step 2 and Step 6 of HCAS methodology (Fig. 1.1) respectively.

When speaking from CS perspective, this work uses the following sampling matrices -

$$\Phi_{B\ 1/2} = \begin{pmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{pmatrix} \tag{4.1}$$

$$\Phi_{B\ 3/4} = \begin{pmatrix} 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 \end{pmatrix} \tag{4.2}$$

Sampling by $\Phi_{B\ 1/2}$ means simple addition of 2 rows or 2 columns to get a single row or column respectively (See Fig. 4.2). It can be implemented using either horizontal or vertical binning. For

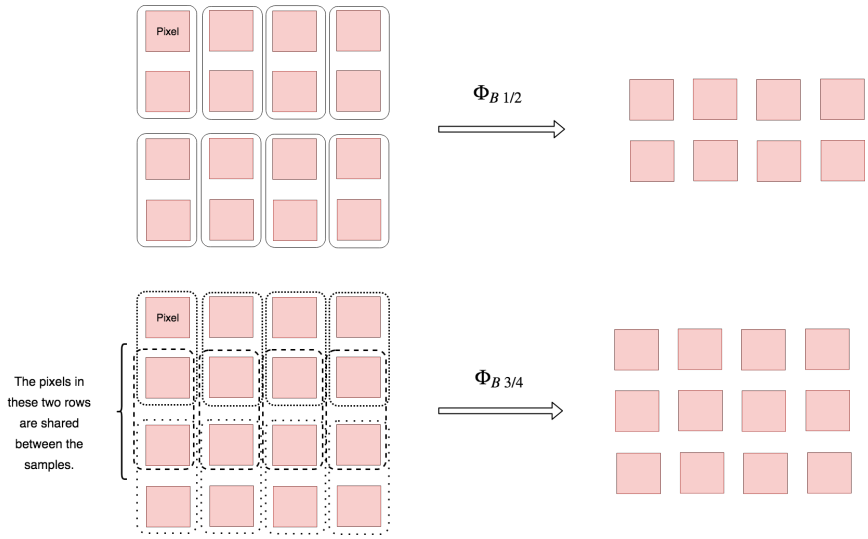


Figure 4.2: Diagram showing the sampling process. This diagram shows sampling using vertical binning process. Enclosed pixels are binned together. Horizontal binning can be implemented similarly.

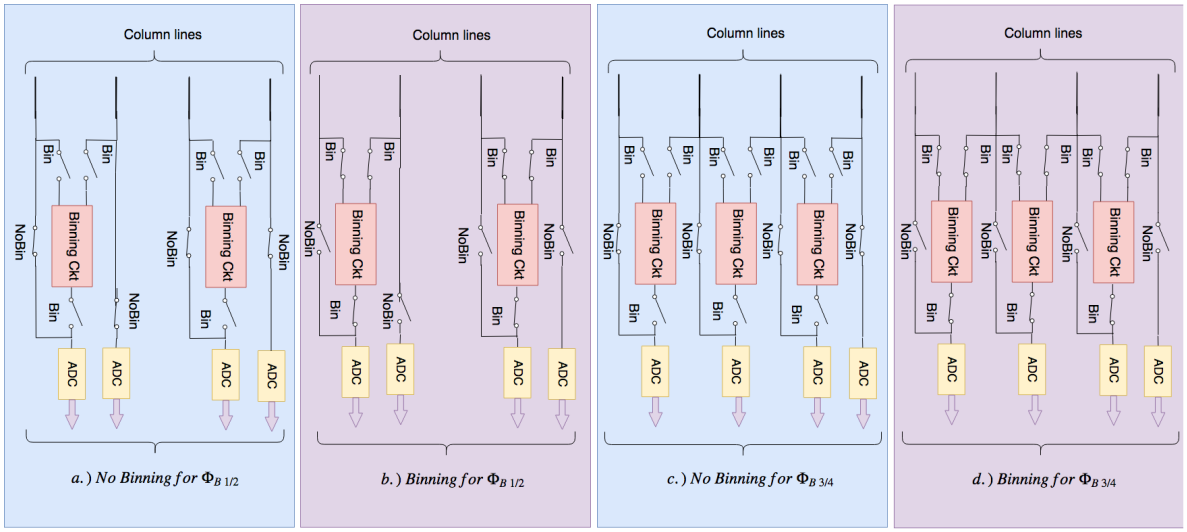


Figure 4.3: Block level schematic diagram for implementation of sampling by horizontal binning. (a) No binning for $\Phi_{B\ 1/2}$. (b) Binning for $\Phi_{B\ 1/2}$. (c) No binning for $\Phi_{B\ 3/4}$. (d) Binning for $\Phi_{B\ 3/4}$.

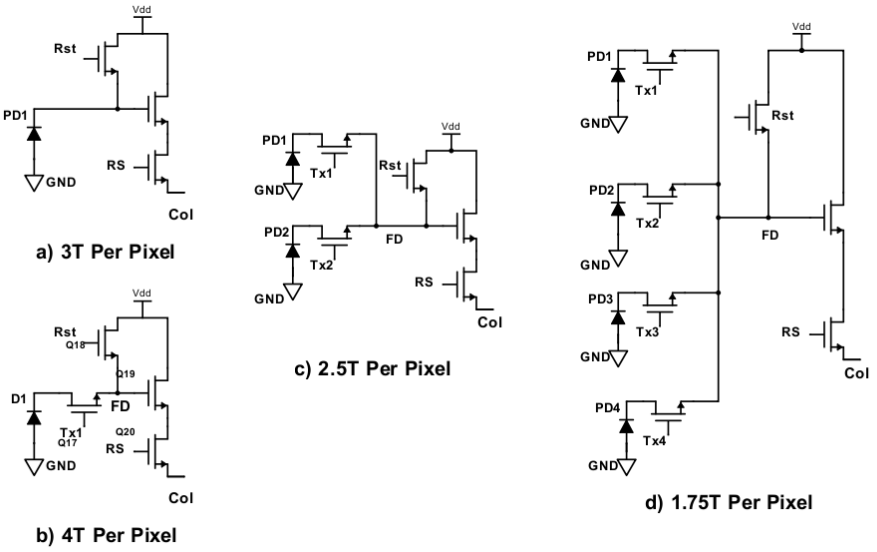


Figure 4.4: Diagram showing schematic for (a) 3T, (b) 4T, (c) 2.5T and (d) 1.75T pixel design. PD refers to photodiode, Tx refers to transfer signal, Rst refers to Reset signal, RS refers to row select signal, FD refers to floating diffusion node and Col refers to column line.

horizontal binning, binning circuits can used in a fashion shown in Fig. 4.3 (a) and (b). These binning circuits will perform either addition of signal or averaging depending upon the circuit implementation design. Horizontal binning can be done for all the pixels architectures shown in Fig. 4.4 i.e. 1.75T, 2.5T, 3T and 4T.

For operation in CS mode, all Bin switches shown in the Fig. 4.3 should be high and all $NoBin$ switches should be low and vice-versa for non-CS mode. Vertical binning can also be used to implement sampling by $\Phi_{B\ 1/2}$ for 2.5T and 1.75T pixels. For 2.5T pixel this can be done by switching Tx1 and Tx2 lines to high simultaneously as shown in Fig. 4.5(b). For 1.75T pixel, this can be done by pulling high Tx1 and Tx2 simultaneously followed by Tx3 and Tx4 as shown in Fig. 4.5(d). Simultaneous switching results in averaging operation. For operating in non-CS mode, Tx signals are switched to high one at a time as shown in Fig. 4.5(a) and (c).

$\Phi_{B\ 3/4}$ also represents simple addition, however there is an overlap in the binning window. The second and third row/column overlaps between samples for every four rows/columns as shown in

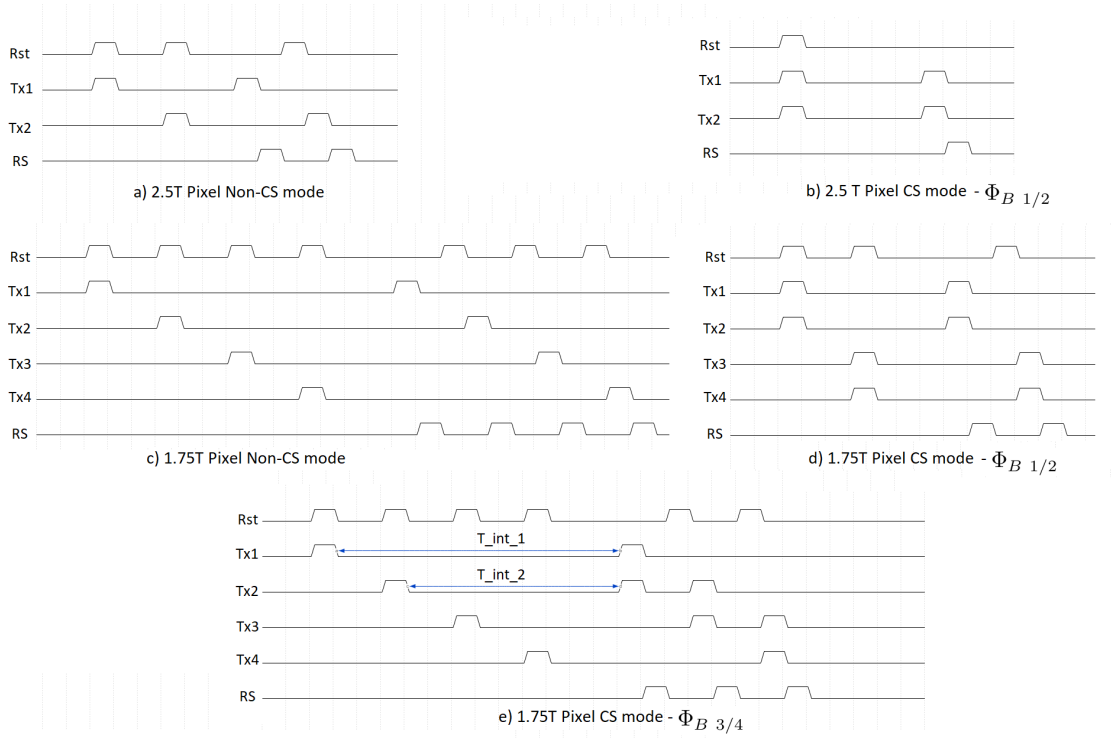


Figure 4.5: Timing diagram for pixel operation for both non-compressed and compressed sensing mode using vertical binning. (a) 2.5T Pixel Non-CS mode. (b) 2.5T Pixel CS mode - $\Phi_{B\ 1/2}$. (c) 1.75T Pixel Non-CS mode. (d) 1.75T Pixel CS mode - $\Phi_{B\ 1/2}$. (e) 1.75T Pixel CS mode - $\Phi_{B\ 3/4}$.

Fig. 4.2. The block level schematic for horizontal binning implementation is shown in Fig. 4.3(c) and (d). This can be implemented for all the pixels architectures shown in Fig. 4.4 i.e. 1.75T, 2.5T, 3T and 4T. The pattern shown in Fig. 4.3 should be repeated every 4 columns to achieve sampling desired by matrix $\Phi_{B\ 3/4}$. For operation in CS mode, all Bin switches shown in the diagram should be high and all $NoBin$ switches should be low and vice-versa for non-CS mode.

Sampling by $\Phi_{B\ 3/4}$ can also be implemented using vertical binning. For 1.75T pixel, this can be done by switching Tx1 and Tx2 to high simultaneously followed by Tx2 and Tx3 followed by Tx3 and Tx4. The timing diagram for this scheme is shown in Fig. 4.5(d). One can see from the diagram that integration time for 2 adjacent photodiodes are not same. For example, integration time marked as T_int_1 and T_int_2 are not same. Since $T\_int\_2 < T\_int\_1$, our sampling matrix will become following -

Figure 4.6: The set of 30 images used to perform simulation.

$$\Phi_{3/4} = \begin{pmatrix} 1 & 1-\delta & 0 & 0 \\ 0 & 1 & 1-\delta & 0 \\ 0 & 0 & 1 & 1-\delta \end{pmatrix}, \tag{4.3}$$

where $0 < \delta < 1$. The actual value of δ will depend upon implementation details and its estimation is outside the scope of this paper. However, we show that reconstruction performance will not be affected much by different values of delta. The reconstruction performance using various values of δ is show in Table 4.1.

It is not necessary to do binning on entire image. Only selected regions of image can be binned by turning the binning mode on for selected rows or columns. There are numerous implementation of binning circuits available in the literature (Ref. [16, 17, 18, 19, 15]). Its implementation is outside the scope of this paper. Also, there are numerous other binning techniques that can be applied to achieve sampling using $\Phi_{B\ 1/2}$ and $\Phi_{B\ 3/4}$ for different pixel architectures. For example, one can turn adjacent row select signals to high to perform vertical binning using the technique mentioned by Leitner et al. (Ref. [49, 48]).

Apart from binning, compression can also be performed by decreasing the number of bits used

to represent a pixel in digitized image. This happens at Step 2 (see Fig. 6.1) which takes place at ADC. Decrease in bitwidth means decrease of resolution at ADC. This can result in huge savings in power because at lower resolutions, noise and linearity requirements are relaxed and voltage scaling can help us achieve an exponential reduction in power consumption (Ref. [42]). Since ADC is responsible for a major share of power consumption during the process of raw image acquisition (Ref. [36, 43]), this will lead to a significant reduction in the power consumption. It will also make readout faster resulting in an increase of frame rate. Reconstruction performance with different bitwidth is reported in Table 4.1.

Finally at Step 3 (see Fig. 6.1), we either store or reconstruct the image. Since sampling matrices perform either averaging or addition of adjacent pixels, the sampled image still has image-like properties. Hence conventional image compression algorithms like JPEG etc. can be applied to sampled images (Ref. [50]). However, storage performance has not been measured as it is outside the scope of this work. For measuring performance of reconstruction process, we report the PSNR (Peak Signal to Noise Ratio) and SSIM (Structural Similarity Index Metric). We use raw image as a baseline to calculate these metrics. We use $bitwidth = 8$ for baseline image. Raw data compression is also reported in Table 4.1 and can be calculated as follows -

$$Raw\ Data\ Compression = 100 \times \frac{(8 - S.R. \times Bitwidth)}{8}, \tag{4.4}$$

where $S.R.$ represents sampling rate. Sampling rate is $1/2$ and $3/4$ for $\Phi_{B\ 1/2}$ and $\Phi_{B\ 3/4}$ respectively. A set of 30 images used to perform simulation is shown in Fig. 4.6. The best reconstructed images in terms of PSNR and SSIM is shown in Fig. 4.7. A plot of PSNR vs. Bitwidth and SSIM vs Bitwidth for different values of δ is shown in Fig. 4.8 and Fig. 4.9 respectively. The plots only show results for $\Phi_{B\ 3/4}$ sampling matrix.

4.3 Discussion

This chapter discussed about programmable implementation of CS by binning which is implemented through simple manipulation of control signals of pixel. It also discussed binning using

a) Best PSNR

b) Best SSIM

Figure 4.7: The best reconstructed image. (a) Best PSNR case. PSNR = 44.7dB, SSIM = 0.953. (b) Best SSIM case. PSNR = 42.53 dB, SSIM = 0.982.

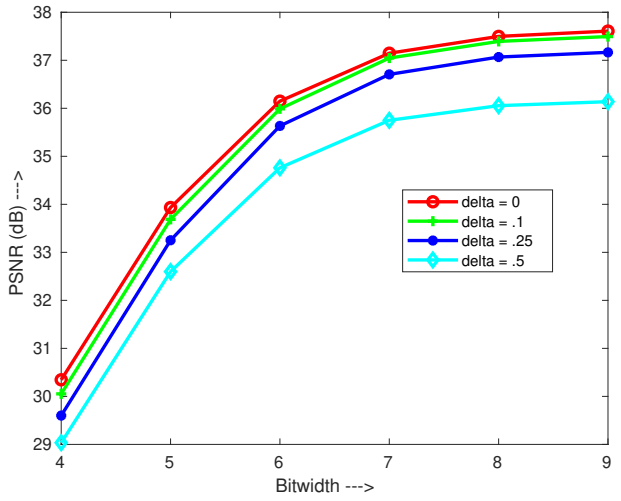


Figure 4.8: Graph showing reconstruction PSNR vs Bitwidth for different values of δ. Results for only $\Phi_{B\ 3/4}$ sampling matrix is shown.

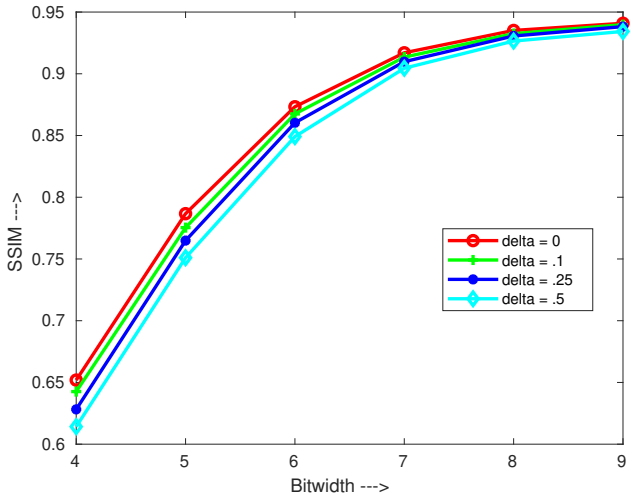


Figure 4.9: Graph showing reconstruction SSIM vs Bitwidth for different values of δ. Results for only $\Phi_{B\ 3/4}$ sampling matrix is shown.

column read lines in pixel grid. However it requires extra circuitry. Apart from making CS programmable which is the main takeaway of this chapter, there are other important observations. The prime one being that overlap of measurements in sensing matrix requires more circuitry as pixel values are shared between pixels. Another important observation is that small block size is the key enabler to achieve in-pixel CS. This is because if there are overlaps in measurement in blocks and block size is very large then the value of each pixel has to be routed to multiple measurements which will require additional circuitry. Thus binning is the simplest and most flexible way to achieve CS sampling. It has the potential to reduce raw data generated by more than half while maintaining the PSNR close to 34 dB resulting power savings by atleast half or decreasing clock frequency by roughly half. While this chapter used MatLab simulations to perform experiments, the next chapter will focus on full system simulation. Full system simulations provide a more accurate picture as they take into account all the non-ideal behavior and noise that occurs in practical settings.

Table 4.1: Reconstruction Results

$Sampling\ Matrix$	δ	$Bitwidth$	$PSNR\,(dB)$	$SSIM$	$Raw\ Data\ Compression\ (\%)$
$\Phi_{B\ 3/4}$	0	9	37.6075	0.9409	15.625
$\Phi_{B\ 3/4}$	0	8	37.4977	0.9351	25
$\Phi_{B\ 3/4}$	0	7	37.1471	0.917	34.375
$\Phi_{B\ 3/4}$	0	6	36.1485	0.8733	43.75
$\Phi_{B\ 3/4}$	0	5	33.9331	0.7867	53.125
$\Phi_{B\ 3/4}$	0	4	30.3456	0.6519	62.5
$\Phi_{B\ 3/4}$	0.1	9	37.4945	0.9398	15.625
$\Phi_{B\ 3/4}$	0.1	8	37.3934	0.9325	25
$\Phi_{B\ 3/4}$	0.1	7	37.0466	0.9135	34.375
$\Phi_{B\ 3/4}$	0.1	6	35.9889	0.8675	43.75
$\Phi_{B\ 3/4}$	0.1	5	33.687	0.7754	53.125
$\Phi_{B\ 3/4}$	0.1	4	30.0526	0.6426	62.5
$\Phi_{B\ 3/4}$	0.25	9	37.1652	0.9382	15.625
$\Phi_{B\ 3/4}$	0.25	8	37.0675	0.9306	25
$\Phi_{B\ 3/4}$	0.25	7	36.7065	0.9098	34.375
$\Phi_{B\ 3/4}$	0.25	6	35.6331	0.8603	43.75
$\Phi_{B\ 3/4}$	0.25	5	33.2519	0.7649	53.125
$\Phi_{B\ 3/4}$	0.25	4	29.6	0.6282	62.5
$\Phi_{B\ 3/4}$	0.5	9	36.138	0.9343	15.625
$\Phi_{B\ 3/4}$	0.5	8	36.0549	0.9265	25
$\Phi_{B\ 3/4}$	0.5	7	35.7498	0.9045	34.375
$\Phi_{B\ 3/4}$	0.5	6	34.7607	0.8491	43.75
$\Phi_{B\ 3/4}$	0.5	5	32.5984	0.751	53.125
$\Phi_{B\ 3/4}$	0.5	4	29.0348	0.6142	62.5
$\Phi_{B\ 1/2}$	$N.A.$	9	32.4509	0.8526	43.75
$\Phi_{B\ 1/2}$	$N.A.$	8	32.4359	0.8503	50
$\Phi_{B\ 1/2}$	$N.A.$	7	32.376	0.8393	56.25
$\Phi_{B\ 1/2}$	$N.A.$	6	32.1984	0.8146	62.5
$\Phi_{B\ 1/2}$	$N.A.$	5	31.5833	0.7566	68.75
$\Phi_{B\ 1/2}$	$N.A.$	4	29.8868	0.6515	75

5. CS ON AN IMAGING SYSTEM [1]

The previous chapter performed CS experiments at algorithmic simulation level. Those experiments did not take into account the noise and non-linearity present in an actual system. To get a more accurate picture of CS performance, this chapter simulates the proposed imaging pipeline using a full system simulator known as ISET [2]. ISET is a very popular simulator used by imaging system companies and it has recently become open-source. ISET simulations serves as a proof of concept that these algorithms can be deployed to an actual imaging system.

5.1 Related Work

In general, the work in CS fall in two categories - Algorithmic simulation using clean images or actual system implementation. Simple simulations do not take into account the noise and non-ideal behaviour in present in actual settings. A crucial missing link is a full system simulation. Full system simulation establishes the efficacy of proposed algorithm in real world deployment scenario and allows one to quickly iterate and improve the algorithm to suit the needs of actual system. These works, such as by Oike et al. (Ref. [36]), Dadkhah et al. (Ref. [37]) and Katic et al. (Ref. [14]), perform the actual implementation while the works by Leitner et al. (Ref. [48, 49]) fall in algorithmic simulation category. To the best of our knowledge this is the first full system simulation of CS using ISET like simulator.

5.2 Simulation and Experiment

This section discusses the implementation of our HCAS pipeline using ISET for more accurate simulations. A block diagram of our proposed methodology is shown in Fig. 5.1. Step 1 and Step 2 of the methodology shown in Fig. 5.1 correspond to Step 1 and Step 2 of HCAS methodology (Fig. 1.1). The Step 3 of Fig. 5.1, which corresponds to reconstruction is actually Step 6 of 1.1). Compressed Sampling is done by binning (using averaging operation) in $Sensor$ stage (see Fig.

[1] Parts of this chapter are reprinted with permission from "Accurate simulation of on-sensor compressed sensing using ISET," by P. S. Gupta and G. S. Choi, Image Sensing Technologies: Materials, Devices, Systems, and Applications VI, vol. 10980, p. 1098012, © 2019, International Society for Optics and Photonics

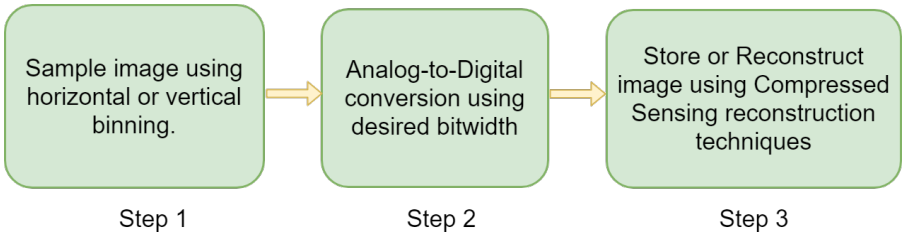


Figure 5.1: Block diagram of our proposed methodology. Step 2 and Step 3 of this methodology correspond to Step 1, Step 2 and Step 6 of HCAS methodology (Fig. 1.1) respectively.

5.1). From a CS perspective, this represents sampling using following sensing matrix -

$$\Phi_B = \begin{pmatrix} 0.5 & 0.5 & 0 & 0 \\ 0 & 0 & 0.5 & 0.5 \end{pmatrix} \tag{5.1}$$

The sampling matrix used in Eq. 5.1 combines 2 column pixels to 1 using averaging. One of the requirements for CS reconstruction to work is that -

$$\Phi\Phi^T = I. \tag{5.2}$$

However, one can see that $\Phi_B\Phi_B^T \neq I$. This can be solved by multiplying Φ_B with a scaling factor as follows -

$$\Phi_{B-Scaled} = \frac{1}{\sqrt{N}}\Phi_B, \tag{5.3}$$

where $N = Sum\ of\ squares\ of\ row\ elements\ of\ matrix$. For our case $N = \sqrt{0.5}$. However, instead of multiplying Φ_B with scaling factor one can multiply the image sampled by Φ_B with scaling factor. This can be done right before the reconstruction process. The reconstruction process is then done with $\Phi_{B-Scaled}$. For reconstruction, the algorithm mentioned in Section 2.3.4 was used with Dual Tree Discrete Wavelet Transform as sparsity basis. The experiments were done using the $Caucasian\ Male$ image provided by the ISET which is shown in Fig. 5.3.

Most of the parameters used for imaging system in this experiment were derived from the ISET

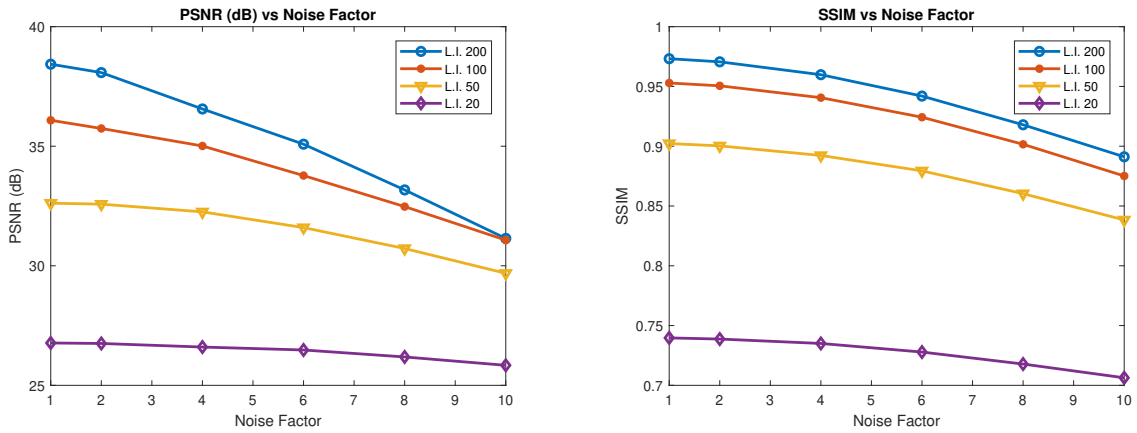


Figure 5.2: Graph of PSNR(dB) vs. Noise Factor and SSIM vs. Noise Factor. L.I. refers to Luminous Intensity in cd.

tutorials and examples. The parameters for sensor and pixel are shown in Table 5.1. For spectral quantum efficiency, the values of Nikon D100 were used which were provided by ISET. A diffraction limited lens model with focal length of $3\ mm$, $f\ number$ of 4 and $cosine-fourth$ for $off-axis\ vignetting$ was used. The noise parameters of pixels and image sensor are shown in last row of Table 5.1. These are scaled using a single factor, called $Noise\ Factor$ in this work, and its effect is studied on CS reconstruction performance. This means if noise factor is n, then all noise values shown in last row of Table 5.1 are multiplied with n. The values of noise shown in the table are for noise factor equal to 1. D65 illumination was used in all cases. The light intensity was also varied from $200\ cd$ to $20\ cd$ and reconstruction performance measured. The exposure time was same for all the cases. For image processor, $illuminant\ correction\ method$ was set to $Gray\ World$, $internal\ color\ space$ was set to $XYZ\ Color\ Space$ and $sensor\ data\ conversion\ method$ was set to $Macbeth\ Color\ Checker\ (MCC)\ optimized$. For benchmark, the image generated without application of binning and noise factor as 1 at corresponding illumination was used. PSNR (Peak Signal to Noise Ration) and SSIM (Structural Similarity) index were used as quality metrics. The PSNR and SSIM results were averaged for 5 experiments. The results are shown in the form of graph in Fig. 5.2 for both PSNR and SSIM. One can see that as noise factor increases, the per-

formance of CS algorithm decreases. PSNR and SSIM also decrease with decrease in intensity of light. However at low light conditions, loss due to decrease in Signal to Noise Ratio (SNR) at image sensor dominates the performance of CS and noise factor has little effect. This can be seen from the graph as curve becomes more flat as light intensity decreases. The best case and worst case reconstruction results are shown in top row and bottom row of Fig. 5.3 respectively. The left column of Fig. 5.3 represents images generated without binning and right column represents binned images reconstructed using CS. One can see from Fig. 5.3 that even for the worst case, the reconstructed image is very similar to the image generated without binning.

The low PSNR and SSIM is due to low lighting condition. Since the number of image samples are reduced by half due to binning, there is approximately 50% power savings in the process of image acquisition (Ref. [50, 51]). Additionally, binning process can be controlled using software which provides flexibility to the user to switch between CS and no CS modes.

Table 5.1: Image Sensor and Pixel Parameters

Resolution	***Pixel Size***	***Fill Factor***	***ADC Resolution***
1024×1024 $Bayer\ GBRG$	$2.2\ \mu m$	.45	$8\ bits$
Voltage Swing	***Well Capacity***	***Exposure***	***CDS***
$1.15\ V$	$9000\ e^-$	$61.9\ ms$	off
Dark Voltage	***Read Noise***	***PRNU***	***DSNU***
$10^{-5}\ V/s$	$.96\ mV$	$.2218\ \%$	$1\ mV$

PRNU refers to Photo Response Non-Uniformity, DSNU refers to Dark Signal Non-Uniformity and CDS stands for correlated double sampling.

5.3 Discussion

This chapter implemented proposed imaging pipeline in ISET framework. This allows one to measure the performance of CS algorithm under variety of different conditions and configuration

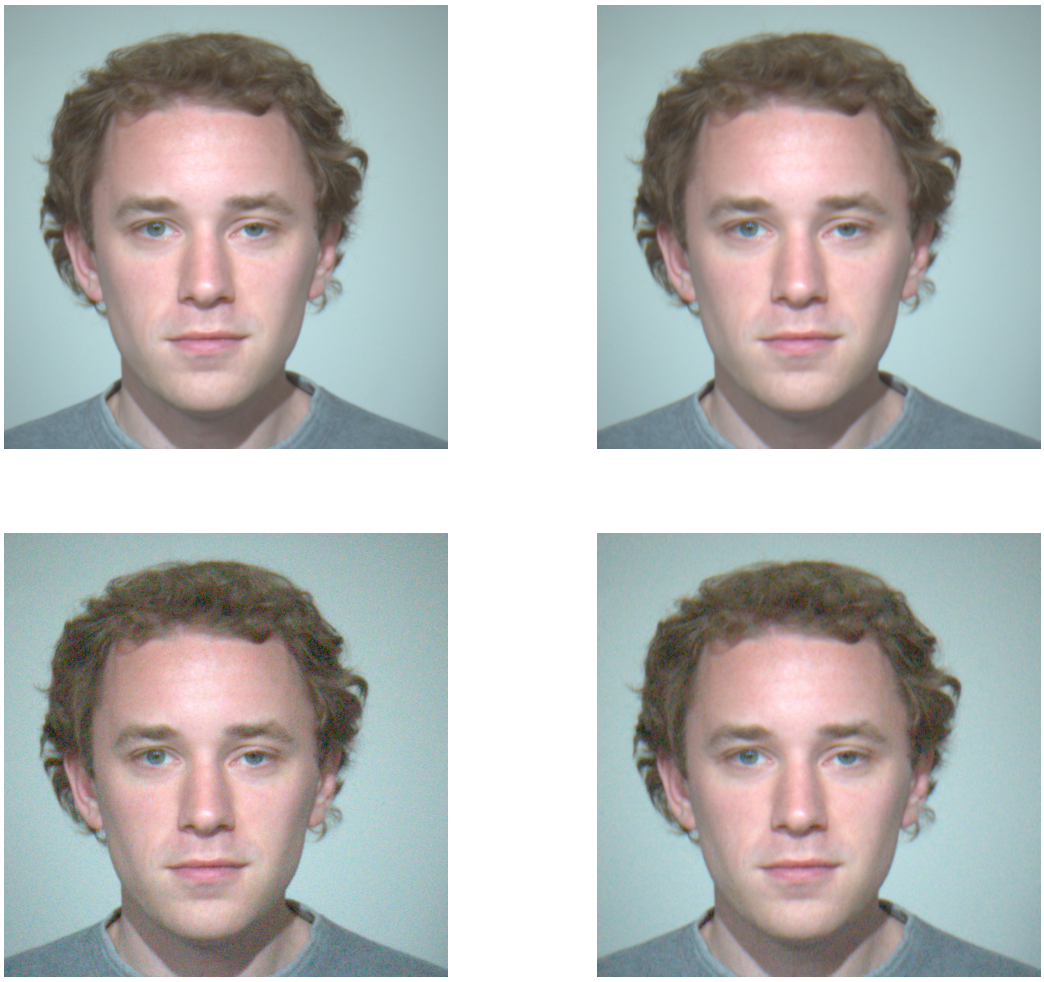


Figure 5.3: Caucasian Male image from ISET simulator for best case and worst case condition. Top-Left: Image taken at 200 cd luminous intensity and noise factor 1 (Best case, no binning). Top-Right: CS reconstructed image at 200 cd luminous intensity and noise factor 1 (PSNR = 38.57 dB, SSIM = 0.9733. Best reconstruction case). Bottom-Left: Image taken at 20 cd luminous intensity and noise factor 10 (Worst case, no binning). Bottom-Right: CS reconstructed image at 20 cd luminous intensity and noise factor 1 (PSNR = 25.84 dB, SSIM = 0.7066. Worst reconstruction case).

like different noise levels, lighting conditions, pixel size etc. . This allows one to study the effect of noise and non-linearity due to external factors on reconstruction algorithm whose performance is crucial to our proposed pipeline. One can see that under very noisy conditions, the performance of reconstruction algorithm drops as noise due to external factors dominate the process. Depending upon the application and end user requirement, the system designer can then take informed decisions on modifications to improve the performance of system. For example one can use large pixel sizes to improve low light sensitivity if that is the requirement of end user. All this modification and studies can be done prior to actual design resulting in huge savings in development cost and time.

Until now we have contended that reconstruction process can happen in cloud etc. . However sometimes there is a need to run reconstruction in the edge device itself. The next chapter is devoted to such efforts.

6. FIXED POINT CS RECONSTRUCTION [1]

The previous chapter discussed the full system implementation of our proposed pipeline using CS. However it used a floating point reconstruction algorithm. This chapter adds the fixed point implementation of the reconstruction algorithm to the ISET implementation proposed in previous chapter making the entire CS based imaging pipeline operating in fixed point mode. Fixed points computations are much more efficient than their floating point counterparts. There might be a need to implement reconstruction algorithm in the edge device itself so this is a must step to do. Also even if reconstruction is carried out in cloud, it does not hurt to have an efficient implementation of algorithm. However, conversion to fixed point does not come without cost. There is a trade-off between reconstruction quality and bit precision. Sometimes a good trade-off exists where the loss in quality is not significant and the bitwidth of computations is also reasonable and such scenarios should be exploited to gain maximum efficiency. We study this trade-off with simple averaging based sampling matrices and compare the complexity with Gaussian based sampling matrix.

6.1 Related Work

To our best knowledge this is the first implementation of CS using ISET which utilizes fixed point computation only. The works, such as by Oike et al. (Ref. [36]), Dadkhah et al. (Ref. [37]) and Katic et al. (Ref. [14]), use floating point reconstruction methods and random sampling matrices. The work by Leitner et al. (Ref. [48, 49]) uses deterministic matrices closer to our sampling matrices but still use reconstruction algorithm in floating point mode. Also, they use DCT as sparsity basis while we use DDWT as sparsity basis which is superior. None of the works used ISET for studying the implementation of their system.

[1] Parts of this chapter are reprinted with permission from "Fixed point simulation of compressed sensing and reconstruction," by P. S. Gupta and G. S. Choi, Computational Imaging IV, vol. 10990, p. 109900I, © 2019, International Society for Optics and Photonics.

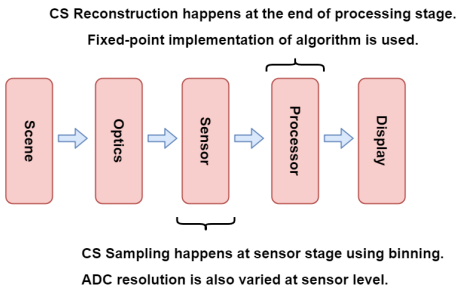


Figure 6.1: Block diagram of our proposed methodology.

6.2 Simulation and Experiment

As a first step, this work converts the reconstruction algorithm to fixed point representation without the use of ISET. This avoids the need for multispectral images. It then uses this fixed point reconstruction algorithm in ISET for performing accurate simulations of CS acquisition and reconstruction. Using ISET, this work studies the effect of ADC bitwidth and image sensor noise on reconstruction performance.

For CS, Gaussian random matrices are generally used. However, from previous chapters we know that CS reconstruction works quite good for super-resolution tasks too (Ref. [52, 50, 51]). CS super-resolution requires use of deterministic sensing matrix which can be easily implemented in hardware using pixel binning (Ref. [51]). This work also uses CS super-resolution. From a CS point of view, binning (averaging operation) means sampling using following matrix,

$$\Phi_B = \begin{pmatrix} 0.5 & 0.5 & 0 & 0 \\ 0 & 0 & 0.5 & 0.5 \end{pmatrix}. \tag{6.1}$$

The sampling matrix used in Eq. 6.1 takes the average value of 2 pixels. One of the requirements of sampling matrix is that,

$$\Phi\Phi^T = I. \tag{6.2}$$

This can be solved by multiplying Φ_B with scaling factor as shown below,

$$\Phi_{B-Scaled} = K \times \Phi_B, \tag{6.3}$$

where,

$$K = \frac{1}{\sqrt{Sum\ of\ squares\ of\ row\ elements\ of\ matrix}} = \frac{1}{\sqrt{0.5}}. \tag{6.4}$$

However, this work does some additional rearrangement. It takes out the common factor of 0.5 from Φ_B and includes it in the scaling factor. So the sampling matrix becomes as follows,

$$\Phi_{B-Scaled} = \underbrace{\frac{1}{\sqrt{0.5}} \times 0.5}_{\text{New Scaling factor K}} \times \begin{pmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{pmatrix} \tag{6.5}$$

So the new scaling factor becomes,

$$K = \frac{1}{\sqrt{0.5}} \times 0.5 = \sqrt{0.5}. \tag{6.6}$$

And the new sampling matrix becomes,

$$\Phi_{B-new} = \begin{pmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{pmatrix}. \tag{6.7}$$

Instead of multiplying the sampling matrix with scaling factor, this work multiplies the sampled image with scaling factor. This reduces the number of multiplications by half, since the sampled image has only half the number of samples compared to original. This simple transformation has a huge impact in hardware resource usage. Reconstruction using the matrix in Eq. 6.7 requires only addition of rows of image and scalar multiplication of result with K mentioned in Eq. 6.6. Where as for Gaussian random sampling matrices, expensive matrix multiplication is required. For a sampling rate of 0.5, a comparison of addition and multiplication operations for super-resolution and Gaussian case is shown in Table 6.1.

One can see from the table that CS super-resolution reduces the multiplication and addition operations by a factor of N. For traditional block CS, a block size of 32 is common. Thus there is 32 times decrease in number of addition and multiplication operations for super-resolution case.

Even if the matrix is sparse like $\Phi_{B\ 3/4}$ introduced in Chapter 4, K becomes a matrix as follows -

$$K = \begin{pmatrix} -0.2706 & -0.3827 & -0.2706 \\ 0.5000 & 0.0000 & -0.5000 \\ -0.6533 & 0.9239 & -0.6533 \end{pmatrix} \tag{6.8}$$

Thus using simplistic sampling matrix like Φ_B makes reconstruction simpler.

Table 6.1: Comparison of complexity between CS Super-Resolution and CS using Gaussian Matrix. Block Size = N. Subrate = 0.5

$\boldsymbol{Operation}$	$\boldsymbol{Super-Resolution}$	$\boldsymbol{Random-Gaussian}$
$Addition$	$O(N^2)$	$O(N^3)$
$Multiplication$	$O(N^2)$	$O(N^3)$

From a hardware implementation perspective, sparsity of sampling matrix has additional advantages too apart from ones discussed already. Since only two pixels are added to produce one resulting value it reduces the bitwidth of the associated adders compared to non-sparse case where multiple values are added to produce one result.

For reconstruction, this work uses the algorithm mentioned in Section 2.3.4 with Dual Tree Discrete Wavelet Transform as sparsity basis. The operations as well as wavelets used in the reconstruction algorithm were converted to fixed point representation. This work uses 2's complement format. Since coefficients of wavelet transform can differ from image to image, a set of 30 images used in Ref [50] were used to arrive at an appropriate fixed-point transformation using trials which is shown in Table 6.2. After fixed point conversion the PSNR dropped by only 0.05 db

for the set of 30 images.

Table 6.2: Fixed point conversion of Reconstruction process

$\boldsymbol{Step}$	$\boldsymbol{Integer\ Bits, Decimal\ Bits}$
$Step\ 1, 2, 5, 6$	$9, 4$
$Step\ 3, 4$	$16, 4$
$Filter\ coefficients$	$1, 16$
$\boldsymbol{Loss}$	$\boldsymbol{0.05\ dB}$

$Steps$ refer to the steps of algorithm in Section 2.3.4

Next, this fixed point reconstruction algorithm was used in ISET framework. This study was done to find out the effect of ADC bit resolution and sensor noise on CS reconstruction performance in an actual system. A block diagram of our proposed methodology is shown in Fig. 6.1. Compressed Sampling is done by binning (using averaging operation) in $Sensor$ stage (see Fig. 6.1). This happens before ADC operation in image sensor. The ADC resolution was varied from 8 bits to 5 bits and its effect on CS reconstruction was studied. Most of the parameters used for imaging system in this experiment were derived from the ISET tutorials and examples. The parameters for sensor and pixel are shown in Table 6.3. For spectral quantum efficiency, the values of Nikon D100 provided by ISET were used. A diffraction limited lens model with focal length of $3\ mm$, $f\ number$ of 4 and $cosine - fourth$ for $off - axis\ vignetting$ was used. The last row of Table 6.3 shows the noise parameters of pixels and image sensor. These are scaled using a single factor, called $Noise\ Factor$ in this work, and its effect is studied on CS reconstruction performance. This means if noise factor is n, then all noise values shown in last row of Table 6.3 are multiplied with n. The values of noise shown in the table are for noise factor equal to 1. D65 illumination and a light intensity of $200\ cd$ was used. For image processor, $illuminant\ correction\ method$ was set to $Gray\ World$, $internal\ color\ space$ was set

to $XYZ\ Color\ Space$ and $sensor\ data\ conversion\ method$ was set to Macbeth Color Checker (MCC) optimized. The experiments were done using the $Caucasian\ Male$ image provided by the ISET which is shown in Fig. 6.2. For baseline image, the image generated without application of binning and noise factor as 1 and bitwidth as 8 was used. This is shown in Fig. 6.2(a). This work uses PSNR and SSIM index as quality metrics to measure reconstruction performance. The PSNR and SSIM results were averaged for 5 experiments. The results are shown in the form of graph in Fig. 6.3 for both PSNR and SSIM.

As expected, as the noise factor increases, the performance of CS algorithm decreases. PSNR and SSIM also decrease with decrease in bitwidth. However, the decrease is very small from bitwidth 8 to 6. Performance falls sharply for bitwidth of 5. At bitwidth of 5, the loss of quality due to decrease in number of bits dominates and noise factor has little effect. This can be observed from the graph as the curve is almost flat. Some reconstruction results are shown in Fig. 6.2. The baseline image is shown in Fig. 6.2(a). One can see from the figure that for bitwidth of 8 to 6 (Fig. 6.2(c-h)), the perceptual quality of image is quite good. However for bitwidth of 5 (Fig. 6.2(b), one can see that there is lot of degradation in the picture.

As the number of image samples are reduced by half due to binning, this results in approximately 50% power savings in the process of image acquisition (Ref. [50, 51]). Fixed point implementation allows hardware designers to avoid expensive floating point execution units and use simpler, faster and power efficient fixed point units in the design. Additionally, bitwidth reduction results in a simpler and power efficient ADC. If 6 bits are used for CS, it results in 62.5% reduction in raw data bits generated at image sensor compared to no CS case. At lower ADC resolution, noise and linearity requirements are relaxed. Thus voltage scaling can be applied to achieve an exponential reduction in power consumption (Ref. [42]). Since ADC is responsible for a major chunk of power consumption during the process of raw image acquisition (Ref. [36, 43]), more than 50% power can be saved.

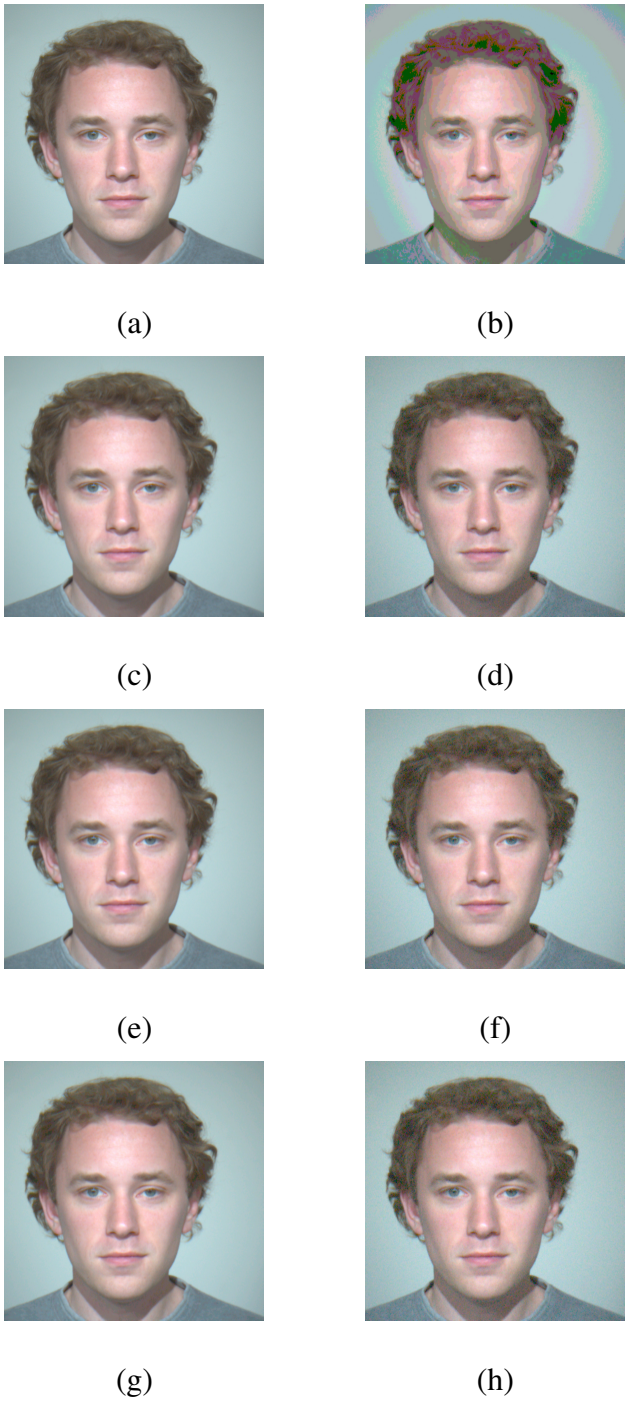


Figure 6.2: Results of CS Reconstruction Experiment for Caucasian Male Image. (a) Baseline Image. N.F. = 1, No CS. (b) B.W. = 5, N.F. = 1, PSNR = 25.69 dB, SSIM = 0.8013 (c) B.W. = 8, N.F. = 1, PSNR = 38.18 dB, SSIM = 0.9732 (d) B.W. = 8, N.F. = 10, PSNR = 31.43 dB, SSIM = 0.8896 (e) B.W. = 7, N.F. = 1, PSNR = 37.79 dB, SSIM = 0.9710 (f) B.W. = 7, N.F. = 10, PSNR = 31.44 dB, SSIM = 0.8877 (g) B.W. = 6, N.F. = 1, PSNR = 36.83 dB, SSIM = 0.9609 (h) B.W. = 6, N.F. = 10, PSNR = 31.04 dB, SSIM = 0.8812. N.F. stands for noise factor and BW stands for bitwidth of ADC output.

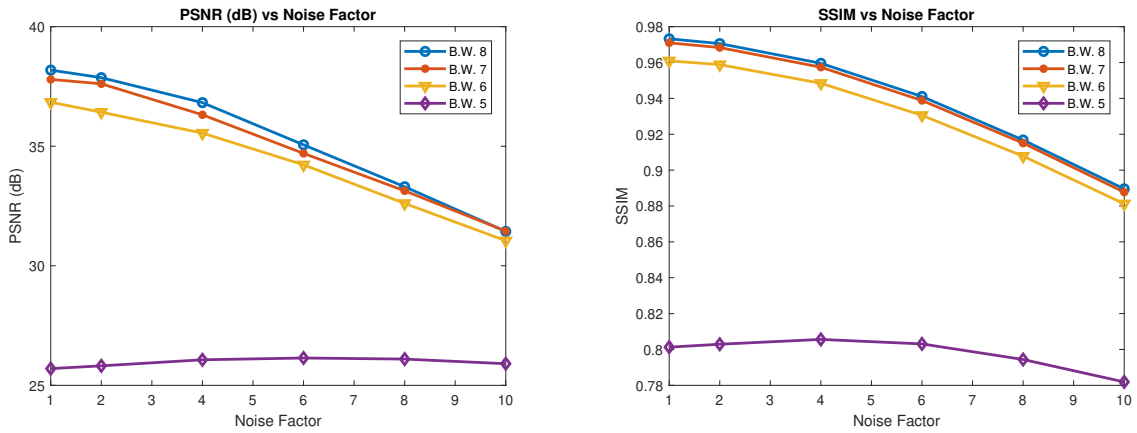


Figure 6.3: Graph of PSNR(dB) vs. Noise Factor and SSIM vs. Noise Factor for different ADC bitwidth. B.W. refers to bitwidth/resolution of ADC.

6.3 Discussion

This chapter finishes the fixed point implementation of entire imaging pipeline based on CS implementation. An implementation of the entire pipeline was presented using ISET. A key takeaway is that having a simple averaging sampling matrix simplifies the reconstruction process as well. For a block size of $N \times N$ the complexity of sampling step (Step 2 and Step 6 in Section 2.3.4) in the reconstruction algorithm is reduced by a factor of N as compared to Gaussian random sampling matrices. Even for sparse matrix like $\Phi_{B\ 3/4}$ introduced in Chapter 6 the complexity increases because of overlap between the measurements. Thus having a simple sensing matrix is beneficial for entire imaging pipeline. We also see that SPL (Section 2.3.4) reconstruction algorithm can be easily converted to fixed point implementation with negligible loss (approx 0.05 dB) in reconstruction quality.

While the last few chapters focused extensively on CS and it performed quite well for compression in the entire imaging pipeline. For example we can easily achieve a performance of more than 30 dB while maintaining raw data compression above 50% levels. However as seen in Chapter 2 the performance falls sharply when we try to achieve higher compression in raw data. Inspired by the success of DL algorithms in Computer Vision field we decided to pursue it for our proposed

Table 6.3: Image Sensor and Pixel Parameters

Resolution	***Pixel Size***	***Fill Factor***	***ADC Resolution***
1024×1024 $Bayer\ GBRG$	$2.2\ \mu m$	0.45	$8\ bits$
Voltage Swing	***Well Capacity***	***Exposure***	***CDS***
$1.15\ V$	$9000\ e^-$	$61.9\ ms$	off
Dark Voltage	***Read Noise***	***PRNU***	***DSNU***
$10^{-5}\ V/s$	$0.96\ mV$	$0.2218\ \%$	$1\ mV$

PRNU refers to Photo Response Non-Uniformity, DSNU refers to Dark Signal Non-Uniformity and CDS stands for correlated double sampling.

imaging pipeline. The next chapter is devoted to this purpose.

7. DEEP LEARNING FOR RECONSTRUCTION

The earlier chapter focused on CS based methods for reconstruction of images acquired using HCAS. This system had a limitation that if one tried to average 4 pixels instead of 2, the reconstruction performance falls off sharply as explained in Chapter 3. To overcome this issue and achieve extremely high compression rates, we employ DL based network to reconstruct the image. Deep Learning methods have gained popularity in recent times due to their superior performance and have outpaced tradition computer vision algorithms in the task they perform.

7.1 Related Work

While DNN has been used for tasks like super-resolution (Ref. [53, 54, 55, 56, 57]) and image denoising [58], this work is novel in the following perspectives:

i) Our acquisition method (see Fig. 7.1) uses realistic and hardware based compression schemes from imaging system perspective like binning, bit truncation and JPEG compression. It performs compression on entire image acquisition pipeline i.e. from raw data at source (image sensor) to processed data (using JPEG).

ii) Downsampling operation in our framework is *averaging and rounding* instead of bicubic which is more popular for superresolution tasks [59]. Again our method is more realistic as averaging operation is easy to implement in hardware especially at image sensor level using pixel binning technique available in commercial image sensors.

iii) By performing in-situ compression on raw data using binning and bit truncation, HCAS performs power savings in downstream power hungry components like ADC (Analog to Digital Converters) and DSP units.

There are some other works in the literature which have proposed pixel bit depth enhancements [60, 61, 62]. However these works were targeted at converting low bitdepth images to high bitdepth display. Another work proposed super-resolution and bitdepth enhancement [63], how-

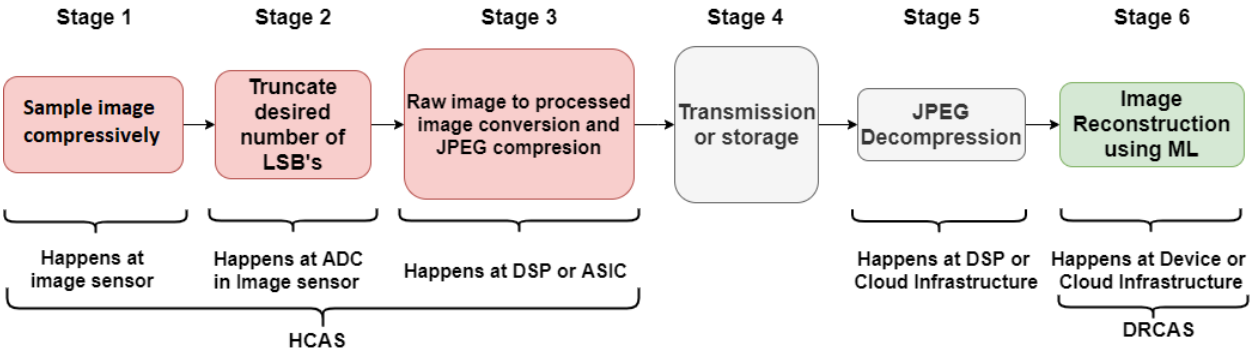


Figure 7.1: Proposed Image Acquisition Methodology: It consists of 6 stages. In the stages 1-3 (HCAS), an image gets compressed using downsampling, bit truncation and JPEG. Stage 4 represents transmission of image which can be a wireless medium or an on-chip bus or even storage. Stage 5 performs JPEG decompression and stage 6 consists of the proposed DRCAS to restore the desired image.

ever, the authors used A+ (Adjusted anchored neighborhood regression algorithm) [64] method for super-resolution instead of Neural Networks. Some works have also focused on denoising and compression artifact removal tasks [65, 66, 58]. To the best of our knowledge, this is the first work to investigate the DNN based image restoration considering the combination of downsampling, bit truncation and JPEG with intention of compression and power savings.

7.2 DRCAS

Table 7.1: Comparison of networks.

Network	Residual Blocks	Trainable Weights
This Work	6	$0.5M$
EDSR	32	$43M$

The entire methodology for image acquisition is shown in Fig. 7.1. We propose the DRCAS, denoting Deep Restoration network for hardware based Compressed Acquisition Scheme, to finish the task in Stage 6, with architecture shown in Fig. 7.2. DRCAS is inspired by the original ResNet architecture [3], EDSR architecture [4] and SRCNN architecture [55] with some differences. A

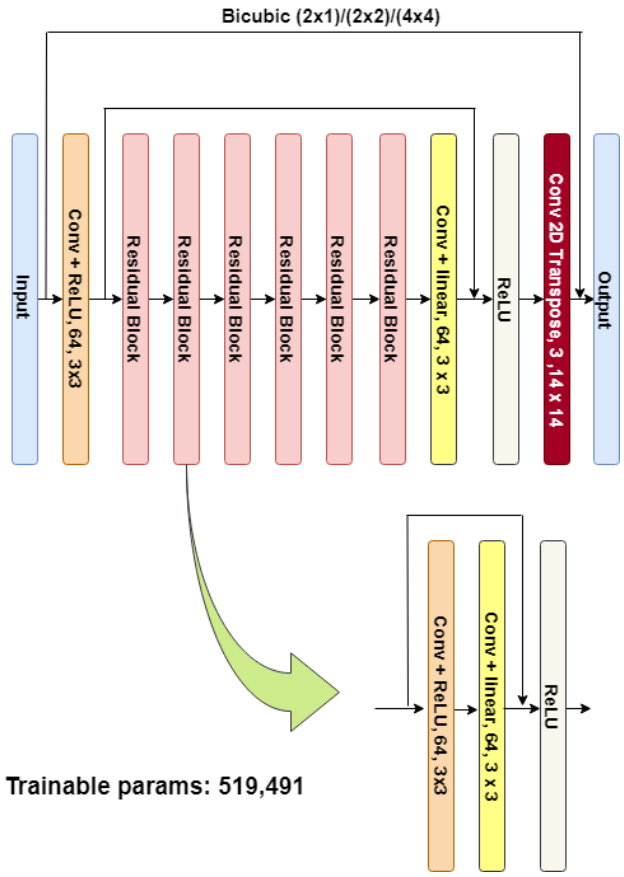


Figure 7.2: DRCAS network proposed in this work.

comparison of ResNet block is shown in Fig. 7.3. To start, the basic ResNet block used in this work uses ReLU (Rectified Linear Unit) layer in the end like original ResNet network. However it gets rid of Batch Normalization network as in EDSR Network. Also, unlike EDSR, our DRCAS avoids learning a complete image; it only learns the residual between the bicubic interpolated image and the actual image making the model much smaller in comparison. This is achieved by making a bypass connection between input and output using bicubic interpolation function as shown in Fig. 7.2. Since the residuals are mostly close to zero, the training is speed up and the model complexity gets significantly reduced.

We train a separate network for a given downsampling factor, bit truncation and JPEG Quality factor to restore the image quality and resolution. Thus there are 48 different training tasks (4 cases of JPEG Quality, 4 cases of bit truncation and 3 cases of downsampling). The hyper-parameters

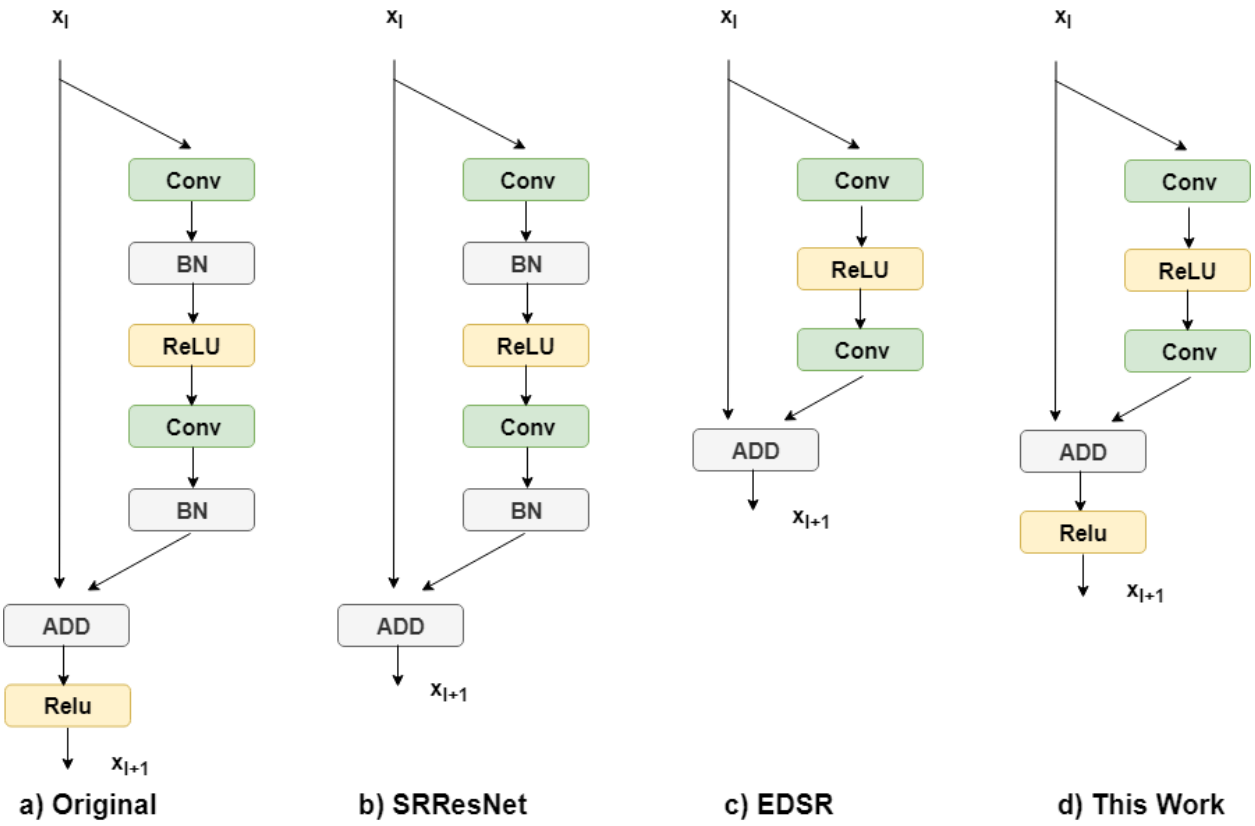


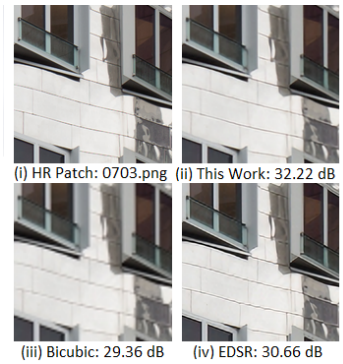
Figure 7.3: A comparison of basic ResNet blocks

are kept same for each training task.

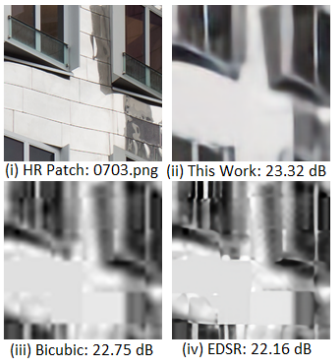
7.3 Experimental results

We use DIV2K dataset [67] for training and evaluation; DIV2K dataset is a high-quality (2K resolution) image dataset, which consists of 800 training images, 100 validation images and 100 testing images. Since the testing images are not made public, we use the last 100 images from training set (i.e. image 0701.png to 0800.png) as the testing set. The network uses patch size of 128×128 and $70,000$ training samples in a batch size of 64. The network is trained for 24 epochs.

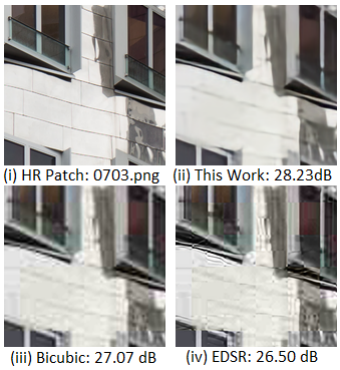
PSNR (Peak Signal to Noise Ratio) is used as a metric to measure reconstruction performance with original high resolution image as the baseline. We also compare the performance of our DRCAS with CS which is shown in Table 7.2 for 2×1 superresolution and a patch size of 128×128. We see that DRCAS is much better than CS reconstruction by atleast $.76\ dB$ and atmost $3\ dB$. Next



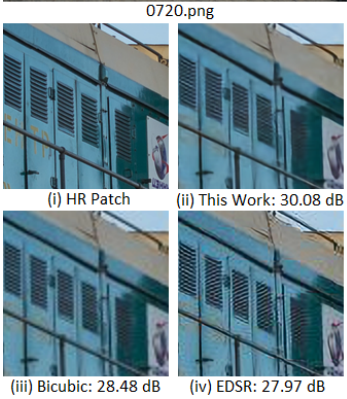
(a) 2×2 $S.R.$, $Q = 100$
(a) $B.W. = 8$



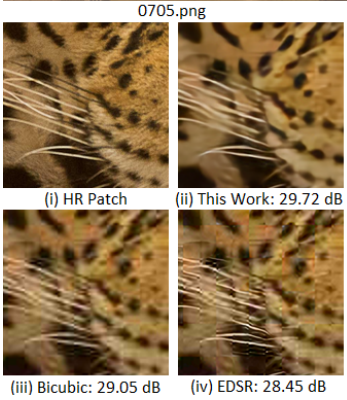
(b) 4×4 $S.R.$, $Q = 70$
(b) $B.W. = 5$



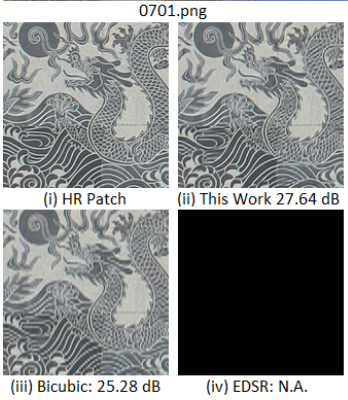
(c) 2×2 $S.R.$, $Q = 80$
(c) $B.W. = 6$



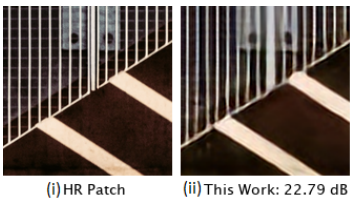
(d) 2×2 $S.R.$, $Q = 90$
$B.W. = 7$

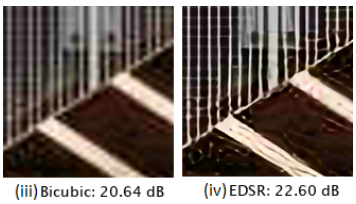


(e) 2×2 $S.R.$, $Q = 80$
$B.W. = 6$



(f) 2×1 $S.R.$, $Q = 100$
$B.W. = 8$





(g) 4×4 $S.R.$, $Q = 90$, $B.W. = 6$

Figure 7.4: Selected reconstructed images. For Fig. (f) EDSR result is not available as it is not designed for 2×1 super resolution.

Table 7.2: Comparison between ML and CS reconstruction performance. Format: ML | CS | Difference, in dB.$2 \times 1S.R.$ Patch Size = 128×128.

	$B.W. = 8$	$B.W. = 7$	$B.W. = 6$	$B.W. = 5$
Q 100	37.36 | 36.60 | 0.76	36.71 | 35.35 | 1.36	35.56 | 33.38 | 2.18	33.56 | 30.47 | 3.09
Q 90	35.65 | 34.37 | 1.28	34.23 | 32.80 | 1.43	32.56 | 30.80 | 1.76	30.47 | 28.28 | 2.19
Q 80	34.46 | 33.24 | 1.22	33.07 | 31.56 | 1.51	31.28 | 29.56 | 1.72	29.26 | 27.20 | 2.06
Q 70	33.82 | 32.38 | 1.44	32.21 | 30.79 | 1.42	30.38 | 28.68 | 1.70	28.11 | 26.17 | 1.94

we show the results for reconstruction using DRCAS in Table 7.3 for full sized testing dataset. One can see from the result that our DRCAS outperforms bicubic too. For testing purposes, this work also predicted the output of EDSR network for the downsampled images used in our work. As expected the performance of the EDSR network falls sharply as it is not trained to handle the noise due to bit truncation, averaging and JPEG. One can also see that the performance of EDSR gets worse than bicubic as the image quality degrades. It also serves as a proof that our DRCAS does train itself properly to handle the degradation induced by JPEG, bit truncation and binning. Some samples of reconstructed images including best case (Q = 100 and B.W. = 8) and worst case (Q = 70 and B.W. = 5) are shown in Fig. 7.4. One can see that bicubic interpolated images are more noisy and less sharper than the images generated by DRCAS and EDSR.

While the performance of the network proposed in this work might seems inferior to numbers reported in the EDSR paper [4], the focus in this work is to *perform on-sensor compression to reduce data traffic and save energy*. This is achieved by bit truncation and pixel binning both of which can be performed on commercially available image sensors. Almost all existing works of super-resolution use bicubic downsampling method which yields better image but cannot be performed on the image sensor. Thus bicubic downsampling fails to perform compression of raw data. Our proposed DRCAS is also simpler than EDSR network and a comparison is shown in Table 7.1. Pixel binning and bit truncation lead to significant reduction in raw data generated from image sensor. An analysis of compression of raw data is provided in Table 7.4. It is measured as

Table 7.3: Reconstruction results for DIV2K dataset (0701.png-0800.png). PSNR metric in dB. Bicub. refers to bicubic interpolation, EDSR refers to the EDSR network in paper [4] and B.W. referes to the bitwidth of the image.

Quality	*B.W.*	*This Work* 4×4	*BiCub.* 4×4	*EDSR* 4×4	*This Work* 2×2	*BiCub.* 2×2	*EDSR* 2×2	*This Work* 2×1	*BiCub.* 2×1
100	8	**27.91**	26.75	27.28	**32.74**	30.97	31.61	**34.96**	33.19
100	7	**27.80**	26.68	27.12	**32.43**	30.78	31.14	**34.58**	32.86
100	6	**27.50**	26.44	26.59	**31.71**	30.09	30.04	**33.60**	31.96
100	5	**26.59**	25.72	25.26	**29.97**	28.76	27.87	**31.30**	29.98
90	8	**27.21**	26.36	26.29	**31.39**	30.18	29.88	**33.41**	32.12
90	7	**26.58**	25.87	25.51	**30.32**	29.35	28.69	**32.27**	31.11
90	6	**25.76**	25.12	24.63	**29.03**	28.14	27.38	**30.72**	29.64
90	5	**24.61**	24.05	23.48	**27.74**	26.49	25.75	**28.62**	27.63
80	8	**26.60**	25.90	25.57	**30.42**	29.44	28.82	**32.37**	31.24
80	7	**25.84**	25.23	24.80	**29.23**	28.38	27.68	**31.01**	29.98
80	6	**24.93**	24.37	23.92	**27.88**	27.06	26.43	**29.40**	28.40
80	5	**23.76**	23.20	22.69	**26.13**	25.36	24.76	**27.33**	26.37
70	8	**26.18**	25.55	25.15	**29.76**	28.88	28.20	**31.68**	30.60
70	7	**25.39**	24.80	24.38	**28.53**	27.73	27.09	**30.25**	29.24
70	6	**24.43**	23.87	23.44	**27.13**	26.35	25.78	**28.56**	27.58
70	5	**23.16**	22.60	22.11	**25.37**	24.58	24.03	**26.52**	25.50

follows

$$Raw\ Data\ Compression = \frac{8 \cdot N - B}{8 \cdot N}, \tag{7.1}$$

where N represents number of pixels binned together and B represents bitwidth (B.W.) of pixel of downsampled image. One can achieve $50\% - 96\%$ reduction in raw data. Reduction in raw data means significant energy saving in downstream processing. One can achieve approximately proportional savings in energy for the same frequency of operation or one can employ DVFS to achieve quadratic scaling in energy reduction. Apart from savings in power, the system also becomes faster as there is less data to process. Reduction in bitwidth can also result in exponential reduction in power consumed at ADC (Chapter 2.2). Additionally, it can lead to more than proportional savings in energy in image processing circuits as LSB's switch more from one pixel to another than MSB in an image.

Table 7.4: Raw data Compression Results. B.W. refers to the bitwidth of image. Raw Compression does not depend on JPEG Quality factor Q.

	$B.W. = 8$	$B.W. = 7$	$B.W. = 6$	$B.W. = 5$
2×1	50%	56.25%	62.5%	68.75%
2×2	75%	78.12%	81.25%	84.37%
4×4	93.75%	94.53%	95.31%	96.09%

Table 7.5: Switching activity analysis of images. For DIV2K dataset (0701.png-0800.png).

$Bit\ Position$	$Swiching\ Activity\ (\alpha)$	$\%age\ of\ total$
$0 = LSB$	0.48	21.5
1	0.46	20.6
2	0.41	18.4
3	0.33	14.8
4	0.25	11.2
5	0.17	7.6
6	0.09	4.0
$7 = MSB$	0.04	1.8

Let us assume that image is being readout to an 8-bit wide data bus in column-wise fashion for each color channel. A table of switching activity measurement for such a case is shown in Table 7.5. One can see that the last three LSB's contribute roughly 60% of switching activity. This will lead to significant savings in dynamic power consumption as explained earlier Chapter 2.1. Reduction in raw data also leads to reduction in processed image size after JPEG compression. The results for this are shown in Table 7.6, which shows the size of the resulting image as a percentage of the size of the original high resolution image stored in lossless JPEG format. One can see that compressed image size ranges from 22.7% to less than 1% of the size of lossless image. An estimate of power savings due to compression of both raw and finished data is provided in Table

Table 7.6: Size Comparison. For DIV2K dataset (0701.png-0800.png). Measured as percentage with respect to original image in lossless JPEG format.

$Quality$	$Bitwidth$	2×1	2×2	4×4
100	8	22.7%	12.27%	3.48%
100	7	17.38%	9.41%	2.66%
100	6	12.88%	6.95%	1.92%
100	5	9.41%	5.11%	1.41%
90	8	7.77%	4.29%	1.21%
90	7	5.32%	3.07%	0.84%
90	6	3.68%	1.92%	0.57%
90	5	2.45%	1.27%	0.39%
80	8	5.32%	2.86%	0.82%
80	7	3.48%	1.88%	0.55%
80	6	2.25%	1.23%	0.37%
80	5	0.78%	0.80%	0.25%
70	8	4.09%	2.25%	0.65%
70	7	2.86%	1.47%	0.43%
70	6	1.76%	0.96%	0.29%
70	5	1.17%	0.61%	0.18%

7.7 and one can achieve 50-90% power savings depending on compression levels.

As mentioned before, this work does not take into account the energy spent in reconstruction of the image as the aim is to reduce the energy for acquisition. However we have provided an approximate analysis of energy using the number of MAC (Multiply and Accumulate) operations which is the basic unit of calculations [68] in CNNs. It is difficult to estimate the power/energy of data movement because it is highly architecture dependent [68]. An energy estimation for computations is provided in Table 7.8 using 16 bit FP MAC units proposed in [5].

The images can be reconstructed either on edge devices or on cloud, and an example of edge

Table 7.7: Estimated power savings for the process of image acquisition. DL stands for our proposed acquisition using Deep Learning based method and C.R. stands for on-chip Compression Ratio.

Operation	**Design 1 (mW) (Ref. [36])**	**Design 2 (mW) (Ref. [43])**	**DL for Design 1 (mW)** $C.R.: 68.75\% - 25\%$	**DL for Design 2 (mW)** $C.R.: 68.75\% - 25\%$
I/O	27	70	$1.08 - 13.5$	$2.8 - 35$
ADC	60	209	$2.4 - 30$	$8.4 - 104.5$
Pixel	1.8	23	1.8	23
Other	4.2	20	4.2	20
JPEG (Ref. [47])	13.18	386.3	$0.52 - 6.6$	$15.4 - 193.2$
Total	106.18	708.3	$10.0 - 56.1$	$69.6 - 375.6$
Power Savings	0%	0%	$90.6\% - 47.1\%$	$90.1\% - 47\%$

Table 7.8: Approximate energy estimation for computation in DRCAS.The energy for each MAC operation is obtained from [5]. Each MAC consumes 1.35×10^{-12} J of energy. Final image resolution is assumed to be 1024×1024. Frequency can be calculated depending upon frame rate.

$Layer$	2×1	2×2	4×4
$Layer\ 1$	$905\ M$	$453\ M$	$113\ M$
$Each\ Res.\ Block$	$19,327\ M$	$9,663\ M$	$2416\ M$
$Last\ Layer$	$39,460\ M$	$39,460\ M$	$39,460\ M$
$Total\ Comp.$	$165,990.5\ M$	$102,722.5\ M$	$55,277\ M$
$Total\ Energy$	$0.224\ J$	$0.138\ J$	$0.074\ J$

device can be SmartTV. One can reduce the image/video data transmission bandwidth. The image/video can then be reconstructed using the GPU available in smart TVs. For the case of smartphones, image can be acquired in low resolution mode and can be reconstructed back in the cloud. Since the images and photos are generally uploaded to cloud storage nowadays, reconstruction in cloud is technically feasible. For the case of drones transmitting video surveillance footage, the compression can reduce the power consumption in drone. It can also make transmission feasible

in noisy environment or over longer distance because small image size offers an opportunity to aggressively encode the message packet with ECC.

7.4 Discussion

This chapter establishes the use of DNN for data compression in the entire imaging pipeline. Using DL we can perform much more aggressive raw data compression which was not possible using CS because DL achieves much better reconstruction results than CS. It is the performance of reconstruction which determines how much raw data compression one can achieve. The proposed DRCAS network performs the task of superresolution as well as artifact removal and performs better than the baseline methods used in this chapter. One can see that the DL based generated images are much sharper than the baseline methods used in this chapter. Apart from downsampling by averaging, we also measure the switching activity reduction due to bit truncation. As explained earlier, flipping bits contribute to dynamic power consumption (Section 2.1.1). One can achieve significant savings in switching activity reduction over bus wires inside an SoC by getting rid of few LSB's since LSB's flip the most and contain the least amount of information. Thus is makes DL a good candidate for our proposed imaging pipeline. It helps one achieve huge data compression (see Tables 7.6, 7.4), both raw data and finished image data, resulting in significant savings in power during acquisition. Motivated by the superior performance of DL in terns of superresolution and artifact removal, we explore the use of DL to improve HDR imaging in the next chapter.

8. SUPERRESOLUTION IN HDR IMAGING

Motivated by the performance of DL for superresolution and artifact removal in previous chapter, we propose the use of DL networks in HDR imaging. One of the issues in HDR imaging is that it requires multiple LDR images of the scene. This increases the amount of data linearly by factor of the number of LDR images as well as the total exposure time. The total exposure time generally increases by a factor of 2 for each LDR image. This presents serious issues as the LDR images need to be aligned properly and during the exposure time any motion or handshake blur will deteriorate the quality of the final HDR image. We utilize the DRCAS network for superresolution and artifact removal introduced in the previous chapter for reducing the amount of data required for HDR imaging and as well as total exposure time.

8.1 Related Work

There are multiple works related to HDR available in literature. Some of the works like Ref. [69, 70] focus on correcting the motion of objects in HDR images. There are some other works like Ref. [71, 72, 73] focus on HDR image through a singe LDR image. These networks work by generating synthetic images of different exposures or by estimating the missing information due to over-exposure or under-exposure. These methods generally require properly exposed images so that missing information is not substantial. However, our method proposed in this chapter is more system specific. Our idea is to acquire a single image using by exposing each pixel in a block of 4 pixels to a different exposure value (see Fig. 8.1). Thus we get 4 different downsampled images of 1/4 size each as compared to full size image. Thus our work in effect generates only a single image and thus it can act as an input to the existing works described above while achieving raw data compression as explained in previous chapters.

8.2 Simulation and Experiments

In this experiment we change the downsampling method from averaged binning to decimation. Lets assume a Bayer RGGB filter. Our idea is to choose alternate RGGB blocks along rows and

columns of image. Thus it leads to decimation by 4. This scheme is shown in Fig. 8.1. While decimation results in poorer quality than averaging, averaging cannot be implemented here as we need 4 images at different exposure times. Averaging can be implemented if we need only 1 image for a given exposure time. A comparison between performance of averaging and decimation for 2×2 superresolution for 128×128 sized patches from DIV2K dataset used in previous chapter is provided in Table 8.1.

Table 8.1: Comparison between Averaging and Decimation.128×128 sized patches from DIV2K dataset.

S.R.	Averaging (dB)	Decimation (dB)
2×2	35.46	34.36

Decimation is roughly 1 dB poorer in performance but it allows us to capture 4 images in the same frame. This is expected as decimation is more noisy than averaging. These pixels are marked as 1,2,3 and 4 in Fig. 8.1. Each of the 4 pixel blocks will have different exposure levels. In this experiment we assume that each image is separated by 2 stops in exposure. Thus instead of waiting for 4 sequence of frames, we are trading off the resolution to get 4 frames simultaneously. These 4 LDR images will then be used to construct an HDR image. As expected, the HDR image will also have 4 times less resolution. This is where our DRCAS algorithm introduced in previous section will be used. A simple schematic of this methodology is shown in Fig. 8.2. For the purpose of super-resolution we retrained the DRCAS algorithm using DIV2K dataset. Unlike the methodology in previous chapters, we did not use any bit truncation and lossy compression like JPEG while training. This was done to just demonstrate this concept. The lossy compression can be used to perform additional data compression and its affect on reconstruction performance can be studied. For HDR image dataset we use the 'HDR Photographic Survey' dataset by Mark Fairchild. We used a set of 41 images from this dataset. Each HDR image has 9 LDR images generally separated by one stop in exposure. Since we are using only 4 LDR images to form HDR

image, we choose images separated apart by 2 stops in exposure. This is just for demonstration purpose only. One can choose as many images as one wants. A sample image of reconstruction is shown in Fig. 8.3. We compare the construction performance by using full resolution HDR images constructed from full resolution LDR images as baseline. For construction HDR images we use the HDR function provided by MatLab as baseline. We compare DRCAS restoration with Bicubic interpolation method. DRCAS beats bicubic by roughly 3dB. Simultaneous capture of LDR also helps to improve total exposure time. Lets assume that the base exposure time is x seconds. Since our images are separated by 2 stops in exposure -

$$sequential\ exposure = x + 4x + 16 + 64x = 85x \tag{8.1}$$

$$simultaneous\ exposure = 64x \tag{8.2}$$

$$improvement = 100 \times (21x/85x)\% = 25.9\% \tag{8.3}$$

Thus we see that we achieve rougly 26% improvement in total exposure time apart from 4x compression in raw data

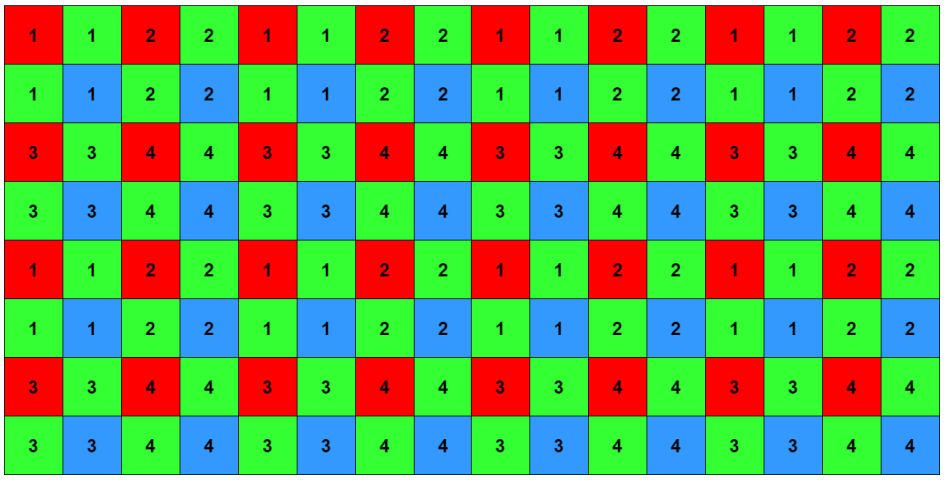
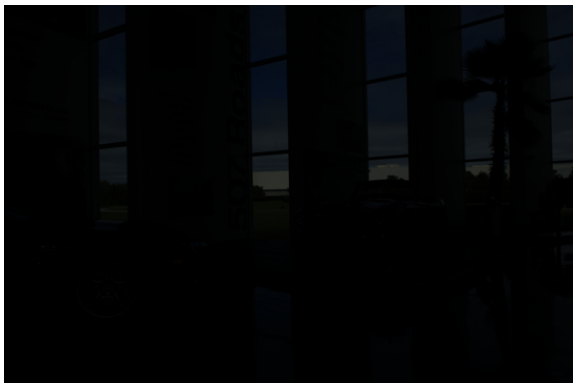


Figure 8.1: Proposed sampling using decimation.

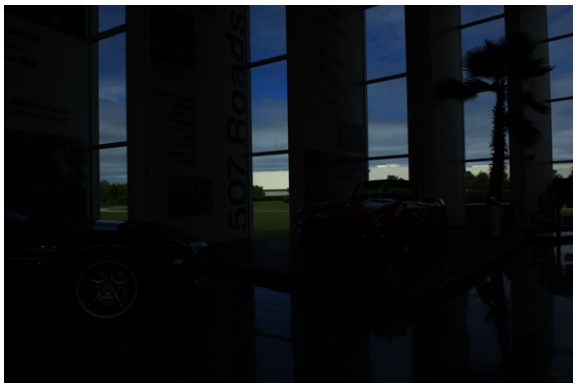
Figure 8.2: Proposed system methodology.

8.3 Discussion

In this chapter we saw an application of our DRCAS network proposed in previous chapter for HDR imaging. It helps us to reduce raw data by 4X and also leads to 26% improvement in total exposure time. While we use only 4 LDR images, we will get a better image than a full resolution single exposure image. These 4 images can then be utilized by single shot HDR imaging techniques mentioned in Section 8.1. Thus our proposed acquisition pipeline augments the existing imaging system without much modification to existing components of imaging system while achieving raw data compression. New techniques, in particular single shot HDR imaging, can benefit from proposed work in this chapter while achieving power and performance benefits due to reduction in raw data.



(a) $Exposure\ time\ =\ 3.1\ ms$



(b) $Exposure\ time\ =\ 12.5\ ms$

(d) $Exposure\ time\ =\ 50\ ms$

(e) $Exposure\ time\ =\ 200\ ms$

(d) $MatLab\ HDR\ Reconstruction$

(e) $Proposed\ Method.\ PSNR\ =\ 43.13\ dB$

Figure 8.3: Reconstruction Example

9. CONCLUSION

This work started with the goal to perform data compression in the entire image acquisition pipeline. The motivation behind this objective was to achieve system efficiency and enable designers to meet power and performance criteria in the age of slowing Moore's law. On top of slowing device scaling, the demand from consumers for high resolution videos and pictures is growing day by day which only serves to worsen this problem. The increase in demand from consumers means more data and higher processing speed requirement from an imaging system designers perspective. That is precisely the reason why designers tend to prefer advanced technology nodes which happen to be faster and less power consuming. To add to the problems the newer technology nodes in VLSI are very expensive [7] and only high volume of production can make it profitable. Thus even if Moore's law is made to kept going slowly somehow, the cost associated with the newer technology nodes would be difficult to justify for a lot of applications and consumer electronic goods.

In view of slowing down Moore's Law as well as increasing cost of advanced technology nodes, this thesis focused its efforts on algorithmic and system design innovations to reduce data by the means of compression at the source thereby reducing dataflow in entire system pipeline. This directly translates to atleast proportional savings in power and performance improvement. Looking from another perspective, this compression means that existing technology nodes can be utilized to process more data resulting in increase in shelf life of the technology node and cost savings. Our proposed HCAS acquisition method results in compression throughout the image acquisition pipeline. However, compression implies that one has to do reconstruction when the image needs to be viewed or processed/operated by some software and this process of reconstruction can be quite energy consuming resulting in net energy loss. To handle this issue, we take advantage of the fact that a lot of mobile devices like our smartphones etc. use cloud as storage and we can run the reconstruction on cloud thereby saving power on our edge device like smartphone. Power consumption in the cloud is not an issue and cloud computing lowers the cost of computation by

exploiting the high volume and scale of operation. Thus they can also afford newer and expensive technologies. On the otherhand, video streaming services can also take advantage of our proposed method to save bandwidth of transmission as reconstruction of video in Smart-TV is not an issue because it has ample access to power. Similar is the case with a drone transmitting a surveillance footage to a base station. In the case of drones, reduction in data can be used to encode the data packet more aggressively using ECC to transmit over noisy environments or over longer distances. Thus in all these cases, our proposed image sensing pipeline helps in multitude of ways, the prime being slowing Moore's law and expensive technology nodes, justifying the need of our proposed system.

To perform compression this work downsampled the image using CS matrices presented in Chapter 3 through the use of novel pixel design and also proposed to truncate the LSBs of the resulting image. The proposed CS matrices behave superior as compared to traditional Gaussian random matrices for compression level of 50% and are also easy to implement. Our CS sampling does not alter distribution of image like Gaussian random matrices making application of JPEG feasible. However, this proposed pixel will only operate in CS configuration making it inflexible. Nevertheless, fixed configuration systems have definite advantages like they are more simpler. As seen in Chapter 3, the proposed pixel reduce wiring requirement related to addressing, address decoder, power supply, reset and global shutter control wiring in image sensor by half. This results in improvement in fill factor and reduction in transistor count per pixel by 1 for global shutter. Looking from another perspective, the savings in area can also be utilized to increase resolution of image sensors. The proposed pixel achieve raw data compression between 25-69% resulting in power savings of 23-65% approximately.

To address the issue of inflexibility we proposed to use pixel binning feature of image sensors in Chapter 4 as one may require to operate the imaging system in non-CS mode as well. This is because reconstruction process is not exact and results in some noise in the image and depending upon application this noise might not be suitable. The proposed pixel binning for CS can be controlled just by the manipulation of control signals of pixels which are generally exposed to system

programmer. User can also control the type of sampling matrix for example $\Phi_{3/4B}$ or $\Phi_{1/2B}$ for raw data compression of 15.5 - 62.5% and 43.75-75% respectively. This will allow user to control the quality of resulting image as maximum performance of $\Phi_{3/4B}$ is around 37.6 dB and for $\Phi_{1/2B}$ it is 32.45 dB. Going further, we also studied the performance of our proposed system using CS reconstruction algorithm in the presence of noise and non-linearities present in actual system in Chapter 5. For this purpose we used ISET toolbox. We showed that even with noise and lighting intensity variation of 10x, the reconstruction generated quite good quality perceptual images. Thus it makes CS reconstruction suitable for actual systems too. Motivated by this we came up with a fixed point implementation of reconstruction algorithm in Chapter 6 in case one wants to implement an entire system on ASIC/FPGA. We show that using the simplified CS matrices based on averaging and by avoiding the overlap between successive CS samples, we can reduce the sampling process complexity by N as shown in Chapter 6. Thus having a complicated sampling matrix not only makes sampling difficult to implement but also makes reconstruction more complex. The entire system was simulated in ISET making our complete system in fixed point implementation.

However as seen from Chapter 3 the performance of CS reconstruction falls below 30 dB for compression ratio of sampling matrix greater than 2 (i.e. sampling rate of 0.5). This limits the compression one can achieve at raw data level. To achieve even more compression at sensor level and inspired by the success of deep learning in Computer Vision tasks, we studied Deep Learning based networks for reconstruction in Chapter 7. The superior performance of our DL algorithm resulted in raw data compression in the range 50-96% resulting in PSNR of reconstruction between 35-23 dB respectively. Because of compression in raw data, the final image size also gets reduced significantly. As compared to lossless JPEG, we obtain image sizes which is 22%-1% of original image in lossless JPEG format for raw data compression levels of 50-96% and reconstruction performance of 35-23 dB respectively. Thus according to the requirements of final image quality user can tradeoff reconstruction performance with quality using Tables 7.3, 7.4 and 7.6.

In the same chapter we also measured the switching activity of images getting transmitted over a bus. From Table 7.5 we can see that the 3 LSBs contribute to over 60% of switching activity.

Considering that modern NoCs consume upto 20-30% power, LSB truncation can lead to upto 60% reduction in dynamic power consumption over transmission.

Lastly in Chapter 8 we extended the use of our DRCAS reconstruction algorithm to HDR imaging. From Chapter 7 we saw that 2×2 super-resolution (i.e. reduction in total resolution by a factor of 4) generates good perceptual results. This fact was utilized to generate 4 images at $1/4$ of total resolution but at different exposure levels. The simultaneous generation of 4 images helped us to cut total exposure time by 26% while achieving a raw data reduction of 4x.

Thus our proposed imaging pipeline in Fig. 1.1 esp. with DRCAS reconstruction has a very good potential to achieve compression in the entire imaging pipeline from source to destination. It helps to solve the issues discussed in Chapter 1 effectively because of following main features -

- It achieves compression in the entire imaging pipeline. It reduces raw data by atleast 4x and still maintains good reconstruction quality (above 30dB) perceptually.
- It utilizes JPEG too after raw data compression. For 4x reduction in raw data it achieves more than 10x compression of image when compared to lossless JPEG version while maintain reconstruction above 30dB for many quality factor cases.
- For NoC based systems one can easily achieve switching activity reduction upto 60% while maintaining reconstruction performance above 30 dB.
- The acquisition methodology can be very easily implemented in existing imaging system as it uses very simplistic and hardware friendly techniques.
- It is programmable and user can control the amount of compression and hence final reconstruction quality at runtime.

As a part of future work, new DL based networks can be explored to improve performance. One can also look into low power on-chip DL reconstruction too [68]. Another interesting area can be designing low power custom machine vision systems. With rapid advances in the field of

DL as well as huge demand of products and services related to it, all these areas carry a lot of potential and should be explored extensively.

REFERENCES

[1] G. Köklü, R. Etienne-Cummings, Y. Leblebici, G. De Micheli, and S. Carrara, "Characterization of standard cmos compatible photodiodes and pixels for lab-on-chip devices," in *2013 IEEE International Symposium on Circuits and Systems (ISCAS2013)*, pp. 1075–1078, IEEE, 2013.

[2] J. E. Farrell, P. B. Catrysse, and B. A. Wandell, "Digital camera simulation," *Appl. Opt.*, vol. 51, pp. A80–A90, Feb 2012.

[3] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

[4] B. Lim, S. Son, H. Kim, S. Nah, and K. Mu Lee, "Enhanced deep residual networks for single image super-resolution," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pp. 136–144, 2017.

[5] H. Zhang, J. He, and S.-B. Ko, "Efficient posit multiply-accumulate unit generator for deep learning applications," in *2019 IEEE International Symposium on Circuits and Systems (ISCAS)*, pp. 1–5, IEEE, 2019.

[6] "Cisco visual networking index: Global mobile data traffic forecast update, 2016–2021 white paper,"

[7] M. Lapedus, "Big trouble at 3nm," June 2018.

[8] M. B. Taylor, "Is dark silicon useful? harnessing the four horsemen of the coming dark silicon apocalypse," in *DAC Design Automation Conference 2012*, pp. 1131–1136, IEEE, 2012.

[9] D. Zhu, Y. Li, and L. Chen, "On trade-off between static and dynamic power consumption in noc power gating," in *2019 IEEE/ACM International Symposium on Low Power Electronics and Design (ISLPED)*, pp. 1–6, IEEE, 2019.

[10] E. R. Fossum *et al.*, "Cmos image sensors: electronic camera-on-a-chip," *IEEE transactions on electron devices*, vol. 44, no. 10, pp. 1689–1698, 1997.

[11] S. Lauxtermann, A. Lee, J. Stevens, and A. Joshi, "Comparison of global shutter pixels for cmos image sensors," in *2007 International Image Sensor Workshop*, 2007.

[12] K. Yasutomi, S. Itoh, S. Kawahito, and T. Tamura, "Two-stage charge transfer pixel using pinned diodes for low-noise global shutter imaging," in *IISW*, 2009.

[13] K. Murari, R. Etienne-Cummings, N. Thakor, and G. Cauwenberghs, "Which photodiode to use: A comparison of cmos-compatible structures," *IEEE sensors journal*, vol. 9, no. 7, pp. 752–760, 2009.

[14] N. Katic, M. H. Kamal, A. Schmid, P. Vandergheynst, and Y. Leblebici, "Compressive image acquisition in modern cmos ic design," *International Journal of Circuit Theory and Applications*, vol. 43, no. 6, pp. 722–741, 2015.

[15] Z. Zhou, B. Pain, and E. R. Fossum, "Frame-transfer cmos active pixel sensor with pixel binning," *IEEE Transactions on electron devices*, vol. 44, no. 10, pp. 1764–1768, 1997.

[16] H.-Y. Huang, P. A. Conge, and L.-W. Huang, "Cmos image sensor binning circuit for low-light imaging," in *Industrial Electronics and Applications (ISIEA), 2011 IEEE Symposium on*, pp. 586–589, IEEE, 2011.

[17] J. Cho, J. Choi, S.-J. Kim, J. Shin, S. Park, J. D. Kim, and E. Yoon, "A 5.9 μm-pixel 2d/3d image sensor with background suppression over 100klx," in *VLSI Circuits (VLSIC), 2013 Symposium on*, pp. C6–C7, IEEE, 2013.

[18] O. Kumagai, A. Niwa, K. Hanzawa, H. Kato, S. Futami, T. Ohyama, T. Imoto, M. Nakamizo, H. Murakami, T. Nishino, *et al.*, "A 1/4-inch 3.9 mpixel low-power event-driven back-illuminated stacked cmos image sensor," in *Solid-State Circuits Conference-(ISSCC), 2018 IEEE International*, pp. 86–88, IEEE, 2018.

[19] S. Yoshihara, Y. Nitta, M. Kikuchi, K. Koseki, Y. Ito, Y. Inada, S. Kuramochi, H. Wakabayashi, M. Okano, H. Kuriyama, *et al.*, "A 1/1.8-inch 6.4 mpixel 60 frames/s cmos image

sensor with seamless mode change," *IEEE Journal of Solid-State Circuits*, vol. 41, no. 12, pp. 2998–3006, 2006.

[20] X. Li, X. Lan, M. Yang, J. Xue, and N. Zheng, "Efficient lossy compression for compressive sensing acquisition of images in compressive sensing imaging systems," *Sensors*, vol. 14, no. 12, pp. 23398–23418, 2014.

[21] M. A. Davenport, M. F. Duarte, Y. C. Eldar, and G. Kutyniok, "Introduction to compressed sensing," *Preprint*, vol. 93, no. 1, p. 2, 2011.

[22] E. J. Candes, "The restricted isometry property and its implications for compressed sensing," *Comptes Rendus Mathematique*, vol. 346, no. 9, pp. 589–592, 2008.

[23] E. Candes and J. Romberg, "Sparsity and incoherence in compressive sampling," *Inverse problems*, vol. 23, no. 3, p. 969, 2007.

[24] E. J. Candès and M. B. Wakin, "An introduction to compressive sampling," *IEEE signal processing magazine*, vol. 25, no. 2, pp. 21–30, 2008.

[25] D. L. Donoho, "Compressed sensing," *IEEE Transactions on information theory*, vol. 52, no. 4, pp. 1289–1306, 2006.

[26] E. J. Candes, J. K. Romberg, and T. Tao, "Stable signal recovery from incomplete and inaccurate measurements," *Communications on pure and applied mathematics*, vol. 59, no. 8, pp. 1207–1223, 2006.

[27] L. Gan, T. T. Do, and T. D. Tran, "Fast compressive imaging using scrambled block hadamard ensemble," in *Signal Processing Conference, 2008 16th European*, pp. 1–5, IEEE, 2008.

[28] Z. He, T. Ogawa, and M. Haseyama, "The simplest measurement matrix for compressed sensing of natural images," in *2010 IEEE International Conference on Image Processing*, pp. 4301–4304, IEEE, 2010.

[29] L. Gan, "Block compressed sensing of natural images," in *2007 15th International conference on digital signal processing*, pp. 403–406, IEEE, 2007.

[30] I. W. Selesnick, R. G. Baraniuk, and N. C. Kingsbury, "The dual-tree complex wavelet transform," *IEEE signal processing magazine*, vol. 22, no. 6, pp. 123–151, 2005.

[31] J. Yang, Y. Wang, W. Xu, and Q. Dai, "Image coding using dual-tree discrete wavelet transform," *IEEE Transactions on Image Processing*, vol. 17, no. 9, pp. 1555–1569, 2008.

[32] S. Mun and J. E. Fowler, "Block compressed sensing of images using directional transforms," in *2009 16th IEEE international conference on image processing (ICIP)*, pp. 3021–3024, IEEE, 2009.

[33] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*. MIT press, 2016.

[34] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[35] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1–9, 2015.

[36] Y. Oike and A. El Gamal, "Cmos image sensor with per-column σ δ adc and programmable compressed sensing," *IEEE Journal of Solid-State Circuits*, vol. 48, no. 1, pp. 318–328, 2013.

[37] M. Dadkhah, M. J. Deen, and S. Shirani, "Cmos image sensor with area-efficient block-based compressive sensing," *IEEE Sensors Journal*, vol. 15, no. 7, pp. 3699–3710, 2015.

[38] R. A. DeVore, "Deterministic constructions of compressed sensing matrices," *Journal of complexity*, vol. 23, no. 4-6, pp. 918–925, 2007.

[39] R. R. Naidu, P. Jampana, and C. S. Sastry, "Deterministic compressed sensing matrices: Construction via euler squares and applications.," *IEEE Trans. Signal Processing*, vol. 64, no. 14, pp. 3566–3575, 2016.

[40] N. Kulkarni, P. Nagesh, R. Gowda, and B. Li, "Understanding compressive sensing and sparse representation-based super-resolution," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 22, no. 5, pp. 778–789, 2012.

[41] P. Sen and S. Darabi, "Compressive image super-resolution," in *2009 Conference Record of the Forty-Third Asilomar Conference on Signals, Systems and Computers*, pp. 1235–1242, IEEE, 2009.

[42] M. Yip and A. P. Chandrakasan, "A resolution-reconfigurable 5-to-10-bit 0.4-to-1 v power scalable sar adc for sensor applications," *IEEE Journal of Solid-State Circuits*, vol. 48, no. 6, pp. 1453–1464, 2013.

[43] M.-S. Shin, J.-B. Kim, M.-K. Kim, Y.-R. Jo, and O.-K. Kwon, "A 1.92-megapixel cmos image sensor with column-parallel low-power and area-efficient sa-adcs," *IEEE Transactions on Electron Devices*, vol. 59, no. 6, pp. 1693–1700, 2012.

[44] S. Sukegawa, T. Umebayashi, T. Nakajima, H. Kawanobe, K. Koseki, I. Hirota, T. Haruta, M. Kasai, K. Fukumoto, T. Wakano, *et al.*, "A 1/4-inch 8mpixel back-illuminated stacked cmos image sensor," in *2013 IEEE International Solid-State Circuits Conference Digest of Technical Papers*, pp. 484–485, IEEE, 2013.

[45] B. E. Bayer, "Color imaging array," July 20 1976. US Patent 3,971,065.

[46] M. Innocent, "General introduction to cmos image sensors,"

[47] Y. Shichao, H. Zhizhong, and C. Xin, "A scalable multi-pipeline jpeg encoding architecture," in *Microelectronics (ICM), 2016 28th International Conference on*, pp. 369–372, IEEE, 2016.

[48] S. Leitner, H. Wang, and S. Tragoudas, "Compressive image sensor technique with sparse measurement matrix," in *System-on-Chip Conference (SOCC), 2016 29th IEEE International*, pp. 223–228, IEEE, 2016.

[49] S. Leitner, H. Wang, and S. Tragoudas, "Design of scalable hardware-efficient compressive sensing image sensors," *IEEE Sensors Journal*, vol. 18, no. 2, pp. 641–651, 2018.

[50] P. S. Gupta and G. S. Choi, "Image acquisition system using on sensor compressed sampling technique," *Journal of Electronic Imaging*, vol. 27, no. 1, p. 013019, 2018.

[51] P. S. Gupta and G. S. Choi, "Programmable compressed sensing using simple deterministic sensing matrices," in *Optoelectronic Imaging and Multimedia Technology V*, vol. 10817, p. 108170C, International Society for Optics and Photonics, 2018.

[52] P. Sen and S. Darabi, "Compressive rendering: A rendering application of compressed sensing," *IEEE Transactions on Visualization and Computer Graphics*, vol. 17, no. 4, pp. 487–499, 2011.

[53] C. Dong, C. C. Loy, K. He, and X. Tang, "Image super-resolution using deep convolutional networks," *IEEE transactions on pattern analysis and machine intelligence*, vol. 38, no. 2, pp. 295–307, 2016.

[54] J. Kim, J. Kwon Lee, and K. Mu Lee, "Accurate image super-resolution using very deep convolutional networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1646–1654, 2016.

[55] C. Ledig, L. Theis, F. Huszár, J. Caballero, A. Cunningham, A. Acosta, A. Aitken, A. Tejani, J. Totz, Z. Wang, *et al.*, "Photo-realistic single image super-resolution using a generative adversarial network," *arXiv preprint*, 2017.

[56] Z. Wang, J. Chen, and S. C. Hoi, "Deep learning for image super-resolution: A survey," *arXiv preprint arXiv:1902.06068*, 2019.

[57] W. Yang, X. Zhang, Y. Tian, W. Wang, and J.-H. Xue, "Deep learning for single image super-resolution: A brief review," *arXiv preprint arXiv:1808.03344*, 2018.

[58] 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.

[59] R. Timofte, S. Gu, J. Wu, and L. Van Gool, "Ntire 2018 challenge on single image super-resolution: methods and results," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pp. 852–863, 2018.

[60] J. Liu, W. Sun, and Y. Liu, "Bit-depth enhancement via convolutional neural network," in *International Forum on Digital TV and Wireless Multimedia Communications*, pp. 255–264, Springer, 2017.

[61] J. Liu, W. Sun, Y. Su, P. Jing, and X. Yang, "Be-calf: bit-depth enhancement by concatenating all level features of dnn," *IEEE Transactions on Image Processing*, 2019.

[62] Y. Su, W. Sun, J. Liu, G. Zhai, and P. Jing, "Photo-realistic image bit-depth enhancement via residual transposed convolutional neural network," *Neurocomputing*, 2019.

[63] S. Umeda, H. Watanabe, T. Ikai, T. Hashimoto, T. Chujoh, and N. Ito, "Joint super-resolution and bit depth extension by dnn," in *International Workshop on Advanced Image Technology (IWAIT) 2019*, vol. 11049, p. 1104925, International Society for Optics and Photonics, 2019.

[64] R. Timofte, V. De Smet, and L. Van Gool, "A+: Adjusted anchored neighborhood regression for fast super-resolution," in *Asian conference on computer vision*, pp. 111–126, Springer, 2014.

[65] X. Mao, C. Shen, and Y.-B. Yang, "Image restoration using very deep convolutional encoder-decoder networks with symmetric skip connections," in *Advances in neural information processing systems*, pp. 2802–2810, 2016.

[66] K. Yu, C. Dong, C. C. Loy, and X. Tang, "Deep convolution networks for compression artifacts reduction," *arXiv preprint arXiv:1608.02778*, 2016.

[67] E. Agustsson and R. Timofte, "Ntire 2017 challenge on single image super-resolution: Dataset and study," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, July 2017.

[68] V. Sze, Y.-H. Chen, T.-J. Yang, and J. S. Emer, "Efficient processing of deep neural networks: A tutorial and survey," *Proceedings of the IEEE*, vol. 105, no. 12, pp. 2295–2329, 2017.

[69] N. K. Kalantari and R. Ramamoorthi, "Deep high dynamic range imaging of dynamic scenes.," *ACM Trans. Graph.*, vol. 36, no. 4, pp. 144–1, 2017.

[70] S. Wu, J. Xu, Y.-W. Tai, and C.-K. Tang, "Deep high dynamic range imaging with large foreground motions," in *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 117–132, 2018.

[71] B. Masia, S. Agustin, R. W. Fleming, O. Sorkine, and D. Gutierrez, "Evaluation of reverse tone mapping through varying exposure conditions," *ACM transactions on graphics (TOG)*, vol. 28, no. 5, pp. 1–8, 2009.

[72] Y. Endo, Y. Kanamori, and J. Mitani, "Deep reverse tone mapping," *ACM Transactions on Graphics (Proc. of SIGGRAPH ASIA 2017)*, vol. 36, Nov. 2017.

[73] G. Eilertsen, J. Kronander, G. Denes, R. Mantiuk, and J. Unger, "Hdr image reconstruction from a single exposure using deep cnns," *ACM Transactions on Graphics (TOG)*, vol. 36, no. 6, 2017.