Modelling and calibration of logarithmic CMOS image sensors

Dileepan Joseph, Keble College
DPhil in Engineering Science
University of Oxford

September 30, 2002

Abstract

Logarithmic CMOS image sensors capture high dynamic range scenes without saturation or loss of perceptible detail but problems exist with image quality. This thesis develops and applies methods of modelling and calibration to understand and improve the fixed pattern noise (FPN) and colour rendition of logarithmic imagers. Chapter 1 compares CCD and CMOS image sensors and, within the latter category, compares linear and logarithmic pixel designs. Chapter 2 reviews the literature on multilinear algebra, unifying and extending approaches for analytic and numeric manipulation of multi-index arrays, which are the generalisation of scalars, vectors and matrices. Chapter 3 defines and solves the problem of multilinear regression with linear constraints for the calibration of a sensor array, permitting models with linear relationships of parameters across the array. Chapter 4 develops a steady state model for the digital response of a logarithmic pixel to light stimulus and uses it to characterise and correct FPN, which proves to depend nonlinearly on illuminance, by calibration of simulated and experimental data. Chapter 5 models the transient response of logarithmic imagers, for typical source follower readout circuits, and shows with simulation and experiment how transient operation and design may cause FPN, which may partially be corrected by a steady state calibration. Chapter 6 extends the steady state model of the image sensor to examine and reduce the dependence of FPN on temperature, comparing in simulation and experiment methods of calibration that use pixel responses under both dark and light conditions. Chapter 7 describes the calibration of pixel responses in terms of a standard colour space, extending previous models suitable for FPN correction but unsuitable for colour rendition, and shows that colour rendition of a Fuga 15RGB logarithmic camera competes with that of conventional digital cameras. Finally, Chapter 8 discusses and summarises the main results of this thesis and outlines future theoretical, simulation and experimental work.

Acknowledgements

People

Steve Collins, Lionel Tarassenko, Satoshi Aoyama, Sunay Shah, Alistair McEwan, Gari Clifford, Simukai Utete, Mayela Zamora, Jan Minchington, Stephen Payne, Neil Townsend, Paul Hayton, Christopher Rabson, other members of the SPANN and MCAD research groups, other friends and my family all helped me to realise my doctorate.[1]

I was one of Steve's first DPhil students. Steve was readily available when needed, quick to see what mattered and what didn't to solve an engineering problem, insightful in all things hardware and an observant reader. Without his critical eye, my thesis would certainly have been less accurate and readable. I remember thinking, when I first started working with him, that it would be a great achievement to impress someone who is not easily impressed. I owe him many thanks for the apprenticeship.

Lionel was my first DPhil supervisor. The work I did with him on delta-sigma modulators led to my first journal publication [1]. We thought those ideas could be applied to logarithmic imaging. However, the fixed pattern noise of the latter proved to be problematic and became the focus of my thesis. Lionel always maintained an interest in my progress. In the course of his regular group meetings, I learned much about signal processing, neural networks and the management of large projects.

Satoshi collaborated with me for one especially memorable year. He taught me valuable skills in ASIC design and layout. Moreover, he contributed greatly to my understanding of readout circuits for image sensors. Together, we sought improvements to the readout and pixel circuits, culminating in a prototype camera that we designed and built, which is now being tested. On a personal level, Satoshi and I played Go in Oxford and explored Budapest during an IEEE conference.

Sunay and Alistair were great lab-mates, ready and competent with technical advice, critical worldviews and boyish humour. We shared PC management, UNIX resources and hot chocolate seamlessly. Gari and I had valuable discussions on tensor calculus and stress relief. He also ran the tea and biscuits club for ages, which provided timely injections of caffeine and sugar while working.

Simukai had the privilege or misfortune of being the only girl in the hardware boys club. Her office overheated while I chilled under the common A/C unit, which meant we contested the temperature setting. Mayela and I shared numerous genes with

[1] SPANN, the name of Lionel's group, stands for Signal Processing and Artificial Neural Networks and MCAD, the name of Steve's group, stands for Microelectronic Circuits and Analogue Devices.

nocturnal animals. It was nice to have her next door when I worked late at night. Both Simukai and Mayela had a cheerful interest in my personal and academic well-being.

I made use of Jan's well-kept photocopier, fax machine, printer and stationary cupboard often. Jan also helped me to place orders for hardware and software and to book rooms for tutorials. She coordinated many of the aspects that defined Lionel's research group as a unit, such as organising meetings and communicating news. She didn't mind too much that I kept asking her for a stapler—the lab one disappeared regularly.

Apart from being a friendly face, Stephen read Chapter 2 of my thesis and wrote useful comments. Neil and Paul volunteered much time and effort to manage and upgrade the local computing facilities while Kit has been a fantastic administrator of the wider computing facilities. Members of Lionel's and Steve's research groups added to a pleasant working environment and to lively social events outside work.

In my time at Oxford, I've met several disappointed DPhil students. I was never one of them because my friends and family were supportive, encouraging and distracting throughout. Participation in various extracurricular activities also gave me maturity. I've enjoyed my time at Keble greatly and am indebted to people in the MCR who made my degree seem less like work and more like play.

Sponsors

Thank you to the Natural Sciences and Engineering Research Council of Canada for funding most of my DPhil. I must also thank Lionel for writing me a letter of reference when I applied to NSERC and for completing biannual reports. Thank you to Doug Thomson, Greg Bridges and Howard Card of the University of Manitoba for writing me supporting letters and for staying interested in my career.

Thank you also to the University of Oxford for employing me during the final years of my DPhil. I must especially thank Steve, who successfully applied for funding from the Engineering and Physical Sciences Research Council of Britain. The EPSRC grant paid for my employment, for my participation in two international conferences and for equipment I used in the course of our research.

I am also very grateful to several other organisations that supported me financially at one time or another, especially Universities UK, the British Council, Celanese Canada, the Keble Association and Keble College. My gratitude extends to the people who wrote me supporting letters for these and other awards.

Publications

Some of the work reported in this thesis has been published in a different form elsewhere. The principal concepts of Chapter 4 and most of Chapter 7 were reported in papers presented at two IEEE Instrumentation and Measurement Technology Conferences [2, 3]. Both papers were subsequently accepted for publication in the IEEE Transactions on Instrumentation and Measurement.

Chapter 4 differs from the first paper, published in 2001, by its integration with the rest of the thesis, a reorganisation of the section on calibration, the inclusion of simulation results and the use of new experimental data. The conclusions, nonetheless, remain the same. Chapter 7 differs from the second paper, published in 2002, by virtue of its integration with the rest of the thesis and the space for detail.

Contents

List of Figures

List of Tables

List of Abbreviations

ADC Analogue-to-digital converter

AMS Austria Micro Systems

APS Active pixel sensor

ASIC Application-specific integrated circuit

BSIM MOS model from Berkeley University

CCD Charge coupled device

CIE Commission Internationale de l'Eclairage

CMOS Complementary MOS

DCM Double current mirror

FPN Fixed pattern noise

Fuga 15RGB Camera designed by IMEC

HDRC VGA Camera designed by IMS

HDTV High-definition television

HSPICE An integrated circuit simulator

IEC International Electrotechnical Commission

IMEC Interuniversity MicroElectronics Center

IMS Institute for Microelectronics Stuttgart

Lab CIE perceptual colour space

MOS Metal-oxide-semiconductor

NMOS N-channel MOS

PMOS P-channel MOS

PPS Passive pixel sensor

RGB Red, green and blue

SSE Sum square error

SNR Signal-to-noise ratio

sRGB IEC standard colour space

T# Transistor identified by number

XYZ CIE linear colour space

Chapter 1

Introduction

1.1 Motivation

The importance of visual information to society may be measured by the technological endeavour over millenia to record observed scenes on an independent medium. Artistic license aside, amateurs and professionals have sought to render images with a maximum of perceptual accuracy and a minimum of effort. The culmination of this undertaking is the digital camera. However, the development of the digital camera is far from complete.

Although digital cameras have in many ways surpassed the capabilities of film cameras, the human eye remains the ultimate standard for comparison and it vastly outperforms the best cameras in many respects. Furthermore, widespread economic interest in cameras, with a market demand expected to reach 60 million by the year 2002 [7], has sustained reseach and development in a variety of image sensor designs, which make up the operational core of the digital camera. The various designs may be broadly grouped into two categories: charge coupled device (CCD) sensors and complementary metal-oxide-semiconductor (CMOS) sensors. Table 1.1 compares these electronic sensors to photographic film and the human eye.

The eye is a remarkable organ not simply because of its ability to sense light but especially because of its ability to process light information before even sending a signal to the brain. Far more information enters the eye, in terms of the positions and wavelengths of observed photons, than can realistically be transmitted down the optic nerve, or processed by the visual cortex, in real time. By genetic design, the eye encodes the vast visual input in such a way that the limited neural output retains the most significant descriptors of the scene while the rest are discarded [8].

With his work on the silicon retina, Carver Mead helped bring biological inspiration into the image sensor community [9]. This effort sought to replicate biological structures of the eye, concerned with information encoding, using analogue electronics. Although the focus of many years of research, some of it still ongoing, the work did not lead to an economical camera that renders images realistically (but this was not always the goal). Endeavours at biological inspiration in image sensing that were

Table 1.1: The human eye versus silicon (and film). Numbers given are typical values, as of 1999, following Dierickx [6].

Criterion	Eye	CCD	CMOS	Film
Spectral resp.	400–700nm	400–1000nm	400–1000nm	300–700nm
Peak quant. eff.	$< 20\%$	$> 50\%$	$> 50\%$	
Dynamic range	40–120dB	80dB	66–120dB	20–80dB
Dark limit	10^{-3}lux	10^{-4}–0.1lux	10^{-3}–1lux	≈ 0lux
Noise photons	10	10	100	100
Integ. time	0.3s	40ms–5min	40ms–10s+	Unlimited
Max. frame rate	≈ 15Hz	10kHz	$\gg 10$kHz	1 shot only
No. of pixels	$12 \cdot 10^7$ cones	$8 \cdot 10^5$–$9 \cdot 10^7$	$8 \cdot 10^5$	$> 10^6$
Pixel pitch	2–3μm	5–10μm	5–10μm	10–20μm
Image size	3cm	1mm–11cm	1mm–2cm	Film size
Rad. hardness	1mrad	10krad	10krad–?	
Op. temperature	36°C	73K–200°C	0K–200°C	0K–100°C
Power dissip.	< 1mW	500mW	50mW	None
Colour quality	Ideal	Poor	Poor	Poor
Photometry	Impossible	Easy	Easy	Possible
Preprocessing	Extensive	None	None	None
Access method	Data driven	Serial only	Serial/random	Optical only
Data path	$5 \cdot 10^6$ nerves	8–10 bits	8 bits	None
Unit price	Invaluable	100 euros	10 euros	0.1 euros
Dev. cycle	$5 \cdot 10^8$ years	5 years	2 years	20 years
Number of fabs	$3 \cdot 10^9$	10	1000	10

commercially successful sought less to reproduce biological structure and more to reproduce biological function and relied less on analogue electronics and more on digital electronics [10, 11]. Semiconductor physics is vastly different from cellular biology and therefore information processing structures must be tailored to the medium. Furthermore, the reliability and flexibility of digital over analogue electronics has led a general trend in the semiconductor industry to favour the former over the latter.

This thesis concerns the biologically inspired digital cameras composed of logarithmic as opposed to linear image sensors. These sensors, which may be built in CMOS but not in CCD technology, are semi-successful in that they are available commercially but remain of interest only to researchers and developers because of problems with image quality. The hypothesis advocated here is that by deriving a model of the logarithmic CMOS image sensor, supported by semiconductor theory, and by deriving a method to calibrate the model, validated with simulated and experimental data, it will be possible to understand precisely how these digital cameras fall short of rendering an image with a maximum of perceptual accuracy and a minimum of effort. Such an understanding may be used to improve the image quality, as shall be shown, at an expense of digital processing. Such an understanding may also be used, in the future, to design a better logarithmic CMOS image sensor.

Section 1.2 gives a background to CCD and CMOS image sensors, to linear and

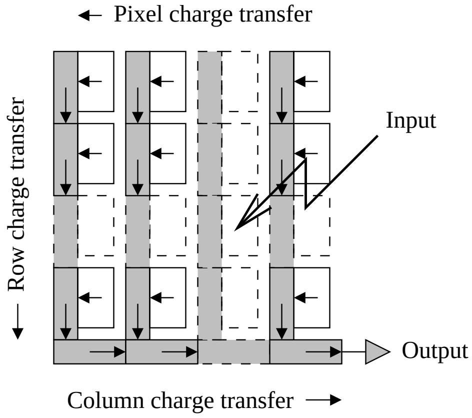


Figure 1.1: CCD image sensors march photogenerated charge systematically from an array of pixels to an output amplifier.

logarithmic CMOS designs and to problems with image quality in the latter, outlining promises and challenges of various options. Section 1.3 previews the theory, simulations and experiments that comprise the remaining chapters.

1.2 Background

1.2.1 CCD versus CMOS

Figure 1.1 depicts the architecture of an *interline transfer* CCD image sensor [12], typical of video rate CCD imagers [13]. Light striking the photosensitive area of each pixel creates charge carriers in the doped silicon substrate and these carriers collect in a potential well, which is created in each pixel by a voltage applied to a gate electrode. After a programmable period of time elapses, the charge is shifted to another well in the pixel, shielded from light by opaque metallisation, by modulation of gate voltages. As a result, all photosensitive wells are simultaneously emptied of charge. While the collection process resumes from scratch in each pixel, the charges stored in the shielded wells are shifted repeatedly in parallel from one row to the next by modulation of gate voltages. The charges in the bottom row are shifted into a separate row of shielded wells called the output register. In the time between shifting of rows into this register, the charge in each well of the register is shifted repeatedly in parallel from one column to the next by modulation of gate voltages. The charge in the last column of the register is shifted to an amplifier where it is converted to a voltage for driving external electronics. In this manner, an array of photogenerated charge is marched systematically to the amplifer before the next image is available [14, 15].

Figure 1.2 shows the architecture of a CMOS image sensor [15], which is similar to a memory array. Each pixel consists of a photodetector, usually a photodiode or a photogate, and one or more transistors and capacitors depending on the design (which

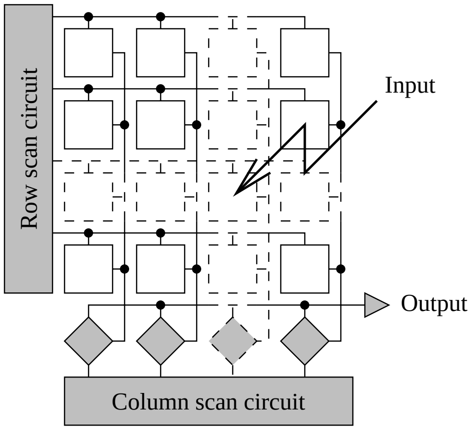


Figure 1.2: CMOS image sensors operate like memory arrays with photosensitive pixels instead of memory cells.

varies considerably). As before, light striking the photodetector creates charge carriers that are used to produce a signal, which may be a voltage or current. The way in which the signal is produced and the type of signal depends on the pixel design. For example, an early design called the passive pixel sensor (PPS) integrated the charge onto a capacitor to produce a voltage. To read a pixel, a row scan circuit and a column scan circuit decode a supplied address and enable the row and column lines of the pixel. As with cells in a memory array, all pixels in a column drive a common buffer via a shared column bus. When a particular row is selected, all pixels in that row drive their respective column buffers. All column buffers drive a common amplifer via a single output bus. Only one buffer, selected by the column scan circuit, operates at a time. Thus, by scanning of the address space, each pixel may drive its photogenerated signal to the output amplifier. In a PPS, the capacitor in each pixel is connected to its column bus by a transistor switch (the column buffer is connected to the output bus by another transistor switch). Modern pixel designs, called active pixel sensors (APS), have additional transistors in each pixel and column circuit to amplify the signal.

CCDs dominate the image sensor market, taking 90% of share in 2001, for many reasons [16]. The semiconductor industry has had three decades of experience in the making and selling of CCD sensors, whereas CMOS sensors have been viable for less than a decade [14]. Investment and development of CCDs continues apace with Sony, Matsushita, NEC and Texas Instruments announcing megapixel sensors in recent years. In applications demanding high resolution and sensitivity, CCD sensors are preferred because they deliver a better image quality than CMOS sensors, especially for still photography [7, 17, 15]. Variations in device characteristics, such as feature dimensions and silicon doping levels, from pixel to pixel and column to column leads to substantial fixed pattern noise with CMOS [13, 15]. There is also a high temporal noise from thermal and $1/f$ sources with CMOS sensors because signals are transferred to the outside world via multiple transistor stages [18]. Fixed and temporal noise is smaller with CCD sensors because charge packets are transferred almost perfectly within the sensor and pass to the outside world via a single output stage [15, 18]. In CCD technology, the percentage of the pixel area devoted to photodetection, called the fill factor, is high compared to CMOS technology, giving a better photoelectric efficiency [14]. Furthermore, CCD sensors are fabricated in dedicated processes that are fully optimised for imaging [15]: junction and depletion depths are positioned for optimal spectral sensitivity and minimum crosstalk [19]; special attention is paid to minimising dark current, a parasitic effect in photodetectors, so that it is lower with CCD technology than with CMOS [19, 7]; and CCD technology is designed to make good capacitors, maximise signal-to-noise ratio (SNR) and achieve a high charge transfer efficiency (up to 99.999%) [14]. Typical CCD sensors achieve SNRs better than $60\mathrm{dB}$, where a $42\mathrm{dB}$ SNR is the threshold for a VHS-quality still picture.

Nonetheless, CCD technology faces many challenges. CCD manufacturers use specialised fabrication processes that have their roots in the early days of MOS integrated circuits and are incompatible with modern CMOS processes that make today's more complex mixed-signal integrated circuits [13]. As a result, CCDs cannot be integrated easily with CMOS circuits, such as timing and control logic, because of additional fabrication complexity and increased cost [20, 13]. Even with integration, CCDs are high capacitance devices so adjacent CMOS circuits would dissipate too much power [20]. Therefore, these circuits are added externally, requiring extra components and board space. For high charge transfer efficiency, CCDs require specialised processes with large voltage swings and multiple supply and bias voltages, which further complicates the system [20, 14, 13]. CCDs also suffer from blooming and smear, especially when imaging a scene containing bright lights. Blooming occurs when the depleted region under a gate fills with charge and excess charge spills into neighbouring depleted regions. Smear, appearing as vertical stripes, occurs when photogenerated charge leaks into shielded wells during the parallel transfer of charge packets in each column to the output register. Most modern CCDs have structures to reduce these effects.

The gap between CCD and CMOS in terms of sensitivity is diminishing and is expected to close for future high performance multimegapixel sensors [7]. As CCDs transport their charge packets to external electronics through a single output stage, high frequencies of charge-to-voltage conversion are required for a high pixel count and frame rate [15]. Already, CCDs are having difficulties meeting the demands of the high performance video market since CCD noise performance, which is the main advantage over CMOS for still photography, worsens by a factor of five to ten at higher speeds [17]. For good quality images, the CCD readout rate is limited by sequential access and the need for nearly perfect charge transfer [20]. These factors are exacerbated with shrinking feature sizes because more pixels need reading in the same time with smaller pixel sizes but the same image size and frame rate [7]. With millions of pixels, random accessibility may become important as the data flow requirements for full frame video may challenge microprocessors. Due to their intrinsically serial readout, CCDs cannot support random access [20, 13]. For the same reason, the CCD process suffers from poor yields (and CCD sensors suffer from susceptibility to radiation damage) [20, 14]. If a defect appears in a single pixel of a CCD sensor then it interrupts the charge transfer process of the column, rendering most of the column useless.

Because of CCD limitations, CMOS technology has been gaining ground where system integration may be traded against the moderate image quality that is readily available [7]. CMOS image sensors may be fabricated in standard CMOS processes, allowing analogue and digital signal processing circuits to be integrated on the same die as the sensor array [14, 15]. The PPS technology, developed before the APS technology, was unpopular because of poor image quality. However, APS technology raises the SNR and photoelectric efficiency of CMOS imagers near to those of CCDs. APS offers lower noise readout, improved scalability and higher speed compared to PPS [20]. As standard CMOS processing allows sensors to take advantage of the enormous infrastructure and learning curve of the semiconductor industry, CMOS imagers are beginning to compete against CCDs in many areas of the consumer market [14]. Industry sources predict CMOS sensors will take over CCD sensors in consumer-grade digital cameras, the worldwide market for which is estimated to be 8.5 million in 2001. The estimated market for CMOS imagers in general is 60 million in 2002 [7]. Applications include robotics, machine vision, guidance and navigation, automotive technology and consumer electronics (e.g. video phones, computer inputs and home surveillance) [20].

Many large players have entered the emerging market for CMOS imagers, including Texas Instruments, Motorola, Toshiba and Rockwell [14, 7]. Intel plans to market digital cameras for PCs using CMOS sensors that it will manufacture. STMicroelectronics and Photobit are the leading suppliers of CMOS sensors, which take up 10% of the imager market in 2001 [16]. A trend in the semiconductor industry is to outsource to achieve economies of scale (the CEO of Photobit expects 50% of all integrated circuits to be fabricated by the world's three leading foundries by 2002) so there are also 56 fabless CMOS imaging companies. Though CCDs presently dominate the market, there are only about five large manufacturers [14]. The fact that CMOS sensors can be built by more people means there will be more competition and ultimately lower prices. For CMOS imagers, product differentiation will increasingly be found in the circuit design, chip architecture and system integration levels whereas CCD product differentiation is mostly found at the device and process design levels [19].

CMOS pixels will scale better with technology not because the photodetectors are any better than CCDs but because more and more additional circuitry can be placed in each pixel without affecting pixel size, fill factor or sensitivity [14]. In theory, pixel sizes do not need to go below $5\mu\mathrm{m} \times 5\mu\mathrm{m}$ because of the diffraction limit of the camera lens [19]. However, since in a common Bayer patterned colour imager, a 2×2 mosaic of pixels (with red, green and blue filters) defines an effective colour pixel, further downscaling of single pixels may be useful to fit one effective colour pixel into the optical lens resolution limit [7]. Apart from adding circuits to pixels, analogue-to-digital converters (ADCs) can be integrated on the image sensor die and, with digital signal processing and other functions, a one-chip camera becomes possible [14]. This has the advantage that all off-chip communication can be digital but, more importantly, the integration of circuits on one die reduces the power, size and cost of the system.

There are several other advantages of CMOS over CCD. The similarity in readout between CMOS imagers and memory arrays means pixels may be randomly addressed, which is desirable [20, 13]. For the same reason, defects tend to affect individual pixels in a CMOS sensor leading to better yields and hence a cheaper product [14]. Since modern CMOS imagers have amplifiers present in each pixel, charge to voltage conversion operates at low frequencies even in the case of a high pixel count [15].

Video performance relies more on signal processing and driving than on low noise performance of pixels [17]. Even in aspects of light sensation, CMOS may exceed CCD technology [13]. CMOS photodiode imagers typically have 60–65% absolute quantum efficiency at peak. For standard CCD, the absolute peak quantum efficiency is about 35% because of semitransparent polysilicon gates above each pixel, which partially obstruct incoming light especially in the blue portion of the spectrum.

Despite numerous advantages and enormous interest in CMOS technology, a dominant changeover from CCD to CMOS will not be soon [14]. Presently, CMOS sensor offerings are at the low end of the market, e.g. webcams, where cost is more important than performance [17]. Whereas CMOS foundries are sufficient for acceptable resolution sensors, achieving high resolution and quality comparable to CCDs requires dedicated processes [16]. Front-end process modifications such as additional implants and back-end process modifications such as color filters and microlenses are essential for implementing commercially viable image sensors. Fundamentally, as multiple metal layers hamper high resolution imaging, spacings need to be made as thin as possible. Fabless providers will have to make arrangements with foundries for add-on CMOS process modules tweaked at least for general imaging. Furthermore, using standard CMOS technology to make an image sensor does not automatically result in a major price advantage in the finished camera [19]. A non-negligible fixed cost with all digital cameras is due to optical related processes such as optical testing, optical cleanliness, optical packaging, on-chip colour filter arrays and on-chip microlenses.

Since standard CMOS technology develops to optimise the power-delay, reliability and cost-performance of logic and memory circuits, if device characteristics germane to imaging are not considered as the technology evolves then CMOS imagers may not benefit from device scaling [19]. However, to reach the resolution standards that CCDs today dictate, CMOS sensors must use downscaled processes [7]. Yield, die cost and lens cost also benefit from a small pixel and therefore a small die size. While standard CMOS may provide adequate imaging at the 2–$1\mu\mathrm{m}$ generation without any process change, modifications to the fabrication process and innovations of the pixel architecture are needed to enable good quality imaging at the $0.5\mu\mathrm{m}$ generation and beyond [19]. Optimisation of CMOS imagers begins to diverge from that of CMOS logic and memory at the 0.35–$0.25\mu\mathrm{m}$ generation. If foundries are willing to tailor the junction and/or channel implants and selectively or globally removing the opaque silicide module, accepting the cost and/or performance degradation associated with doing so, CMOS imagers may be scaled to 0.25–$0.18\mu\mathrm{m}$. The use of silicon-on-insulator then poses a significant problem. As CMOS technologies approach the 0.13–$0.1\mu\mathrm{m}$ generation, parasitic off currents, gate tunnelling currents and p-n junction tunnelling currents begin to approach the dark current density observed today and both tunnelling currents increase exponentially with further device scaling. Related to device scaling, voltage scaling reduces the dynamic range of standard CMOS imagers by decreasing the full signal charge capacity. Enhancements and deviations from standard processes will be necessary to keep up sensitivity with downscaled generations [7].

The long-term challenges facing CMOS imaging have not escaped the attention of academia and industry. Various specialised devices have been developed to increase sensitivity without costing too much in pixel area, including the photogate, the pinned photodiode and the thin-film-on-ASIC pixels [7]. Foundries are recognising the mar-

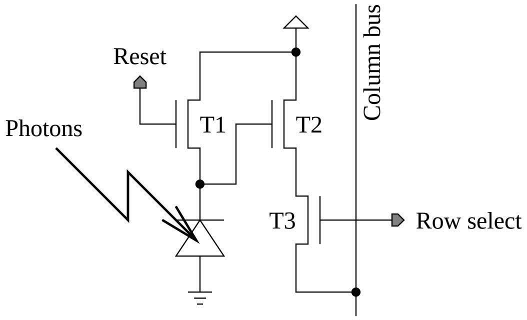


Figure 1.3: Linear CMOS pixels integrate photogenerated charge, sensed by a diode, onto a capacitor, i.e. the depletion capacitance of the diode.

ket potential of CMOS image sensors and are responding to the needs of the technology [16]. The Taiwan Semiconductor Manufacturing Corporation, the world's leading foundry, announced that it will use its entire process family, from $0.8\mu\mathrm{m}$ down to $0.35\mu\mathrm{m}$ and below to support CMOS image sensor production. Foundries can optimise for 20 different processes simultaneously and may expand for up to 100 processes. In the long run, lower costs associated with using *standard* CMOS processes may not be the winning advantage of CMOS over CCD. The real advantage of CMOS imaging is the high level of on-chip logic, memory and signal processing possible, as well as the capability for random access, all of which remain basically impossible with CCDs [19, 7]. In addition, and perhaps more importantly, the lower operating voltage and lower power consumption will be the determining factor in many applications. especially consumer electronics and mobile computing.

1.2.2 Linear versus logarithmic

There are many different types of CMOS pixel designs in the literature and the market. Two concepts of particular distinction, however, are integrated versus continuous response pixels [21, 22]. The former, which is by far more common, is normally characterised by a linear response. The latter, which is the focus of this thesis, is normally characterised by a logarithmic response.

Figure 1.3 shows a typical APS design, which is a linear pixel that employs integration [14, 7, 15, 18]. Light incident on the photodiode generates charge carriers, which are collected on the capacitor formed by the gate of the amplifying transistor $\mathrm{T2}$. After a programmable integration time has elapsed, the voltage on the capacitor is read out on the column bus by enabling the row select line of the pixel, turning on the switch transistor $\mathrm{T3}$. This voltage is linearly related to the total charge, which in turn is linearly related to the incident illuminance. By pulsing the reset line high, the voltage at the gate of $\mathrm{T2}$ may be reset to the supply level via the switch transistor $\mathrm{T1}$.[1]

[1] When an NMOS transistor is used for reset instead of a PMOS transistor, as shown in Figure 1.3, steady state is not always reached during reset (for typical reset times) and the final gate voltage of $\mathrm{T2}$ depends on

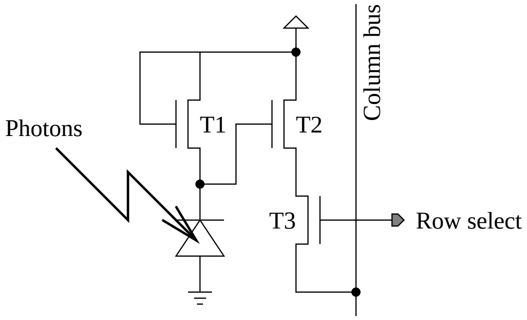


Figure 1.4: Logarithmic CMOS pixels convert photogenerated current, sensed by a diode, into a voltage using a load transistor in weak inversion.

The integration may then be repeated.

Figure 1.4 shows a typical logarithmic pixel [23, 24, 25, 26]. Remarkably, the only difference between the circuit schematics of Figures 1.3 and 1.4 is the diode connection of transistor T1 and the lack of a reset line in the latter. These changes mean that light incident on the photodiode generates a current, linearly dependent on the light intensity, that continuously flows through the load T1. Because this current is small relative to the load, T1 operates in weak inversion leading to a logarithmic current-to-voltage conversion. As before, the signal voltage appears at the gate of transistor T2 and transistor T3 is a switch used for connecting T2 to the column bus when the row select line is enabled.

Both types of sensor are susceptible to fixed pattern noise (FPN), which is caused by a variation of device parameters, especially threshold voltages, from pixel to pixel or column to column [21, 22]. The linear pixel of Figure 1.3 has a substantial advantage in this respect, owing to integration. By modifying the column buffer to read the pixel response after reset and subtracting this result from the pixel response after integration, a method known as double sampling, FPN due to pixel variations may be reduced [20]. Double sampling also reduces transistor $1/f$ noise, which is a temporal rather than a spatial effect. Furthermore, by subtracting from the signal level the reset level prior to integration, rather than the reset level after integration, the reset noise or the uncertainty in the gate voltage of T2 upon reset (also called kTC noise) may be reduced. Such an operation is termed *correlated* double sampling. By introducing another reset level at the column buffers, delta difference sampling reduces FPN due to column variations. Due to its continuous response, the logarithmic pixel of Figure 1.4 suffers greatly from FPN as there is normally no reset level in the pixel to enable double sampling [21, 22].

However, logarithmic pixels have an advantage over linear pixels in terms of dynamic range [27, 25, 28, 4]. Real scenes span over eight decades of illuminance, ranging from 10^{-3}lux in starlight to 10^{2}–10^{3}lux for indoor lighting, to 10^{5}lux for bright sunlight and to higher levels for specularities or direct viewing of bright sources (such

its initial value, which may cause image lag [18]. However, an NMOS is often used because it leaves more room in the pixel layout for the photosensitive diode.

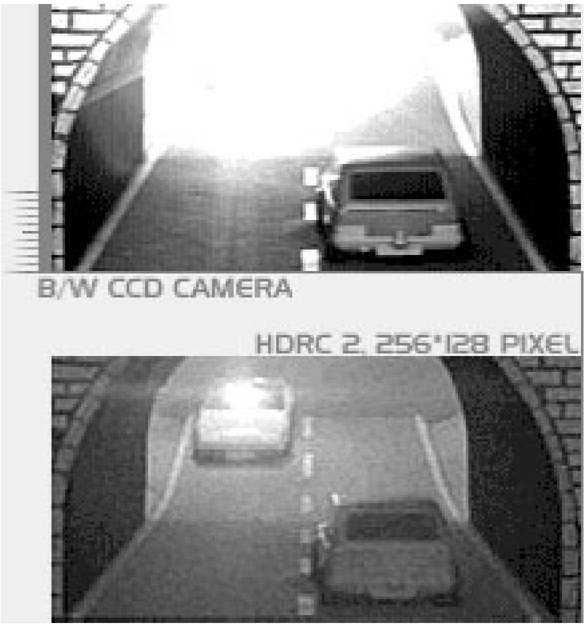


Figure 1.5: An image from IMS Chips shows how linear cameras (CCD or CMOS) saturate when they encounter a high dynamic range scene whereas logarithmic cameras (CMOS only) capture perceptible detail in the bright and dark parts of the scene [4].

as oncoming headlights or the sun).[2] Under normal conditions, the useful dynamic range does not exceed five decades at once (shadows to sunlight) but a sixth may be added to discriminate highlights [29]. Typical linear CCD and CMOS APS sensors may capture three decades of dynamic range whereas logarithmic CMOS sensors may capture six decades [4]. Figure 1.5 compares images of a high dynamic range scene, defined to encompass over three decades of light intensity, taken by a linear CCD sensor (linear CMOS sensors have comparable performance) and a logarithmic CMOS sensor. The linear sensor can adapt over a high dynamic range by aperture adjustment or global control of integration time but saturated patches of black or white appear when imaging a high dynamic range at once [25]. The logarithmic sensor can capture detail in bright and dark parts of a scene simultaneously, approximating human perception.

Human perception roughly approximates Weber's law, which states that the threshold to sense the difference between the illuminance of a fixation point and its surroundings is a fraction, about 1–10%, of the surrounding illuminance [30]. Even if signal-to-noise ratios of linear sensors could be improved to resolve six decades of dynamic range, it would be difficult to meet the quantisation requirements [25]. For example, while it takes 14 bits to quantise illuminance with 10% accuracy over a three decade range, it would take 24 bits to do the same over six decades. Achieving the

[2] Illuminance measures light power per square metre weighted by the spectral response of the human eye.

latter degree of quantisation would be costly for still photography and very difficult at video rates. Even if it were possible and economical to digitise a scene with 24 bits per pixel (per colour channel) most bits would be wasted when the image is displayed to a human. By Weber's law, human perception has less absolute sensitivity to bright illuminances than to dim ones [30]. An alternative is to use for six decades the same degree of quantisation used for three decades (normally less than 14 bits) but this would lead to a lack of perceptible detail, especially at dim illuminances. The best solution is to encode illuminances on a logarithmic scale so that a fractional threshold becomes a constant threshold, ideal for uniform quantisation over a high dynamic range [4]. On a logarithmic scale, capturing six decades of illuminance with 1% accuracy requires only 12 bits of quantisation.

There have been other approaches to achieving a high dynamic range image sensor but most result in a low fill factor (the percentage of pixel area devoted to light collection) or a large pixel [25]. For example, embedding multiple amplifiers within linear pixels, multimode sensors permit varying sensitivity levels, configured by switches, from pixel to pixel. Still, with few sensitivity levels, saturated patches may appear or there may be a failure to capture perceptible detail. Another approach converts the photogenerated signal in a linear pixel to a pulse frequency. Every time the integrated charge reaches a threshold, a pulse is generated and the pixel is reset thereby avoiding saturation. The illuminance is measured by the counting of pulses. Unfortunately, threshold voltage mismatch causes frequency errors, which are multiplicative rather than additive. Sensors with local exposure control are similar to pulse frequency sensors in that a reset is generated when the integrated charge exceeds a threshold but, here, the threshold is high instead of low. The time taken to reset a pixel is used to measure illuminance. In dim lighting, the response is quite slow.

Two methods to increase the dynamic range of integrating sensors that show promise are well capacity adjustment and multiple sampling but both have undesirable dips in the SNR as a function of illuminance (these drops are smaller for multiple sampling) [25, 31]. With well capacity adjustment, at any point in time, photogenerated charge in excess of the limit imposed by a potential barrier flows over the barrier into a charge sink. Normally, this results in clipping and it was originally implemented to suppress blooming (a phenomenon worse than saturation whereby charge overflows from saturated pixels into adjacent unsaturated pixels). However, by starting with a lower potential barrier and increasing it with time, this method can be used to create a monotonic compression curve. In other words, well capacity adjustment implements a nonlinear response with an integrating sensor. There is a decrease in fill factor, or an increase in pixel size, with this method as well as the addition of a control mechanism.

Multiple sampling, of which dual sampling is a specific and common example, involves reading the signal level of each pixel at multiple instants of the integration period [25, 31]. These multiple samples are post-processed to produce a single image. The idea is that bright illuminances will be sampled without saturation at the earlier instants and that dim illuminances will be sampled with less noise at the later instants. Multiple sampling does not affect the fill factor or pixel size with photodiode APS circuits because readout is nondestructive. However, multiple column bus processing chains are needed with photogate APS circuits because of destructive readout, which makes the method impractical beyond dual sampling. Even with photodiode APS circuits, it

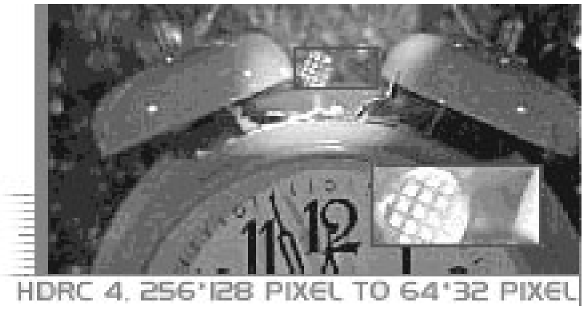


Figure 1.6: Since logarithmic pixels operate continuously, they permit high speed imaging especially when frame size is traded for frame rate. In this example from IMS Chips, the subframe rate of $4000\mathrm{Hz}$ is 16 times faster than the full frame rate [4].

is difficult to achieve more than two samples per frame because of the high readout speeds that are required, especially at video rates. On the other hand, two samples may not be sufficient to represent the areas of the scene that are too dark to be captured in the first image and too bright to be captured in the second.

Although multiple sampling may enable linear sensors to capture high dynamic range scenes, it limits the frame rate and may suffer from blur with scenes that contain motion. Furthermore, linear sensors are not randomly accessible in time unlike logarithmic sensors [32, 24]. This is because of the integrating nature of linear sensors, which means responses are available at discrete intervals of time, versus the continuous nature of logarithmic sensors, which means responses are available at any moment. Availability of random access in both space and time makes logarithmic sensors ideal for motion detection and tracking [33, 26]. As the readout of a logarithmic imager mirrors that of a memory array, pixel responses may be read in any order at any time [4]. It is not necessary to read an entire frame if only a subframe contains the interesting information, as shown in Figure 1.6. Logarithmic sensors easily permit a tradeoff between frame size and frame speed, useful in applications such as optical inspection, robotics, navigation, character or code recognition, position feedback systems, ranging and sizing, very fast dimensional measurements on continuous production lines, and web or wire thickness measurements [32, 24]. Even at low frame rates, the ability to select and read subframes reduces the data flow requirements on microprocessors.

The response of a logarithmic pixel is available continuously, i.e. at any moment in time, but the response is not instantaneous [24, 25]. The time a logarithmic pixel takes to respond to a change in illuminance depends on filtering effects associated with the charging or discharging of capacitances in the pixel [26]. Due to the weak inversion operation of the load transistor, the response time is a nonlinear function of illuminance. However, this is not a problem because high photocurrents give a fast response to intensity modulations while low photocurrents average the photon shot noise with a slow response [33]. Furthermore, despite the variation, the response time is typically fast. Using modulated lasers, Tabet et al measured the small signal $3\mathrm{dB}$ bandwidth of a logarithmic pixel to be $97.5\mathrm{kHz}$ at an indoor light level ($437\mathrm{lux}$) [26].

IMS Chips measured the large signal settling time, with its logarithmic sensor, for a 10^4-to-$1\mathrm{lux}$ change in illuminance to be $8\mathrm{ms}$ (a step change in the reverse direction settles in $0.8\mu\mathrm{s}$) [4]. Because of the combined capability of high dynamic range and high frame rate, the automotive industry is increasingly looking to logarithmic sensors to fulfil the requirements of traffic applications [33].

1.2.3 Logarithmic CMOS image sensors

Logarithmic CMOS image sensors are useful for high dynamic range and high speed imaging [4]. However, the problem of FPN needs addressing, especially for industrial and commercial applications involving safety of personnel and the public. Furthermore, colour rendition with logarithmic sensors is a contentious issue, as observed by Yadid-Pecht, because the nonlinear output makes subsequent signal processing difficult [25]. Indeed, the theory of colour rendition has been developed for linear sensors, something overlooked by both C-Cam Technologies and IMS Chips in their commercial versions of colour logarithmic image sensors, which display responses as if they were from linear sensors [34, 4]. Nonetheless, FPN remains the primary concern for both monochromatic and colour imagers.

Various approaches for dealing with FPN have been suggested and may be broadly categorised into analogue and digital techniques. Analogue techniques to reduce FPN modify the pixel and/or readout circuit operation. For example, Ricquier et al developed an image sensor that permitted hot carrier degradation of the threshold voltage of the amplifying transistor in each pixel (T2 in Figure 1.4) [23]. In addition to dissipating a lot of power, this method was very slow and needed repetition because the threshold voltages would initially relax back towards their original values.

Kavadias et al developed a method to reduce FPN by modifying the pixel and readout circuitry to include a reset level [27, 28]. Each pixel may be calibrated against a reference current in place of the normal photodiode current. With double sampling, this method removes offsets due to threshold voltage variations. As subtraction of the two levels is done in analogue at the end of each column, additional offsets created by the column amplifiers must be minimised. Furthermore, the current source for the reset level needs to be constant from pixel to pixel, which can be difficult. A disadvantage of double sampling is that it interrupts the continuous operation of the pixel, since calibration occurs every frame, and reduces the response time especially at low light levels (when the time to recover from reset is longer). Additionally, the calibration process is performed in a current regime different to the actual operating conditions and there is a noticeable residual FPN due to leakage current, doping density and gate-oxide thickness variation. Sensitivity variations of each pixel are pronounced because of the small dimensions of the photodiode but they may not be corrected with this approach.

Loose et al also developed a method for analogue reduction of FPN [21, 22]. As with Kavadias et al, this method replaces pixel photocurrents with reference current sources once per frame by careful use of switch transistors. However, instead of the usual double sampling, an amplifier feeds a voltage back, during calibration, to the gate of the weak inversion load transistor (which, unlike T1 in Figure 1.4, is not tied to the supply). The gate voltage is adjusted so that the pixel response to the reference current equals a reference voltage, which compensates for threshold voltage variation.

The cost is a large pixel size due to a high number of transistors and a capacitor, for storing the correction locally, per pixel. As many feedback amplifiers and reference current sources as there are columns are needed. These circuits have to match precisely to avoid additional variations between individual columns. Unfortunately, parasitic photocurrents discharge the capacitors that store the offset correction. Discharge time is inversely proportional to ambient illumination, making it difficult to set the time between calibration and readout. It should be short for high illuminances because of a fast discharge but long for low illuminances because of a slow recovery of the response. Furthermore, a residual variation exists due to capacitance mismatch and current mirror mismatch from column to column and switch transistor variation from pixel to pixel.

Although research and development of analogue methods to reduce FPN continue, digital methods have been employed in two commercial versions of logarithmic CMOS image sensors—the Fuga 15 series originally developed by the Interuniversity Micro-Electronics Center (IMEC) but now supplied by C-Cam Technologies and the HDRC series developed by the Institute for Microelectronics Stuttgart (IMS) and marketed by IMS Chips [35, 4]. Both approaches use three transistors and a photodiode per pixel, as in Figure 1.4. The Fuga 15d sensor, an early commercial sensor, had an array of 512×512 pixels, manufactured with a 50% yield in a $0.7\mu\mathrm{m}$ $5\mathrm{V}$ technology [32, 36]. IMEC also reported the fabrication of a 2048×2048 sensor in a $0.5\mu\mathrm{m}$ $5\mathrm{V}$ technology, with a high yield if a small number of bad pixels are acceptable [24]. The Fuga 15d and the 2048×2048 sensor, which was not commercialised, had a full frame rate of about $8\mathrm{Hz}$ but both could be subsampled to increase the frame rate. The HDRC VGA 2 sensor, manufactured in a $0.35\mu\mathrm{m}$ $3.3\mathrm{V}$ technology, delivers 640×480 pixels at a full frame rate of $45\mathrm{Hz}$ but it can also be subsampled [4].

Both the Fuga 15 and HDRC series of logarithmic image sensors implement digital reduction of FPN [35, 4]. An image of a uniform scene, such as a white sheet of paper under uniform illumination, is taken and stored, usually off-chip in an EEPROM. This image captures the lowest order variation of pixel responses, called offset variation, and is subtracted from subsequent images that are captured. However, Marshall and Collins have noted that FPN reduction degrades as the illumination of captured scenes departs from the illumination of the uniform scene used for calibration [10]. Hoefflinger et al considered a digital correction of gain variation, as well as offset variation, with an early HDRC sensor but no results were published comparing this method to offset correction only [33]. Yadid-Pecht suggested that FPN had a nonlinear dependence on illumination but she neither characterised this dependency nor sought to correct it [25].

Marshall and Collins and Loose et al suggested that threshold voltage variation would be affected by temperature [21, 10, 22]. A variation between the temperature dependences of pixel responses would be more problematic than a uniform temperature dependence. However, none of these dependences were characterised. Instead, Marshall and Collins suggested a digital method for FPN correction that considered both temperature and illumination dependence [10]. They advocated using an autofocus system to defocus a scene to obtain a calibration image that may then be subtracted from the focused image of the scene. This approach required frequent mechanical operation and introduced spatial high pass filtering to the image, unsuitable when rendering images for human observers in a perceptually acceptable way.

Evident by the commercial examples, digital approaches to correct FPN are promising for images taken with logarithmic CMOS image sensors. However, considering the ubiquity of linear sensors (CCD or CMOS) in the marketplace, widespread use of logarithmic sensors remains curtailed. Thus, a comprehensive model of pixel responses is required to understand the cause and nature of problems with image quality, including colour rendition. In addition, a way to calibrate logarithmic sensors is required to render images with a maximum of perceptual accuracy, robust to temperature and illumination changes, without sacrificing the capability for high dynamic range and high frame rate that makes the technology attractive. Indeed, a combination of digital and analogue approaches may ultimately be needed to achieve this challenging task.

1.3 Method

1.3.1 Theory

The modelling and calibration of image sensors involves the analytical and numerical manipulation of images. While a single image has the two dimensional structure of a matrix, a collection of images, e.g. taken with varying temperature or illuminance, may be naturally represented by an array of higher order than the matrix. Antzoulatos and Sawchuk [37], Blaha [38] and Snay [39] argue that an algebra of multiple index arrays facilitates the analytical and numerical manipulation of certain data. Such an approach is applied in this thesis and entails a review and extension of the subject of multilinear algebra, which formulates the basic operations on arrays.

Calibration of an image sensor involves specifying a model to relate the output to the input and estimating the parameters of the model from image data. Under certain conditions, multilinear regression is a suitable technique for estimation, which is useful even with nonlinear models to reduce the number of parameters that require nonlinear optimisation. Since an image sensor is also an array of pixel sensors, the task of modelling and calibration should consider possible relationships between the parameters of sensors in an array, as it leads both to better understanding of the cause and nature of parameter variation as well as to robust parameter estimation [40, 41, 42]. For this purpose, constrained regression is required and, to efficiently process the vast quantities of image data used in this thesis, attention must be given to the formulation so that computation takes a reasonable amount of processor time and memory space. All computations were done using MATLAB 5.3 on Sun Sparc workstations.

Because image sensors are composed of electronic circuits, the relationship between the output and the input are described using conventional models of electronic devices. Many models exist for these devices (transistors, diodes etc.) at varying levels of complexity and accuracy [43]. To facilitate analysis, Level 1 models omitting the finite output resistance in saturation (or the Early effect) are used to model transistors in the saturation or triode region. Level 3 models are used for transistors in the subthreshold region (there is no Level 1 model and the Level 2 and 3 models are identical) and the Shockley model is used for diodes. Further simplifications are often made. Transistors configured as switches are usually assumed to be ideal open or short circuits in the off and on states. Sometimes, more complex models are employed, when for example the Level 1 model fails to describe the temperature dependence of a parameter.

Occasionally, the limitations of such models are discussed when they prove significant.

While models of image sensors derived in this fashion permit an understanding of the physical factors involved in a relationship, these models often contain too many parameters for estimation, both from a practical and theoretical perspective. Apart from the computational complexity of estimating too many parameters, it may be impossible to distinguish one parameter from another, e.g. when they are added or multiplied together, purely from input versus output considerations. Thus, physical models are abstracted to mathematically equivalent but simpler models prior to calibration.

1.3.2 Simulation

In order to calibrate a model of an image sensor, it is necessary to have data. One way to produce this data is by simulating a circuit schematic of an image sensor, a group of pixels or a single pixel. A simulation is limited mainly in two ways. First, the schematic may not contain all the circuit elements present in a real sensor such as parasitic resistances, capacitances, diodes and transistors. Second, the models used by the simulator to describe the behaviour of circuit elements are only approximations of the behaviour of real elements. These models, however, are far more sophisticated than the Level 1–3 models used for theoretical analysis [43].

Nonetheless, simulation has many advantages. The cost of simulation in time and money, especially for variations in circuit design or over broad test conditions, is small compared to that of experiment. More importantly, simulation allows the study of circuits under controlled and well defined circumstances, which helps to disentangle cause and effect when many causes and effects exist simultaneously. Thirdly, simulation allows the observation of many states and variables internal to a circuit or device that could not be observed in experiment without either specialised equipment, foresight prior to circuit fabrication or disruption of circuit operation in the process.

Simulations were done using the Spectre simulator in Cadence 4.4.5 for a $0.35\mu\mathrm{m}$ $3.3\mathrm{V}$ AMS CMOS process (for a p-type substrate with three metal layers and one polysilicon layer) [44]. Transistors and diodes were modelled using BSIM3 Version 3 with parameters supplied by AMS [43]. The nominal width of all transistors was set to $1\mu\mathrm{m}$, as that was the width of the substrate contact (so hardly any space would be saved in a layout using smaller widths), and the nominal length was set to $0.6\mu\mathrm{m}$, as that was the minimum length recommended by AMS for transistors in circuits sensitive to threshold voltage variation [45]. A parasitic diode model, which describes the p-n junction formed between n-type diffusion and p-type substrate, was used to represent diodes. As these diodes were used to simulate photodiodes in pixels, they were set to a $6.32\mu\mathrm{m} \times 6.32\mu\mathrm{m}$ size that corresponds to a photosensitive square in a $10\mu\mathrm{m} \times 10\mu\mathrm{m}$ pixel with a 40% fill factor, which are the specifications of the HDRC VGA 2 logarithmic pixels built in a $0.35\mu\mathrm{m}$ $3.3\mathrm{V}$ process by IMS Chips [4].

The Spectre simulator permitted various types of analyses, four of which were used for the simulations reported in this thesis. DC analysis calculates the voltages and currents of all nodes and branches in the circuit schematic assuming a steady state condition. This analysis may be performed while sweeping the voltage or current of an independent source, either in linear or logarithmic steps. Transient analysis, on the other hand, calculates voltages and currents of nodes and branches as a function of time, where the time step is selected by the simulator and may vary during the simulation. These voltages and currents depend on the initial conditions specified by the user and steady state values are reached only when the simulation runs for a sufficient duration. Although independent sources cannot be swept directly with transient analysis, arbitrary voltage and current waveforms may be used as stimuli. The third type—parameteric analysis—repeats a simulation but each time changes a parameter according to a given sequence of values. This analysis can be used with DC or transient analysis to sweep, for example, a voltage or current source or the ambient temperature.

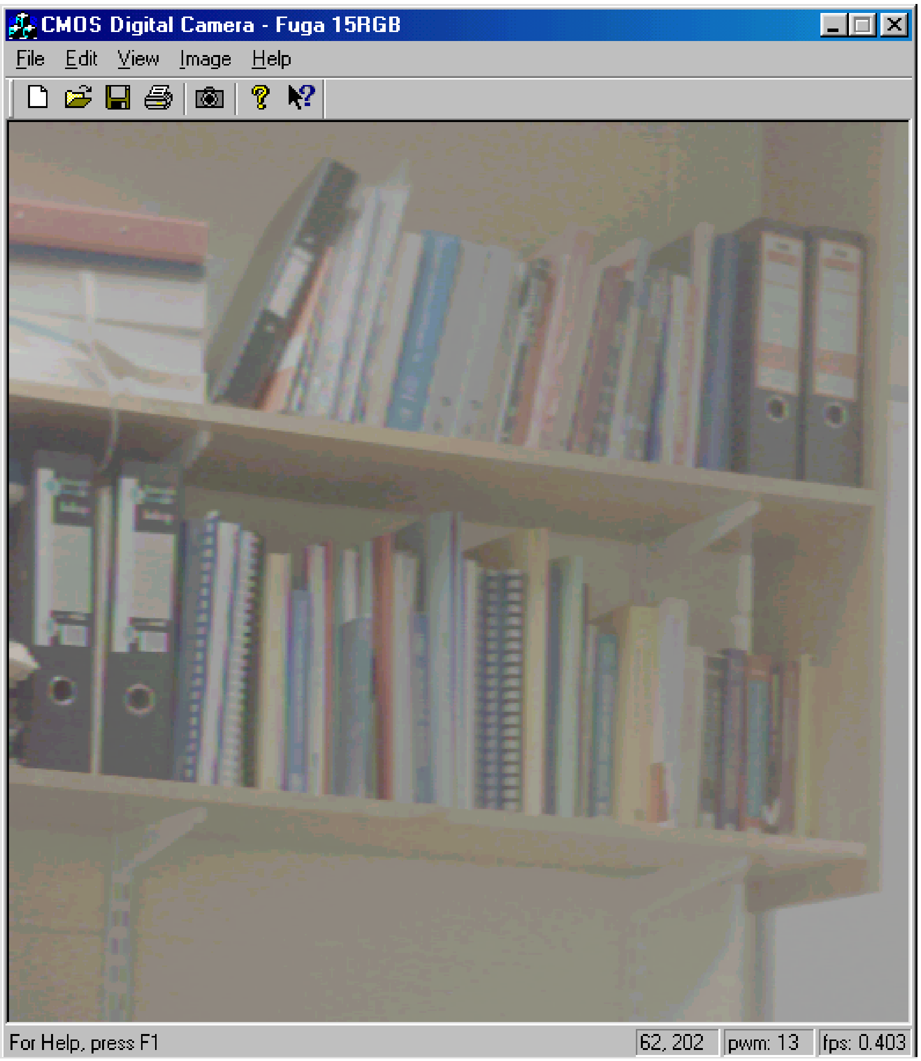
The fourth type of analysis, used in this thesis alongside the DC and parameteric (but not the transient) analyses, is Monte Carlo analysis. Normally, in a simulation, all circuit devices have exactly the same values for model parameters, although the node and branch voltages and currents may differ. Monte Carlo analysis chooses parameter values by mathematical functions on pseudorandom samples from statistical distributions. The functions and distributions are tailored to the simulated process and, thus, are provided by AMS. There are three types of Monte Carlo analysis in Spectre. Process variation simulates the statistical distribution of parameters assuming the electrical properties of devices are uniform across a die but non-uniform from one process run to the next. Mismatch variation, on the other hand, simulates the variation of electrical parameters on a die from device to device, neglecting the distance between devices, but ignores the variation from process run to process run. Lastly, process and mismatch variation includes both effects. As this thesis concerns the individual calibration of image sensors and each sensor consists of one die from one process run, only mismatch variation is simulated.

1.3.3 Experiment

Experiments were performed using a Fuga 15RGB camera from C-Cam Technologies [35]. Although this sensor does not represent the latest or best technology in logarithmic CMOS imaging, it belongs to the most sucessful generation of the Fuga series developed by IMEC, being a colour version of the Fuga 15d [32]. IMEC was a pioneer in the field, developing logarithmic imagers with publications as early as 1992 and releasing the Fuga 15 series commercially in the late 1990s. The Fuga 15d has long been the subject of independent research in logarithmic imaging and is still sold today [10]. Nonetheless, strong competition has appeared from IMS Chips in the last two years with its commercial series of HDRC sensors [4]. This series, which also offers colour, originates from work at IMS with publications as early as 1993.

The Fuga 15RGB was supplied as a camera system complete with lens and housing [35]. However, the camera needed to be operated by an external computer via a PCI card and a ribbon cable. C-Cam Technologies supplied a device driver and sample code to run the camera [34]. For the experiments in this thesis, a Microsoft Visual C++ application was created, giving control over camera parameters such as readout timing and frame size, implementing image processing and display operations and permitting the export of captured images in bitmap format. Figure 1.7 shows a screenshot of the application. The screenshot demonstrates the problem with colour rendition.

Figure 1.8 shows an unprocessed image taken with the Fuga 15RGB. The manufacturer provides a rudimentary way to reduce the grainy distortion of the image, which is



For Help, press F1 | 62, 202 | pwm: 13 | fps: 0.403

Figure 1.7: A Microsoft Visual C++ application was developed to run the Fuga 15RGB camera. As shown in this example, colour rendition is poor with logarithmic sensors without image processing beyond FPN reduction.

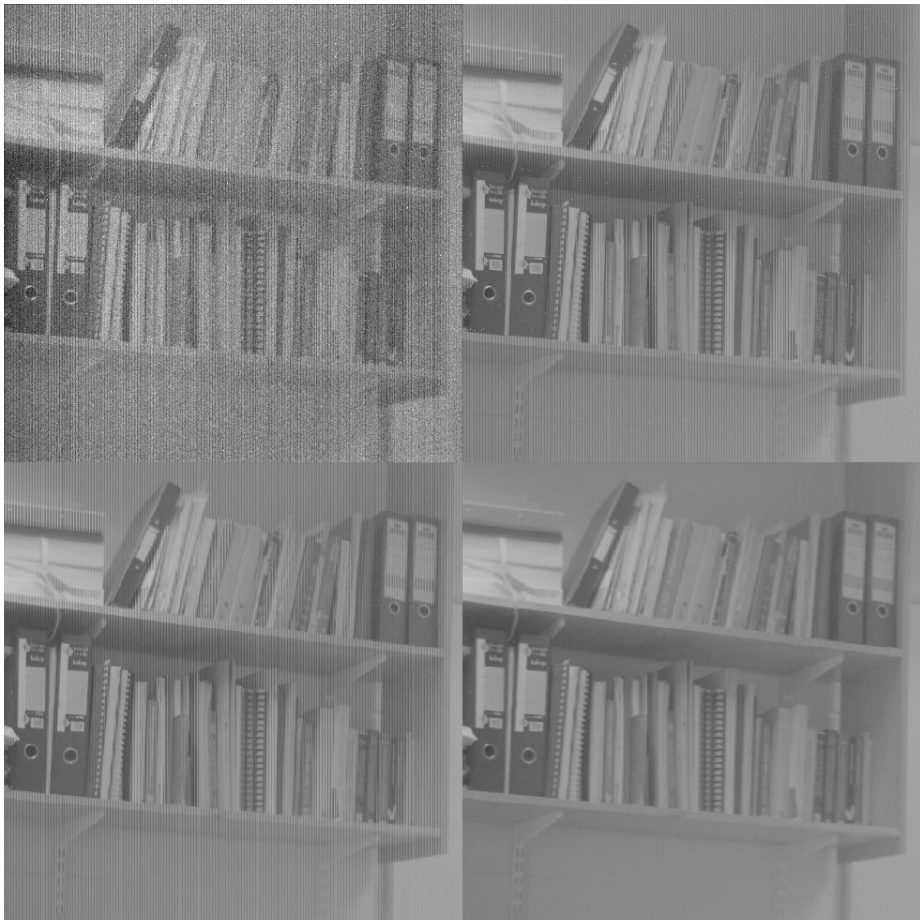


Figure 1.8: An image taken with the Fuga 15RGB, displayed unprocessed (top left), with built-in offset correction (top right), with additional median filtering (bottom left) and further greyscale interpolation (bottom right).

due to FPN [35, 34]. The PCI card can subtract a frame of 8-bit integers, stored in an EEPROM on the card, from captured images. The feature may be calibrated by imaging a uniform scene and saving the data in the EEPROM (the PCI card subtracts the mean from this data). Figure 1.8 shows the result of the built-in offset correction (after calibration with a white sheet of paper). The result contains speckle, most visible in the shadow under the top shelf, and vertical stripes. The speckle is caused by dead pixels, which appear not to respond to scene stimulus. In reality, they do respond but only very weakly. The stripes appear because the Fuga 15RGB was made by depositing red, green and blue colour filters on alternating columns of a Fuga 15d. Median filtering removes the speckle effectively, as shown in Figure 1.8. The filter replaces each pixel value by the median value of itself and the two nearest vertical neighbours. This design minimises the effect on resolution and does not corrupt the colour information. Median filtering is used only when images are displayed in this thesis but not prior to any calibration. Thus, dead pixels are modelled as having statistically extreme parameter values. By interpolating the corrected response of a pixel and its four or two nearest horizontal neighbours, a colour or greyscale image may be derived without stripes, as shown in Figures 1.7 and 1.8 for colour and greyscale respectively. However, rendition may be poor without colour or contrast processing. Furthermore, the manufacturer observes that FPN calibration needs to be repeated when illumination conditions change or when timing parameters are changed.

The Fuga 15RGB sensor has an on-chip 8-bit ADC [35, 34]. Therefore, analogue pixel responses are quantised with eight bits of accuracy. Because of FPN, which causes a wide variation in pixel responses even for a uniform scene, and because experiments reported in this thesis drive the camera from two to three and a half decades of dynamic range, pixel responses often saturate the ADC range. However, the camera allows the ADC range to be shifted by a programmable offset. By changing the ADC offset, saturated pixels may be brought into the ADC range. This feature offers an extra two bits of information per pixel. Denoting the response of a pixel over the actual 8-bit range as y' and the response of the pixel over the effective 10-bit range as y then (1.1) gives the relationship between the two, where G is a gain parameter (determined by regression analysis to be about -1.56), Δy is the 8-bit ADC offset and ϵ accounts for error in the relationship due to temporal noise and ADC nonlinearity. The standard deviation of this residual error was estimated to be 0.9LSB over a wide range of Δy values (from 10 to 255LSB) and over about two decades of illuminance (using overhead fluorescent lighting) at room temperature. This shows that the temporal noise and ADC nonlinearity are small.

$$y' = y + G\Delta y + \epsilon \tag{1.1}$$

If y' does not saturate for a pixel, i.e. $1 \leq y' \leq 254$, then y may be estimated within the limits of the error using the previously estimated value of G and the known value of Δy, as in (1.2). Rather than choosing Δy carefully for each pixel to avoid saturation, which is slow, a more practical approach is to take a few images of a scene for different values of Δy, spread out to capture the range of y. The actual response y' should not saturate for one or more of these frames (unless the effective response y is outside the 10-bit range, in which case the pixel is assigned 0 or 1023LSB if it is

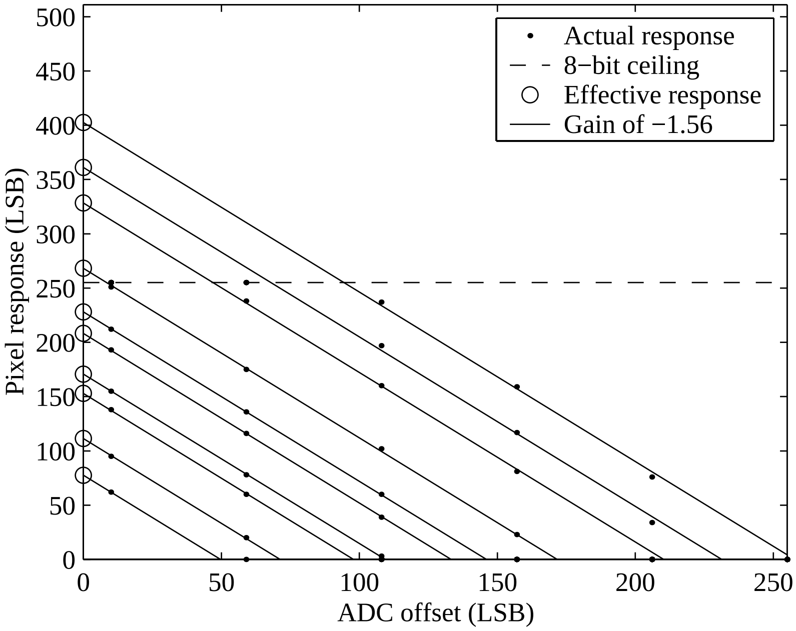


Figure 1.9: Actual responses y'_k of ten pixels for multiple ADC offset settings Δy_k. Actual responses may saturate at 0 or 255LSB but, if responses do not saturate for at least one ADC offset, effective responses y may be estimated for no ADC offset.

always dark or bright respectively). If a pixel is unsaturated in P images of a scene for ADC offsets Δy_k, where $1 \leq k \leq P$, then the corresponding P actual responses y'_k may be used to estimate the effective response y of the pixel, as in (1.3). Such an averaging reduces the effects of temporal noise and ADC nonlinearity.

$$y \approx y' - G\Delta y \qquad 1 \leq y' \leq 254 \tag{1.2}$$

$$y \approx \frac{1}{P}\sum_{k=1}^{P}(y'_k - G\Delta y_k) \qquad 1 \leq y'_k \leq 254 \tag{1.3}$$

Figure 1.9 gives an example of this multiframing approach for ten pixels, each responding to a different stimulus. Six values of the ADC offset are used ranging from 10 to 255LSB, typical of the experiments in this thesis, and the figure plots the actual response of each pixel for each image. Note that these responses sometimes saturate the 8-bit range. The effective response of each pixel, calculated according to (1.3), is projected onto the ordinate axis (i.e. $\Delta y = 0$). Note that pixels may have different values for P in (1.3). This multiframing approach is used for all experiments to avoid unneccessary saturation of responses. Each effective image in an experiment is computed from several actual images, taken with different ADC offsets. Furthermore, all subsequent modelling and calibration refers to the effective response y of each pixel and not to any of the actual responses y'_k. Although this procedure is an experimental inconvenience, it brings the number of bits per pixel of the Fuga 15RGB in line with that of the HDRC VGA 2, which uses a 10-bit ADC [4].

1.3.4 Organisation

The rest of this thesis is organised as follows. Chapter 2 reviews the literature on multilinear algebra, unifying and extending approaches for analytic and numeric manipulation of multi-index arrays, which are the generalisation of scalars, vectors and matrices. Chapter 3 defines and solves the problem of multilinear regression with linear constraints for the calibration of a sensor array, permitting models with linear relationships of parameters across the array. Chapter 4 develops a steady state model for the digital response of a logarithmic pixel to light stimulus and uses it to characterise and correct FPN, which proves to depend nonlinearly on illuminance, by calibration of simulated and experimental data. Chapter 5 models the transient response of logarithmic imagers, for typical source follower readout circuits, and shows with simulation and experiment how transient operation and design may cause FPN, which may partially be corrected by a steady state calibration. Chapter 6 extends the steady state model of the image sensor to examine and reduce the dependence of FPN on temperature, comparing in simulation and experiment methods of calibration that use pixel responses under both dark and light conditions. Chapter 7 describes the calibration of pixel responses in terms of a standard colour space, extending previous models suitable for FPN correction but unsuitable for colour rendition, and shows that colour rendition of a Fuga 15RGB logarithmic camera competes with that of conventional digital cameras. Finally, Chapter 8 discusses and summarises the main results of this thesis and outlines future theoretical, simulation and experimental work.

Chapter 2

Multilinear algebra

2.1 Introduction

What is essentially one concept has variously been called *array* [38], *hypermatrix* [37], *multidimensional array* [46], *multidimensional matrix* [47], *multilinear* [48] and (erroneously) *tensor* [48] algebra in the literature. Although disagreeing in terminology and notation, authors have agreed on the usefulness of multilinear algebra, a generalisation of linear algebra that includes arrays of higher order than scalars, vectors and matrices. Multilinear algebra was originally invented as a means of performing matrix differentiation [37] but applications have included the block analysis of system sensitivity [47], the analysis of variance [48], the modelling of distributed parameter systems [49] and the analysis and synthesis of massively parallel computing structures [37]. Of relevance to the modelling of image sensors, Antzoulatos and Sawchuk argue that algebraic manipulation of planar data structures—typical in image processing—requires operations more powerful than those afforded by classical linear algebra [37]. Of relevance to the calibration of image sensors, Blaha and Snay argue, giving examples from least squares estimation, that array equations are sometimes more efficient, in terms of the processor time and memory space required to compute a solution, than corresponding matrix equations [38, 39].

There are important similarities and differences between multilinear algebra and tensor calculus but this connection is either avoided or dealt with superficially in the literature [48]. The contents of either an array or a tensor are numbers that correspond to a point in a specific multidimensional space. These numbers form a tensor only if they obey certain transformation laws under a change of the coordinate system used to describe the point [50]. Tensors have certain properties that are independent of the underlying coordinate system. For these reasons, they are used to represent various fundamental laws in physics and mathematics [50]. On the other hand, arrays may not be tensors and multilinear algebra has little to do with differential geometry. Nonetheless, the index notation and conventions of classical tensor calculus [51] are more powerful for describing operations on arrays than the notation and conventions of various definitions of multilinear algebra in the literature.[1]

The multilinear algebras described by Blaha and Snay [38, 39], Milov [47], Takemura [48] and Suzuki and Shimizu [49] limit the rich types of multiplication possible with tensor notation and conventions, which permit an arbitrary combination of inner and outer products between two arrays [50]. Libkin et al [46] and Baumann [52] define algebras suitable for describing and executing powerful queries on array databases but these algebras are too far from tensor calculus to be of general use mathematically. Antzoulatos and Sawchuk restrict tensor notation and conventions on purpose in the process of defining a powerful but complex algebra [37]. Equivalence and assignment are different in this algebra, which may lead to confusion and error in derivations. For example, multiplication between two arrays may be ambiguous without assignment of the result to a third array, which means binary products are actually ternary operations. Despite these weaknesses, some definitions of multilinear algebra have strong features that standard descriptions of tensor calculus lack.

In linear algebra, matrix inversion is no less important than matrix multiplication. By contrast, conventional expositions of tensor calculus omit inversion though multiplication is a central concept [51, 50]. The reason is because tensor equations may in many cases be rewritten as matrix equations, often done when operations such as inversion are required [40]. Such an approach is not suitable for multilinear algebra where inversion, though neglected by many authors in the field, is no less fundamental than it is in linear algebra, especially when a derivation involves the manipulation of inverses. Blaha and Snay [38, 39] and Suzuki and Shimizu [49] discuss inversion but their consideration is limited by restrictions on multiplication present in their respective algebras. Antzoulatos and Sawchuk implicitly consider array inversion since their algebra imposes a specific mapping between an array and a matrix [37]. This duality serves as a means to transfer an equation into the domain that is most convenient for a particular type of operation or representation and then to transfer back [37]. Despite this feature, inversion like multiplication suffers from the complexity of the algebra. Furthermore, array expressions may not mix freely with matrix expressions in an equation due to the separation of domains.

Antzoulatos and Sawchuk define an array operation that has no analogue in classical linear algebra or tensor calculus—element-wise multiplication [37]. However, their particular definition is inconsistent with tensor notation and conventions, which is one of the reasons why they invent a more complex algebra, and they do not explore the properties of the operation. From an analytical viewpoint, they do not consider its relevance to inner and outer products nor do they appreciate its connection to unary operations such as array contraction. From a computational viewpoint, they do not realise the advantages of element-wise multiplication in calculating the variance of a stochastic array. Lastly, Antzoulatos and Sawchuk [37], like Blaha and Snay before them [38, 39], observe that an automatic mapping exists between array equations and matrix equations but they do not account for element-wise multiplication in their mapping.

Unifying and extending various concepts in the literature of multilinear algebra, this chapter defines a multilinear algebra that is compatible with tensor calculus, formalises

[1] Modern definitions of tensor calculus use an index-free approach that, although elegant, is undesirable in terms of computational applicability [48].

inversion in the array domain, includes element-wise multiplication and encompasses linear algebra (meaning scalars, vectors and matrices may be used with arrays easily). Section 2.2 defines an array formally and outlines the fundamentals of multilinear algebra. Section 2.3 provides implementations of array multiplication and inversion in Matlab. Section 2.4 describes the analytical and computational applications of stochastic, sparse and cell arrays. Specific applications are found in Chapter 3.

2.2 Fundamentals

Linear algebra is a calculus for scalars, vectors and matrices. A scalar needs no indices for it always represents a single element whereas the scalar elements of a vector are identified by one index. The scalar elements of a matrix are identified by two indices—the row and column numbers. Multilinear algebra, by extension, is a calculus for arrays where scalar elements are identified by multiple indices. Data organised into multiple index arrays arises naturally in a variety of scientific disciplines [46]. Furthermore, a variety of artificial sources such as simulators, image renderers and data warehouse population tools generate array data [52].

Formally, a scalar array $\underset{\mathbf{d}}{a_{\mathbf{i}}}$ of order N, for a nonnegative integer N, and positive integer dimensions $\mathbf{d} = (\, d_1 \;\, d_2 \;\, \dots \;\, d_N \,)$ is a function that maps every vector of N integers $\mathbf{i} = (\, i_1 \;\, i_2 \;\, \dots \;\, i_N \,)$, where $1 \leq i_k \leq d_k$, to a scalar element. The dimensionality of an array $\underset{\mathbf{d}}{a_{\mathbf{i}}}$, denoted $\dim \underset{\mathbf{d}}{a_{\mathbf{i}}}$, is the product $\prod_{k=1}^{N} d_k$ of its dimensions and should not be confused with the order of an array. For example, scalars, vectors and matrices are analogous to arrays of order zero, one and two respectively. On the other hand, an array with a dimensionality of one, two or three may represent a point in a geometry of one, two or three spatial dimensions. The dimensionality of a scalar is one by definition.

For simplicity, the array $\underset{\mathbf{d}}{a_{\mathbf{i}}}$ may be referred to as a, $a_{\mathbf{i}}$ or $\underset{\mathbf{d}}{a}$ when the omitted vector of dimensions $\mathbf{d}$ or indices $\mathbf{i}$ is either implied by the context or irrelevant to the discussion. Furthermore, the dimensions may be written as $d_1 \times d_2 \cdots d_N$ to emphasise dimensionality and the indices may be written with no punctuation as $i_1 i_2 \dots i_N$ for brevity. Indices may also be written as superscripts to make a distinction in tensor calculus, relevant to differential geometry, between covariant and contravariant indices.

For equality, addition or subtraction of arrays to be meaningful, as in (2.1), arrays must have corresponding dimensions. In other words, the dimensions of corresponding indices, i.e. indices identified by the same variable, must be the same and all indices must correspond. The equality, addition or subtraction of arrays means the equality, addition or subtraction of their elements, identified by corresponding indices over the entire domain of index values. If a variable is assigned to an index in an array expression then the meaning of the expression does not change with a substitution of the variable. Thus, (2.1) is equivalent to (2.2), where the variable h has been replaced by the variable k. In general, the positions of variables that are assigned to indices of an array matter, as in (2.3), just as the positions of the arguments of a function matter.

$$\underset{L\times M\times N}{c_{hij}} = \underset{M\times N\times L}{a_{ijh}} + \underset{N\times M\times L}{b_{jih}} \tag{2.1}$$

$$\underset{L\times M\times N}{c_{kij}} = \underset{M\times N\times L}{a_{ijk}} + \underset{N\times M\times L}{b_{jik}} \tag{2.2}$$

$$\underset{N\times N}{a_{ij}} \neq \underset{N\times N}{a_{ji}} \tag{2.3}$$

Most unary operations on an array, such as negation, operate element-wise without changing the order or dimensions of the array. Two exceptions are the operations of contraction and attraction, as in (2.4) and (2.5). Contraction of an array over two or more indices of equal dimension, identified by a repeated variable, is equivalent to a summation of array elements over the domain of the variable. Attraction of an array over two or more indices of equal dimension, identified by a repeated and underlined variable, is equivalent to a selection of array elements over the domain of the variable.

$$b_{ij} = a_{ikjk} \tag{2.4}$$

$$b_{ijk} = a_{i\underline{k}j\underline{k}} \tag{2.5}$$

Contraction or attraction of an array results in an array of fewer indices and dimensions by the number of identified indices less one. Multiple contractions and/or attractions over disjoint sets of indices are possible and are distinguished by the use of different variables. In tensor calculus, an array may be contracted over only two indices and attraction does not exist. As seen in the next section, these changes are introduced in multilinear algebra because of element-wise multiplication.

So far, only scalar arrays have been considered. Vector and matrix arrays may be defined in analagous terms to scalar arrays. A homogenous array of vectors $\underset{\mathbf{d}}{\mathbf{a}_{\mathbf{i}}}$ or matrices $\underset{\mathbf{d}}{\mathbf{A}_{\mathbf{i}}}$ is denoted by (optional) dimensions $\mathbf{d}$ and indices $\mathbf{i}$ as with an array of scalars $\underset{\mathbf{d}}{a_{\mathbf{i}}}$. The vectors or matrices indexed by a homogenous array must be of uniform size and this size may be indicated by superscripts. Section 2.4.3 discusses heterogenous arrays, where vectors and/or matrices indexed by an array may not have the same size. Vectors of size N are assumed to be $N \times 1$ column vectors unless specified to be $1 \times N$ row vectors. Equality, addition, subtraction, contraction and attraction of vector and matrix arrays proceed as with scalar arrays.

Note that vectors and matrices are not equal to scalar arrays of order one or two. A distinction exists because the row and/or column indices of vectors and matrices, unlike array indices, are required to obey the rules of linear algebra. However, any index or pair of indices of a scalar array may become a row and/or a column index, as described in the next section, to make a vector or matrix array.

2.2.1 Multiplication

An inner product of two arrays of corresponding dimensions, indicated by repeated index variables as in tensor calculus, is a scalar equal to the sum of all products of corresponding elements. An outer product of two arrays of order M and N, indicated by differing index variables as in tensor calculus, is an array of order $M + N$ that indexes the product of every pair of elements with one element taken from each operand. The dimensionality of the result equals the product of the operand dimensionalities.

Examples of inner and outer products are given in (2.6) and (2.7) respectively.

$$c = a_{ij} b_{ij} \tag{2.6}$$

$$c_{hijk} = a_{hi} b_{jk} \tag{2.7}$$

An element-wise product of two arrays of corresponding dimensions, indicated by repeated and underlined index variables, is an array of corresponding dimensions that indexes the product of every pair of corresponding elements. Since element-wise products represent an <u>inter</u>mediate concept between <u>in</u>ner and ou<u>ter</u> products, they are called inter products hereafter for brevity. An example is given in (2.8).

$$c_{ij} = a_{\underline{ij}} b_{\underline{ij}} \tag{2.8}$$

Inner, outer and inter products are the fundamental types of array multiplication. Note that the product of an array with a scalar is an outer product, as in (2.9). A mixed product of two arrays, as in (2.10), indicates a combination of inner, outer and inter products, each applied over specific indices according to the above conventions.

$$c_{ij} = a_{ij} b \tag{2.9}$$

$$c_{jj'k} = a_{ij\underline{k}} b_{ij'\underline{k}} \tag{2.10}$$

Multiplication of vector or matrix arrays, as in (2.11), obeys the rules of both linear algebra and multilinear algebra. Thus, products of vector and/or matrix arrays do not commute in general and the number of columns on the left side of a product must equal the number of rows on the right side. However, any product of two scalar arrays is always commutative, which proves useful to simplify expressions. This distinction between vector or matrix arrays and scalar arrays is one reason why vectors and matrices are not the same as first and second-order scalar arrays.

$$\mathbf{C}^{M\times N}_{\substack{ijk \\ P\times Q\times R}} = \mathbf{A}^{M\times L}_{\substack{i\underline{k} \\ P\times R}} \mathbf{B}^{L\times N}_{\substack{j\underline{k} \\ Q\times R}} \tag{2.11}$$

Any product between one array and the sum or difference of two other arrays is distributive. Outer products of multiple arrays are associative. Inner, inter and mixed products of multiple arrays are not always strictly associative but always have association identities. For example, the ternary inner product $a_i b_i c_i$ differs from either the left or right associations in (2.12), which also differ from each other. The left association is the outer product of c with the inner product of a and b whereas the right association is the outer product of a with the inner product of b and c.

$$(a_i b_i) c_i \neq a_i (b_i c_i) \tag{2.12}$$

However, the ternary inner product $a_i b_i c_i$ equals the left and right association identities in (2.13), which replace the inner products inside the parentheses of (2.12) with inter products. Similarly, the ternary inter product $a_{\underline{i}} b_{\underline{i}} c_{\underline{i}}$ equals the left and right association identities in (2.14), which indicate an inter product between the array outside parentheses and the inter product inside parentheses with a second underline.

$$(a_{\underline{i}} b_{\underline{i}}) c_i = a_i (b_{\underline{i}} c_{\underline{i}}) \tag{2.13}$$

$$(a_{\underline{\underline{i}}} b_{\underline{\underline{i}}}) c_{\underline{i}} = a_{\underline{i}} (b_{\underline{\underline{i}}} c_{\underline{\underline{i}}}) \tag{2.14}$$

In tensor calculus, which lacks inter products, an index variable may not repeat more than once in a product, not counting pairs that disappear within parentheses. Otherwise, products of multiple arrays, e.g. $a_i b_i c_i$, may not associate into an equivalent sequence of binary operations, which is important for derivations and computations. Mixtures of inner and outer products are strictly associative, as in (2.15), when no index variable repeats more than once. The same may be said for mixtures of inter and outer products, as in (2.16), or for any mixed product. Multilinear algebra does not restrict the repetition of index variables since inter products enable association identities.

$$(a_i b_i) c_j = a_i (b_i c_j) \tag{2.15}$$
$$(a_{\underline{i}} b_{\underline{i}}) c_j = a_{\underline{i}} (b_{\underline{i}} c_j) \tag{2.16}$$

The binary operations of inner and inter products logically follow from the unary operations of contraction and attraction and the binary operation of outer products. An inner product may be rewritten as the contraction of an outer product, as in (2.17), and an inter product may be rewritten as the attraction of an outer product, as in (2.18). These properties help to derive association identities for mixed products, since outer products are strictly associative, and to simplify array expressions. Tensor calculus does not allow an array to be contracted over more than two indices because of the connection to inner products, which are restricted for the sake of associativity.

$$a_k b_k = c_{kk} \,|\, c_{ij} = a_i b_j \tag{2.17}$$
$$a_{\underline{k}} b_{\underline{k}} = c_{\underline{kk}} \,|\, c_{ij} = a_i b_j \tag{2.18}$$

The advantage of defining inner and inter products directly, i.e. without resorting to the use of outer products, lies in computation. Computing an outer product of two arrays, each of dimensionality N, requires the product of every pair of elements, with one element taken from each array, and takes $O(N^2)$ time and space. Computing an inner or inter product of the same two arrays, however, requires the product of only corresponding elements and takes $O(N)$ time and space, with or without summation.

Scalar arrays may transform into vector or matrix arrays via an inner product with a vector or matrix basis array. The vector basis array, denoted $()_i^N$, indexes over $1 \leq i \leq N$ the vectors of size N that are zero except for the i^{th} element, which is one. Likewise, the matrix basis array, denoted $()_{ij}^{M \times N}$, indexes over $1 \leq i \leq M$ and $1 \leq j \leq N$ the matrices of size $M \times N$ that are zero except for the $(i,j)^{\text{th}}$ element, which is one. An inner product of a scalar array with a basis array, as in (2.19) and (2.20), assigns one or two array indices to vector or matrix indices (i.e. rows or columns) respectively.

$$\mathbf{a}_{ij} = a_{hij} ()_h \tag{2.19}$$
$$\mathbf{A}_h = a_{hij} ()_{ij} \tag{2.20}$$

The basis arrays also serve a tabular purpose by arranging the elements of a scalar array into a vector or matrix for convenient display, as in (2.21) and (2.22). The superscript T in (2.21) denotes transposition of the column vectors indexed by the basis

Table 2.1: Possible binary operations on arrays where the operands and result have an order of zero, one or two. New operators are needed in linear algebra to express operations that involve inter products. Assume that $\mathbf{a}$, $\mathbf{b}$ and $\mathbf{c}$ equal $a_i()_i$, $b_i()_i$ and $c_i()_i$ and that $\mathbf{A}$, $\mathbf{B}$ and $\mathbf{C}$ equal $a_{ij}()_{ij}$, $b_{ij}()_{ij}$ and $c_{ij}()_{ij}$ respectively.

Op.	Multilinear	Linear (old)	Linear (new)
1	$c = ab$	$c = ab$	$c = ab$
2	$c = a_i b_i$	$c = \mathbf{a}^{\mathrm{T}}\mathbf{b}$	$c = \mathbf{a} \bullet \mathbf{b}$
3	$c = a_{ij} b_{ij}$	$c = \mathrm{tr}\,\mathbf{A}^{\mathrm{T}}\mathbf{B}$	$c = \mathbf{A} \bullet \mathbf{B}$
4	$c_i = a_i b$	$\mathbf{c} = \mathbf{a}b$	$\mathbf{c} = \mathbf{a}b$
5	$c_i = a_{\underline{i}} b_{\underline{i}}$		$\mathbf{c} = \mathbf{a} \circ \mathbf{b}$
6	$c_i = a_{ij} b_j$	$\mathbf{c} = \mathbf{A}\mathbf{b}$	$\mathbf{c} = \mathbf{A}\mathbf{b}$
7	$c_i = a_{\underline{i}j} b_{\underline{i}j}$		$\mathbf{c} = \mathbf{A} \diamond \mathbf{B}$
8	$c_{ij} = a_i b_j$	$\mathbf{C} = \mathbf{a}\mathbf{b}^{\mathrm{T}}$	$\mathbf{C} = \mathbf{a}\mathbf{b}^{\mathrm{T}}$
9	$c_{ij} = a_{ij} b$	$\mathbf{C} = \mathbf{A}b$	$\mathbf{C} = \mathbf{A}b$
10	$c_{ij} = a_{\underline{ij}} b_{\underline{i}}$		$\mathbf{C} = \mathbf{A} \triangleright \mathbf{b}$
11	$c_{ij} = a_{ik} b_{kj}$	$\mathbf{C} = \mathbf{A}\mathbf{B}$	$\mathbf{C} = \mathbf{A}\mathbf{B}$
12	$c_{ij} = a_{\underline{ij}} b_{\underline{ij}}$		$\mathbf{C} = \mathbf{A} \circ \mathbf{B}$

array. Transposition swaps row and column indices but has no effect on array indices.

$$\underset{L\times M\times N}{a_{hij}} \ ()_h^{\mathrm{T}} = \begin{pmatrix} a_{1ij} & a_{2ij} & \cdots & a_{Lij} \end{pmatrix} \tag{2.21}$$

$$\underset{L\times M\times N}{a_{hij}} \ ()_{ij} = \begin{pmatrix} a_{h11} & a_{h12} & \cdots & a_{h1N} \\ a_{h21} & a_{h22} & \cdots & a_{h2N} \\ \vdots & \vdots & \ddots & \vdots \\ a_{hM1} & a_{hM2} & \cdots & a_{hMN} \end{pmatrix} \tag{2.22}$$

In this manner, vector and matrix arrays are related very simply to scalar arrays. Because scalar arrays of order zero, one and two are analogous to scalars, vectors and matrices, linear algebra may express some (but not all) products of such arrays that result in an array of order zero, one or two. Table 2.1 demonstrates that classical linear algebra may express eight out of twelve possible products. The remaining operations, which involve inter products, may be expressed with three new operators because Operation 5 is equivalent to Operations 7, 10 and 12 when the dimension of j is one.

Operation 3 in Table 2.1 exists in classical linear algebra but, for $N \times N$ matrices $\mathbf{A}$ and $\mathbf{B}$, computing the trace of a matrix product takes $O(N^3)$ time whereas an inner product of the equivalent arrays needs $O(N^2)$ time. The symbol $\bullet$ often denotes the inner product of vectors, as in Operation 2, and may be used to denote the inner product of matrices. Operations 2 and 3 may be used to transform a vector or matrix array into a scalar array, as in (2.23) and (2.24), using the vector or matrix basis arrays. The complementary pairs of transformations in (2.19) and (2.20) and in (2.23) and (2.24)

Table 2.2: Useful unary operations on arrays of order one or two. The results are arrays of order zero or one. New operators are needed in linear algebra to express operations that involve inter products. Assume that $\mathbf{a}$ and $\mathbf{A}$ equal $a_i()_i$ and $a_{ij}()_{ij}$ respectively.

Op.	Multilinear	Linear (old)	Linear (new)
1	$b = a_{ii}$	$b = \operatorname{tr}\mathbf{A}$	$b = \operatorname{tr}\mathbf{A}$
2	$b = a_i a_i$	$b = \lVert\mathbf{a}\rVert^2$	$b = \lVert\mathbf{a}\rVert^2$
3	$b = a_{ij} a_{ij}$	$b = \operatorname{tr}\mathbf{A}^{\mathrm{T}}\mathbf{A}$	$b = \lVert\mathbf{A}\rVert^2$
4	$b_i = a_{\underline{ii}}$		$\mathbf{b} = \operatorname{diag}\mathbf{A}$
5	$b_i = a_{\underline{i}} a_{\underline{i}}$		$\mathbf{b} = \langle\mathbf{a}\rangle^2$
6	$b_i = a_{\underline{i}j} a_{\underline{i}j}$		$\mathbf{b} = \langle\mathbf{A}\rangle^2$

both involve inner products, either over array indices or over vector or matrix indices.

$$a_{hij} = \mathbf{a}_{ij} \bullet ()_h \tag{2.23}$$

$$a_{hij} = \mathbf{A}_h \bullet ()_{ij} \tag{2.24}$$

Operation 7 in Table 2.1, denoted by the symbol $\diamond$, takes an inter product over the row indices of $\mathbf{A}$ and $\mathbf{B}$ and an inner product over the column indices. Operation 10, denoted by the symbol $\triangleright$, takes an inter product over the row indices of $\mathbf{A}$ and $\mathbf{b}$ and an outer product over the column index of $\mathbf{A}$. A similar product may be defined, denoted by the symbol $\triangleleft$, that takes the operands in the reverse order. Operation 12, denoted by the symbol $\circ$, takes an inter product of $\mathbf{A}$ and $\mathbf{B}$. Properties of commutation, distribution and association of these operators may be derived readily in light of their multilinear equivalents and earlier discussion. Minor variations of the operations in Table 2.1 exist, which may be expressed with the listed operators and transposition.

As contraction and attraction operate on two or more indices of an array, there are analogues for these unary operations with matrices, which have two indices. Furthermore, several binary operations in Table 2.1 imply unary operations when both operands are the same. Table 2.2 lists unary operations that appear in this thesis, three of which do not exist in classical linear algebra. Operation 1 gives the trace of a matrix, analogous to contraction. Operation 2 gives the squared norm of a vector. For an $N \times N$ matrix in Operation 3, computing the trace of a matrix product takes $O(N^3)$ time whereas an inner product of the equivalent arrays needs only $O(N^2)$ time. Thus, the squared norm of a matrix is defined for efficiency. Operation 4 gives the diagonal elements of a matrix, analogous to attraction. Operations 5 and 6 give the squared form of a vector and matrix, defined to be the squared norm of each row.

2.2.2 Inversion

Multiplication and inversion are connected. The purpose of finding an inverse is usually to cancel one term in a product via multiplication. Alternately, inverses are connected to identities as the product of an array and its inverse should yield an identity, which is an array that leaves another array unchanged upon multiplication. In multilinear algebra, there are often more than one identity for a given array, depending on the type of multiplication, and so there are often more than one inverse. Some identities and types of multiplication specify non-unique inverses of certain arrays. For example, the unit scalar is an identity that defines the inner product inverse of a vector $\mathbf{a}$, denoted $\mathbf{a}^{-1}$ in (2.25), which is not unique in general.

$$\mathbf{a} \bullet \mathbf{a}^{-1} = 1 \tag{2.25}$$

A useful class of identities and types of multiplication are those that define unique inverses of given arrays. For example, the $N \times N$ identity matrix $\mathbf{I}$ and ordinary matrix multiplication define a unique inverse of any $N \times N$ matrix $\mathbf{A}$, denoted $\mathbf{A}^{-1}$ in (2.26), if it exists. Usually, the definition of an inverse includes a dual relation, as in (2.27), where a complement of the operation (e.g. by commutation) also produces an identity.

$$\mathbf{A}\mathbf{A}^{-1} = \mathbf{I} \tag{2.26}$$

$$\mathbf{A}^{-1}\mathbf{A} = \mathbf{I} \tag{2.27}$$

An identity that leaves an array unchanged with only an outer product is the unit scalar. There is no identity that leaves an array unchanged with only an inner product because a strict inner product of two arrays results in a scalar. An identity that leaves an array $a_{\mathbf{i}}$, i.e. $a_{i_1 i_2 \ldots i_N}$, unchanged with only inner and outer products is the product of delta arrays $\delta_{i_1 i_1'} \delta_{i_2 i_2'} \ldots \delta_{i_N i_N'}$, denoted $\delta_{\mathbf{i},\mathbf{i}'}$ for short, where elements indexed by $\delta_{ii'}$ are zero when $i \neq i'$ and one when $i = i'$. Multiplication by a delta array implies a substitution of index variable, as in (2.28). An identity that leaves an array $a_{\mathbf{i}}$ unchanged with only an inter product is the unit array $1_{\mathbf{i}}$, which is one for all values of $\mathbf{i}$. Thus, in multilinear algebra, identities are products of delta arrays and unit arrays.

$$a_i \delta_{ij} = a_j \tag{2.28}$$

Let a and a^{-1} be arrays of equal dimension with index variables $\mathbf{h}$ composed of distinct variables $\mathbf{i}$, $\mathbf{j}$ and $\mathbf{k}$, arranged in any sequence. Then $\tilde{a}$ and $\tilde{a}^{-1}$ in (2.29) and (2.30) are identical to a and a^{-1} with a permutation of the index variable sequence. These permutations facilitate the definition of inverse.

$$\tilde{a}_{\mathbf{ijk}} = a_{\mathbf{h}} \tag{2.29}$$

$$\tilde{a}^{-1}_{\mathbf{ijk}} = a^{-1}_{\mathbf{h}} \tag{2.30}$$

An inverse of the array $a_{\mathbf{h}}$ for the identity $\delta_{\mathbf{j},\mathbf{j}'} 1_{\mathbf{k}}$ is the array $a^{-1}_{\mathbf{h}}$ when relations (2.31) and (2.32) hold. The symmetry of these relations means that an inverse of $a_{\mathbf{h}}$ for $\delta_{\mathbf{j},\mathbf{j}'} 1_{\mathbf{k}}$ is also an inverse for $\delta_{\mathbf{i},\mathbf{i}'} 1_{\mathbf{k}}$. Thus, $a^{-1}_{\mathbf{h}}$ may also be called an inverse of $a_{\mathbf{h}}$ over $\mathbf{i}$ and $\mathbf{j}$, where $\mathbf{k}$ and the two identities are implied.

$$\tilde{a}_{\mathbf{ij}\underline{\mathbf{k}}} \tilde{a}^{-1}_{\mathbf{ij}'\underline{\mathbf{k}}} = \delta_{\mathbf{j},\mathbf{j}'} 1_{\mathbf{k}} \tag{2.31}$$

$$\tilde{a}_{\mathbf{ij}\underline{\mathbf{k}}} \tilde{a}^{-1}_{\mathbf{i}'\mathbf{j}\underline{\mathbf{k}}} = \delta_{\mathbf{i},\mathbf{i}'} 1_{\mathbf{k}} \tag{2.32}$$

More than one inverse may exist for a given array. For example, a^{-1}_{hij} is an inverse of a_{hij} for $\delta_{hh'}$ or $\delta_{ii'}\delta_{jj'}$, or over h and ij, when (2.33) and (2.34) hold. However,

a^{-1}_{hij} is also an inverse of a_{hij} for 1_{hij}, or over no indices, when (2.35) holds. In the latter case, the relations and identities in (2.31) and (2.32) are identical.

$$a_{hij}a^{-1}_{h'ij} = \delta_{hh'} \tag{2.33}$$

$$a_{hij}a^{-1}_{hi'j'} = \delta_{ii'}\delta_{jj'} \tag{2.34}$$

$$a_{\underline{hij}}a^{-1}_{\underline{hij}} = 1_{hij} \tag{2.35}$$

The context of a derivation usually implies which particular inverse and identity is being used. For example, given a and c in (2.36), b may be derived in (2.37) using a^{-1} as defined by (2.35) but not by (2.33) and (2.34). Nonetheless, the inverse may be specified explicitly, as in (2.38), by including the implied identity.

$$a_{\underline{hij}}b_{\underline{hij}k} = c_{hijk} \tag{2.36}$$

$$b_{hijk} = a^{-1}_{\underline{hij}}c_{\underline{hij}k} \tag{2.37}$$

$$b_{hijk} = (1_{\underline{\underline{hij}}}/a_{\underline{\underline{hij}}})c_{\underline{hij}k} \tag{2.38}$$

An inverse of $a_{\mathbf{h}}$ for $\delta_{\mathbf{j},\mathbf{j}'}1_{\mathbf{k}}$ is unique if it exists. If different inverses $b_{\mathbf{h}}$ and $c_{\mathbf{h}}$ exist then (2.39) and (2.40) hold by definition (2.31) for each inverse, where $\tilde{a}$, $\tilde{b}$ and $\tilde{c}$ equal a, b and c with a permutation of index variables $\mathbf{h}$ into the sequence $\mathbf{i}$, $\mathbf{j}$ and $\mathbf{k}$.

$$\tilde{a}_{\mathbf{ij}\underline{\mathbf{k}}}\tilde{b}_{\mathbf{ij}'\underline{\mathbf{k}}} = \delta_{\mathbf{j},\mathbf{j}'}1_{\mathbf{k}} \tag{2.39}$$

$$\tilde{a}_{\mathbf{ij}\underline{\mathbf{k}}}\tilde{c}_{\mathbf{ij}'\underline{\mathbf{k}}} = \delta_{\mathbf{j},\mathbf{j}'}1_{\mathbf{k}} \tag{2.40}$$

The left sides of (2.39) and (2.40) are equal because the right sides are the same. Equating the left sides and multiplying by $\tilde{b}$ with an inner product over $\mathbf{j}$ and an inter product over $\mathbf{k}$ gives (2.41), which may be rewritten in (2.42) with an association identity.

$$\tilde{b}_{\mathbf{i}'\mathbf{j}\underline{\mathbf{k}}}(\tilde{a}_{\mathbf{ij}\underline{\underline{\mathbf{k}}}}\tilde{b}_{\mathbf{ij}'\underline{\underline{\mathbf{k}}}}) = \tilde{b}_{\mathbf{i}'\mathbf{j}\underline{\mathbf{k}}}(\tilde{a}_{\mathbf{ij}\underline{\underline{\mathbf{k}}}}\tilde{c}_{\mathbf{ij}'\underline{\underline{\mathbf{k}}}}) \tag{2.41}$$

$$(\tilde{b}_{\mathbf{i}'\mathbf{j}\underline{\underline{\mathbf{k}}}}\tilde{a}_{\mathbf{ij}\underline{\underline{\mathbf{k}}}})\tilde{b}_{\mathbf{ij}'\underline{\mathbf{k}}} = (\tilde{b}_{\mathbf{i}'\mathbf{j}\underline{\underline{\mathbf{k}}}}\tilde{a}_{\mathbf{ij}\underline{\underline{\mathbf{k}}}})\tilde{c}_{\mathbf{ij}'\underline{\mathbf{k}}} \tag{2.42}$$

The product in parentheses on each side of (2.42) equals $\delta_{\mathbf{i},\mathbf{i}'}1_{\mathbf{k}}$, as in (2.43), by definition (2.32) for the inverse b (with commutation). Multiplication of $\tilde{b}$ and $\tilde{c}$ on each side of (2.43) by this identity proves that $\tilde{b}$ equals $\tilde{c}$, as in (2.44), contradicting the premise that b and c are different. Thus, the inverse of $a_{\mathbf{h}}$ over $\mathbf{i}$ and $\mathbf{j}$ is unique if it exists.

$$\delta_{\mathbf{i},\mathbf{i}'}1_{\underline{\mathbf{k}}}\tilde{b}_{\mathbf{ij}'\underline{\mathbf{k}}} = \delta_{\mathbf{i},\mathbf{i}'}1_{\underline{\mathbf{k}}}\tilde{c}_{\mathbf{ij}'\underline{\mathbf{k}}} \tag{2.43}$$

$$\tilde{b}_{\mathbf{i}'\mathbf{j}'\mathbf{k}} = \tilde{c}_{\mathbf{i}'\mathbf{j}'\mathbf{k}} \tag{2.44}$$

2.3 Implementation

As described in Section 2.1, several authors in the literature have discussed the mapping of array (and tensor) expressions to matrix expressions and vice versa. Some authors describe a manual mapping for a specific problem whereas others describe an automatic mapping for more general problems. In the latter case, the authors do not appreciate that many automatic mappings exist. This section summarises all possible mappings, neatly described by a class of arrays called encoding and decoding arrays.

An array $i_{\mathbf{i}}$ of order N, i.e. $i_{i_1 i_2 \ldots i_N}$, and dimensionality D is called an encoding array if it is a one-to-one mapping of index vectors $\mathbf{i}$ to integer elements i, where $1 \leq i \leq D$. For any encoding array $i_{\mathbf{i}}$, there is a corresponding decoding (vector) array $\mathbf{i}_i$, which defines the converse one-to-one mapping. An encoding array $i_{\mathbf{i}}$ may be used to encode multiple indices $\mathbf{i}$ of another array $a_{\mathbf{i}}$ into one index i, as in (2.45), creating an array b_i of lower order but equal dimensionality. Conversely, a decoding array $\mathbf{i}_i$ may be used to decode one index i of another array b_i into multiple indices $\mathbf{i}$, as in (2.46), creating an array $a_{\mathbf{i}}$ of higher order but equal dimensionality. Thus, encoding and decoding of indices are complementary and reversible operations.

$$b_{i_{\mathbf{i}}} = a_{\mathbf{i}} \tag{2.45}$$

$$a_{\mathbf{i}_i} = b_i \tag{2.46}$$

Any array expression may be automatically transformed with encoding arrays to a lattice expression, where a lattice is the name given to a first order matrix array. Lattice expressions may be evaluated by a sequence of ordinary matrix expressions, one for each tab of the lattice, which is the name given to the third index of the lattice (after the row and column indices). The results may be transformed with decoding arrays back into the original array domain. By making a specific choice of encoding and decoding that is optimal in the language, array multiplication and inversion are implemented in MATLAB. Users of MATLAB may therefore work with the high level description of multilinear algebra, given in Section 2.2, with the confidence that an accurate and efficient implementation exists. Implementations of multilinear algebra for other programming languages may be readily derived.

2.3.1 Multiplication

Consider two arbitrary arrays $a_{\mathbf{x}}$ and $b_{\mathbf{y}}$ and the array $c_{\mathbf{z}}$ resulting from an arbitrary product, as in (2.47), indicated by repeated and non-repeated variables between indices $\mathbf{x}$ and $\mathbf{y}$ where some repeated variables are underlined.[2] Denoting the repeated variables that specify inner and inter products by indices $\mathbf{h}$ and $\mathbf{k}$ respectively and the non-repeated variables that specify outer products of a and b by indices $\mathbf{i}$ and $\mathbf{j}$ respectively, the original product may be rewritten with a mapping as the product of two lattices $\mathbf{A}_k$ and $\mathbf{B}_k$, as in (2.48), resulting in a third lattice $\mathbf{C}_k$.

$$c_{\mathbf{z}} = a_{\mathbf{x}} b_{\mathbf{y}} \tag{2.47}$$

$$\mathbf{C}_k = \mathbf{A}_{\underline{k}} \mathbf{B}_{\underline{k}} \tag{2.48}$$

The mapping is given by (2.49)–(2.51) using the matrix basis array and four encoding arrays $h_{\mathbf{h}}$, $i_{\mathbf{i}}$, $j_{\mathbf{j}}$ and $k_{\mathbf{k}}$, which are arbitrary except for their dimensions. In these mappings, multiple indices $\mathbf{i}$ of $a_{\mathbf{x}}$ and $\mathbf{j}$ of $b_{\mathbf{y}}$ are encoded into single row and column

[2] For example, $\mathbf{x} = h_1 \ldots h_L i_1 \ldots i_M \underline{k_1 \ldots k_P}$ and $\mathbf{y} = h_1 \ldots h_L j_1 \ldots j_N \underline{k_1 \ldots k_P}$, which means $\mathbf{z} = i_1 \ldots i_M j_1 \ldots j_N k_1 \ldots k_P$. In general, the indices of each array may be in any sequence.

indices of $\mathbf{A}_k$ and $\mathbf{B}_k$ respectively to achieve an outer product in (2.48). Multiple indices $\mathbf{h}$, repeated in $a_\mathbf{x}$ and $b_\mathbf{y}$, are encoded into single column and row indices of $\mathbf{A}_k$ and $\mathbf{B}_k$ respectively to achieve an inner product in (2.48). Lastly, multiple indices $\mathbf{k}$, repeated and underlined in $a_\mathbf{x}$ and $b_\mathbf{y}$, are encoded into single tab indices of $\mathbf{A}_k$ and $\mathbf{B}_k$ respectively to achieve an inter product in (2.48).

$$\mathbf{A}_{k_\mathbf{k}} = a_\mathbf{x}()_{i_\mathbf{i} h_\mathbf{h}} \tag{2.49}$$

$$\mathbf{B}_{k_\mathbf{k}} = b_\mathbf{y}()_{h_\mathbf{h} j_\mathbf{j}} \tag{2.50}$$

$$\mathbf{C}_{k_\mathbf{k}} = c_\mathbf{z}()_{i_\mathbf{i} j_\mathbf{j}} \tag{2.51}$$

The lattice equation in (2.48) implies a sequence of ordinary matrix multiplications, indexed by k, that may be implemented efficiently in MATLAB, as in Figure 2.1. In asymptotic terms, the original scalar equation and the final lattice equation require the same number of floating point operations $O(HIJK)$ and byte storage $O(HIK + HJK + IJK)$, where H, I, J and K are the dimensionalities of encoding arrays $h_\mathbf{h}$, $i_\mathbf{i}$, $j_\mathbf{j}$ and $k_\mathbf{k}$ respectively. This is because the lattice equation does not introduce any additions or multiplications and elements of lattices $\mathbf{A}_k$, $\mathbf{B}_k$ and $\mathbf{C}_k$ have a one-to-one correspondence with elements of arrays $a_\mathbf{x}$, $b_\mathbf{y}$ and $c_\mathbf{z}$. Furthermore, the mapping in (2.49)–(2.51) may be implemented in $O(HIK + HJK + IJK)$ time and space with MATLAB, using the `permute` and `reshape` functions.

Note from (2.48) that if no inter product is involved in an array multiplication then the operation is equivalent to a single matrix multiplication. Therefore, matrix equations (which are effectively second order) underlie array equations involving only inner and outer products and lattice equations (which are effectively third order) underlie array equations that also involve inter products.

2.3.2 Inversion

With the permutations in (2.29) and (2.30), the relations in (2.31) and (2.32) that define the inverse $a_\mathbf{h}^{-1}$ of an array $a_\mathbf{h}$ for the identity $\delta_{\mathbf{j},\mathbf{j}'}1_\mathbf{k}$ or $\delta_{\mathbf{i},\mathbf{i}'}1_\mathbf{k}$, or over $\mathbf{i}$ and $\mathbf{j}$, transform to lattice equations (2.52) and (2.53) by the mapping given below.

$$\mathbf{A}_{\underline{k}}\mathbf{A}_{\underline{k}}^{-1} = 1_k\mathbf{I} \tag{2.52}$$

$$\mathbf{A}_{\underline{k}}^{-1}\mathbf{A}_{\underline{k}} = 1_k\mathbf{I} \tag{2.53}$$

For encoding arrays $i_\mathbf{i}$, $j_\mathbf{j}$ and $k_\mathbf{k}$ that are arbitrary except for their dimensions, lattices $\mathbf{A}_k$ and $\mathbf{A}_k^{-1}$ in (2.52) and (2.53) are one-to-one mappings in (2.54) and (2.55) of arrays $a_\mathbf{h}$ and $a_\mathbf{h}^{-1}$. The difference of index order in the matrix basis arrays of (2.54) and (2.55) serve to avoid a transposition in (2.52) and (2.53).

$$\mathbf{A}_{k_\mathbf{k}} = a_\mathbf{h}()_{i_\mathbf{i} j_\mathbf{j}} \tag{2.54}$$

$$\mathbf{A}_{k_\mathbf{k}}^{-1} = a_\mathbf{h}^{-1}()_{j_\mathbf{j} i_\mathbf{i}} \tag{2.55}$$

The lattice transformation shows that array inverses may be computed by mapping the relations (2.31) and (2.32) to a sequence of matrix equations, indexed by k in (2.52)

Function `c = atimes(a,b,ha,hb,ka,kb)`, where `a` and `b` are arrays of order `M` and `N` with dimensions `da` and `db`, `ha` and `ka` consist of distinct integers in `[1,M]` with `i` taking those integers that remain, `hb` and `kb` consist of distinct integers in `[1,N]` with `j` taking those integers that remain, returns an array `c` that is the multiplication of `a` and `b`, with an inner product over indices `ha` of `a` and `hb` of `b` and an inter product over indices `ka` of `a` and `kb` of `b`. The indices of `c` correspond, in sequence, to indices `i` of `a`, `j` of `b` and `ka` of `a` (or `kb` of `b`). Inter products may be omitted with the syntax `c = atimes(a,b,ha,hb)`.

```
function c = atimes(a,b,ha,hb,ka,kb)

if nargin <= 4
   ka = [];
   kb = [];
end

M = max([ha ka ndims(a)]);
N = max([hb kb ndims(b)]);
da = [size(a) ones(1,M-ndims(a))];
db = [size(b) ones(1,N-ndims(b))];
dh = da(ha);
dk = da(ka);

if isequal(dh,db(hb)) & isequal(dk,db(kb))
   x = 1:M;
   y = 1:N;
   x([ha ka]) = 0;
   y([hb kb]) = 0;
   i = x(logical(x));
   j = y(logical(y));
   di = da(i);
   dj = db(j);
   H = prod(dh);
   I = prod(di);
   J = prod(dj);
   K = prod(dk);
   a = permute(a,[i ha ka]);
   b = permute(b,[hb j kb]);
   a = reshape(a,I,H,K);
   b = reshape(b,H,J,K);
   c = zeros(I,J,K);

   for k = 1:K
      c(:,:,k) = a(:,:,k)*b(:,:,k);
   end

   c = reshape(c,[di dj dk 1 1]);
else
   error('Incompatible dimensions.');
end
```

Figure 2.1: Array multiplication implemented in MATLAB.

and (2.53), each of which may be solved independently by ordinary matrix inversion. The lattice inverse may be mapped back to the array domain using the matrix basis array and decoding arrays. Although the lattice inverse $\mathbf{A}_k^{-1}$ will depend on the encoding arrays used in the transformation, the array inverse $a_{\mathbf{h}}^{-1}$ will not because of its uniqueness. Array inversion may be implemented efficiently in MATLAB, as in Figure 2.2, since the forward and backward mappings are insignificant compared to the matrix inversions. A single function call abstracts the details of inversion from the user.

Existence of the array inverse in (2.31) and (2.32) hinges on the existence of each matrix inverse in the sequence of matrix equations implied by (2.52) and (2.53). Thus, for the array $a_{\mathbf{h}}$ to be invertible over $\mathbf{i}$ and $\mathbf{j}$, the number of rows and columns of the lattice $\mathbf{A}_k$ in (2.54) must equate, which implies the dimensionalities of $i_{\mathbf{i}}$ and $j_{\mathbf{j}}$ must equate. Providing this holds, the array is invertible if and only if the determinant of each matrix indexed by the lattice is nonzero. With this observation, the squared determinant of an array $a_{\mathbf{h}}$ over indices $\mathbf{i}$ and $\mathbf{j}$, denoted $\det^2_{\mathbf{i},\mathbf{j}} a_{\mathbf{h}}$ in (2.56), is defined as the cumulative product of the squared determinant of matrices indexed by the lattice $\mathbf{A}_k$, which is a mapping of $a_{\mathbf{h}}$ using encoding arrays $i_{\mathbf{i}}$, $j_{\mathbf{j}}$ and $k_{\mathbf{k}}$ and the matrix basis array. Therefore, the inverse of an array $a_{\mathbf{h}}$ over indices $\mathbf{i}$ and $\mathbf{j}$ exists if and only if the squared determinant of $a_{\mathbf{h}}$ over $\mathbf{i}$ and $\mathbf{j}$ exists and is nonzero.

$$\det{}^2_{\mathbf{i},\mathbf{j}} a_{\mathbf{h}} = \begin{cases} \prod_{k=1}^{\dim k_{\mathbf{k}}} (\det \mathbf{A}_k)^2 \mid \mathbf{A}_{k_{\mathbf{k}}} = a_{\mathbf{h}}()_{i_{\mathbf{i}} j_{\mathbf{j}}}, & \dim i_{\mathbf{i}} = \dim j_{\mathbf{j}} \\ \text{undefined}, & \text{otherwise} \end{cases} \tag{2.56}$$

The reason for the square in (2.56) is that lattices $\mathbf{A}_k$ that result from different encodings of the same array $a_{\mathbf{h}}$ are related to each other by a permutation of row, column and tab (the third index of the lattice) numbers. As the sign of a matrix determinant may change with a permutation of the rows and columns of the matrix, the sign of the cumulative product of matrix determinants depends on the choice of encoding arrays. Squaring the determinants ensures uniqueness. In general, a unique definition for an array operation that is independent of any lattice transform seems preferable.

2.4 Applications

As described in Section 2.1, predecessors of the multilinear algebra described herein have had several applications. Although it would be possible to review these applications in terms of the formulation given here, showing its efficiency in deriving previously complex results, these applications have little relevance to the modelling and calibration of image sensors. Thus, new applications are considered below, some straightforward and complete and others difficult and incomplete. These ideas are employed in Chapter 3 to find an efficient solution of the generic and raster sensor array problems.

2.4.1 Statistical variance

A stochastic array is an array with elements drawn randomly from some joint probability density function. Alternately, a stochastic array is a sample from a (possibly infinite) population of arrays of equal dimensions. As with scalar random variables, the

Function `ainv(a,i,k)`, where `a` is an array of order `N` and dimensions `d`, vectors `i` and `k` consist of distinct integers in `[1,N]` and vector `j` consists of those integers that remain, returns an array of equal order, dimensions and index sequence to `a` that is the inverse of `a` over index positions `i` and `j`. The syntax `ainv(a,i)` assumes an empty vector `k`.

```
function a = ainv(a,i,k)

if nargin <= 2
   k = [];
end

N = max([i k ndims(a)]);
d = [size(a) ones(1,N-ndims(a))];
h = 1:N;
h([i k]) = 0;
j = h(logical(h));
di = d(i);
dj = d(j);
I = prod(di);

if I == prod(dj)
   dk = d(k);
   K = prod(dk);
   a = permute(a,[i j k]);
   a = reshape(a,I,I,K);

   for k = 1:K
      a(:,:,k) = inv(a(:,:,k));
   end

   a = reshape(a,[dj di dk 1 1]);
   a = ipermute(a,[j i k]);
else
   error(’Impossible inversion.’)
end
```

Figure 2.2: Array inversion implemented in MATLAB.

expected value $\mathcal{E}\{x\}$ of a stochastic array x is simply the mean array $\bar{x}$ of the population. The expected value of a linear function of a stochastic array x with non-stochastic coefficient arrays a and b, as in (2.57), may be simplified in the usual way.

$$\mathcal{E}\{a_{\mathbf{ijk}} + b_{\mathbf{hi\underline{k}}} x_{\mathbf{hj\underline{k}}}\} = a_{\mathbf{ijk}} + b_{\mathbf{hi\underline{k}}} \mathcal{E}\{x_{\mathbf{hj\underline{k}}}\} \tag{2.57}$$

Armed with the expectation operator, three different types of variance may be defined for a stochastic array x: an outer variance $\mathcal{C}\{x\}$ called the *covariance*, an inter variance $\mathcal{V}\{x\}$ called simply the *variance* and an inner variance $\mathcal{S}\{x\}$ called the *scalar-variance*. These are obtained in (2.58)–(2.60) from the expectation of an outer, inter and inner product of x, less the expected value of x (i.e. $\bar{x}$), with itself.

$$\mathcal{C}\{x_{\mathbf{h}}\} = \mathcal{E}\{(x_{\mathbf{h}} - \bar{x}_{\mathbf{h}})(x_{\mathbf{h}'} - \bar{x}_{\mathbf{h}'})\} \tag{2.58}$$

$$\mathcal{V}\{x_{\mathbf{h}}\} = \mathcal{E}\{(x_{\underline{\mathbf{h}}} - \bar{x}_{\underline{\mathbf{h}}})(x_{\underline{\mathbf{h}}} - \bar{x}_{\underline{\mathbf{h}}})\} \tag{2.59}$$

$$\mathcal{S}\{x_{\mathbf{h}}\} = \mathcal{E}\{(x_{\mathbf{h}} - \bar{x}_{\mathbf{h}})(x_{\mathbf{h}} - \bar{x}_{\mathbf{h}})\} \tag{2.60}$$

In terms of linear algebra, if the stochastic array x is effectively a stochastic vector then the outer variance is the covariance matrix of the vector, the inter variance is a vector consisting of the diagonal elements of the covariance matrix and the inner variance is a scalar equal to the trace of the covariance matrix. If the stochastic vector is of size N then calculating the covariance matrix needs at least $O(N^2)$ time and space. Because linear algebra has no provision for the inter product of vectors, calculating the inter variance using only the operators of classical linear algebra implies calculating the whole covariance matrix with a minimum $O(N^2)$ complexity.

In cases where the outer variance is unnecessary, which is often true in the statistical description of stochastic variables estimated by regression, then multilinear algebra provides an opportunity to calculate the inter variance in $O(N)$ time and space. In general, calculating the inner variance also requires at least $O(N)$ time and space, counting the time and space required to process and store the arguments of the inner product. Calculating the inner variance may be more efficient with multilinear algebra than with classical linear algebra as the former provides an operator to compute the inner product of two matrices (the stochastic vector may be a function of other matrices) without requiring matrix multiplication. The inter variance, however, provides much more information than the inner variance at possibly the same computational cost.

2.4.2 Exploitation of sparsity

Arrays of order N were defined as functions over vectors of N integers. An alternative definition of an array is a collection of elements arranged in a rectangular fashion such that the coordinates of each element is given by a vector of N integers. The reason a functional definition is preferable is because it does not imply a particular storage class. In some situations, it is more efficient to store a mathematical description of an array without storing a single element. This is often true for the encoding arrays used to transform array equations to lattice equations. For example, h_{ij} in (2.61) is an encoding array that may be used to vectorise a second order array, as in (2.62).

$$\underset{M\times N}{h_{ij}} \equiv i + M(j-1) \tag{2.61}$$

$$\mathbf{a} = a_{ij}()_{h_{ij}} \tag{2.62}$$

An important class of arrays, as with matrices, are those arrays with a minority of nonzero elements, called sparse arrays. Sparsity may be exploited to improve both the time and space performance of an algorithm, by limiting arithmetic operations where possible to nonzero elements and by storing only nonzero elements. One way to store a sparse array is to keep a list of nonzero elements together with their corresponding indices. If there are M nonzero elements in an array of order N then storage requires $O(MN)$ space since the indices are vectors of N integers. Using an encoding array $i_{\mathbf{i}}$ that is a linear function of indices $\mathbf{i}$, as in (2.63), a storage method that requires $O(M + N)$ space transforms the sparse array $a_{\mathbf{i}}$ into a sparse vector $\mathbf{a}$, as in (2.64). The M nonzero elements of $\mathbf{a}$ and their corresponding row numbers are stored with the $N + 1$ coefficients $\mathbf{j}$ and k of the linear function in (2.63).

$$i_{\mathbf{i}} \equiv \mathbf{i} \bullet \mathbf{j} + k \tag{2.63}$$

$$\mathbf{a} = a_{\mathbf{i}}()_{i_{\mathbf{i}}} \tag{2.64}$$

Sparse vector or matrix arrays may be transformed to sparse scalar arrays (using the vector or matrix basis arrays) and stored by the method described above with the additional storage of the vector or matrix size. This approach is efficient when the vector or matrix array contains a minority of nonzero vectors or matrices, which are dense themselves, when the vectors or matrices indexed by the array are sparse themselves or when there is a combination of the two. Since array operations may be implemented by a sequence of matrix operations, sparse arrays were stored as native sparse vectors in MATLAB, using the mapping in (2.63) and (2.64), and were transformed to native sparse matrices when performing array multiplication or inversion. MATLAB's implementation of sparse vectors and matrices is efficient in the sense that the time taken for vector or matrix operations is generally proportional to the number of arithmetic operations on nonzeros or the number of nonzeros in the result, whichever is greater, and the space taken for storage is generally proportional to the number of nonzeros [53].

However, when used to represent sparse arrays, the implementation of sparse vectors and matrices in MATLAB is sometimes inefficient. Transforming a sparse array equation into a sequence of sparse matrix equations (or a single sparse matrix equation when there are no inter products) involves rearranging the elements of the sparse vector or matrix used to represent the array. Because MATLAB stores the elements of a sparse vector or matrix in column major order [53], this implementation requires an implicit sorting operation on the indices of the nonzero elements. If there are $O(N)$ nonzero elements in a sparse array then the time required to perform an array operation is at least $O(N \log N)$ in MATLAB because of the rearrangement of elements. If the time required to perform the underlying arithmetic operations of the sparse array operation is less than $O(N \log N)$ then sorting is the limiting factor. Furthermore, MATLAB stores extra information to optimise sparse vector and matrix operations [53] that may exceed $O(N)$ space when used to represent sparse arrays of $O(N)$ nonzeros.

Although the present implementation of sparse arrays is not optimal, a more detailed study of sparse arrays may, in future, improve the time and space performance so that they are not limited by bookkeeping but by arithmetic operations and nonzero storage. One property of array operations that may be exploited is that, although they are implemented by a sequence of matrix operations, they are independent of the particular choice of encoding arrays. Furthermore, ongoing database research into sparse array manipulation and storage may prove fruitful, particularly in the use of set operations on nonzero indices, which may be optimised using hashing (instead of sorting). Indeed, various researchers have advocated using hashing with sparse matrices.

2.4.3 Systems of equations

So far, arrays of scalars, vectors and matrices have been discussed, where the latter two are effectively arrays of higher order because the vectors or matrices indexed by the array were all required to have the same size. The concept of an array of homogenous arrays, therefore, is essentially an array of higher order. A different and useful concept, however, is the array of heterogenous elements, be they scalars, vectors, matrices or arrays of scalars, vectors or matrices. These collections are called cell arrays.

Cell arrays are indicated using Greek instead of Arabic subscripts. For example, $\underset{N}{\mathbf{X}_\alpha}$ denotes a cell array of order one and dimension N (i.e. with one variable α used to index N matrices of possibly different sizes). Elements of a cell array are cells that may be arbitrary scalar, vector or matrix arrays. Normally, the symbol for a cell array represents the extent of actual cells in the array so that $\mathbf{X}_\alpha$ represents an array of cells that are either scalars, vectors or matrices, as in (2.65). As another example, $\mathbf{b}_{\alpha\beta ij}$ in (2.66) is a 2×4 array of cells, indexed by α and β, that are scalar or vector arrays of order zero, one or two, using neither index i nor j, either index i or j or both indices i and j. The cell basis arrays $\{\}_\alpha$ and $\{\}_{\alpha\beta}$ are used to display other cell arrays like the vector and matrix basis arrays $()_i$ and $()_{ij}$ are used to display vectors and matrices.

$$\mathbf{X}_\alpha\{\}^{\mathrm{T}}_\alpha = \{x_1 \quad \mathbf{x}_2 \quad \mathbf{X}_3\} \tag{2.65}$$

$$\mathbf{b}_{\alpha\beta ij}\{\}_{\alpha\beta} = \begin{Bmatrix} b_{11} & \mathbf{b}_{12i} & b_{13j} & \mathbf{b}_{14ij} \\ \mathbf{b}_{21i} & b_{22j} & \mathbf{b}_{23i} & b_{24j} \end{Bmatrix} \tag{2.66}$$

Cell arrays are particularly useful in simplifying the representation and manipulation of array equations. For example, consider the linear algebra equation in (2.67).

$$\mathbf{y}^M = \mathbf{X}_1^{M\times P_1}\mathbf{b}_1^{P_1} + \mathbf{X}_2^{M\times P_2}\mathbf{b}_2^{P_2} + \mathbf{X}_3^{M\times P_3}\mathbf{b}_3^{P_3} \tag{2.67}$$

If the matrices and vectors on the right side of (2.67) are homogenous (i.e. $P_1 = P_2 = P_3$) then the equation may be rewritten as (2.68) with matrix and vector arrays. If they are heterogenous then (2.67) may still be simplified using cell arrays, as in (2.69).

$$\mathbf{y}^M = \mathbf{X}_i^{M\times P}\mathbf{b}_i^{P} \tag{2.68}$$

$$\mathbf{y}^M = \mathbf{X}_\alpha^{M\times P_\alpha}\mathbf{b}_\alpha^{P_\alpha} \tag{2.69}$$

Although the matrices and vectors indexed by $\mathbf{X}_\alpha$ and $\mathbf{b}_\alpha$ in (2.69) may be heterogenous, their sizes are constrained. The number of rows in each matrix indexed by $\mathbf{X}_\alpha$ must equal the number of rows in $\mathbf{y}$. For each α, the number of columns in $\mathbf{X}_\alpha$ must equal the number of rows in $\mathbf{b}_\alpha$. These constraints are specific to the inner product (over α) $\mathbf{X}_\alpha\mathbf{b}_\alpha$ and vary for the inter product $\mathbf{X}_{\underline{\alpha}}\mathbf{b}_{\underline{\alpha}}$ and the outer product $\mathbf{X}_\alpha\mathbf{b}_\beta$.

In addition to rules governing multiplication, much may be said about the inversion of cell arrays, especially in the solution of systems of array equations (a problem which occupies Chapter 3). Consider the example of two matrix equations (2.70) and (2.71) that may be expressed by a single cell equation (2.72).

$$\mathbf{y}_1 = \mathbf{X}_{11}\mathbf{b}_1 + \mathbf{X}_{12}\mathbf{b}_2 \tag{2.70}$$

$$\mathbf{y}_2 = \mathbf{X}_{21}\mathbf{b}_1 + \mathbf{X}_{22}\mathbf{b}_2 \tag{2.71}$$

$$\mathbf{y}_\alpha = \mathbf{X}_{\alpha\beta}\mathbf{b}_\beta \tag{2.72}$$

Suppose $\mathbf{y}$ and $\mathbf{X}$ are known and $\mathbf{b}$ is required in (2.72). Constraints on the cells permit the cell equation to be transformed to the partitioned matrix equation in (2.73).

$$\begin{pmatrix} \mathbf{y}_1 \\ \mathbf{y}_2 \end{pmatrix} = \begin{pmatrix} \mathbf{X}_{11} & \mathbf{X}_{12} \\ \mathbf{X}_{21} & \mathbf{X}_{22} \end{pmatrix} \begin{pmatrix} \mathbf{b}_1 \\ \mathbf{b}_2 \end{pmatrix} \tag{2.73}$$

If there is a solution for the vector made up of $\mathbf{b}_1$ and $\mathbf{b}_2$ in (2.73) then there is a unique cell array $\mathbf{X}_{\alpha\beta}^{-1}$ that is the inverse of $\mathbf{X}_{\alpha\beta}$ in (2.72) such that (2.74) holds.

$$\mathbf{b}_\beta = \mathbf{X}_{\alpha\beta}^{-1}\mathbf{y}_\alpha \tag{2.74}$$

Thus, cell arrays obey certain algebraic properties that may be exploited by a more comprehensive (metalinear) algebra, yet to be developed. The possibility to automate the forward and backward transformation of cell array equations to partitioned matrix equations is enticing because cell array equations, e.g. (2.72), are no less useful for large systems of array equations than for small systems, e.g. (2.70) and (2.71). However, because there still remain many unanswered questions regarding their properties, cell arrays are used in this thesis only for their notational convenience.

2.5 Conclusion

An array of order N is a functional mapping from a vector of N integers to a scalar, vector or matrix field. What linear algebra is to scalars, vectors and matrices, multilinear algebra is to scalar, vector and matrix arrays. Vector and matrix arrays may easily be converted to scalar arrays and vice versa. Multilinear algebra derives from tensor calculus but permits attraction, arbitrary combinations of inner, inter and outer products and inversion of arrays for certain identities (with existence and uniqueness theorems). Multiplication and inversion of arrays may be transformed to lattice equations with encoding arrays, where lattices are first order matrix arrays. Lattice equations may be solved by a sequence of matrix multiplications or inversions. The results may be transformed into the original array domain with decoding arrays and do not depend on the choice of encoding and decoding arrays. The underlying mechanics of array multiplication and inversion are easily and efficiently automated in MATLAB.

Multilinear algebra shows that four basic binary operations are missing from classical linear algebra because the latter does not permit inter (or element-wise) products. Operators are introduced to define these operations for vectors and matrices. Three useful unary operations are also defined for vectors and matrices, involving attraction and inter products. Stochastic, sparse and cell arrays were considered. For stochastic arrays, three types of variance were defined—outer, inter and inner variance—which have different minimum computing time and storage space requirements. Sparse arrays are arrays where only a minority of elements are nonzero. It pays in computing time and storage space to exploit this sparsity and a simple MATLAB implementation was discussed, although it is not optimal in time and space because of an internal sort and bookkeeping. Lastly, cell arrays provide a convenient way to describe and manipulate arrays of heterogenous elements, useful in solving systems of array equations.

Chapter 3

Constrained regression

3.1 Introduction

An image sensor with N pixels is essentially an array of N sensors. Consider an array of N sensors where the response of each sensor is a linear function of P inputs plus Gaussian noise. Calibration of this sensor array may be accomplished by estimating the PN coefficients of the multiple linear functions from the M responses of each sensor to M input vectors, where $M > P$. Assuming all sensors respond to the same input vector, for each observation of the calibration, these conditions may be modelled by (3.1), where $\mathbf{Y}$ is an $M \times N$ matrix of sensor responses, $\mathbf{X}$ is an $M \times P$ matrix of input vectors, $\mathbf{B}$ is a $P \times N$ matrix of linear coefficients and $\mathbf{\Sigma}$ is an $M \times N$ matrix of Gaussian noise, assumed to be independent from sample to sample.

$$\mathbf{Y} = \mathbf{X}\mathbf{B} + \mathbf{\Sigma} \tag{3.1}$$

The parameters $\mathbf{B}$ of the sensor array may be estimated by multilinear regression, as $\hat{\mathbf{B}}$ in (3.2), where $\mathbf{Y}$ is pre-multiplied by the pseudo-inverse of $\mathbf{X}$ [42, 54].

$$\hat{\mathbf{B}} = (\mathbf{X}^{\mathrm{T}}\mathbf{X})^{-1}\mathbf{X}^{\mathrm{T}}\mathbf{Y} \tag{3.2}$$

Note that the formulation in (3.2) is the solution of N independent multilinear regression problems, one for each column of $\mathbf{Y}$, with a single matrix equation. Assuming $N > M$, this solution takes $O(PMN)$ floating point operations (flops) and requires $O(MN)$ bytes of storage. It is effectively an $O(N)$ time and space algorithm if the number of sensors is much larger than the number of observations, i.e. $N \gg M$, as shall be assumed in the context of imaging.

Suppose parameters of the sensor array obey certain linear constraints. Enforcing these constraints during calibration leads to more accurate parameter estimates, especially in the presence of noise [40, 41, 42]. The constraints may relate parameters of one sensor to parameters of another sensor so calibration ceases to be one of N independent regressions but one of a single constrained regression, called the *generic sensor array problem*. In addition to parameter estimation, the generic problem requires estimation of the variance of the Gaussian noise, which measures the residual error in

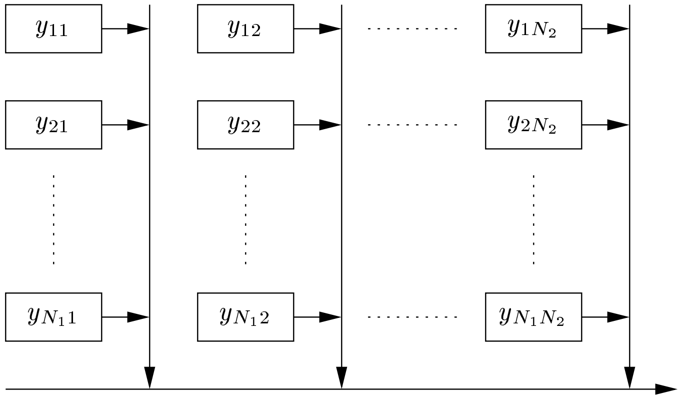


Figure 3.1: A rectangular array of $N_1 \times N_2$ sensors. The response of each sensor is denoted $y_{j_1j_2}$ and responses are scanned in raster fashion.

the model, and the variance of each estimated parameter, which is the uncertainty of the calibration due to noise. In cases where the exact constraints on parameters are unknown, these statistics help to distinguish good hypotheses from bad ones [40].

Ideally, a solution to the generic problem would take $O(N)$ time and space, as with the unconstrained sensor array problem, but this chapter shows it is not always possible. One class of the generic problem of particular interest to imaging, where it is always possible to do so, is called the *raster sensor array problem*. Consider an $N_1 \times N_2$ rectangular array of sensors with responses $y_{j_1j_2}$, where $1 \leq j_1 \leq N_1$ and $1 \leq j_2 \leq N_2$, that are scanned in raster fashion, as depicted in Figure 3.1. The response at each sensor in each column is read serially to a column buffer and the response at each column buffer is read serially to an output buffer. Assume the response $y_{j_1j_2}$ at the output buffer, which is read for row j_1 and column j_2 of the sensor array, depends on parameters that may vary from sensor to sensor, parameters that may vary from column to column and parameters that may not vary. This is a special case of the generic problem, where parameters must satisfy one of these three constraints.

Methods to solve the generic and raster problems, which are useful in the modelling and calibration of image sensors, are derived in Sections 3.2 and 3.3. The generic problem is investigated because it is more flexible than the raster problem and, as the equations are simpler, it facilitates an explanation of the solution. Though it may be solved in $O(N)$ time and space as a special case of the generic problem, assuming an efficient sparse array implementation in MATLAB, the raster problem is investigated separately to optimise the $O(N)$ time and space performance. Extensions of the generic (and raster) problems are considered in Section 3.4, where sensors may respond to different input vectors for each observation in the calibration or when responses may depend on parameters in a nonlinear fashion. Section 3.5 simulates a raster sensor array problem to illustrate the usefulness of constrained regression and to demonstrate the time and space performance of various methods of solution.

3.2 Generic methods

Magnus and Neudecker derive a solution to the multilinear regression problem with linear constraints using best affine unbiased estimation, which yields the same result as least squares estimation at greater theoretical complexity [42]. The authors suggest that least squares is not a method of estimation but of approximation and that it is a remarkable coincidence that the results are equal. However, Wang and Rhee note that the maximum likelihood estimator of a multilinear regression problem (with or without constraints) always equals the least square estimator when the error belongs to a Gaussian distribution [55]. Wang and Rhee also discuss maximum likelihood estimation when the error belongs to any l_p distribution (e.g. l_1 is a Laplacian distribution and l_2 is a Gaussian distribution) [55]. Because the error is always assumed to be Gaussian in this chapter, least squares estimation is used throughout.

This chapter assumes that all matrices needing inversion have full rank. Magnus and Neudecker discuss the solution of rank deficient problems (e.g. redundant constraints or insufficient linearly independent observations) [42]. This chapter also assumes that there are only equality constraints on the regression parameters. Wang and Rhee [55] and Ghiorso [41] consider problems that involve inequality constraints. Such problems require a simplex-tableau approach in the theory of linear programming. Lastly, this chapter assumes that Gaussian errors are statistically independent from sample to sample. Magnus and Neudecker discuss ways of including known statistical dependencies of the error in the regression problem [42].

To solve the problem of constrained regression, Magnus and Neudecker use a Lagrangian method, where the constraints are explicit [42]. Ghiorso uses a different method, where the constraints are implicit—a regression problem with constrained parameters is equivalent to another regression problem with unconstrained but fewer parameters [41]. Both methods are applied in this section to solve the generic problem because the method of explicit constraints is the most obvious approach whereas the method of implicit constraints has better time and space performance.

3.2.1 Explicit constraints

Equation (3.3) models the generic problem for an array of N sensors, indexed by the variable j. Observations of the response of each sensor are given by vectors of size M, indexed by the N-dimensional array $\mathbf{y}_j$. Each row of the $M \times P$ matrix $\mathbf{X}$ gives the input vector seen by all sensors for each observation. Parameters of the sensor functions are given by vectors of size P, indexed by the N-dimensional array $\mathbf{b}_j$. The vector array $\boldsymbol{\epsilon}_j$ represents the Gaussian error. The L constraints on the parameters are given explicitly in (3.4), where $\mathbf{A}_j$ is an N-dimensional array of $L \times P$ matrices and $\mathbf{c}$ is a vector of size L. Note that the inner product over j in (3.4) permits constraints relating parameters in one sensor to parameters in another.

$$\mathbf{y}_j = \mathbf{X}\mathbf{b}_j + \boldsymbol{\epsilon}_j \tag{3.3}$$

$$\mathbf{A}_j\mathbf{b}_j = \mathbf{c} \tag{3.4}$$

Equations (3.5) and (3.6) assert that the stochastic error has zero mean and that the error, with an unknown standard deviation of σ_ϵ, is statistically independent from sample to sample. The regression problem is to find an estimate $\hat{\mathbf{b}}_j$, of the actual parameters $\mathbf{b}_j$, that minimises the sum square error (SSE) defined in (3.7).

$$\mathcal{E}\{\boldsymbol{\epsilon}_j\} = \mathbf{0}_j \tag{3.5}$$

$$\mathcal{C}\{\boldsymbol{\epsilon}_j\} = \sigma_\epsilon^2 \delta_{jj'} \mathbf{I} \tag{3.6}$$

$$SSE(\mathbf{b}_j) = 1_j ||\mathbf{y}_j - \mathbf{X}\mathbf{b}_j||^2 \tag{3.7}$$

If this problem is formulated using linear algebra, as in (3.8), then the parameters $\mathbf{b}_j$ must be rearranged as a vector so that the constraints $\mathbf{A}_j$ in (3.4), when rearranged as a matrix, may relate parameters of one sensor to parameters of another.

$$\begin{pmatrix} \mathbf{A}_1 & \mathbf{A}_2 & \dots & \mathbf{A}_N \end{pmatrix} \begin{pmatrix} \mathbf{b}_1 \\ \mathbf{b}_2 \\ \vdots \\ \mathbf{b}_N \end{pmatrix} = \mathbf{c} \tag{3.8}$$

However, vectorisation of $\mathbf{b}_j$ requires (3.3) to be rewritten in matrix form, as in (3.9), where the input matrix $\mathbf{X}$ is repeated N times in a larger $MN \times PN$ sparse input matrix. This N-fold redundancy wastes time and space during computation.

$$\begin{pmatrix} \mathbf{y}_1 \\ \mathbf{y}_2 \\ \vdots \\ \mathbf{y}_N \end{pmatrix} = \begin{pmatrix} \mathbf{X} & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \mathbf{X} & \ddots & \vdots \\ \vdots & \ddots & \ddots & \mathbf{0} \\ \mathbf{0} & \cdots & \mathbf{0} & \mathbf{X} \end{pmatrix} \begin{pmatrix} \mathbf{b}_1 \\ \mathbf{b}_2 \\ \vdots \\ \mathbf{b}_N \end{pmatrix} + \begin{pmatrix} \boldsymbol{\epsilon}_1 \\ \boldsymbol{\epsilon}_2 \\ \vdots \\ \boldsymbol{\epsilon}_N \end{pmatrix} \tag{3.9}$$

To minimise the SSE in (3.7) subject to the constraints in (3.4), a Lagrangian is defined in (3.10) with a vector $\boldsymbol{\lambda}$ of L multipliers. The partial derivatives of the Lagrangian with respect to $\mathbf{b}_j$ and $\boldsymbol{\lambda}$ are given in (3.11) and (3.12). The SSE is minimised subject to the constraints when the partial derivatives of the Lagrangian equal zero.

$$\mathcal{L}(\mathbf{b}_j, \boldsymbol{\lambda}) = SSE(\mathbf{b}_j) + (\mathbf{A}_j\mathbf{b}_j - \mathbf{c})^{\mathrm{T}}\boldsymbol{\lambda} \tag{3.10}$$

$$\frac{\partial \mathcal{L}}{\partial \mathbf{b}_j} = -2\mathbf{X}^{\mathrm{T}}(\mathbf{y}_j - \mathbf{X}\mathbf{b}_j) + \mathbf{A}_j^{\mathrm{T}}\boldsymbol{\lambda} \tag{3.11}$$

$$\frac{\partial \mathcal{L}}{\partial \boldsymbol{\lambda}} = \mathbf{A}_j\mathbf{b}_j - \mathbf{c} \tag{3.12}$$

Setting (3.11) and (3.12) equal to zero and solving for $\mathbf{b}_j$ gives the estimate $\hat{\mathbf{b}}_j$ in (3.17), with intermediates listed in (3.13)–(3.16). The estimator that solves the unconstrained regression problem is $\hat{\mathbf{b}}_{0j}$ in (3.16).

$$\mathbf{D} = (\mathbf{X}^{\mathrm{T}}\mathbf{X})^{-1} \tag{3.13}$$

$$\mathbf{E}_j = \mathbf{D}\mathbf{A}_j^{\mathrm{T}} \tag{3.14}$$

$$\mathbf{F}_j = \mathbf{E}_j(\mathbf{A}_j\mathbf{E}_j)^{-1} \tag{3.15}$$

$$\hat{\mathbf{b}}_{0j} = \mathbf{D}\mathbf{X}^{\mathrm{T}}\mathbf{y}_j \tag{3.16}$$

Table 3.1: Asymptotic time and space performance for a dense and sparse solution to the generic and raster problem, using explicit constraints.

	Time $O(\cdot)$		Space $O(\cdot)$	
Equation	Dense	Sparse	Dense	Sparse
(3.13)	P^2M		PM	
(3.14)	LP^2N	P^2N	LPN	PN
(3.15)	L^2PN		LPN	
(3.16)	PMN		MN	
(3.17)	LPN		LPN	
(3.18)	PMN		MN	
(3.19)	LPN	PN	LPN	
Total	L^2PN		LPN	

$$\hat{\mathbf{b}}_j = \hat{\mathbf{b}}_{0j} - \mathbf{F}_j(\mathbf{A}_j\hat{\mathbf{b}}_{0j} - \mathbf{c}) \tag{3.17}$$

Equation (3.18) estimates the variance of the error $\hat{\sigma}_\epsilon^2$ using the estimated parameters $\hat{\mathbf{b}}_j$. The SSE is divided by the degrees of freedom, which equals the number of equations (i.e. L constraints and MN observations) minus the number of variables (i.e. PN parameters). This ensures that the estimated error variance is unbiased. Equation (3.19) estimates the inter variance of the parameters $\hat{\mathcal{V}}\{\hat{\mathbf{b}}_j\}$, using the estimated error variance. The exact inter variance may be known only with the exact error variance. Note that the inter variance in (3.19) involves a mixed product of lattices $\mathbf{E}_j$ and $\mathbf{F}_j$, with an inter product over the row indices and an inner product over the column indices, as in Operation 7 of Table 2.1, and an inter product over the tab indices.

$$\hat{\sigma}_\epsilon^2 = \frac{SSE(\hat{\mathbf{b}}_j)}{L + MN - PN} \tag{3.18}$$

$$\hat{\mathcal{V}}\{\hat{\mathbf{b}}_j\} = \hat{\sigma}_\epsilon^2(1_j \operatorname{diag}\mathbf{D} - \mathbf{E}_{\underline{j}} \diamond \mathbf{F}_{\underline{j}}) \tag{3.19}$$

Assuming the number of constraints is proportional to the number of sensors, i.e. $L \propto N$, the number of observations is much less than the number of sensors, i.e. $M \ll N$, and the number of parameters per sensor is less than the number of observations, i.e. $P < M$, Table 3.1 gives the time and space requirements for the explicit method, i.e. to compute (3.13)–(3.19). Except for (3.14), (3.15) and (3.17), all equations take $O(N)$ time and space. With a dense implementation, (3.14) is $O(N^2)$ in time and space. However, for the raster problem, $\mathbf{A}_j$ and $\mathbf{E}_j$ have $O(N)$ nonzero elements, which means (3.14) has $O(N)$ performance when implemented with sparse arrays.

Although $\mathbf{A}_j$ and $\mathbf{E}_j$ have $O(N)$ nonzero elements for the raster problem, their product in (3.15) has $O(N^2)$ nonzero elements. Inversion of this dense product takes $O(N^3)$ time. The inverse in (3.15) has $O(N^2)$ nonzero elements, which means the product of $\mathbf{E}_j$ with the inverse takes $O(N^2)$ time (considering the sparsity of $\mathbf{E}_j$). The result $\mathbf{F}_j$ has $O(N^2)$ nonzero elements, which means (3.17) takes $O(N^2)$ time and space. Therefore, the explicit method takes $O(N^3)$ time and $O(N^2)$ space for the raster problem with a dense or sparse implementation, which is also the worst case performance of this method for the generic problem.

3.2.2 Implicit constraints

Enforcing linear constraints on a space of PN parameters $\mathbf{b}_j$ is equivalent to defining $\mathbf{b}_j$ as the linear transformation, with coefficients $\mathbf{U}_j$ and $\mathbf{b}_{0j}$, of an unconstrained subspace of Q parameters $\mathbf{a}$, as in (3.21). The transformation implicitly ensures that the constraints are satisfied. With this idea, minimising the SSE in (3.7) subject to (3.3)–(3.6) is equivalent to minimising the SSE in (3.24) subject to (3.20)–(3.23).

$$\mathbf{y}_j = \mathbf{X}\mathbf{b}_j + \boldsymbol{\epsilon}_j \tag{3.20}$$

$$\mathbf{b}_j = \mathbf{U}_j\mathbf{a} + \mathbf{b}_{0j} \tag{3.21}$$

$$\mathcal{E}\{\boldsymbol{\epsilon}_j\} = \mathbf{0}_j \tag{3.22}$$

$$\mathcal{C}\{\boldsymbol{\epsilon}_j\} = \sigma_\epsilon^2 \delta_{jj'}\mathbf{I} \tag{3.23}$$

$$SSE(\mathbf{b}_j) = 1_j \|\mathbf{y}_j - \mathbf{X}\mathbf{b}_j\|^2 \tag{3.24}$$

The required equivalence between explicit constraints (3.4) and implicit constraints (3.21) means that (3.25) holds for the entire space spanned by the vector $\mathbf{a}$.

$$\forall \mathbf{a} : \mathbf{A}_j(\mathbf{U}_j\mathbf{a} + \mathbf{b}_{0j}) = \mathbf{c} \tag{3.25}$$

If (3.25) holds then so does (3.26), which means $\mathbf{U}_j$ is in the null space or kernel of $\mathbf{A}_j$ and may be determined by row reduction of $\mathbf{A}_j$. Alternatively, $\mathbf{A}_j$ may be determined if $\mathbf{U}_j$ is given. Because of the null space relationship, the number of explicit parameters PN in (3.3) or (3.20) equals the sum of the number of explicit constraints L in (3.4) and the number of implicit parameters Q in (3.21), as in (3.27).

$$\mathbf{0} = \mathbf{A}_j\mathbf{U}_j \tag{3.26}$$

$$PN = L + Q \tag{3.27}$$

Using (3.25) and (3.26), the vector $\mathbf{c}$ and vector array $\mathbf{b}_{0j}$ may be calculated from each other as in (3.28) and (3.29). The time and space cost of the transformations are immaterial because the user is assumed to be capable of providing either representation of the constraints, which is true for the raster problem.

$$\mathbf{c} = \mathbf{A}_j\mathbf{b}_{0j} \tag{3.28}$$

$$\mathbf{b}_{0j} = \mathbf{A}_j^{\mathrm{T}}(\mathbf{A}_j\mathbf{A}_j^{\mathrm{T}})^{-1}\mathbf{c} \tag{3.29}$$

An advantage of implicit over explicit constraints is that no Lagrangian is needed to minimise the SSE with the former. With the substitutions in (3.30) and (3.31), the SSE in (3.24) may be reformulated in (3.32) as an exact function of the implicit parameters $\mathbf{a}$. The minimum of the SSE with respect to $\mathbf{a}$ is found by solving for the vector that makes the partial derivative in (3.33) equal to zero.

$$\mathbf{z}_j = \mathbf{y}_j - \mathbf{X}\mathbf{b}_{0j} \tag{3.30}$$

Table 3.2: Asymptotic time and space performance of a dense and sparse solution to the generic and raster problem, using implicit constraints.

	Time $O(\cdot)$		Space $O(\cdot)$	
Equation	Dense	Sparse	Dense	Sparse
(3.30)	PMN		MN	
(3.31)	$PQMN$	PMN	QMN	MN
(3.34)	Q^2MN	Q^2	QMN	Q^2
(3.35)	QMN	Q^2	QMN	Q^2
(3.36)	PQN	PN	PQN	PN
(3.37)	QMN	MN	QMN	MN
(3.38)	PQ^2N	PQN	PQN	
Total	Q^2MN	PQN	QMN	PQN

$$\mathbf{W}_j = \mathbf{X}\mathbf{U}_j \tag{3.31}$$

$$SSE = 1_j||\mathbf{z}_j - \mathbf{W}_j\mathbf{a}||^2 \tag{3.32}$$

$$\frac{\partial SSE}{\partial \mathbf{a}} = -2\mathbf{W}_j^{\mathrm{T}}(\mathbf{z}_j - \mathbf{W}_j\mathbf{a}) \tag{3.33}$$

The solution $\hat{\mathbf{a}}$ that minimises the SSE in (3.32) is given by (3.34) and (3.35), which are similar to (3.13) and (3.16). This parameter estimate, in a subspace of the original parameter space, may be linearly transformed into an estimate in the original space, as in (3.36). Because of the equivalence of the representations, the estimator $\hat{\mathbf{b}}_j$ in (3.17) equals the estimator $\hat{\mathbf{b}}_j$ in (3.36), although the equations are substantially different.

$$\mathbf{V} = (\mathbf{W}_j^{\mathrm{T}}\mathbf{W}_j)^{-1} \tag{3.34}$$

$$\hat{\mathbf{a}} = \mathbf{V}\mathbf{W}_j^{\mathrm{T}}\mathbf{z}_j \tag{3.35}$$

$$\hat{\mathbf{b}}_j = \mathbf{U}_j\hat{\mathbf{a}} + \mathbf{b}_{0j} \tag{3.36}$$

As before, (3.37) estimates the variance of the error $\hat{\sigma}_\epsilon^2$ using the estimated parameters $\hat{\mathbf{b}}_j$. Note that (3.18) and (3.37) are identical because (3.7) and (3.24) are identical and (3.27) ensures the denominator (i.e. the degrees of freedom) is the same. Equation (3.38) estimates the inter variance $\hat{\mathcal{V}}\{\hat{\mathbf{b}}_j\}$ of the estimated parameters.

$$\hat{\sigma}_\epsilon^2 = \frac{SSE(\hat{\mathbf{b}}_j)}{MN - Q} \tag{3.37}$$

$$\hat{\mathcal{V}}\{\hat{\mathbf{b}}_j\} = \hat{\sigma}_\epsilon^2(\mathbf{U}_{\underline{j}}\mathbf{V}) \diamond \mathbf{U}_{\underline{j}} \tag{3.38}$$

Assuming the number of implicit parameters is proportional to the number of sensors, i.e. $Q \propto N$, Table 3.2 gives the time and space requirements of the implicit method, i.e. to compute (3.30), (3.31) and (3.34)–(3.38). With a dense implementation of all arrays, the implicit method takes $O(N^3)$ time and $O(N^2)$ space like the explicit method. The limiting equations are (3.34) and (3.38) as all others take $O(N^2)$ time.

However, with a sparse implementation of arrays $\mathbf{U}_j$ and $\mathbf{W}_j$, these equations take $O(N^2)$ time for the raster problem. The number of nonzeros in $\mathbf{U}_j$ is $O(PN)$ and $\mathbf{W}_j$, calculated in (3.31), has a similar sparsity with $O(MN)$ nonzeros.

Exploiting the sparsity leads to a reduction in time and space requirements for (3.31), (3.36) and (3.37) from $O(N^2)$ to $O(N)$. However, (3.34), (3.35) and (3.38) retain $O(N^2)$ time and space requirements because matrix $\mathbf{V}$ in (3.34) remains dense (it is related to the covariance of parameters, which tends to be dense because of relationships between parameters). The product of $\mathbf{W}_j$ with itself in (3.34) is sparse with $O(N)$ nonzeros but leads to a dense matrix upon inversion. Nonetheless, the sparsity of $\mathbf{U}_j$ and $\mathbf{W}_j$ enables the raster problem to be solved in $O(N^2)$ time with implicit constraints whereas it takes $O(N^3)$ time with explicit constraints.

3.2.3 Cholesky factorisation

While an $O(N^2)$ time and space solution to the raster problem is better than an $O(N^3)$ time and $O(N^2)$ space solution, an $O(N)$ time and space solution is by far more useful when N is large (as it often is with image sensors). Such a solution is possible with the implicit method and Cholesky factorisation. For the implicit method, if the number of nonzeros in $\mathbf{U}_j$ is $O(PN)$ then a sparse implementation of $\mathbf{U}_j$ and $\mathbf{W}_j$ leads to $O(PMN)$ time requirements and $O(MN)$ space requirements except for (3.34), (3.35) and (3.38), as in Table 3.2, because the $Q \times Q$ matrix $\mathbf{V}$ tends to be dense. However, the product of $\mathbf{W}_j$ with itself in (3.34) remains sparse and has $O(Q)$ nonzero elements. Only upon inversion does sparsity disappear.

Because $\mathbf{W}_j^{\mathrm{T}}\mathbf{W}_j$ is a positive definite matrix then a Cholesky factor $\mathbf{W}_{\mathcal{C}}$ of the product exists, which is an upper triangular square matrix that satisfies (3.39) [56].

$$\mathbf{W}_{\mathcal{C}}^{\mathrm{T}}\mathbf{W}_{\mathcal{C}} = \mathbf{W}_j^{\mathrm{T}}\mathbf{W}_j \tag{3.39}$$

Since the product $\mathbf{W}_j^{\mathrm{T}}\mathbf{W}_j$ is sparse of $O(Q)$ nonzeros, its Cholesky factor is also sparse with $O(Q)$ nonzeros and takes $O(Q)$ time to compute. Moreover, the inverse of the Cholesky factor, denoted $\mathbf{V}_{\mathcal{C}}$, is also sparse with $O(Q)$ nonzeros and inversion takes $O(Q)$ time to compute. Most importantly, the dense matrix $\mathbf{V}$ in (3.34) is a product of the inverse Cholesky factor with itself, as in (3.40).

$$\mathbf{V} = \mathbf{V}_{\mathcal{C}}\mathbf{V}_{\mathcal{C}}^{\mathrm{T}} \tag{3.40}$$

With Cholesky factorisation, there is no need to compute $\mathbf{V}$ because (3.35) and (3.38) may be rewritten in (3.41) and (3.42) using the inverse Cholesky factor to avoid producing this dense matrix. As a result of the Cholesky factorisation, (3.41) takes $O(MN)$ time and uses $O(MN)$ space and (3.42) takes $O(PN)$ time and uses $O(PN)$ space. The order of operations in (3.41), indicated by parentheses, is crucial to achieve this performance improvement. Thus, the implicit method with Cholesky factorisation takes $O(PMN)$ time and $O(MN)$ space to solve the raster problem. Since N is much larger than P or M, this method essentially takes $O(N)$ time and space.

$$\hat{\mathbf{a}} = \mathbf{V}_{\mathcal{C}}(\mathbf{V}_{\mathcal{C}}^{\mathrm{T}}(\mathbf{W}_j^{\mathrm{T}}\mathbf{z}_j)) \tag{3.41}$$

$$\hat{\mathcal{V}}\{\hat{\mathbf{b}}_j\} = \hat{\sigma}_\epsilon^2\langle\mathbf{U}_j\mathbf{V}_{\mathcal{C}}\rangle^2 \tag{3.42}$$

As for the explicit method, if the positive definite product $\mathbf{A}_j\mathbf{E}_j$ in (3.15) is sparse then it is profitable to find its Cholesky factor $\mathbf{G}_\mathcal{C}$, as in (3.43). This factor may be used to simplify (3.15), (3.17) and (3.19), as in (3.44)–(3.46). However, the explicit method with Cholesky factorisation does not yield any improvement for the raster problem because $\mathbf{A}_j\mathbf{E}_j$ is dense, causing a dense Cholesky factor and a dense inverse. Nonetheless, other classes of the generic problem may result in a sparse product $\mathbf{A}_j\mathbf{E}_j$, in which case Cholesky factorisation would be profitable.

$$\mathbf{G}_\mathcal{C}^\mathrm{T}\mathbf{G}_\mathcal{C} = \mathbf{A}_j\mathbf{E}_j \tag{3.43}$$

$$\mathbf{F}'_j = \mathbf{E}_j\mathbf{G}_\mathcal{C}^{-1} \tag{3.44}$$

$$\hat{\mathbf{b}}_j = \hat{\mathbf{b}}_{0j} - \mathbf{F}'_j(\mathbf{G}_\mathcal{C}^{-1\mathrm{T}}(\mathbf{A}_j\hat{\mathbf{b}}_{0j} - \mathbf{c})) \tag{3.45}$$

$$\hat{\mathcal{V}}\{\hat{\mathbf{b}}_j\} = \hat{\sigma}_\epsilon^2(1_j \operatorname{diag}\mathbf{D} - \langle\mathbf{F}'_j\rangle^2) \tag{3.46}$$

3.3 Raster method

A sparse implementation of the implicit method with Cholesky factorisation requires $O(N)$ time and space to solve the generic problem when the coefficient array $\mathbf{U}_j$ and related arrays $\mathbf{W}_j$ and $\mathbf{W}_j^\mathrm{T}\mathbf{W}_j$ have $O(N)$ nonzeros. The raster problem meets these criteria. However, the expected $O(N)$ performance may not be realised in MATLAB because of a sub-optimal sparse array implementation. Instead of using a generic method to solve the raster problem, an $O(N)$ time and space method to solve the raster problem may be derived that does not require sparse arrays or Cholesky factorisation. Even if a better sparse array implementation existed, this raster method is more efficient than a generic method to solve the raster problem in non-asymptotic terms (i.e. it is a faster and smaller $O(N)$ method). Nonetheless, the generic method is much more flexible for the analysis of sensor arrays.

Equations (3.47)–(3.51) formulate the raster problem using cell arrays, with the sensor index j of Section 3.2 decoded into row and column indices j_1 and j_2, as in Figure 3.1. The PN parameters of the $N_1 \times N_2$ sensor array are partitioned into three heterogenous arrays, corresponding to those P_1N parameters $\mathbf{b}_{1j_1j_2}$ that vary from sensor to sensor, those P_2N parameters $\mathbf{b}_{2j_1j_2}$ that vary from column to column and those P_3N parameters $\mathbf{b}_{3j_1j_2}$ that do not vary. These constraints may be imposed either explicitly, analogous to (3.4), or implicitly, analogous to (3.21). Both formulations lead to the same solution but implicit constraints are chosen in (3.48), where $\mathbf{a}_{1j_1j_2}$, $\mathbf{a}_{2j_2}$ and $\mathbf{a}_3$ represent the subspace of fewer parameters. Equations (3.49) and (3.50) state the assumptions on the Gaussian error and (3.51) defines the SSE.

$$\mathbf{y}_{j_1j_2} = \mathbf{X}_\alpha\mathbf{b}_{\alpha j_1j_2} + \boldsymbol{\epsilon}_{j_1j_2} \tag{3.47}$$

$$\mathbf{b}_{\alpha j_1j_2}\{\}_\alpha^\mathrm{T} = \left\{\mathbf{a}_{1j_1j_2} \quad 1_{j_1}\mathbf{a}_{2j_2} \quad 1_{j_1j_2}\mathbf{a}_3\right\} \tag{3.48}$$

$$\mathcal{E}\{\boldsymbol{\epsilon}_{j_1j_2}\} = \mathbf{0}_{j_1j_2} \tag{3.49}$$

$$\mathcal{C}\{\boldsymbol{\epsilon}_{j_1j_2}\} = \sigma_\epsilon^2\delta_{j_1j'_1}\delta_{j_2j'_2}\mathbf{I} \tag{3.50}$$

$$SSE(\mathbf{b}_{\alpha j_1j_2}) = 1_{j_1j_2}||\mathbf{y}_{j_1j_2} - \mathbf{X}_\alpha\mathbf{b}_{\alpha j_1j_2}||^2 \tag{3.51}$$

The maximum likelihood estimator of $\mathbf{b}_{\alpha j_1 j_2}$ in (3.47) is found by minimising the SSE in (3.51) with respect to $\mathbf{a}_{1j_1j_2}$, $\mathbf{a}_{2j_2}$ and $\mathbf{a}_3$. The partial derivatives of the SSE with respect to these variables are given in (3.52)–(3.54).

$$\frac{\partial SSE}{\partial \mathbf{a}_{1j_1j_2}} = -2\mathbf{X}_1^{\mathrm{T}}(\mathbf{y}_{j_1j_2} - \mathbf{X}_\alpha \mathbf{b}_{\alpha j_1 j_2}) \tag{3.52}$$

$$\frac{\partial SSE}{\partial \mathbf{a}_{2j_2}} = -2_{j_1}\mathbf{X}_2^{\mathrm{T}}(\mathbf{y}_{j_1j_2} - \mathbf{X}_\alpha \mathbf{b}_{\alpha j_1 j_2}) \tag{3.53}$$

$$\frac{\partial SSE}{\partial \mathbf{a}_{3}} = -2_{j_1j_2}\mathbf{X}_3^{\mathrm{T}}(\mathbf{y}_{j_1j_2} - \mathbf{X}_\alpha \mathbf{b}_{\alpha j_1 j_2}) \tag{3.54}$$

In the process of finding the estimates $\hat{\mathbf{a}}_{1j_1j_2}$, $\hat{\mathbf{a}}_{2j_2}$ and $\hat{\mathbf{a}}_3$ that make (3.52)–(3.54) equal zero, a number of intermediates are derived, given in (3.55)–(3.63), which could be avoided if a metalinear algebra existed to automatically solve the system of equations (3.52)–(3.54) subject to (3.48). Note that (3.55) and (3.56) involve inter products of cell arrays and that (3.55) implies a sequence of heterogenous matrix inversions.

$$\mathbf{R}_\alpha = (\mathbf{X}'^{\mathrm{T}}_{\underline{\alpha}}\mathbf{X}_{\underline{\alpha}})^{-1} \tag{3.55}$$

$$\mathbf{S}_\alpha = \mathbf{R}_{\underline{\alpha}}\mathbf{X}'^{\mathrm{T}}_{\underline{\alpha}} \tag{3.56}$$

$$\mathbf{S}_{\alpha\beta} = \mathbf{S}_\alpha \mathbf{X}_\beta \tag{3.57}$$

$$\mathbf{S}_{123} = \mathbf{S}_{13} - \mathbf{S}_{12}\mathbf{S}_{23} \tag{3.58}$$

$$\mathbf{X}'_1 = \mathbf{X}_1 \tag{3.59}$$

$$\mathbf{X}'_2 = \mathbf{X}_2 - \mathbf{X}_1\mathbf{S}_{12} \tag{3.60}$$

$$\mathbf{X}'_3 = \mathbf{X}_3 - \mathbf{X}_1\mathbf{S}_{123} - \mathbf{X}_2\mathbf{S}_{23} \tag{3.61}$$

$$\bar{\mathbf{y}}_{j_2} = 1_{j_1}\frac{\mathbf{y}_{j_1j_2}}{N_1} \tag{3.62}$$

$$\bar{\mathbf{y}} = 1_{j_2}\frac{\bar{\mathbf{y}}_{j_2}}{N_2} \tag{3.63}$$

Although $\mathbf{S}_{\alpha\beta}$ in (3.57) describes nine matrices, only three are important—the ones on the right hand side of (3.58). The remaining six are either identity matrices (when $\alpha = \beta$) or zero matrices (when $\alpha > \beta$). Equations (3.62) and (3.63) are averages of the MN sensor responses $\mathbf{y}_{j_1j_2}$, taken over each column and over all sensors respectively.

Equations (3.64)–(3.66) give the implicit parameters $\hat{\mathbf{a}}$ that minimise the SSE. These may be transformed in (3.67) to determine the maximum likelihood estimator $\hat{\mathbf{b}}_{\alpha j_1 j_2}$ for the parameters of the raster sensor array problem. Note that the parameters $\hat{\mathbf{a}}_3$ in (3.64), which do not vary from sensor to sensor, depend only on the average responses over all sensors. The parameters $\hat{\mathbf{a}}_{2j_2}$ in (3.65), which vary from column to column, depend on the average responses over each column and over all sensors respectively. The parameters $\hat{\mathbf{a}}_{1j_1j_2}$ in (3.64), which vary from sensor to sensor, depend on all responses as well as the averages mentioned above.

$$\hat{\mathbf{a}}_3 = \mathbf{S}_3\bar{\mathbf{y}} \tag{3.64}$$

$$\hat{\mathbf{a}}_{2j_2} = \mathbf{S}_2\bar{\mathbf{y}}_{j_2} - 1_{j_2}\mathbf{S}_{23}\hat{\mathbf{a}}_3 \tag{3.65}$$

$$\hat{\mathbf{a}}_{1j_1j_2} = \mathbf{S}_1\mathbf{y}_{j_1j_2} - 1_{j_1}\mathbf{S}_{12}\hat{\mathbf{a}}_{2j_2} - 1_{j_1j_2}\mathbf{S}_{13}\hat{\mathbf{a}}_3 \tag{3.66}$$

Table 3.3: Asymptotic time and space performance of a (dense) solution to the raster problem (using implicit constraints).

Equation(s)	Time $O(\cdot)$	Space $O(\cdot)$
(3.55)–(3.57), (3.60) and (3.61)	P^2M	PM
(3.58)	P^3	P^2
(3.59) and (3.64)	PM	PM
(3.62) and (3.63)	MN	MN
(3.65), (3.66) and (3.68)	PMN	MN
(3.67) and (3.70)	PN	PN
Total	PMN	MN

$$\hat{\mathbf{b}}_{\alpha j_1 j_2}\{\}_\alpha^{\mathrm{T}} = \left\{\hat{\mathbf{a}}_{1 j_1 j_2} \quad 1_{j_1}\hat{\mathbf{a}}_{2 j_2} \quad 1_{j_1 j_2}\hat{\mathbf{a}}_3\right\} \tag{3.67}$$

As before, (3.68) uses the estimated parameters $\hat{\mathbf{b}}_{\alpha j_1 j_2}$ to estimate the error variance $\hat{\sigma}_\epsilon^2$, where the number of implicit parameters Q is given in (3.69). P_1, P_2 and P_3 are the number of parameters per sensor that vary from sensor to sensor, that vary from column to column and that do not vary respectively. As in (3.18) and (3.37), the error variance is the SSE divided by the degrees of freedom and is an unbiased estimate.

$$\hat{\sigma}_\epsilon^2 = \frac{SSE(\hat{\mathbf{b}}_{\alpha j_1 j_2})}{MN - Q} \tag{3.68}$$

$$Q = P_1 N + P_2 N_2 + P_3 \tag{3.69}$$

The estimated error variance is used to estimate the inter variance of the parameters $\hat{\mathcal{V}}\{\hat{\mathbf{b}}_{\alpha j_1 j_2}\}$, given in (3.70). These variances are pre-multiplied by $1_{j_1 j_2}$, which means all sensors have the same parameter variances although the sensors may have different parameters. Such a symmetry exists only for the raster problem. Examples of the generic problem may be constructed, solvable in $O(N)$ time and space using the implicit method with sparse arrays and Cholesky factorisation, that have sensor-dependent parameter variances.

$$\hat{\mathcal{V}}\{\hat{\mathbf{b}}_{\alpha j_1 j_2}\}\{\}_\alpha = \left\{\begin{matrix} \hat{\sigma}_\epsilon^2 1_{j_1 j_2}\left(\frac{\operatorname{diag}\mathbf{R}_1}{1} + \frac{(\mathbf{S}_{12}\mathbf{R}_2)\diamond\mathbf{S}_{12}}{N_1} + \frac{(\mathbf{S}_{123}\mathbf{R}_3)\diamond\mathbf{S}_{123}}{N}\right) \\ \hat{\sigma}_\epsilon^2 1_{j_1 j_2}\left(\frac{\operatorname{diag}\mathbf{R}_2}{N_1} + \frac{(\mathbf{S}_{23}\mathbf{R}_3)\diamond\mathbf{S}_{23}}{N}\right) \\ \hat{\sigma}_\epsilon^2 1_{j_1 j_2}\frac{\operatorname{diag}\mathbf{R}_3}{N} \end{matrix}\right\} \tag{3.70}$$

Table 3.3 gives the time and space requirements for the raster method, i.e. to compute (3.55)–(3.70). There are no sparse arrays and no Cholesky factorisation is needed. Cholesky factorisation of the product $\mathbf{X}_{\underline{\alpha}}^{\prime\mathrm{T}}\mathbf{X}_{\underline{\alpha}}$ in (3.55) may be helpful from a non-asymptotic point of view but dense matrix inversion in MATLAB uses LU factorisation anyway, which is also efficient. As Table 3.3 shows, the raster method needs $O(PMN)$ time and $O(MN)$ space, which effectively means an $O(N)$ performance.

Table 3.4: Asymptotic time and space performance of a dense and sparse solution to the generic problem with sensor-varying input, using implicit constraints.

	Time $O(\cdot)$		Space $O(\cdot)$	
Equation	Dense	Sparse	Dense	Sparse
(3.73)	PMN		PMN	
(3.74)	$PQMN$	PMN	QMN	PMN
(3.34)	Q^2MN	Q^2	QMN	Q^2
(3.35)	QMN	Q^2	QMN	Q^2
(3.36)	PQN	PN	PQN	PN
(3.37)	QMN	MN	QMN	MN
(3.38)	PQ^2N	PQN	PQN	
Total	Q^2MN	PQN	QMN	PQN

3.4 Extensions

3.4.1 Sensor-varying input

Equations (3.3), (3.20) and (3.47) relate the observed responses $\mathbf{y}_j$ or $\mathbf{y}_{j_1 j_2}$ of the sensor array to the M input vectors making up the rows of $\mathbf{X}$ or $\mathbf{X}_\alpha$ so that each sensor in the array receives the same input vector for each observation of the array response. A more liberal formulation would permit each sensor to see a different input vector for each observation. For example, one observation of an image sensor is an image that may represent different luminances at each pixel, rather than the same luminance as is normally done for calibration purposes. Interestingly, such a liberal formulation of the generic problem results only in a minor change to the method of solution. Although it may also be done for the explicit method, the case of sensor-varying input is formulated below for the implicit method.

Equations (3.20) and (3.24) are modified to Equations (3.71) and (3.72), where the lattice $\mathbf{X}_j$ represents the PMN sensor-dependent inputs. This is another example of the usefulness of the inter product. The remaining equations in Section 3.2.2 are unchanged except for the substitutions in (3.30) and (3.31). These are replaced by (3.73) and (3.74), reflecting the sensor-dependence of the input.

$$\mathbf{y}_j = \mathbf{X}_{\underline{j}} \mathbf{b}_{\underline{j}} + \boldsymbol{\epsilon}_j \tag{3.71}$$

$$SSE(\mathbf{b}_j) = 1_j \|\mathbf{y}_j - \mathbf{X}_{\underline{j}} \mathbf{b}_{\underline{j}}\|^2 \tag{3.72}$$

$$\mathbf{z}_j = \mathbf{y}_j - \mathbf{X}_{\underline{j}} \mathbf{b}_{0\underline{j}} \tag{3.73}$$

$$\mathbf{W}_j = \mathbf{X}_{\underline{j}} \mathbf{U}_{\underline{j}} \tag{3.74}$$

Table 3.4 gives the time and space requirements of the implicit method for sensor-varying input, i.e. to compute (3.73), (3.74) and (3.34)–(3.38). The only changes compared to Table 3.2 are in the space requirements for (3.73) and the sparse version of (3.74). The required space is at least as much as it takes to store the dense lattice

$\mathbf{X}_j$. For the raster problem, the time requirements of the sparse version of (3.74) equals that of (3.31) because the sparsity of $\mathbf{U}_j$ dominates the result $\mathbf{W}_j$, which continues to have $O(MN)$ nonzeros. The performance of the remaining equations in Section 3.2.2 as well as the Cholesky factorisation in Section 3.2.3 does not change.

Thus, the sensor-varying input problem is hardly more difficult than the sensor-constant input problem. Similar modifications to those described above for the implicit method may be derived for the explicit or raster methods. These modifications mostly involve multiplications and inversions that entail inter products.

3.4.2 Nonlinear optimisation

Multilinear regression may be applied to estimate parameters of a model so long as the output, which may be nonlinear functions of the dependent variables, is a linear function of the inputs, which may be nonlinear functions of the independent variables, and Gaussian error. For example, multilinear regression may be applied to estimate the parameters a and b in the model given by (3.75), where y is the dependent variable, x is the independent variable, c is a constant and ϵ is the Gaussian error.

$$y = a + b\ln(c + x) + \epsilon \tag{3.75}$$

The nonlinearity in (3.75) is no complication because the equation may be rewritten by (3.76)–(3.78), where $\mathbf{x}$ is a row vector of inputs and $\mathbf{b}$ is a column vector of parameters.

$$y = \mathbf{x}\mathbf{b} + \epsilon \tag{3.76}$$

$$\mathbf{x} = \begin{pmatrix} 1 & \ln(c + x) \end{pmatrix} \tag{3.77}$$

$$\mathbf{b} = \begin{pmatrix} a & b \end{pmatrix}^{\mathrm{T}} \tag{3.78}$$

However, if c in the example of (3.75) is not a constant but a parameter then nonlinear optimisation is required because a linear decomposition of inputs and parameters, as in (3.76), is impossible. Nonlinear optimisation involves minimisation of the SSE in (3.79) over parameters $\mathbf{b}$ in (3.78) and c in (3.77). Nonetheless, for any value of c (providing $c > -x$), the values of $\mathbf{b}$ that minimise the SSE in (3.79), denoted $\hat{\mathbf{b}}$, may be estimated by multilinear regression. Thus, the minimum SSE is a known function of c alone and nonlinear optimisation needs to estimate one parameter instead of three.

$$SSE(\mathbf{b}, c) = ||y - \mathbf{x}(c)\mathbf{b}||^2 \tag{3.79}$$

Generalising the above example, a *nonlinear sensor array problem*, where relationships between dependent and independent variables of sensors in an array include nonlinear parameters and constraints, may be simplified with constrained multilinear regression. Let the vector $\mathbf{w}$ denote the fewest parameters in the nonlinear problem whereby, when $\mathbf{w}$ is constant, the remaining parameters $\mathbf{b}_j$ may be estimated by multilinear regression with linear constraints. Thus, the minimum SSE over $\mathbf{b}_j$ and $\mathbf{w}$ is a known function $f(\mathbf{w})$ over $\mathbf{w}$ alone, as in (3.80), where $\hat{\mathbf{b}}_j$ is an estimate of $\mathbf{b}_j$ derived by the generic or raster methods (possibly with sensor-varying input) for given $\mathbf{w}$. The vector $\mathbf{w}$ may be estimated by minimising $f(\mathbf{w})$ with nonlinear optimisation.

$$f(\mathbf{w}) = SSE(\hat{\mathbf{b}}_j, \mathbf{w}) \tag{3.80}$$

If the nonlinear problem includes either linear or nonlinear constraints on the parameters $\mathbf{w}$ (nonlinear constraints are not permitted on the parameters $\mathbf{b}_j$) then an explicit or implicit method may be used to enforce these constraints, as before. An explicit method involves the nonlinear optimisation of a Lagrangian function $\mathcal{L}(\mathbf{w}, \boldsymbol{\lambda})$, as in (3.81), with a vector function $\mathbf{g}(\mathbf{w})$ of constraints and a vector $\boldsymbol{\lambda}$ of multipliers. An implicit method transforms the constrained optimisation of $f(\mathbf{w})$ over a space of parameters $\mathbf{w}$ to the unconstrained optimisation of another function $\tilde{f}(\tilde{\mathbf{w}})$ over a subspace of parameters $\tilde{\mathbf{w}}$. A combination of the two methods is also possible whereby some constraints, e.g. the linear ones, are expressed implicitly with the rest expressed explicitly (it may not be possible to express some nonlinear constraints implicitly).

$$\mathcal{L}(\mathbf{w}, \boldsymbol{\lambda}) = f(\mathbf{w}) + \mathbf{g}(\mathbf{w}) \bullet \boldsymbol{\lambda} \tag{3.81}$$

The error variance σ_ϵ^2 for nonlinear optimisation may be estimated as before, by dividing the SSE realised with estimated parameters $\hat{\mathbf{b}}_j$ and $\hat{\mathbf{w}}$ by the degrees of freedom. The degrees of freedom in the calibration equals the number of equations, including linear and nonlinear constraints, minus the number of variables, counting linear and nonlinear parameters. Nonlinear optimisation, however, complicates the estimation of parameter variances $\mathcal{V}\{\hat{\mathbf{b}}_j\}$ and $\mathcal{V}\{\hat{\mathbf{w}}\}$. Given that the gradient of the SSE with respect to $\mathbf{w}$ is zero for the estimate $\hat{\mathbf{w}}$, due to nonlinear optimisation, the inter variance of $\hat{\mathbf{b}}_j$ may be estimated by the generic or raster methods (possibly with sensor-varying input) to a first order approximation. A better approximation requires the hessian of the SSE with respect to $\mathbf{w}$ for the estimate $\hat{\mathbf{w}}$, which is also needed to estimate the inter variance of $\hat{\mathbf{w}}$. For simplicity, nonlinear problems considered in this thesis ignore the stochasticity of the estimate $\hat{\mathbf{w}}$ to avoid calculation of the hessian.

3.5 Simulations

In this section, an example of the raster problem is simulated to illustrate the use of constrained regression in the modelling and calibration of sensor arrays. The time and space performance of different methods to solve the problem are compared (all methods give the same solution). The example consists of an array of $N_1 \times N_2$ sensors, as in Figure 3.1, where the output of each sensor is a linear function of a single input. Each sensor therefore has an offset and gain parameter.

3.5.1 Modelling and calibration

Three different models are simulated for the sensor array described above. In the sensor-varying gain (SVG) model, the gain may vary from sensor to sensor. In the column-varying gain (CVG) model, the gain may vary only from column to column. Thirdly, in the non-varying gain (NVG) model, the gain does not vary at all. For each model, the offset may vary from sensor to sensor irrespective of the constraints on the gain. Ten observations were generated for each model and each sensor in a 10×10 array of sensors by varying the input from one to ten in integer steps. The offset and gain parameters were chosen randomly from a uniform probability distribution ranging

Table 3.5: The number of explicit constraints L and implicit parameters Q for three models of a sensor array with 200 explicit parameters PN.

Model	L	Q
Sensor-varying gain (SVG)	0	200
Column-varying gain (CVG)	90	110
Non-varying gain (NVG)	99	101

Table 3.6: The residual error, or square root of the estimated error variance $\hat{\sigma}_\epsilon^2$, when simulated SVG, CVG and NVG scenarios are calibrated for SVG, CVG and NVG hypotheses. Over-constrained models give worse results (which are italicised).

Scenario	Residual error		
SVG	**0.10**	*0.85*	*0.88*
CVG	0.10	**0.10**	*1.00*
NVG	0.10	0.10	**0.10**
Hypothesis:	SVG	CVG	NVG

from zero to one. Finally, Gaussian error with a mean of zero and a standard deviation, i.e. σ_ϵ, of 0.1 was added to the sensor responses.

Each of these models may be calibrated by the generic or raster methods, where the number of observations M is ten, the number of parameters per sensor P is two, the number of sensors N is 100 (as $N_1 = 10$ and $N_2 = 10$) and the number of explicit constraints L and implicit parameters Q depend on the model, as in Table 3.5. Note that L and Q always sum to the number of explicit parameters PN (i.e. 200), as in (3.27). For the SVG model, there are no explicit constraints so the implicit parameters equal the explicit parameters. The CVG and NVG models show increasing numbers of explicit constraints (on the gain) and therefore decreasing numbers of implicit parameters.

The user may not know exactly what constraints apply to the model parameters of a sensor array. However, hypotheses may be tested and compared according to the estimated error and parameter variances $\hat{\sigma}_\epsilon^2$ and $\hat{\mathcal{V}}\{\hat{\mathbf{b}}_j\}$, which are often square rooted to give standard deviations called the *residual error* and *parameter uncertainties* respectively. A hypothesis that over-constrains the parameters results in a higher residual error because the model is incompatible with the scenario. A hypothesis that under-constrains the parameters results in higher parameter uncertainties because the extra degrees of freedom in the model attempt to calibrate the stochastic error.

Table 3.6 gives the residual error when simulated responses for the SVG, CVG and NVG scenarios are calibrated for the SVG, CVG and NVG hypotheses. If the hypothesis is incompatible with the scenario (italicised entries) then the residual error is high. However, if the hypothesis is compatible, even if it is not precise, the residual error approximates the actual standard deviation of the Gaussian error because the estimator is unbiased when the model is compatible. Therefore, the residual error distinguishes

Table 3.7: The parameter uncertainties, or square root of the estimated parameter variances $\hat{\mathcal{V}}\{\hat{\mathbf{b}}\}$, when simulated SVG, CVG and NVG scenarios are calibrated for SVG, CVG and NVG hypotheses. Under-constrained models give worse results (italicised).

Scenario	Parameter uncertainties			
SVG	Offset:	**0.26**	0.56	0.53
	Gain:	**0.10**	0.17	0.10
CVG	Offset:	*0.27*	**0.20**	0.57
	Gain:	*0.11*	**0.06**	0.10
NVG	Offset:	*0.26*	*0.19*	**0.18**
	Gain:	*0.11*	*0.06*	**0.03**
Hypothesis:		SVG	CVG	NVG

only compatible models from incompatible ones.

If there is more than one hypothesis that is compatible with the scenario then a distinction between them may be made by comparing the parameter uncertainties. Table 3.7 shows, for scenarios in Table 3.6 where the residual error is similar between hypotheses, that the parameter uncertainties identify the correct model (boldface entries). When the constraints of the scenario are matched by the constraints of the hypothesis, there is less uncertainty in the estimated parameters. If the hypothesis has fewer constraints than the scenario (italicised entries) then the estimated parameters are more likely to be corrupted by the Gaussian error in the observations. Appropriate constraints in a hypothesis compensate for stochastic error in the observations.

Therefore, constrained regression is useful in the analysis of sensor arrays, both for identifying relationships between parameters of a model and for calibrating the model to minimise, first, the residual error and, second, the parameter uncertainties.

3.5.2 Time and space performance

The previous section demonstrated the usefulness of constrained regression for the modelling and calibration of an array with $N = 10 \times 10$ sensors, where the output of each sensor is a linear function of a single input. The same problem is examined in this section but with an evaluation of the time and space performance in MATLAB of various methods of solution as N varies from 1 to 1000, where N_1 and N_2 approximate $\sqrt{N}$ so that the sensor array is roughly square, as with image sensors.

Figure 3.2 shows the number of flops required as a function of N by the explicit and implicit methods, without and with Cholesky factorisation, and by the raster method. The number of flops, for each point in the figure, represents the total number of arithmetic operations needed to solve the nine multilinear regression problems required to produce Tables 3.6 and 3.7 (all methods give the same results). When N is large enough, the explicit method takes $O(N^3)$ time. In contrast, the implicit method takes $O(N^2)$ time without Cholesky factorisation but $O(N)$ time with Cholesky factorisation. The raster method is the fastest and takes $O(N)$ time.

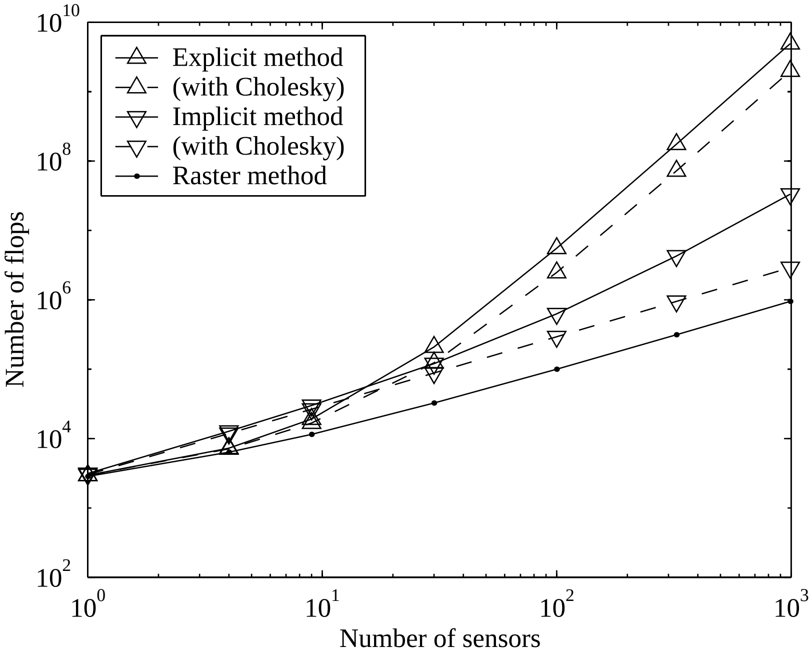


Figure 3.2: Number of flops versus number of sensors to solve a simulated problem by the explicit, implicit and raster methods, without and with Cholesky factorisation.

The explicit and implicit methods were implemented, for the results in Figure 3.2, using sparse arrays. Dense versions were also tested and found to take $O(N^3)$ time. The raster method, however, uses only dense arrays but takes $O(N)$ time. Nonetheless, the explicit and implicit methods may be applied to solve a wider class of generic problems than the raster method, which only solves the raster problem. Furthermore, although the implicit method with Cholesky factorisation uses only $O(N)$ flops, it takes $O(N \log N)$ time because of a sub-optimal sparse array implementation in MATLAB, as described in Chapter 2. MATLAB's flops counter, which was used to produce these results, does not count the $O(N \log N)$ comparisons and swaps involved in an unavoidable internal sort. A better implementation of sparse arrays would yield an $O(N)$ time performance, proportional to the number of arithmetic operations.

Figure 3.3 shows the number of bytes required to solve the nine multilinear regression problems required to produce Tables 3.6 and 3.7 as N varies from 1 to 1000. The explicit method, without or with Cholesky factorisation, and the implicit method, without Cholesky factorisation, take $O(N^2)$ space. However, the implicit method, with Cholesky factorisation, and the raster method need only $O(N)$ space. The raster method is the smallest, almost an order of magnitude better in memory use than the second best method. Memory use affects processing time because a greater memory requirement entails more frequent disk access if the required memory does not all fit

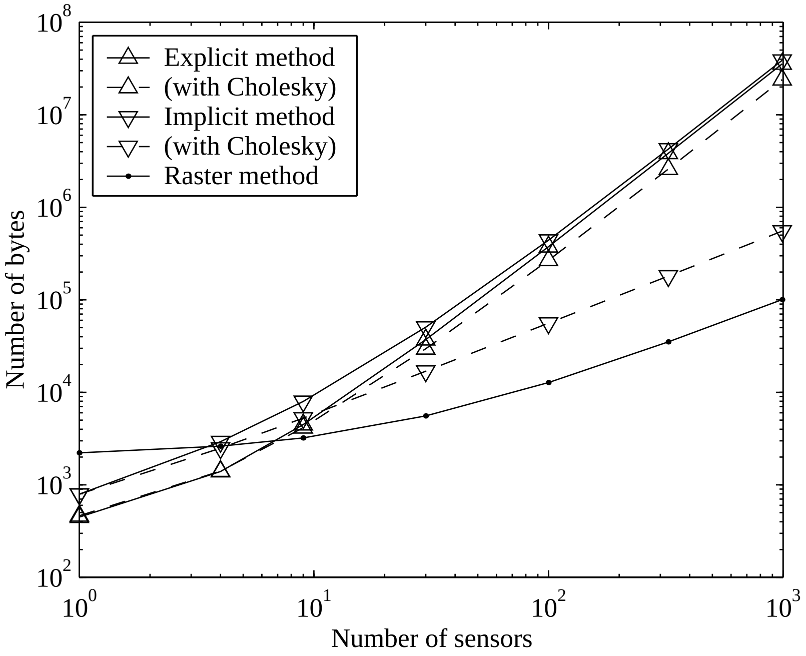


Figure 3.3: Number of bytes versus number of sensors to solve a simulated problem by the explicit, implicit and raster methods, without and with Cholesky factorisation.

inside the working memory of the computer at one time.

Figures 3.4 and 3.5 show the time and space requirements of the explicit and implicit methods implemented using only scalars, vectors and matrices, including sparse vectors and matrices, and the operators of classical linear algebra. The performance of the raster method, which may not be derived with classical linear algebra, is given for comparison. The explicit method takes $O(N^3)$ flops, as in Figure 3.2, and the implicit method takes $O(N^2)$ flops irrespective of Cholesky factorisation. A degradation in performance with the latter occurs because classical linear algebra does not have the operators to compute the inter variance and must necessarily compute the outer variance of the estimated parameters to obtain the parameter uncertainties, which is an $O(N^2)$ task. The covariance of parameters tends to be dense due to relationships across the sensor array. For the same reason, the space requirements of the explicit and implicit methods are $O(N^2)$ in Figure 3.5, regardless of Cholesky factorisation.

Thus, the raster problem may be solved using $O(N)$ flops and bytes as a special case of the generic problem by the implicit method with Cholesky factorisation. However, this solution requires $O(N \log N)$ time because of imperfections in the sparse array routines. Nonetheless, the raster method solves the problem in $O(N)$ time and space using only dense arrays. The best performance that may be obtained using classical linear algebra is $O(N^2)$ in time and space, which is unacceptable for large N.

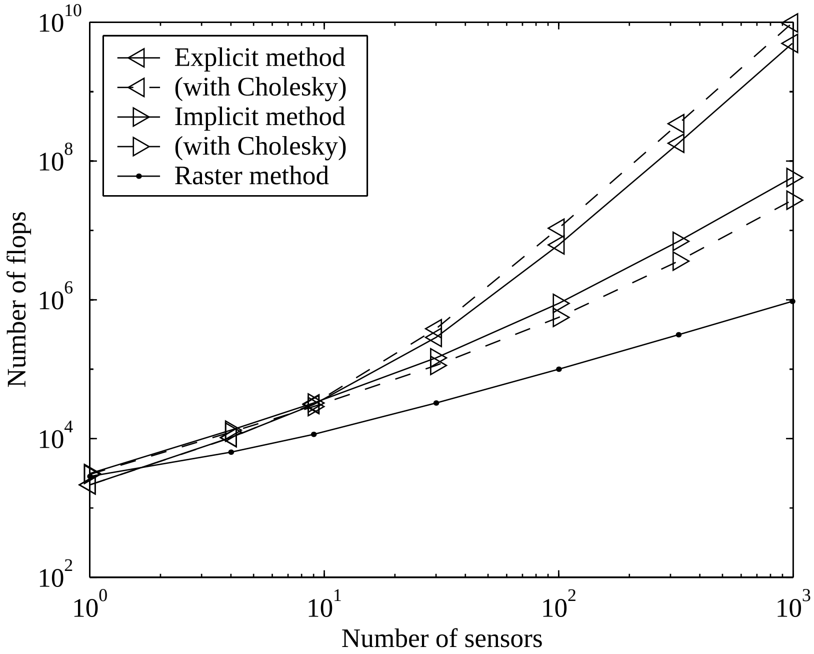


Figure 3.4: Number of flops versus number of sensors to solve a simulated problem, using classical linear algebra, by the explicit and implicit methods, without and with Cholesky factorisation. Performance of the raster method is given for comparison.

3.6 Conclusion

This chapter examined the problem of constrained regression for the analysis of sensor arrays. In the generic problem, the parameters of the array may be linearly constrained in an arbitrary way. In the raster problem, sensor parameters may be linearly constrained in one of three ways, due to raster scanning of the array. The generic problem may be solved by formulating it as a multilinear regression problem with explicit linear constraints on the parameters or as a multilinear regression problem over a linear subspace of the parameter space whereby the constraints are implicit. Performance is expected to be $O(N)$ in time and space when the constraint array of the explicit method or the transformation array of the implicit method has $O(N)$ nonzeros, certain products of these arrays also have $O(N)$ nonzeros and the computing of dense inverses and outer variances are avoided with Cholesky factorisation and inter products. A formulation to solve the raster problem alone, called the raster method, was derived that operates in $O(N)$ time and space with no sparse arrays. These results are useful for the efficient modelling and calibration of sensor arrays.

An example of the raster problem was simulated in MATLAB. The example demonstrated that the relationship between parameters of a model may be deduced by cali-

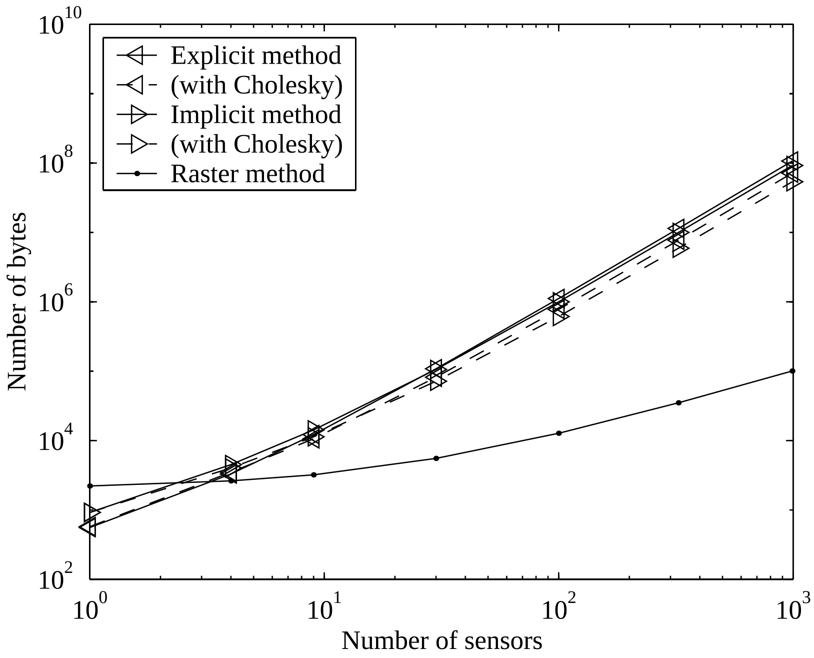


Figure 3.5: Number of bytes versus number of sensors to solve a simulated problem, using classical linear algebra, by the explicit and implicit methods, without and with Cholesky factorisation. Performance of the raster method is given for comparison.

bration of hypotheses, using constrained regression, and comparison of the residual error and parameter uncertainties. The residual error distinguishes hypotheses that are compatible with the scenario from those that are incompatible. The parameter uncertainties identify specific hypotheses from more general (and also compatible) ones. When the correct model is calibrated, the residual error and parameter uncertainties are minimised. The simulation also demonstrated that the explicit method for solving the generic problem performs poorly on the raster problem, needing $O(N^3)$ time and $O(N^2)$ space, because the sparsity conditions for $O(N)$ performance are not met. However, the implicit method for solving the generic problem provides a solution to the raster problem that takes $O(N)$ flops and bytes, although it takes $O(N \log N)$ time because of an imperfect sparse array implementation. The raster method takes $O(N)$ time and space to solve the raster problem, using no sparse arrays. Linear algebraic solutions to the generic problem were also implemented for the explicit and implicit methods (the raster problem cannot be solved directly with classical linear algebra). However, these solutions could not achieve a performance better than $O(N^2)$ on the raster problem because classical linear algebra does not possess inter product operators.

Chapter 4

Fixed pattern noise

4.1 Introduction

As described in Chapter 1, the biggest problem with logarithmic CMOS image sensors is fixed pattern noise (FPN), which is a distortion that appears in an image due to variations of device parameters across the sensor. Dierickx, Scheffer, Loose and others have developed digital and analogue methods to correct FPN by assuming it is independent of illuminance [32, 24, 21]. Loose et al briefly considered FPN as a linear function of illuminance but were unable to compensate for this dependence with their analogue circuit architecture and concluded that it was not significant [21]. However, Yadid-Pecht notes that FPN varies nonlinearly with illuminance in a logarithmic sensor but she neither characterises nor attempts to correct this distortion [25]. This chapter, however, makes a detailed study of FPN in logarithmic CMOS image sensors.

Section 4.2 uses semiconductor theory to model the response of a single logarithmic pixel to illuminance. Section 4.3 considers various models of FPN that may arise in an array of such pixels and derives methods of calibration, using constrained regression and images of a uniform scene. Section 4.4 describes the correction of FPN in arbitrary images using calibrated models. With simulation and experiment, Sections 4.5 and 4.6 compare the calibration and correction of the various FPN models.

4.2 Modelling

Figure 4.1 shows the process by which light stimulus, of illuminance x, falling on a pixel in a typical logarithmic CMOS sensor is converted to a digital response y.[1] Before the light reaches the photodiode in the pixel, it is attenuated due to absorption and reflection by the aperture and lens of the camera, which may be represented by gains G_A and G_L. The attenuation may vary spatially, i.e. from pixel to pixel across the image sensor, which is known as vignetting. Photons absorbed by the photodiode form

[1] In Figure 4.1, the pixel circuit is from Scheffer et al [24] and the remaining circuits are from Mendis et al [20]. The column circuit uses a PMOS source follower to compensate for the voltage shift by the NMOS source follower in the pixel circuit [20].

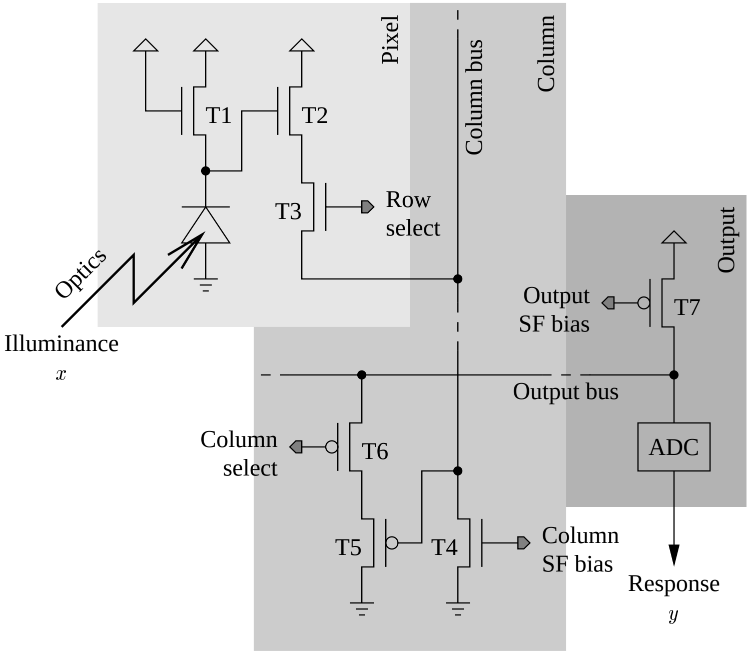


Figure 4.1: From an illuminance x to a digital response y in one pixel of a logarithmic CMOS image sensor. Transistors $\mathrm{T2}$ with $\mathrm{T4}$ and $\mathrm{T5}$ with $\mathrm{T7}$ form an NMOS and PMOS source follower (SF) respectively, when $\mathrm{T3}$ and $\mathrm{T6}$ are turned on.

electron–hole pairs that are swept out by the electric field across the device to produce a current I_P, given in (4.1). This photocurrent is linearly related to the incident light intensity over many orders of magnitude. The relationship depends on the quantum efficiency, which may be represented by a gain G_Q, and the light-sensitive area A of the photodiode.

$$I_P = G_A G_L G_Q A x \tag{4.1}$$

The photodiode in Figure 4.1 is reverse biased to prevent any current flowing to ground through it except for the photocurrent. However, a small leakage current I_S, known as the reverse bias saturation current, also flows to ground through this diode. The total current $I_P + I_S$ sets the gate voltage V_G^{T2}, given in (4.2), of transistor T2 via the diode-connected load transistor T1, where V_{DD} is the supply voltage. Designed to operate in the subthreshold region, T1 has a logarithmic current-to-voltage relationship that is valid over several decades of current amplitude.

$$V_G^{\mathrm{T2}} = V_{DD} - \frac{n^{\mathrm{T1}} kT}{q} \ln \left(\frac{I_P + I_S}{I_{on}^{\mathrm{T1}}} \right) - V_{on}^{\mathrm{T1}} \tag{4.2}$$

Transistor T3 is a switch that is either an open or a short circuit between T2 and the common bus for a column of pixels. This column bus is biased by transistor T4. When

$\mathrm{T}3$ is off, $\mathrm{T}2$ is disconnected from the bus and does not affect its voltage. When $\mathrm{T}3$ is on, a similar switch is off for all other pixels in the column and the gate voltage $V_G^{\mathrm{T}5}$ of transistor $\mathrm{T}5$, given in (4.3), equals the source voltage $V_S^{\mathrm{T}2}$ of $\mathrm{T}2$. As $\mathrm{T}2$ and $\mathrm{T}4$ have the same drain-source current, when $\mathrm{T}3$ is on, and as both operate in saturation, their gate-source minus threshold voltages $V_{GS}^{\mathrm{T}2} - V_T^{\mathrm{T}2}$ and $V_{GS}^{\mathrm{T}4} - V_T^{\mathrm{T}4}$ are linearly related with a dependence on the ratio of current gains $K^{\mathrm{T}2}$ and $K^{\mathrm{T}4}$.

$$V_G^{\mathrm{T}5} = V_G^{\mathrm{T}2} - V_T^{\mathrm{T}2} - \sqrt{\frac{K^{\mathrm{T}4}}{K^{\mathrm{T}2}}}\big(V_{GS}^{\mathrm{T}4} - V_T^{\mathrm{T}4}\big) \tag{4.3}$$

When a pixel is connected to the bus for its column, all pixels in the same row are connected to their respective column buses. However, the analogue-to-digital converter (ADC) processes only one voltage at a time. Therefore, the column buses are switched in sequence onto a common output bus, which is biased by transistor $\mathrm{T}7$, using a two-transistor circuit similar to the one described above. When transistor $\mathrm{T}6$ is switched on, $\mathrm{T}5$ is connected to the output bus and the voltage V_{ADC} at the input of the ADC, given in (4.4), equals the source voltage $V_S^{\mathrm{T}5}$ of $\mathrm{T}5$.

$$V_{ADC} = V_G^{\mathrm{T}5} - V_T^{\mathrm{T}5} - \sqrt{\frac{K^{\mathrm{T}7}}{K^{\mathrm{T}5}}}\big(V_{GS}^{\mathrm{T}7} - V_T^{\mathrm{T}7}\big) \tag{4.4}$$

Rather than getting into the details of ADC circuits, equation (4.5) abstracts the digitisation of voltage V_{ADC} by a clipping function, to limit the maximum and minimum output values, and by rounding off, which introduces quantisation error. Furthermore, the ADC adjusts its input V_{ADC} by an offset F_{ADC} and gain G_{ADC} to fit the domain of voltages to the range of integer codes (e.g. 0–255LSB for an 8-bit ADC).

$$y = \mathrm{round}\big(\,\mathrm{clip}\,\big(F_{ADC} + G_{ADC}V_{ADC}\big)\big) \tag{4.5}$$

If the input voltage does not cause clipping, digitisation may be modelled by a quantisation error term ϵ_Q, with a range of ± 0.5LSB, that is added to the output. Furthermore, the whole process in Figure 4.1 will add noise components at various stages. However, the noise shall be modelled by a single random variable ϵ_N added to the output. A further term ϵ_M may be added to the output to account for error in the underlying device models. Considering these remarks, equation (4.6) gives the digital response y of a pixel.

$$y = F_{ADC} + G_{ADC}V_{ADC} + \epsilon_Q + \epsilon_N + \epsilon_M \tag{4.6}$$

Grouping the equations and physical parameters above, equations (4.7)–(4.11) give the digital response y of a pixel as a logarithm of the illuminance x, with three abstract parameters a, b and c, named the *offset*, *gain* and *bias*, and a stochastic error ϵ. A pixel-to-pixel or column-to-column variation of a, b, c or a combination thereof causes FPN. Therefore, these parameters must be estimated by calibration to correct FPN in an image. Furthermore, the residual error and parameter uncertainties must be estimated to validate the model and determine the accuracy of calibration and correction.

$$y = a + b\ln(c + x) + \epsilon \tag{4.7}$$

$$\begin{aligned} a = F_{ADC} + G_{ADC}\Bigg(&V_{DD} \\ &+ \frac{n^{\mathrm{T1}}kT}{q}\ln\left(\frac{I_{on}^{\mathrm{T1}}}{G_A G_L G_Q A}\right) - V_{on}^{\mathrm{T1}} \\ &- V_T^{\mathrm{T2}} - \sqrt{\frac{K^{\mathrm{T4}}}{K^{\mathrm{T2}}}}\left(V_{GS}^{\mathrm{T4}} - V_T^{\mathrm{T4}}\right) \\ &- V_T^{\mathrm{T5}} - \sqrt{\frac{K^{\mathrm{T7}}}{K^{\mathrm{T5}}}}\left(V_{GS}^{\mathrm{T7}} - V_T^{\mathrm{T7}}\right)\Bigg) \end{aligned} \tag{4.8}$$

$$b = -G_{ADC}\frac{n^{\mathrm{T1}}kT}{q} \tag{4.9}$$

$$c = \frac{I_S}{G_A G_L G_Q A} \tag{4.10}$$

$$\epsilon = \epsilon_Q + \epsilon_N + \epsilon_M \tag{4.11}$$

4.3 Calibration

Calibration of a logarithmic image sensor may be accomplished by minimising the sum square error (SSE) between the actual response y in (4.7) and the estimated response $\hat{y}$ in (4.12), to illuminance x, over parameters a, b and c. The estimated response differs from the actual response by lacking a stochastic error ϵ, which is unpredictable.

$$\hat{y} = a + b\ln(c + x) \tag{4.12}$$

For M different illuminances x_i, where $1 \leq i \leq M$, that are observed uniformly by N pixels in an image sensor, the SSE is given in (4.13), where y_{ij} and $\hat{y}_{ij}$ are the actual and estimated responses with $1 \leq j \leq N$. The stochastic error ϵ_{ij}, which is the difference between y_{ij} and $\hat{y}_{ij}$, is assumed to be statistically independent from sample to sample and to follow a zero-mean Gaussian distribution.

$$SSE = 1_{ij}(y_{ij} - \hat{y}_{ij})^2 \tag{4.13}$$

There are potentially $3N$ variables in a calibration, counting the three explicit parameters a, b and c per pixel. However, since the complexity and robustness of calibration depend on the number of parameters needing estimation, no more variables should be permitted than are absolutely necessary. The number of variables may be reduced by constraining the parameters. Many different types of constraints are possible. The most plausible assume that a variation of the offset, gain, bias or a combination thereof occurs because of a variation in their underlying physical parameters. These physical parameters, given in (4.8)–(4.10), may be divided into three groups: those that belong to the pixel circuit (i.e. photodiode and transistors T1–3), the column circuit (i.e. transistors T4 and T5) or the output circuit (i.e. transistor T7 and the ADC) of Figure 4.1.

Assuming that physical parameters in each circuit group either vary from device to device or remain constant across the die, three possibilities exist for abstract parameters a, b and c—each may vary from pixel to pixel, from column to column or not at all.

Table 4.1: Estimated response $\hat{y}_{ij}$ of the j^{th} logarithmic pixel to illuminance x_i for the four models of FPN with spatially constant bias c, where $l_i = \ln(1_i c + x_i)$. The number of implicit parameters Q is given for cases where x_i is known and unknown.

Model	$\hat{y}_{ij}$	Q	Q
1	$1_{ij}a + 1_j b l_i$	3	M
2	$1_i a_j + 1_j b l_i$	$N + 2$	$M + N - 1$
3	$1_{ij}a + b_j l_i$	$N + 2$	$M + N$
4	$1_i a_j + b_j l_i$	$2N + 1$	$M + 2N - 2$
		Known x_i	Unknown x_i

Strictly speaking, the gain and bias in (4.9) and (4.10) do not depend on column circuit parameters and so may not vary from column to column. However, a consideration of column-to-column variation is deferred until Chapter 5. As a result, there are eight possible hypotheses for constraints on the parameters in a logarithmic image sensor. These may be divided into two groups of four—one for constant bias and one for varying bias. As described below, constant bias models may be calibrated by multilinear regression whereas varying bias models require nonlinear optimisation.

4.3.1 Constant bias

Table 4.1 gives the four models of FPN where the bias does not vary from pixel to pixel, with l_i given in (4.14), and lists the number of implicit parameters Q in each.

$$l_i = \ln(1_i c + x_i) \tag{4.14}$$

If the bias c and illuminance x_i in (4.14) are known then models in the table are examples of the raster problem and calibration may be achieved using the raster method. In general, the bias is unknown. The illuminances may be known if produced by a calibrated light source of variable intensity or if the output of an uncalibrated light source of variable intensity is measured with a light meter. The illuminances may also be known if produced by a constant light source, measured with a light meter, with neutral density filters or aperture settings used to simulate varying intensity.

Nonetheless, it is desirable to avoid measurements where possible. This may be done by taking the illuminances x_i in Table 4.1 to be M unknown parameters, which must be added to Q. Such an action, however, introduces a new complication. Observe that the estimated response $\hat{y}$ in (4.12) is invariant under transformations (4.15)–(4.17), which means that the SSE in (4.13) does not have a unique global minimum for any of the models in Table 4.1. Transformation (4.15) does not apply to the third model, however, because a may not vary in this model but b may vary. These degeneracies mean that there are three fewer implicit parameters for each model in Table 4.1 (two fewer for the third model), which explains the deductions from Q.

$$(a, b, c, x) \equiv (a - b \ln \gamma, b, \gamma c, \gamma x) \tag{4.15}$$

Table 4.2: Estimated response $\hat{y}_{ij}$ of the j^{th} logarithmic pixel in terms of average response $\bar{y}_i$ for the models of FPN with spatially constant bias c, where $l_i = \ln(1_i c + x_i)$. The number of implicit parameters Q is given for the case where x_i is unknown.

Model	$\hat{y}_{ij}$	$\bar{y}_i$	a'_j	b'_j	Q
1	$1_j \bar{y}_i$	$1_i a + b l_i$			M
2	$1_i a'_j + 1_j \bar{y}_i$	$1_i \bar{a} + b l_i$	$a_j - 1_j \bar{a}$		$M + N - 1$
3	$1_i a'_j + b'_j \bar{y}_i$	$1_i a + \bar{b} l_i$	$1_j a - b'_j a$	$b_j / \bar{b}$	$M + 2N - 2$
4	$1_i a'_j + b'_j \bar{y}_i$	$1_i \bar{a} + \bar{b} l_i$	$a_j - b'_j \bar{a}$	$b_j / \bar{b}$	$M + 2N - 2$

$$\equiv (a, b/\gamma, 0, (c + x)^\gamma) \tag{4.16}$$

$$\equiv (a, b, c - \gamma, x + \gamma) \tag{4.17}$$

The requirement for nonlinear optimisation to calibrate the models of Table 4.1, due to nonlinear parameters c and x_i, may be avoided with one assumption. For each illuminance x_i, assume that the average of the actual pixel responses, denoted $\bar{y}_i$ in (4.18), equals the average of the estimated pixel responses, as in (4.19). This assumption is reasonable when the standard deviation σ_ϵ of the zero-mean Gaussian error ϵ_{ij}, which accounts for the difference between actual and estimated responses, is small relative to the number of pixels. Taking an average over N pixels in (4.18) reduces the standard deviation by a factor of $\sqrt{N}$ so that the error may be ignored, as in (4.19).

$$\bar{y}_i = \frac{1_j}{N} y_{ij} \tag{4.18}$$

$$\bar{y}_i \approx \frac{1_j}{N} \hat{y}_{ij} \tag{4.19}$$

Applying the assumption in (4.19) for each hypothesis in Table 4.1, the models of the estimated response $\hat{y}_{ij}$ may be simplified, as in Table 4.2, by making substitutions for the offset and gain, denoted a'_j and b'_j, in some cases. The table also lists the number of implicit parameters Q for each model, which is the total number of variables (i.e. $\bar{y}_i$, a'_j and b'_j, as appropriate) minus degeneracies. The average over all pixels of a'_j and b'_j equals zero and one respectively, which provides one or two degeneracies.

A comparison of Tables 4.1 and 4.2 reveals that the number of implicit parameters are equal (for unknown x_i) except for the third model. The assumption in (4.19) has increased the number of implicit parameters by $N - 2$ for this model, which is a small price to pay for avoiding the nonlinear optimisation required without the assumption. All the models in Table 4.2 may be calibrated using the raster method. The first model doesn't need any calibration as no variables remain after the assumption in (4.19). The third and fourth models are rendered equal by the assumption, which reduces the number of hypotheses to three when the bias does not vary.

Table 4.3: Estimated response $\hat{y}_{ij}$ of the j^{th} logarithmic pixel to illuminance x_i for the four models of FPN with spatially varying bias c_j, where $l_{ij} = \ln(1_i c_j + 1_j x_i)$. The number of implicit parameters Q is given for cases where x_i is known and unknown.

Model	$\hat{y}_{ij}$	Q	Q
1	$1_{ij}a + bl_{ij}$	$N + 2$	$M + N$
2	$1_i a_j + bl_{ij}$	$2N + 1$	$M + 2N - 1$
3	$1_{ij}a + b_{\underline{j}} l_{i\underline{j}}$	$2N + 1$	$M + 2N$
4	$1_i a_j + b_{\underline{j}} l_{i\underline{j}}$	$3N$	$M + 3N - 2$
		Known x_i	Unknown x_i

4.3.2 Varying bias

Table 4.3 gives the four models of FPN where the bias may vary from pixel to pixel, with l_{ij} given in (4.20), and lists the number of implicit parameters Q in each.

$$l_{ij} = \ln(1_i c_j + 1_j x_i) \tag{4.20}$$

If c_j and x_i in (4.20) are known then the estimated response $\hat{y}_{ij}$, for each model in the table, is a linear function of l_{ij} with offset and gain parameters. The parameters may not be estimated by the raster method, as derived in Chapter 3, because of the sensor varying input l_{ij} but the raster method may be extended to account for this condition.

Generally, the biases c_j are unknown, which means the number of implicit parameters for a model in Table 4.3 is $N - 1$ times greater than the corresponding number in Table 4.1 (for known x_i). Although x_i may be known by taking measurements during calibration, the illuminances are taken as M unknown parameters as in Section 4.3.1, which increases Q. Such an approach means the SSE in (4.13) does not have a unique global minimum because transformations (4.15) and (4.17) leave (4.12) unchanged for each model in Table 4.3, although (4.16) does not apply due to bias variation. For the third model in the table, (4.15) does not apply because a may not vary but b may vary. These degeneracies mean there are two fewer implicit parameters for each model, or one fewer for the third model, explaining the deductions from Q in Table 4.3.

Modelling pixel responses in terms of average pixel responses does not facilitate calibration when the bias varies although the assumption in (4.19) remains valid (it depends only on properties of the stochastic error). The reason is because $\bar{y}_i$, known by (4.18), is a linear function of $\bar{l}_i$, given in (4.21), but models in Table 4.3 may not be written as a linear function of $\bar{y}_i$ since l_{ij} is not a linear function of $\bar{l}_i$.

$$\bar{l}_i = \frac{1_j}{N} l_{ij} \tag{4.21}$$

Thus, nonlinear optimisation of c_j and x_i in (4.20), with sensor varying input l_{ij} in Table 4.3, is unavoidable for the calibration of models that have varying bias. However, the raster method, extended to sensor varying input, may be used to estimate offset and gain parameters that minimise the SSE for any choice of c_j and x_i. Thus, the minimum

SSE is a nonlinear function of c_j and x_i alone, which reduces the number of parameters requiring nonlinear optimisation from Q in Table 4.3 (for unknown x_i) to $M + N - 2$, or $M + N - 1$ for the third model, accounting for degeneracies.

Nonlinear optimisation is an iterative process that may be slow when the number of variables is large, which is the case here as N represents the number of pixels in an image sensor. Rather than extend the raster method to sensor varying input, which would be ideal for a class of models (including those that involve columnwise constraints, as in Chapter 5), a specific method may be derived for each model in Table 4.3 to estimate offset and gain parameters given c_j and x_i. However, instead of doing this for each model in Table 4.3, only the fourth model is chosen because it is the most general. From a non-asymptotic point of view, a specific method would be more efficient for a specific problem than the raster method applied to the same problem.

Given c_j and x_i, the minimum of the SSE over a_j and b_j, for the fourth model in Table 4.3, occurs for a'_j and b'_j in (4.22) and (4.23), with intermediates in (4.24)–(4.27). Note that $\bar{y}_j$ in (4.24) and $\bar{l}_j$ in (4.25) are the averages over illuminance of the digital output y_{ij} and the logarithmic input l_{ij} respectively. Additionally, y'_j in (4.26) and l'_j in (4.27) are the correlations over illuminance of y_{ij} and l_{ij} and of l_{ij} and l_{ij} respectively.

$$a'_j = \bar{y}_j - b'_{\underline{j}}\bar{l}_{\underline{j}} \tag{4.22}$$

$$b'_j = y'_{\underline{j}}/l'_{\underline{j}} \tag{4.23}$$

$$\bar{y}_j = \frac{1_i}{M} y_{ij} \tag{4.24}$$

$$\bar{l}_j = \frac{1_i}{M} l_{ij} \tag{4.25}$$

$$y'_j = \frac{1}{M} y_{i\underline{j}} l_{i\underline{j}} - \bar{y}_{\underline{j}}\bar{l}_{\underline{j}} \tag{4.26}$$

$$l'_j = \frac{1}{M} l_{i\underline{j}} l_{i\underline{j}} - \bar{l}_{\underline{j}}{}^2 \tag{4.27}$$

Equations (4.22) and (4.23) imply the minimum SSE, for the fourth model in Table 4.3, is a known function $f(c_j, x_i)$ of only c_j and x_i. Minimisation of $f(c_j, x_i)$ over c_j and x_i may be accomplished from an initial guess c'_j and x'_i using the conjugate gradients method [57]. An initial guess that works well in practise is to assume that all biases are zero, as in (4.28). With this assumption, the assumption in (4.19) and the transformations in (4.15) and (4.16), the logarithm of initial illuminances x'_i becomes a linear function of the average responses $\bar{y}_i$, where $\bar{a}'$ and $\bar{b}'$ are arbitrary coefficients due to the degeneracies. Suitable values for these coefficients are given in (4.30) and (4.31), which normalise $\ln x'_i$ to have zero mean and unit variance over illuminance.

$$c'_j = 0 \tag{4.28}$$

$$\ln x'_i = \frac{\bar{y}_i - 1_i \bar{a}'}{\bar{b}'} \tag{4.29}$$

$$\bar{a}' = \frac{1_i}{M} \bar{y}_i \tag{4.30}$$

$$\bar{b}' = \sqrt{\frac{1_i}{M} (\bar{y}_i - 1_i \bar{a}')^2} \tag{4.31}$$

Because of the transformations in (4.15) and (4.17), the SSE has no unique minimum. These two degeneracies may be eliminated with the constraints in (4.32) and (4.33) on parameter guesses a'_j and c'_j, where $\breve{c}'$ is the minimum of c'_j. Constraint (4.32) is identical to the requirement, for Models 2–4 in Table 4.2, that the offsets a'_j have a zero average. Constraint (4.33) reflects the physical basis of the biases c'_j, due to (4.10), whereby they may not be negative. Both these constraints are satisfied for the initial guesses of c'_j and x'_i, hence a'_j and b'_j, in (4.28) and (4.29). These constraints are not enforced for each guess during nonlinear optimisation but final guesses are transformed with (4.15) and (4.17) so that (4.32) and (4.33) hold. In this manner, parameter estimates $\hat{a}_j$, $\hat{b}_j$, $\hat{c}_j$ and $\hat{x}_i$ are derived that specify a unique minimum of the SSE.

$$\frac{1_j}{N} a'_j = 0 \tag{4.32}$$

$$\breve{c}' = 0 \tag{4.33}$$

There is one more constraint on the parameters that is especially important with a MATLAB implementation, which automatically permits infinite and complex number results. Because the response of a pixel is always finite and never has an imaginary part, the inequality in (4.34) must hold, where $\breve{x}'$ is the minimum of x'_i (this single nonlinear inequality may also be written as MN linear inequalities).

$$\breve{c}' + \breve{x}' > 0 \tag{4.34}$$

The simplest way to satisfy this inequality is to modify the SSE calculation to return a high value (∞ in MATLAB) when (4.34) is not satisfied and to ensure that the line minimiser used by the conjugate gradients method copes with such extreme values. Brent's algorithm for line minimisation was used with the NETLAB implementation of the conjugate gradients method, which succeeded in estimating the unique bias and illuminance parameters $\hat{c}_j$ and $\hat{x}_i$ that minimised the SSE subject to the constraints.

Estimated parameters $\hat{c}_j$ and $\hat{x}_i$ may be used to estimate the error variance $\hat{\sigma}_\epsilon^2$, as in (4.35). In this formula, the numerator is the minimum SSE and the denominator is the degrees of freedom, which is the number of actual responses MN minus the number of implicit parameters Q in (4.36) that are fitted to those responses.

$$\hat{\sigma}_\epsilon^2 = \frac{f(\hat{c}_j, \hat{x}_i)}{MN - Q} \tag{4.35}$$

$$Q = M + 3N - 2 \tag{4.36}$$

Following (4.22) and (4.23), estimates $\hat{a}_j$ and $\hat{b}_j$ may be derived from estimates $\hat{c}_j$ and $\hat{x}_i$. By ignoring the stochasticity of $\hat{c}_j$ and $\hat{x}_i$, the estimated inter variances of $\hat{a}_j$ and $\hat{b}_j$ are derived in (4.37) and (4.38). These results, therefore, are expected to underestimate the actual inter variances. The inter variances of $\hat{c}_j$ and $\hat{x}_i$ are not estimated for simplicity, as they involve the hessian of the SSE with respect to the parameters.

$$\hat{\mathcal{V}}\{\hat{a}_j\} = \hat{\sigma}_\epsilon^2 \frac{1}{M} (1_j + \bar{l}_{\underline{j}}{}^2 / l'_{\underline{j}}) \tag{4.37}$$

$$\hat{\mathcal{V}}\{\hat{b}_j\} = \hat{\sigma}_\epsilon^2 \frac{1}{M} l'_j{}^{-1} \tag{4.38}$$

Table 4.4: Estimated response $\hat{y}_j$ of the j^{th} logarithmic pixel to illuminance x_j for the nil, single and double variation models, where $l_j = \ln(1_j c + x_j)$, and for the triple variation model, where $l_j = \ln(c_j + x_j)$. Spatially varying parameters a_j, b_j and c_j are unknown linear functions of previously estimated parameters $\hat{a}_j$, $\hat{b}_j$ and $\hat{c}_j$.

Variation	$\hat{y}_j$	a_j	b_j	c_j
Nil	$1_j a + b l_j$			
Single	$a_j + b l_j$	$\hat{a}_j + 1_j \bar{a}$		
Double	$a_j + b_{\underline{j}} l_{\underline{j}}$	$\hat{a}_j + \hat{b}_j \bar{a}$	$\hat{b}_j \bar{b}$	
Triple	$a_j + b_{\underline{j}} l_{\underline{j}}$	$\hat{a}_j + \hat{b}_j \bar{a}/\bar{b}$	$\hat{b}_j$	$e^{-\bar{a}/\bar{b}} \hat{c}_j + 1_j \breve{c}$

4.4 Correction

For simplicity, Models 1, 2 and 3/4 of Table 4.2 and Model 4 of Table 4.3 are called the *nil*, *single*, *double* and *triple variation* models respectively. Once these models are calibrated, they may be used to correct FPN present in an image y_j of an arbitrary scene x_j. Image sensors that obey the nil variation model do not really require FPN correction since no parameter varies from pixel to pixel except for the unpredictable stochastic error. The nil variation model, therefore, helps to explain how FPN correction is not about deriving an estimate $\hat{x}_j$ of the scene x_j. Rather, the estimation of a monotonic function of x_j suffices, so long as parameters of the function do not vary from pixel to pixel. Indeed, because of the degeneracies in (4.15)–(4.17), only functions of x_j are determinate with the nil, single, double and triple variation models.

Following Section 4.3, Table 4.4 gives the estimated responses $\hat{y}_j$ of a logarithmic image sensor to a scene x_j, which differ from actual responses y_j by lacking stochastic errors ϵ_j, for the nil, single, double and triple variation models. Because of the degeneracies in (4.15)–(4.17), the calibration described in Section 4.3 does not give unbiased estimates for spatially varying parameters a_j, b_j and c_j and does not estimate spatially constant parameters a and b. As given in Table 4.4, the varying parameters are linear functions of the estimated parameters $\hat{a}_j$, $\hat{b}_j$ and $\hat{c}_j$ with unknown means $\bar{a}$ and $\bar{b}$ and minimum $\breve{c}$, which arise from the assumptions and normalisations of the calibration. Note that $\bar{b}$ may be estimated for triple variation since $\hat{b}_j$ is an unbiased estimate of b_j.

Unknown parameters in Table 4.4 prevent an estimation of the scene x_j without further measurement and calibration, an approach taken in Chapter 7. However, x_j does not need estimation to correct FPN since each model $\hat{y}_j$ in Table 4.4 may be rewritten as a linear function with known coefficients of a model y_{0j} in Table 4.5, where y_{0j} is a logarithmic function of x_j with no offset or gain variation . For nil variation, $\hat{y}_j$ and y_{0j} are the same. For triple variation, y_{0j} includes bias variation but may be rewritten as a logarithmic function with known parameters of a model x_{0j} in Table 4.5, where x_{0j} is a linear function of x_j with no parameter variation.

Thus, FPN may be corrected for any image y_j of a scene x_j by estimating y_{0j} and possibly x_{0j}, according to the type of variation in Table 4.5. This estimation may be performed by minimising the SSE in (4.39) between the actual and estimated responses

Table 4.5: The estimated response $\hat{y}_j$ of the j^{th} logarithmic pixel may be written as a known function of an ideal response y_{0j} for the nil, single and double variation models, where $l_j = \ln(1_j c + x_j)$, or x_{0j} for the triple variation model. The ideal response is an unknown monotonic function of illuminance x_j with no parameter variation.

Variation	$\hat{y}_j$	y_{0j}	x_{0j}
Nil	y_{0j}	$1_j a + b l_j$	
Single	$\hat{a}_j + y_{0j}$	$1_j \bar{a} + b l_j$	
Double	$\hat{a}_j + \hat{b}_{\underline{j}} y_{0\underline{j}}$	$1_j \bar{a} + \bar{b} l_j$	
Triple	$\hat{a}_j + \hat{b}_{\underline{j}} y_{0\underline{j}}$	$\ln(\hat{c}_j + x_{0j})$	$e^{\bar{a}/\bar{b}}(1_j \check{c} + x_j)$

Table 4.6: Estimated ideal response $\hat{y}_{0j}$ or $\hat{x}_{0j}$, as appropriate, of the j^{th} logarithmic pixel to illuminance x_j for the nil, single, double and triple variation models. These estimates use the actual response y_j to illuminance x_j and previously estimated parameters $\hat{a}_j$, $\hat{b}_j$ and $\hat{c}_j$, as appropriate, to invert the models in Table 4.5.

Variation	$\hat{y}_{0j}$	$\hat{x}_{0j}$
Nil	y_j	
Single	$y_j - \hat{a}_j$	
Double	$\hat{b}_{\underline{j}}^{-1}(y_{\underline{j}} - \hat{a}_{\underline{j}})$	
Triple	$\hat{b}_{\underline{j}}^{-1}(y_{\underline{j}} - \hat{a}_{\underline{j}})$	$\exp(\hat{y}_{0j}) - \hat{c}_j$

over the parameters y_{0j} or x_{0j}, as appropriate, giving estimates $\hat{y}_{0j}$ or $\hat{x}_{0j}$. Such a minimisation has a unique analytic solution for each type of variation, given in Table 4.6, which amounts to inversion of the models in Table 4.5. However, there are no degrees of freedom to estimate the error or parameter variances.

$$SSE = 1_j (y_j - \hat{y}_j)^2 \tag{4.39}$$

Note that correction of FPN due to nil, single and double variation takes a linear transformation of the image y_j, giving a nonlinear representation of the scene $\hat{y}_{0j}$. Correction of FPN due to triple variation takes a nonlinear transformation of the image y_j, giving a linear representation of the scene $\hat{x}_{0j}$. The difference arises because the former models assume constant bias whereas the latter model assumes varying bias.

4.5 Simulation

The nil, single, double and triple variation models were calibrated using simulation data for a $0.35\mu\text{m}$ 3.3V AMS process, as described in Chapter 1. Since the simulator considers only electronic devices, optical processes were omitted and the stimulus x of a pixel is simply the photocurrent, as in (4.40). Furthermore, the simulation neither

Table 4.7: The residual error $\hat{\sigma}_\epsilon$, averages $\hat{a}$, $\hat{b}$ and $\hat{c}$ of estimated parameters $\hat{a}_j$, $\hat{b}_j$ and $\hat{c}_j$ and parameter uncertainties $\hat{\sigma}_{\hat{a}}$ and $\hat{\sigma}_{\hat{b}}$ for calibration of the nil, single, double and triple variation models $\hat{y}_{ij}$, where $\hat{l}_{ij} = \ln(1_i\hat{c}_j + 1_j\hat{x}_i)$, with simulation data y_{ij}.

Variation	$\hat{y}_{ij}$	$\hat{\sigma}_\epsilon$	$\hat{a} \pm \hat{\sigma}_{\hat{a}}$	$\hat{b} \pm \hat{\sigma}_{\hat{b}}$	$\hat{c}$
Nil	$1_j\bar{y}_i$	20			
Single	$1_i\hat{a}_j + 1_j\bar{y}_i$	.44	$0.0 \pm .12$		
Double	$1_i\hat{a}_j + \hat{b}_j\bar{y}_i$	.29	0.0 ± 2.4	$1.0\frac{\mathrm{mV}}{\mathrm{mV}} \pm .11\%$	
Triple	$1_i\hat{a}_j + \hat{b}_j\hat{l}_{ij}$	.28	0.0 ± 2.4	$-75\mathrm{mV} \pm .10\%$	.43
In	V	mV	$\mathrm{mV} \pm \mathrm{mV}$		fA

included an ADC nor considered temporal noise, which means the response y of a pixel equals the input voltage of the ADC plus the error in the underlying device models, as in (4.41). These changes imply minor changes to the physical models of the offset, gain, bias and error in (4.8)–(4.11) but do not change the abstract model of the response in Section 4.2 or the calibration and correction described in Sections 4.3 and 4.4.

$$x = I_P \tag{4.40}$$

$$y = V_{ADC} + \epsilon_M \tag{4.41}$$

The photocurrent was simulated by placing an ideal current source in parallel with the pixel diode. Using the DC and Monte Carlo analyses of the simulator, the photocurrent was varied in half decade steps from $1\mathrm{pA}$ to $1\mu\mathrm{A}$ (i.e. $M = 13$) and the circuit in Figure 4.1 was simulated 100 times with randomly varying device parameters (i.e. $N = 100$) according to a mismatch model supplied by AMS. Parameters of T7 were not permitted to vary with iteration as the transistor is common to all pixels. No provision was made to vary parameters of transistors T4–6 every N_1 iterations, while varying parameters of transistors T1–3 every iteration, which would simulate a columnwise variation of some circuit parameters in an array of $N = N_1 \times N_2$ pixels. Instead, parameters of T1–6 were varied every iteration, simulating a random selection of 100 pixels taken from different columns of a larger array. AMS did not provide a mismatch model for diodes so parameters of the photodiode were constant with iteration.

Table 4.7 gives the residual error for calibration of the nil, single, double and triple variation models with simulated responses y_{ij} to uniform photocurrent x_i. Nil variation has the worst result by far, which shows that FPN may not be ignored in logarithmic image sensors. Single variation is much better than nil variation and double variation is almost twice as good as single variation. Thus, gain variation should not be ignored in logarithmic image sensors. The residual errors for double and triple variation are similar, which warrants a comparison of parameter uncertainties.

Table 4.7 gives the average value of estimated parameters alongside the parameter uncertainties. No uncertainty is given for the bias as it is a nonlinear parameter. Parameter uncertainties are constant from pixel to pixel with nil, single and double variation, as in the raster problem of Chapter 3. Parameter uncertainties vary from pixel to pixel

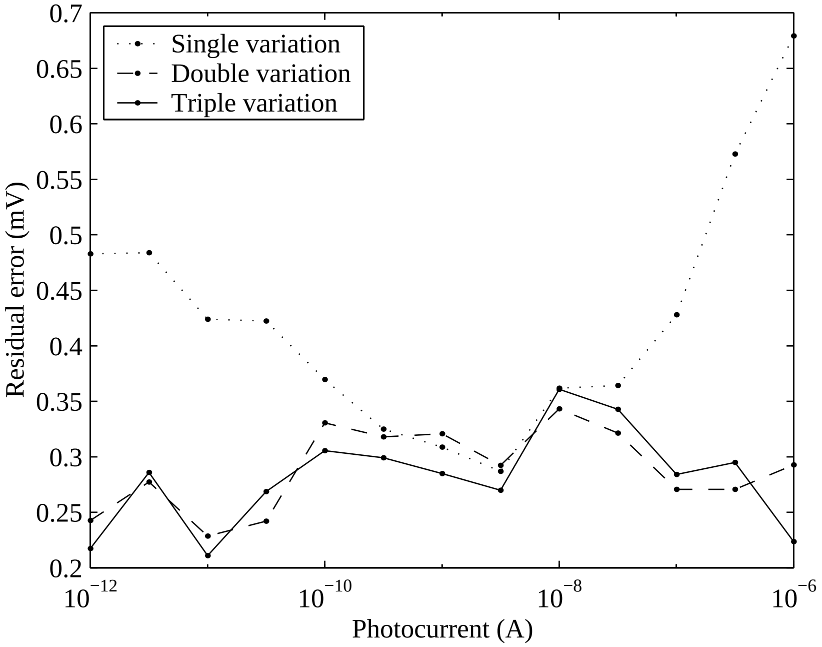


Figure 4.2: The residual error $\hat{\sigma}_{\epsilon_i}$ versus photocurrent x_i for calibration of the single, double and triple variation models with simulated data.

with triple variation, as in (4.37) and (4.38), and so averages are reported. The parameter uncertainties with double and triple variation are comparable. However, figures given for triple variation underestimate uncertainty because the stochasticity of nonlinear estimates $\hat{c}_j$ and $\hat{x}_i$ were not considered. Thus, double variation is the best model of FPN for the simulation, which is logical because the simulator does not consider bias variation. The bias equals the photodiode leakage current I_S, as there are no optical effects, but I_S does not vary in a Monte Carlo simulation of the AMS process.

Note that the average offset is zero in Table 4.7 with single, double and triple variation, which occurs because of degeneracies and normalisations of the calibrations. The average gain is one for double variation because of another degeneracy. However, there are no degeneracies on the gain in triple variation, which means that estimates are unbiased. Lastly, the small magnitude of the average bias reported for triple variation does not mean that the leakage current is insignificant. Estimated biases are an unknown linear function of the true biases due to the degeneracies of the calibration.

Figure 4.2 plots the residual error versus photocurrent for the single, double and triple variation models. This value is the square root of the estimated error variance $\hat{\sigma}^2_{\epsilon_i}$ in (4.42) for each photocurrent, which equals the SSE between actual and estimated responses at one photocurrent divided by the corresponding degrees of freedom (the number of pixel responses minus the fractional number of fitted parameters).

$$\hat{\sigma}^2_{\epsilon_i} = \frac{1_j(y_{ij} - \hat{y}_{ij})^2}{N - Q/M} \tag{4.42}$$

The residual error should be independent of photocurrent in Figure 4.2 as it measures the stochastic error. This is not the case with single variation as the residual error is roughly parabolic. However, the residual error is relatively flat with double and triple variation and the two models do not differ by much. At high photocurrents, double variation worsens and triple variation improves, which suggests a small variation of response due to the onset of saturation in the subthreshold load transistor (i.e. T1 in Figure 4.1). Though not explicitly considered in Sections 4.2 and 4.3, triple variation accommodates some of this variation. The residual error versus photocurrent for calibration of the nil variation model is omitted in the figure for the sake of clarity.

4.6 Experiments

Experiments were done using a 512×512 pixel (i.e. $N = 512^2$) Fuga 15RGB logarithmic image sensor, which was built in a $0.7\mu\mathrm{m}$ $5\mathrm{V}$ process [32]. The camera, which was interfaced to a PC, had an 8-bit ADC with a programmable offset voltage, as described in Chapter 1. By capturing several frames with different offset settings, the resolution was increased to 10 bits in software. Although it is really a colour camera, the Fuga 15RGB is treated here as a monochromatic camera, which does not prejudice results. Chapter 7 considers colour in logarithmic image sensors.

Overhead fluorescent lights provided the illumination for the experiments reported in this section and they did not permit a variation of intensity. Instead, four neutral density filters were placed in sequence over the camera lens to attenuate the illuminance reaching the focal plane in nominal half decade steps for a total variation of two decades. The actual attenuations were measured with a light meter to be 0, 13, 21, 32 and $40\mathrm{dB}$, counting the case of zero attenuation with no filter.

4.6.1 Calibration

The first experiment used five images (i.e. $M = 5$) of a white sheet of paper under uniform illumination, where the measured intensity was varied with neutral density filters. The nil, single, double and triple variation models of responses y_{ij} to illuminances x_i were calibrated according to Section 4.3. Table 4.8 reports the residual error, average values of estimated parameters and parameter uncertainties of each calibrated model (average uncertainties are given for triple variation). Nil variation has the worst residual error by far. The residual error of single variation is almost four times better than that of nil variation and the residual error of double variation is over two times better than that of single variation. These results agree with those of the simulation. Unlike the simulation, the residual error of triple variation is significantly better than that of double variation. Therefore, triple variation is the best model of FPN for the experiment. Note that the average of estimated offsets with single, double and triple variation is zero and the average of estimated gains with double variation is one, as before.

Table 4.8: The residual error $\hat{\sigma}_\epsilon$, averages $\hat{a}$, $\hat{b}$ and $\hat{c}$ of estimated parameters $\hat{a}_j$, $\hat{b}_j$ and $\hat{c}_j$ and parameter uncertainties $\hat{\sigma}_{\hat{a}}$ and $\hat{\sigma}_{\hat{b}}$ for calibration of the nil, single, double and triple variation models $\hat{y}_{ij}$, where $\hat{l}_{ij} = \ln(1_i\hat{c}_j + 1_j\hat{x}_i)$, with experimental data y_{ij}.

Variation	$\hat{y}_{ij}$	$\hat{\sigma}_\epsilon$	$\hat{a} \pm \hat{\sigma}_{\hat{a}}$	$\hat{b} \pm \hat{\sigma}_{\hat{b}}$	$\hat{c}$
Nil	$1_j\bar{y}_i$	20			
Single	$1_i\hat{a}_j + 1_j\bar{y}_i$	5.2	0.0 ± 2.3		
Double	$1_i\hat{a}_j + \hat{b}_j\bar{y}_i$	2.3	0.0 ± 4.7	$1.0\frac{\mathrm{LSB}}{\mathrm{LSB}} \pm 3.1\%$	
Triple	$1_i\hat{a}_j + \hat{b}_j\hat{l}_{ij}$	.68	0.0 ± 1.4	$66\mathrm{LSB} \pm .92\%$	4.8
In	LSB	LSB	LSB ± LSB		lux

Figure 4.3 plots the residual error versus photocurrent for the single, double and triple variation models (the nil variation model is omitted as it has a large error). The single variation model has a minimum error of 2.6LSB in the middle of the domain, with error rising on each side to 6.9 and 7.4LSB. The double variation model has a maximum error of 3.0LSB in the middle, flanked by two minima of 1.1 and 1.6LSB and rising to 2.6LSB at the sides. In contrast, the triple variation model has a relatively flat error of less than $.84$LSB. These results suggest that the triple variation model describes pixel responses very well over the two decade range of illuminance and may be extrapolated to a high dynamic range with little degradation in performance.

The shape of each plot in Figure 4.3 may be readily explained by considering the residual error versus illuminance of selected pixels. Figure 4.4 plots the actual and estimated response of two pixels, for the single, double and triple variation models, versus the average response of all pixels. While the single variation model fits the bottom response well, it fits the top response poorly because of a different response slope. Instead, the estimated response intersects the trend of the actual response in the mid-range of illuminance, minimising the SSE, which explains the v-shaped curve in Figure 4.3. The double variation model matches the response slopes of both pixels but intersects each response trend twice as the actual response follows a curved path (especially the top one), which explains the w-shaped curve in Figure 4.3. For the top response, note that single variation near its intersection is better than double variation, which explains the small region of Figure 4.3 where the former outperforms the latter. The triple variation model has no problem following the curved responses of both pixels and the residual error hardly depends on illuminance, as in Figure 4.3.

4.6.2 Correction

Five images were taken of an office scene illuminated by overhead fluorescent lights, using neutral density filters to simulate intensity variation of the illuminant. Figure 4.5 displays the images after FPN correction, for the nil, single, double and triple variation models, using parameters estimated in the calibration described previously. The histogram of each displayed image has been equalised to facilitate comparison, since the triple variation correction gives a linear representation of the scene whereas the other

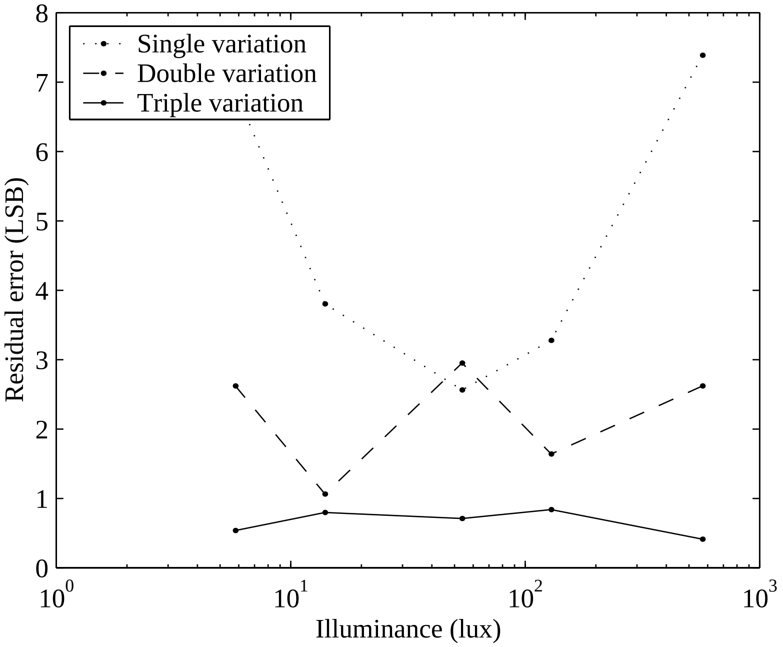


Figure 4.3: The residual error $\hat{\sigma}_{\epsilon_i}$ versus illuminance x_i for calibration of the single, double and triple variation models with experimental data.

corrections give a logarithmic representation of the scene. Table 4.9 lists the illuminances of ten features in the scene for the five attenuations realised with the neutral density filters. The inter-scene dynamic range of any feature is thus 40dB and the intra-scene dynamic range across features is 29dB, for a total of 69dB.

Because the scenes are the same going from top to bottom in Figure 4.5 except getting darker, an ideal logarithmic sensor, apart from lacking FPN, would give identical images with histogram equalisation. By this standard, nil variation gives poor results for two reasons: corrected images have residual FPN and vary with illumination. Single variation reduces FPN substantially in bright lighting but correction and contrast degrade in dim lighting. Double variation performs better than single variation, degrading slowly in dim lighting. Nonetheless, triple variation gives the best results, having little residual FPN and maintaining contrast over the 69dB range of illuminance. Performance does degrade in dim lighting but, as described in Chapter 7, this occurs mainly because of stochastic error and bias magnitude rather than parameter variation.

4.7 Conclusion

This chapter has modelled the response y of a logarithmic CMOS pixel to illuminance x. The model has numerous physical parameters but may be abstracted by a logarithmic

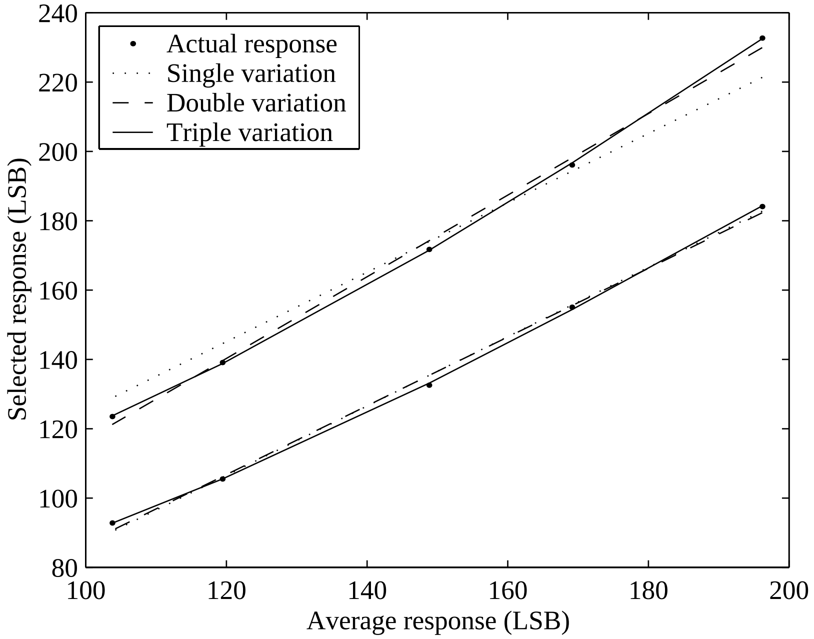


Figure 4.4: The actual and estimated response y_{ij} and $\hat{y}_{ij}$ of two pixels versus the average response $\bar{y}_i$ of all pixels for the single, double and triple variation models.

function $y = a + b\ln(c + x) + \epsilon$ with only three parameters—an offset a, gain b and bias c—and a stochastic error ϵ. A spatial variation of some or all parameters causes fixed pattern noise (FPN). Although it is well known that threshold voltage variation, in the pixel and column source followers, leads to FPN, the model shows other contributions to offset variation and highlights possible sources of gain and bias variation. Bias variation makes FPN calibration and correction a nonlinear problem.

Methods to calibrate various models of FPN, by estimating parameters using images of uniform illuminance, were derived. When the bias is constant from pixel to pixel, for the nil, single and double variation models, calibration may be accomplished with the raster method. When the bias may vary from pixel to pixel, for the triple variation model, multilinear regression may be used to reduce the number of variables substantially but nonlinear optimisation is required to estimate the remaining variables. Calibrated models may be used to correct FPN in images of arbitrary scenes. FPN correction involves the estimation of a monotonic function of the scene illuminance that lacks parameter variation, entailing a linear transformation of images for constant bias models and a nonlinear transformation of images for varying bias models.

Pixel responses to photocurrent or illuminance, taken from simulation and experiment respectively, were used to validate the methods of calibration and correction and to compare the models of FPN. Double variation proved to be the best model of FPN for a simulated image sensor. Although the residual errors of double and triple varia-

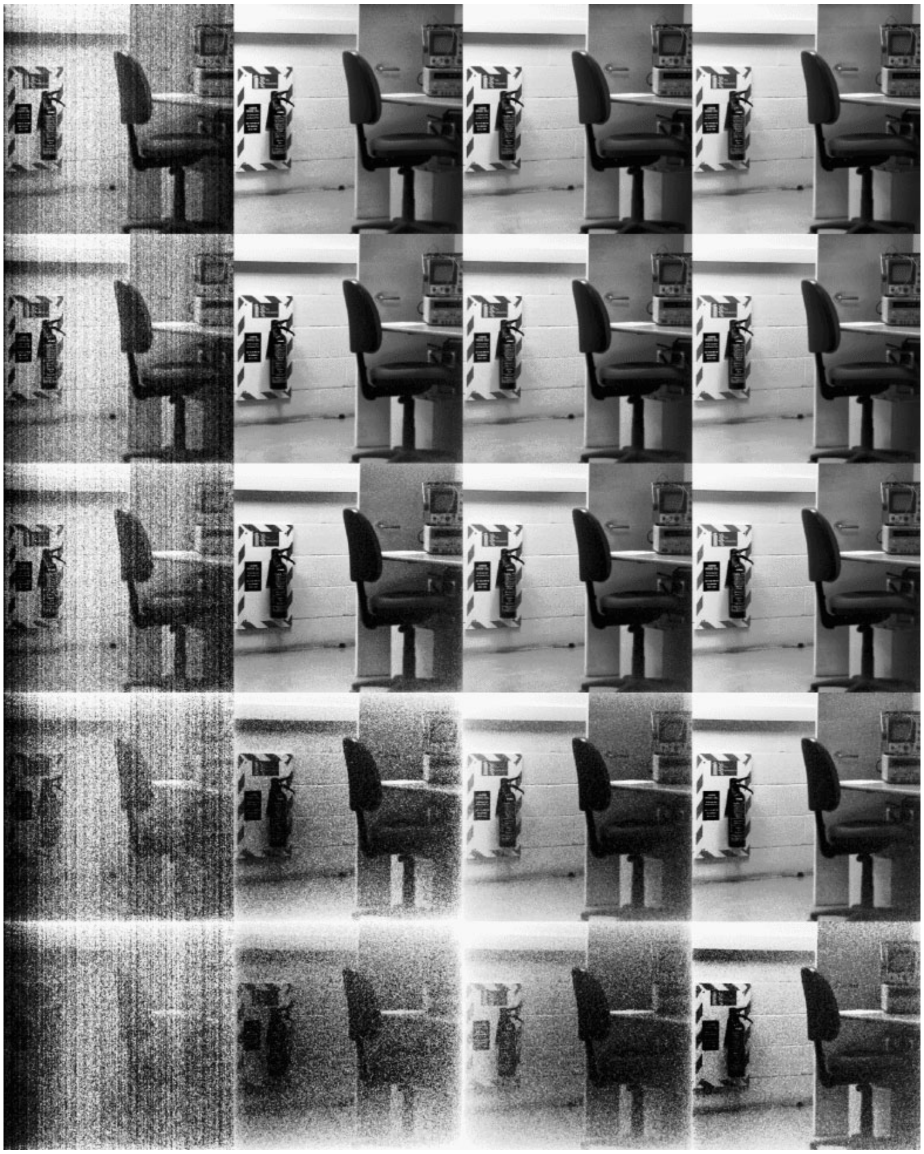


Figure 4.5: FPN correction of Fuga 15RGB images for the nil, single, double and triple variation models (left to right). The images, displayed in greyscale with histogram equalisation, are of one scene with illuminances attenuated by 0, 13, 21, 32 and $40\mathrm{dB}$ (top to bottom) using neutral density filters over the camera lens.

Table 4.9: The measured and calculated illuminance of scene features, in the images of Figure 4.5, for attenuations of 0, 13, 21, 32 and $40\mathrm{dB}$, due to neutral density filters.

Scene feature	Illuminance (lux)				
White bar	380	85	36	9.2	3.8
Desk paper	270	61	25	6.6	2.7
Wall, middle	180	41	17	4.5	1.9
Floor area	150	34	14	3.7	1.5
Door, top	94	21	8.8	2.3	.95
Supply knobs	58	13	5.5	1.4	.59
Extinguisher	41	9.3	3.9	1.0	.42
Scope screen	31	7.1	2.9	.77	.32
Chair, back	22	4.9	2.0	.53	.22
Chair, base	13	2.9	1.2	.32	.13
Attenuation (dB)	0	13	21	32	40

tion were comparable, the parameter uncertainties with the latter were higher. Triple variation proved to be the best model of FPN for the Fuga 15RGB, with a residual error significantly better than that of double variation. The difference between the simulation and experiment occurs because the simulated process did not model the leakage current mismatch responsible for bias variation. Good models of FPN had residual errors that were relatively independent of photocurrent or illuminance, over six and two decades respectively, which suggests they may be extrapolated with good accuracy.

Whether triple variation proves to be a practical model for the calibration and correction of FPN in logarithmic CMOS image sensors remains to be seen. Nonetheless, while analogue techniques to correct pixel and column offset variation, such as double sampling and delta difference sampling, are useful to reduce FPN, they are inadequate to achieve a maximum of perceptual accuracy over a high dynamic range. The same may be said for digital calibration and correction of offset variation or even offset and gain variation. Any linear calibration and correction is a reasonable approximation over only a small region of a nonlinear distortion. The nonlinear effect of bias variation on FPN requires more robust circuits or nonlinear calibration and correction.

Chapter 5

Transient response

5.1 Introduction

The previous chapter dealt with the steady state response of logarithmic CMOS image sensors, showing how pixel-to-pixel variations of device parameters leads to fixed pattern noise (FPN). What this analysis neglected is that voltage changes need time for rising and falling. It is natural to expect that if insufficient time is provided then noise would also appear. Consequentially, this chapter examines the transient response of logarithmic sensors, seeking especially to determine if noise caused by improper timing is purely random or whether it displays a fixed pattern.

A transient analysis of logarithmic sensors may include the response of the photodiode and the subthreshold load transistor but this is unnecessary for three reasons. Firstly, the light falling on the photodiode represents light focused from real world scenes. Such light is normally modulated slowly, except when fast motion is involved. Secondly, a response bandwidth as low as 24–30Hz satisfies the motion sensing capability of the human eye and a bandwidth of 48–75Hz accomodates flicker sensitivity as well [12]. Thirdly, because the logarithmic pixel operates continuously (unlike linear integrating pixels), it provides a very high bandwidth for normal lighting conditions. Studies with pulsed lasers have shown a 3dB bandwidth of about 100kHz [26]. For these reasons, it can be safely assumed that the transient response of the photodiode and load transistor is sufficiently quick to approximate the steady state response for the vast majority of applications.

The transient response of the readout circuit, however, is a crucial factor for the performance of the sensor. In an array of $N_1 \times N_2$ pixels, the pixel responses must be read serially for a full frame image unless more than one ADC is available and unless there is space on the die to allow independent pixel addressing and buffering circuits. Serial readout at a frame rate of R frames per second (fps) requires a pixel scanning rate of N_1N_2R, which would be on the order of 10–100MHz for megapixel sensors operating at video rates. Furthermore, given that switching the ADC from one pixel to the next is necessarily a discontinous process, the transient behaviour of the readout circuit may certainly be a dominant factor of noise in resulting images.

Pixels are normally raster scanned in image sensors, which means that responses are read left-to-right and top-to-bottom across the array, the same way a page of text is read. A row is selected and the responses of all pixels in that row are copied into N_2 parallel buffers, one for each column. This is the first stage of readout. Each buffer is then selected in sequence and copied to another buffer that serves the ADC. This is the second stage of readout. Therefore, the first stage must switch N_1 times, i.e. as many times as there are rows, during the scanning of each frame. The second stage must switch $N_1 \times N_2$ times, i.e. as many times as there are pixels, for each frame read. Thus, the transient response of the second stage needs to be N_2 times, as many times as there are columns, faster than the first stage.

Although this chapter examines the readout circuitry of logarithmic sensors, much of what is said also applies to linear CMOS sensors (but not to CCD sensors as their readout method is much different). Section 5.2 models the transient response of the readout circuit. Section 5.3 describes how insufficient settling time causes FPN and how some of this effect may be accommodated by previous methods of calibration. Section 5.4 uses simulation results to validate the ideal description of the transient response. Section 5.5 uses experimental results of a Fuga 15RGB sensor to demonstrate, with some deviation from the ideal case, the modelled and simulated effects.

5.2 Modelling

Figure 5.1 shows the circuits comprising one column of a typical logarithmic CMOS image sensor, following Chapter 4. Each column consists of N_1 pixels connected to a common bus via N_1 source followers that share a current source, i.e. transistor $\mathrm{T}4$ in the figure, but have separate amplifiers, e.g. transistor $\mathrm{T}2_{j_1}$ for pixel j_1. As each pixel circuit also has a switch, e.g. transistor $\mathrm{T}3_{j_1}$ for pixel j_1, the source follower may be operated in sequence for each pixel by closing the switch for that pixel and opening the switches of all other pixels, as shown in the figure for pixel j_1. In this manner, the source follower output or the column bus voltage, denoted $V_G^{\mathrm{T}5}$ as in Chapter 4, follows the source follower input or the pixel drive voltage, denoted $V_G^{\mathrm{T}2_{j_1}}$.

Chapter 4 presents only a steady state analysis of the above circuit. To perform a transient analysis, assume that no more than one switch is on at a time and that the switches behave in an ideal fashion (except for their capacitance, as described below). An expression is sought for the column bus voltage $v_G^{\mathrm{T}5}(t)$, as a function of time t, for reading pixel j_1 given a pixel drive voltage $V_G^{\mathrm{T}2_{j_1}}$, which does not vary with time, and an initial voltage $v_G^{\mathrm{T}5}(t_0)$ of the column bus, when switch $\mathrm{T}3_{j_1}$ is closed at time t_0.

When switch $\mathrm{T}3_{j_1}$ is closed in the circuit of Figure 5.1, the column bus will charge or discharge towards the steady state result of Chapter 4, i.e. (4.3). The rate of charging or discharging depends on the load impedance seen by the source follower at this node. Such a load includes the distributed resistances and capacitances of the long metal line on the die connecting all the pixels in the column (as well as the gate capacitance of transistor $\mathrm{T}5$ in Figure 4.1). However, these factors are insignificant compared to the source-bulk capacitances of the N_1 switches connected to the node, especially as N_1 is on the order of 1000 for megapixel sensors.

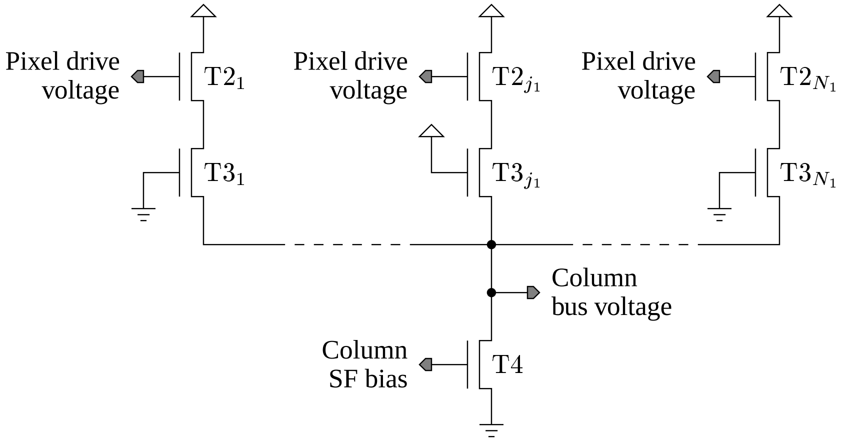


Figure 5.1: The first stage readout of a typical CMOS image sensor consists of N_1 amplifier and switch transistors T2 and T3, one pair in each pixel, and a current source T4, one in each column of pixels. When switch $\mathrm{T3}_{j_1}$ is on, where $1 \leq j_1 \leq N_1$, all other switches are off and $\mathrm{T2}_{j_1}$ forms a source follower (SF) with T4. The second stage readout is similar but uses PMOS instead of NMOS transistors, as in Figure 4.1.

Based on the discussion given above, Figure 5.2 presents a simplification of the circuit in Figure 5.1 for the purpose of transient analysis. The load capacitance C is approximated in (5.1) by taking the source-bulk capacitance C_{SB}^{T3} in (5.2), of a switch T3, in parallel N_1 times. The source-bulk capacitance approximates the depletion capacitance of the reverse biased pn-junction between the source diffusion and bulk substrate of T3, which depends on the area A_D^{T3} and perimeter P_D^{T3} of the diffusion (not the same as the area and perimeter of the transistor) and various process parameters CJ, $CJSW$, MJ, $MJSW$ and PB [58, 44]. Note that C_{SB}^{T3} in (5.2) depends on the source-bulk voltage V_{SB}^{T3} but a worst case capacitance may be obtained by setting this voltage equal to zero. There are many other parasitic capacitances that contribute to the load and source-bulk capacitances but they prove to be small with a detailed simulation.

$$C \approx N_1 C_{SB}^{\mathrm{T3}} \tag{5.1}$$

$$C_{SB}^{\mathrm{T3}} \approx \frac{A_D^{\mathrm{T3}}\, CJ}{\left(1 + \frac{V_{SB}^{\mathrm{T3}}}{PB}\right)^{MJ}} + \frac{P_D^{\mathrm{T3}}\, CJSW}{\left(1 + \frac{V_{SB}^{\mathrm{T3}}}{PB}\right)^{MJSW}} \tag{5.2}$$

Because the drain of T2 in Figure 5.2 is connected to V_{DD} (and the pixel drive voltage is never more than V_{DD}), T2 is always in saturation. On the other hand, T4 is in saturation only if the column bus voltage is sufficiently high so that $V_{GS}^{\mathrm{T4}} - V_T^{\mathrm{T4}} \leq V_{DS}^{\mathrm{T4}}$. Normally, the circuit is designed and the column bias V_G^{T4} is chosen so that T4 is in saturation, for the expected range of the pixel drive voltage, with the switch closed. However, if the column bus is permitted to discharge to ground, as would be the case when all switches in Figure 5.1 are open, then T4 will be in the linear region for any

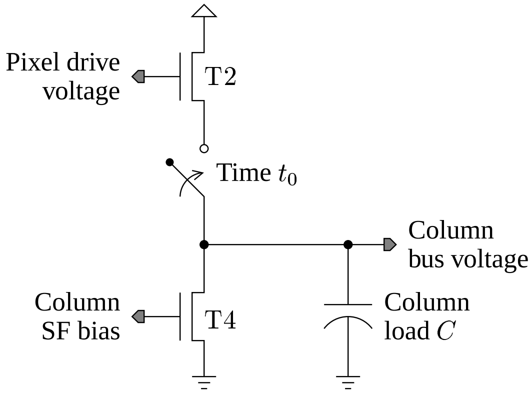


Figure 5.2: The transient response of the first stage readout for a pixel drive voltage may be derived by analysing a two transistor source follower (SF), formed by $\mathrm{T2}$ and $\mathrm{T4}$ when only one switch is turned on in the circuit of Figure 5.1, with a load capacitance C. When the switch is turned on at time t_0, the column bus may have a nonzero voltage due to readout of the previous pixel in the column or a zero voltage due to discharge.

column bias greater than the threshold voltage. In this region, the transistor behaves like a resistor between the drain and source, with a resistance determined by the gate voltage. As this resistance would be small and in parallel with the load capacitance of Figure 5.2, the overall load impedence would be small. Thus, when the switch is closed, $\mathrm{T2}$ will conduct a current to charge this impedence very quickly. Therefore, the column bus voltage will quickly reach a level where $\mathrm{T4}$ enters saturation.

Although it is possible to solve for the transient response analytically for the case where $\mathrm{T4}$ is in the linear region, using Level 1 models and neglecting the output resistance of $\mathrm{T2}$, little of the transient response is affected by assuming $\mathrm{T4}$ is always in saturation regardless of the column bus voltage. Proceeding with this assumption, using Level 1 models and neglecting the output resistance of $\mathrm{T2}$ or $\mathrm{T4}$ in saturation, a differential equation governing the transient response of the circuit is given in (5.3). This differential equation may be solved by making the hyperbolic trigonometric substitution in (5.4) with associated derivative in (5.5).

$$K^{\mathrm{T2}}(V_G^{\mathrm{T2}} - v_G^{\mathrm{T5}}(t) - V_T^{\mathrm{T2}})^2 = K^{\mathrm{T4}}(V_{GS}^{\mathrm{T4}} - V_T^{\mathrm{T4}})^2 + C\frac{\mathrm{d}v_G^{\mathrm{T5}}}{\mathrm{d}t} \tag{5.3}$$

$$v_G^{\mathrm{T5}}(t) = V_G^{\mathrm{T2}} - V_T^{\mathrm{T2}} + \sqrt{\frac{K^{\mathrm{T4}}}{K^{\mathrm{T2}}}}(V_{GS}^{\mathrm{T4}} - V_T^{\mathrm{T4}})\tanh\theta(t) \tag{5.4}$$

$$\frac{\mathrm{d}v_G^{\mathrm{T5}}}{\mathrm{d}t} = \sqrt{\frac{K^{\mathrm{T4}}}{K^{\mathrm{T2}}}}(V_{GS}^{\mathrm{T4}} - V_T^{\mathrm{T4}})\operatorname{sech}^2\theta(t)\frac{\mathrm{d}\theta}{\mathrm{d}t} \tag{5.5}$$

Applying the substitutions in (5.4) and (5.5) and the identity in (5.6), the differential equation in (5.3) may be simplified as in (5.7). Equation (5.7) may be solved easily by integration, giving (5.8) with an arbitrary constant θ_0.

$$\operatorname{sech}^2\theta(t) = 1 - \tanh^2\theta(t) \tag{5.6}$$

$$\frac{\mathrm{d}\theta}{\mathrm{d}t} = -\frac{\sqrt{K^{\mathrm{T2}}K^{\mathrm{T4}}}}{C}(V_{GS}^{\mathrm{T4}} - V_T^{\mathrm{T4}}) \tag{5.7}$$

$$\theta(t) = -\frac{\sqrt{K^{\mathrm{T2}}K^{\mathrm{T4}}}}{C}(V_{GS}^{\mathrm{T4}} - V_T^{\mathrm{T4}})t - \theta_0 \tag{5.8}$$

Substituting $\theta(t)$ in (5.8) back into (5.4), a solution for $v_G^{\mathrm{T5}}(t)$ is obtained in (5.9). The constant θ_0 in (5.9) may be found by noting that at time t_0 the column bus has a known voltage $v_G^{\mathrm{T5}}(t_0)$. With this initial condition, solving for θ_0 results in (5.10).

$$\begin{aligned} v_G^{\mathrm{T5}}(t) = V_G^{\mathrm{T2}} - V_T^{\mathrm{T2}} - \sqrt{\frac{K^{\mathrm{T4}}}{K^{\mathrm{T2}}}}(V_{GS}^{\mathrm{T4}} - V_T^{\mathrm{T4}}) \\ \times \tanh\left(\frac{\sqrt{K^{\mathrm{T2}}K^{\mathrm{T4}}}}{C}(V_{GS}^{\mathrm{T4}} - V_T^{\mathrm{T4}})t + \theta_0\right) \end{aligned} \tag{5.9}$$

$$\begin{aligned} \theta_0 = \tanh^{-1}\left(\sqrt{\frac{K^{\mathrm{T2}}}{K^{\mathrm{T4}}}}\left(\frac{V_G^{\mathrm{T2}} - V_T^{\mathrm{T2}} - v_G^{\mathrm{T5}}(t_0)}{V_{GS}^{\mathrm{T4}} - V_T^{\mathrm{T4}}}\right)\right) \\ - \frac{\sqrt{K^{\mathrm{T2}}K^{\mathrm{T4}}}}{C}(V_{GS}^{\mathrm{T4}} - V_T^{\mathrm{T4}})t_0 \end{aligned} \tag{5.10}$$

The argument of the inverse hyperbolic tangent in (5.10) may sometimes be greater than one or less than minus one, in which case the solution θ_0 is complex. However, the transient response $v_G^{\mathrm{T5}}(t)$ in (5.9) is always real, as may be determined by combining (5.9), (5.10) and the identity (5.11) to give (5.12) with $A(t)$ and B in (5.13) and (5.14).

$$\tanh(\alpha+\beta) = \frac{\tanh\alpha + \tanh\beta}{1 + \tanh\alpha\tanh\beta} \tag{5.11}$$

$$v_G^{\mathrm{T5}}(t) = V_G^{\mathrm{T2}} - V_T^{\mathrm{T2}} - \sqrt{\frac{K^{\mathrm{T4}}}{K^{\mathrm{T2}}}}(V_{GS}^{\mathrm{T4}} - V_T^{\mathrm{T4}})\frac{A(t)+B}{1+A(t)B} \tag{5.12}$$

$$A(t) = \tanh\left(\frac{\sqrt{K^{\mathrm{T2}}K^{\mathrm{T4}}}}{C}(V_{GS}^{\mathrm{T4}} - V_T^{\mathrm{T4}})(t - t_0)\right) \tag{5.13}$$

$$B = \sqrt{\frac{K^{\mathrm{T2}}}{K^{\mathrm{T4}}}}\left(\frac{V_G^{\mathrm{T2}} - V_T^{\mathrm{T2}} - v_G^{\mathrm{T5}}(t_0)}{V_{GS}^{\mathrm{T4}} - V_T^{\mathrm{T4}}}\right) \tag{5.14}$$

Figure 5.3 plots the column bus voltage v_G^{T5} in (5.12) as a function of time t and pixel drive voltage V_G^{T2}, assuming t_0 and $v_G^{\mathrm{T5}}(t_0)$ are zero. Modelling a high-definition television (HDTV) standard, where images have 1080×1920 pixels [12], a load capacitance C equal to 2.4pF was calculated in (5.1) for a 0.35μm 3.3V AMS process [44] with a source diffusion area A_D^{T3} of $1.1\mu\mathrm{m}^2$ and perimeter P_D^{T3} of 4.2μm (for switch transistor T3) and with the number of rows N_1 equal to 1080. Typical for this process, threshold voltages V_T^{T2} and V_T^{T4} were 0.5V and current gains K^{T2} and K^{T4} were $120\frac{\mu\mathrm{A}}{\mathrm{V}^2}$, for 1μm wide by 0.6μm long transistors. The 2.2–2.6V range of the drive voltage V_G^{T2} in Figure 5.3 is typical of a logarithmic pixel in the AMS process when

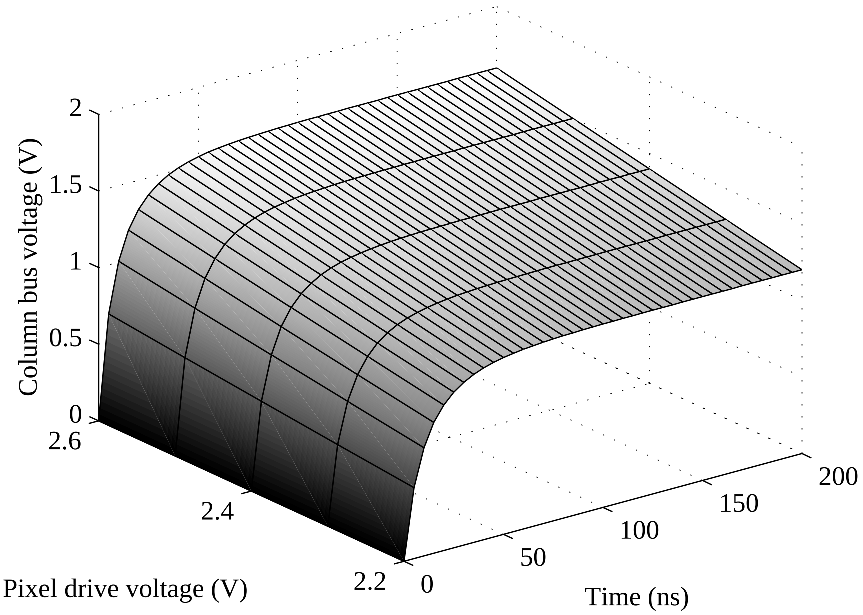


Figure 5.3: The transient response of the first stage readout from a discharged state, as modelled for an HDTV example, where the column bus voltage v_G^{T5} is plotted against pixel drive voltage V_G^{T2} and time t. Note that, for any given time, the column bus voltage is a linear function of the pixel drive voltage.

photocurrents have a $1\mathrm{pA}$–$1\mu\mathrm{A}$ range. The source follower bias voltage V_{GS}^{T4} was $1\mathrm{V}$, or double the threshold voltage, giving a bias current I_{DS}^{T4} of $30\mu\mathrm{A}$.

As time t increases, $A(t)$ in (5.13) approaches unity and $v_G^{\mathrm{T5}}(t)$ in (5.12) approaches the steady state result of Chapter 4. The time it takes for the response to settle depends a little on the value of B in (5.14), which is a function of the pixel drive voltage V_G^{T2} and the initial voltage of the column bus $v_G^{\mathrm{T5}}(t_0)$. However, the settling time depends more closely on parameters of $A(t)$ in (5.13), which means it is proportional to τ in (5.15). This time constant is comprised of factors partly under control of the circuit designer. For the example plotted in Figure 5.3, the time constant in (5.15) equals $40\mathrm{ns}$, which matches the settling time of the response in the figure.

$$\tau = \frac{C}{\sqrt{K^{\mathrm{T2}}K^{\mathrm{T4}}}(V_{GS}^{\mathrm{T4}} - V_T^{\mathrm{T4}})} \tag{5.15}$$

The above derivation gives the transient response of the column bus voltage for any pixel in a column. The column bus voltage is switched in turn to drive an output bus, shared by all columns, via a second stage source follower, as described in Chapter 4. This setup mirrors the parallel source followers of Figure 5.1 except with PMOS transistors instead of NMOS. Analysis of the transient response of the output bus voltage for any column drive voltage is similar to the above analysis and, thus, is not repeated.

5.3 Calibration

If sufficient time is allowed between the switching of a row or column in the sensor array and digitisation then the response of the column buffer or the output buffer should settle at the steady state value. The settling time depends on design parameters of the circuit as well as the initial voltage of the node being charged, i.e. the column bus or the output bus, and the final voltage, as determined by the steady state equation. For example, if the initial and final voltage were the same then no charging or discharging need occur and the settling time would be zero. The settling time allowed by the readout controller should be based on the voltage changes that are likely to occur in a logarithmic sensor, when viewing a typical scene, upon switching from a pixel in one column and row to a pixel in either the next column of the same row or the first column of the next row, depending on the position of the pixel in the raster scan.

Normally, pixel responses are highly correlated with their neighbours except at scene edges in the image, where abrupt changes occur. The settling time allowed must accommodate the variety of charging and discharging demands while meeting the speed requirements of the application. Inevitably, as the circuit theoretically never reaches the steady state without an infinite amount of time, some edges in the image will be slightly smeared along the direction of the raster scan due to insufficient settling. This effect, which happens also in linear CMOS image sensors, may be compensated by digital signal processing to sharpen the scene edges in the image, particularly in the opposite direction of the raster scan.

Care must be taken by the readout controller every time the raster scan completes reading the array and begins again at the first row and every time the raster scan completes reading all columns in one row and switches to the next row. Because these changes often involve extra logic processing in the controller, to generate appropriate addressing signals or to encode synchronisation bits for display purposes, they may permit the column bus or the output bus to discharge. Thus, at the start of every frame readout, the column bus in every column of the array may be required to cover a greater voltage change, than the usual transition from one row to the next, in the usual settling time. Similarly, at the start of every row readout, the output bus may be required to cover a greater voltage change, than the usual transition from one column to the next of a given row, in the usual settling time. The demands on the output bus are more critical than those on the column bus as the former must switch about a thousand times faster (at the pixel scanning frequency) and may have the initial voltage problem a thousand times per frame (as many as there are rows) instead of just once per frame.

These problems may be avoided by ensuring there is no greater delay between reading the last pixel in one frame and the first pixel in the next, or between the last pixel in one row and the first pixel in the next, than there is between reading a pixel in the middle of the array and its neighbour. As this approach may require a low pixel scanning rate, wasting time when reading most pixels in the array, a simpler solution would be for the readout controller to permit extra time to settle at the start of reading a frame or row. Another solution is to precharge each column bus and the output bus to a mid-range voltage when the readout circuit is idle so that readout may resume as if it were scanning from one pixel to the next. A poorly chosen precharge level would be as problematic as a discharge level when digitisation occurs prematurely. Note that increasing the source follower bias currents so that responses settle quickly at the start of reading every frame or row would waste power when reading most pixels.

If the voltage at the column or output bus does not settle prior to digitisation then fixed pattern noise may ensue. Consider an image sensor with $N_1 \times N_2$ pixels, indexed by j_1 and j_2 where $1 \leq j_1 \leq N_1$ and $1 \leq j_2 \leq N_2$. If the time between scanning of rows is T_1 then it will take $N_1 T_1$ time to read one frame. If scanning of a frame begins at time t_0 then scanning of row j_1 begins at time $j_1 T_1 + t_0$ when one period T_1 is given for settling. Assume that the voltage of each column bus at time t_0 is zero. Assume also that circuit parameters (V_T, K etc.) are the same from pixel to pixel and column to column so there is no fixed pattern noise due to stochastic variation. Furthermore, assume the sensor is viewing a uniform scene so that pixel drive voltages $V_G^{\mathrm{T2}_{j_1}}$ are uniform. With these assumptions, the sampled voltage $v_G^{\mathrm{T5}}(t)$ at time $j_1 T_1 + t_0$, denoted $V_{Gj_1}^{\mathrm{T5}}$, on the bus of the first column is given by (5.16)–(5.18).

$$V_{Gj_1}^{\mathrm{T5}} = 1_{j_1}(V_G^{\mathrm{T2}} - V_T) - (V_{GS}^{\mathrm{T4}} - V_T)\frac{A_{\underline{j_1}} + 1_{\underline{j_1}}B}{1_{\underline{j_1}} + A_{\underline{j_1}}B} \tag{5.16}$$

$$A_{j_1} = \tanh\left(\frac{K}{C}(V_{GS}^{\mathrm{T4}} - V_T)j_1 T_1\right) \tag{5.17}$$

$$B = \frac{V_G^{\mathrm{T2}} - V_T}{V_{GS}^{\mathrm{T4}} - V_T} \tag{5.18}$$

Unless T_1 is sufficiently large so that A_{j_1} in (5.17) approximates unity for $j_1 = 1$, the column bus voltage $V_{Gj_1}^{\mathrm{T5}}$ in (5.16) will depend on row number j_1, at least for the first several rows, despite the uniform scene. When j_1 gets large enough, the column bus voltage will settle to the steady state value in (5.19). While these results were derived for the first column, a similar situation exists for all columns. Thus, even with no stochastic variation of device parameters, a row-to-row variation might appear in the digital response of an image sensor due to the transient response of the first stage readout. A similar and simultaneous column-to-column variation would occur due to insufficient settling time in the second stage readout when column bus voltages, indexed by j_2, are switched in sequence to drive the output bus from a discharged state.

$$V_{Gj_1}^{\mathrm{T5}} = 1_{j_1}(V_G^{\mathrm{T2}} - V_{GS}^{\mathrm{T4}}) \tag{5.19}$$

The methods of Chapter 4 to calibrate FPN due to stochastic variation of device parameters may accommodate some of the FPN induced by premature digitisation. Without transient effects, the relationship between the pixel drive voltage V_G^{T2} and the column bus voltage $V_{Gj_1}^{\mathrm{T5}}$ is given by a linear equation, as in (5.19), with constant coefficients from row to row. With transient effects, the same relationship may be approximated by linear equations with offsets a_{j_1} and gains b_{j_1} that vary from row to row (and from column to column for the second stage readout), as in (5.20).

$$V_{Gj_1}^{\mathrm{T5}} = a_{j_1} + b_{j_1} V_G^{\mathrm{T2}} \tag{5.20}$$

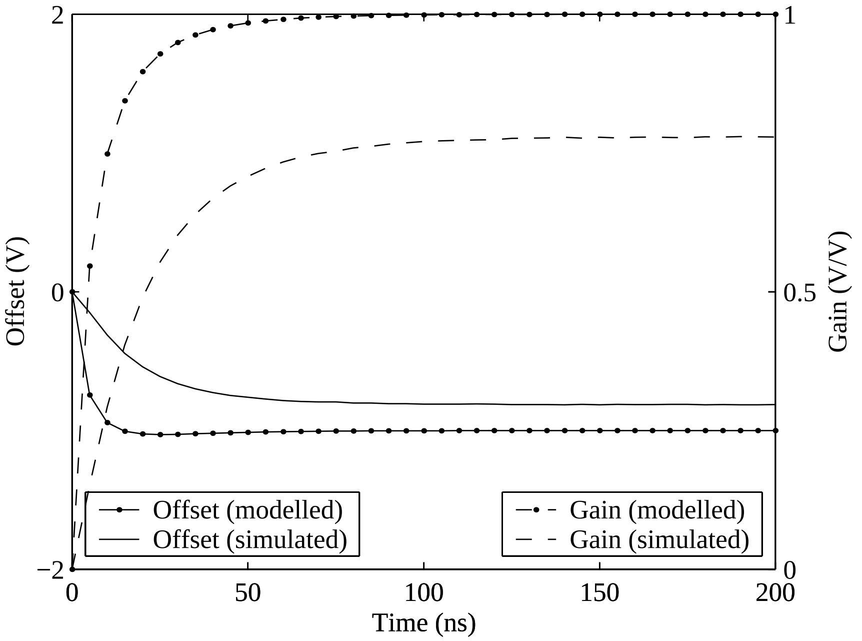


Figure 5.4: The offset and gain of the first stage readout, as modelled and simulated for an HDTV example, that relate the column bus voltage v_G^{T5} linearly to the pixel drive voltage V_G^{T2} as a function of time t. These plots give the offset a_{j_1} and gain b_{j_1} versus row number j_1, where $j_1 T_1$ is the time since discharge when row j_1 is sampled.

Taking (5.16) and performing a first order Taylor expansion of $V_{Gj_1}^{\mathrm{T5}}$ in terms of V_G^{T2} around a reference voltage $\bar{V}_G^{\mathrm{T2}}$ gives the offsets and gains, as in (5.21)–(5.24).

$$a_{j_1} = 1_{j_1}(\bar{V}_G^{\mathrm{T2}} - V_T) - (V_{GS}^{\mathrm{T4}} - V_T)\frac{A_{\underline{j_1}} + 1_{\underline{j_1}}B}{1_{\underline{j_1}} + A_{\underline{j_1}}B} - b_{j_1}\bar{V}_G^{\mathrm{T2}} \tag{5.21}$$

$$b_{j_1} = 1_{j_1} - \frac{1_{\underline{j_1}} - A_{\underline{j_1}}^2}{(1_{\underline{j_1}} + A_{\underline{j_1}}B)^2} \tag{5.22}$$

$$A_{j_1} = \tanh\left(\frac{K}{C}(V_{GS}^{\mathrm{T4}} - V_T)j_1 T_1\right) \tag{5.23}$$

$$B = \frac{\bar{V}_G^{\mathrm{T2}} - V_T}{V_{GS}^{\mathrm{T4}} - V_T} \tag{5.24}$$

Continuing the HDTV example, modelled in Section 5.2, the relationship between the pixel drive voltage and column bus voltage at time t, or $j_1 T_1$ when t_0 is zero, in Figure 5.3 approximates a straight line, with time varying offsets and gains given in Figure 5.4. For small sampling intervals T_1, Figure 5.4 shows that the offset and gain vary for small row numbers j_1 but eventually settle. Furthermore, (5.16)–(5.18) show that the relationship between the pixel drive voltage and the column bus voltage is not perfectly linear. In other words, a first order Taylor expansion of (5.16), as in (5.20), is only an approximation. The higher order terms of the Taylor expansion are expected to vary from row to row, which would appear to the calibration methods of Chapter 4 as a slightly nonlinear FPN in the first few rows. By giving extra settling time at the start of reading each frame, transient-induced FPN may be vastly reduced.

In reality, the period T_1 is large because it represents the time taken to scan one row. For an HDTV sensor with 1080×1920 pixels read at $30\mathrm{Hz}$, this period is about $31\mu\mathrm{s}$. Therefore, the column bus voltage will settle before the first row is read. If scanning of the first row begins as soon as the first row is selected, rather than waiting for one period as assumed above, some pixels in the first row will suffer from insufficient settling time though the effect will disappear by the second row. Thus, row-to-row variation of pixel responses due to insufficient settling time is unlikely or insignificant. However, column-to-column variation is likely and significant with insufficient settling time because the period T_2, representing the time taken by the second stage readout to switch column buffers, is small. For the HDTV sensor, this period is about $16\mathrm{ns}$, a fraction of the settling time of the offsets and gains modelled in Figure 5.4, assuming the second stage transient response is similar to the first.

5.4 Simulation

The circuit in Figure 5.1 was simulated in Cadence using the Spectre simulator and BSIM3 models for an AMS $0.35\mu\mathrm{m}$ $3.3\mathrm{V}$ process. The widths of all transistors were set to $1\mu\mathrm{m}$, the width of the drain and source contacts, and the lengths to $0.6\mu\mathrm{m}$, the minimum length recommended by AMS for transistors in analogue circuits sensitive to threshold voltage variation. Following the HDTV example of Section 5.2, the number of pixels N_1 in the column was set to 1080. This was realised not by having 1079 pixels with open switches, as implied by Figure 5.1, but by having one pixel with an open switch, in addition to pixel j_1, but with amplifier and switch transistors having widths of $1079\mu\mathrm{m}$ and with the source and drain diffusion area and perimeter being 1079 times the usual size. Such wide transistors approximate 1079 transistors in parallel but result in a much faster simulation.

A transient simulation of this setup was performed with the column bias set to $1\mathrm{V}$, resulting in a source follower bias current of about $20\mu\mathrm{A}$, a little lower than the modelled result of Section 5.2, and the pixel drive voltage V_G^{T2} was varied from $2.2\mathrm{V}$ to $2.6\mathrm{V}$, a range typical of a logarithmic pixel simulated in this process. To simulate the condition of a uniform scene presented to a sensor array, the pixel drive voltage of the 1079 parallel pixels with open switches (simulated by a single pixel with wide transistors) was set equal to V_G^{T2} during the simulation.

Figure 5.5 plots the results of the above parametric simulation. The figure shows that the transient response of the column bus voltage behaves as modelled by Figure 5.3, rising like the step-response of a first-order low pass filter to the steady state value. The steady state values of the simulated results are lower than those of the modelled results because the Spectre simulation considers many effects not included in the Level 1 models used for analytical calculations, such as the body effect of $\mathrm{T2}$, the

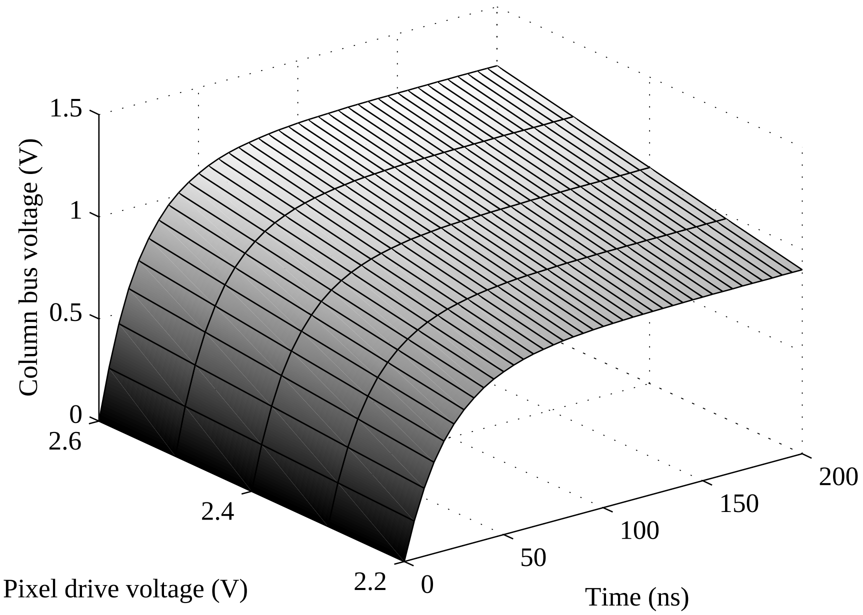


Figure 5.5: The transient response from a discharged state of the first stage readout, as simulated for an HDTV example, where the column bus voltage $v_G^{\mathrm{T}5}$ is plotted against pixel drive voltage $V_G^{\mathrm{T}2}$ and time t. Note that, for any given time, the column bus voltage is a linear function of the pixel drive voltage.

on-resistance of $\mathrm{T}3$ and the finite output resistance of $\mathrm{T}2$ and $\mathrm{T}4$ in saturation.

Figure 5.5 also shows that the dependence of the column bus voltage $v_G^{\mathrm{T}5}$ on the pixel drive voltage $V_G^{\mathrm{T}2}$ is approximately linear at any given time, with the offset and gain varying with time. These time varying coefficients were calculated using linear regression and are plotted in Figure 5.4. The simulated results generally agree with the modelled results, also given in the figure, and show that a variation in the offset and gain of the source follower occurs, from row to row (or column to column), if insufficient time is allowed for the column (or output) bus voltage to settle, especially when it begins from a discharged state as may happen at the start of reading each frame (or row). If sufficient time is allowed for the column (or output) bus to charge then the offset and gain of the linear relationship between $V_G^{\mathrm{T}2}$ and $V_G^{\mathrm{T}5}$ remains constant for all rows (or columns). The offsets and gains in Figure 5.4 of the simulation results in Figure 5.5 are smaller in magnitude than those of the modelled results in Figure 5.3 because of the greater accuracy of the BSIM3 models.

5.5 Experiments

Experiments were conducted using a 512×512 pixel Fuga 15RGB logarithmic sensor, described in Chapter 1. This imager is a colour version of the Fuga 15d, where pixels are overlaid in columnwise fashion with red, green and blue filters. As Chapter 7 focuses on colour in logarithmic sensors, the Fuga 15RGB is treated here as if it were monochromatic. Results presented in this section have been filtered columnwise, after calibration, by a three point moving average filter to cancel the variation introduced by the colour filters. Such an operation does not prejudice the results but facilitates explanation by avoiding unnecessary detail and qualification.

Although the Fuga 15RGB, interfaced to a computer by a PCI card, was capable of a full frame rate of about $8\mathrm{Hz}$ [35], images were very noisy at this speed. Workable performance could be achieved only below $4\mathrm{Hz}$. The camera offered four timing settings to the programmer, called the X1, X2, Y and ADC delay by the manufacturer [34]. The X1 delay controlled the time permitted for settling after a change in the column number (or X-address). This setting had the greatest effect on the speed and image, apart from the ADC delay, and was used to control the frame rate. For reasons that remain unclear, as circuit details of the second stage readout were not supplied, the X2 delay provided an extra delay every 32^{nd} column of the raster scan. However, this setting had almost no effect on the speed or image (Fourier analysis of sample images did not reveal any patterns at 32 column intervals) and was set to the maximum value. The Y delay, possibly a feature not fully implemented in the device driver, had absolutely no effect though it was supposed to control the time permitted for settling after a change in the row number (or Y-address). Lastly, the ADC delay controlled the time permitted for settling at the ADC input and was set to the maximum value. Above a critical value, the setting had little effect on overall speed or image quality but, below this value, both speed and noise increased sharply.

After setting the X1 delay, which was an integer between 0 and 255, the frame rate was computed by measuring the time taken, in Microsoft Windows 98, between readout of consecutive frames. Some variability existed in this measurement as the multitasking operating system used preemptive scheduling but it was compensated for using a moving average filter, a fast processor and by not running other applications in the background. Images were taken of a sheet of white paper, in fluorescent office lighting, to provide a uniform scene. The aperture setting of the lens was varied to simulate intensity variation of the illuminant.

5.5.1 Settling time

Eight images were taken of a sheet of white paper, varying the aperture from 1.8 to 16 f-stops to simulate a two decade intensity variation of the illuminant. These images were taken at the slowest speed of the Fuga 15RGB, at which the frame rate was $0.49\mathrm{Hz}$. The sensor responses, denoted $y_{ij_1j_2}$, where i ranges over the images ($1 \leq i \leq 8$), j_1 ranges over the rows ($1 \leq j_1 \leq 512$) and j_2 ranges over the columns ($1 \leq j_2 \leq 512$), were averaged over the columns and rows respectively to give rowwise and columnwise

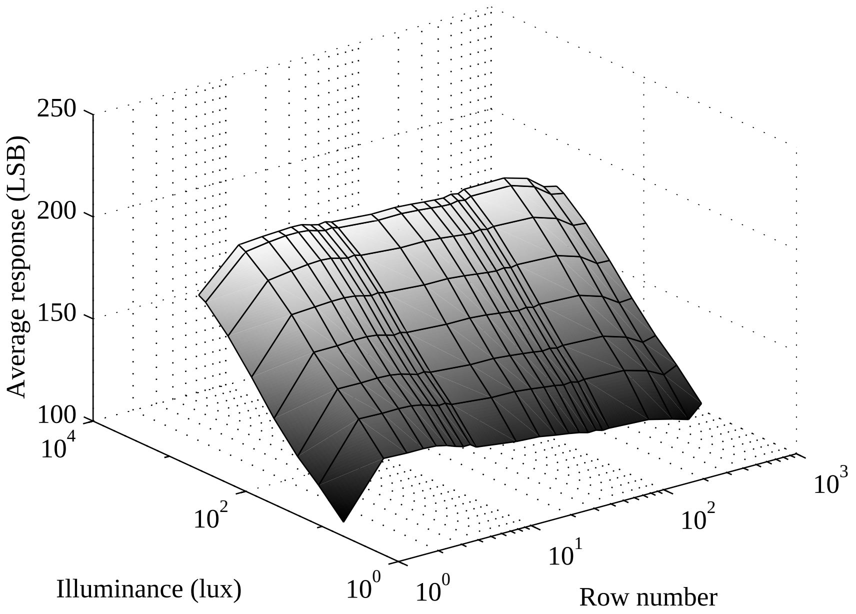


Figure 5.6: The average response $\bar{y}_{ij_1}$ of each row of the Fuga 15RGB versus illuminance x_i and row number j_1. For any row, the average response depends linearly on the logarithm of illuminance. Each row number corresponds to the time the row is digitised from the start of frame scanning by the first stage readout.

response profiles $\bar{y}_{ij_1}$ and $\bar{y}_{ij_2}$ respectively as in (5.25) and (5.26).

$$\bar{y}_{ij_1} = \frac{1_{j_2} y_{ij_1j_2}}{N_2} \tag{5.25}$$

$$\bar{y}_{ij_2} = \frac{1_{j_1} y_{ij_1j_2}}{N_2} \tag{5.26}$$

Figure 5.6 plots the average response $\bar{y}_{ij_1}$ of each row versus illuminance (calculated using the f-stop settings and the measured illuminance of the paper) and row number j_1. The row number, which is proportional to the time the row was read after the start of reading each frame, is on a logarithmic scale to highlight the first few rows while showing all rows. To avoid cluttering the plot with too many lines, as there are 512 rows, responses were averaged rowwise in exponentially increasing bins.[1] The figure shows an insufficient settling time for the first stage readout. Unlike in Figures 5.3 and 5.5, responses of the first row in Figure 5.6 depend on illuminance rather than equal a constant value, which means the Fuga 15RGB permits some settling from the initial condition (though not quite enough). The effective settling time for the first stage

[1] Responses in rows one to nine were not averaged whereas responses in rows 10 to 99, 100 to 499 and 500 to 512 were averaged in bins of 10, 100 and 13 rows respectively.

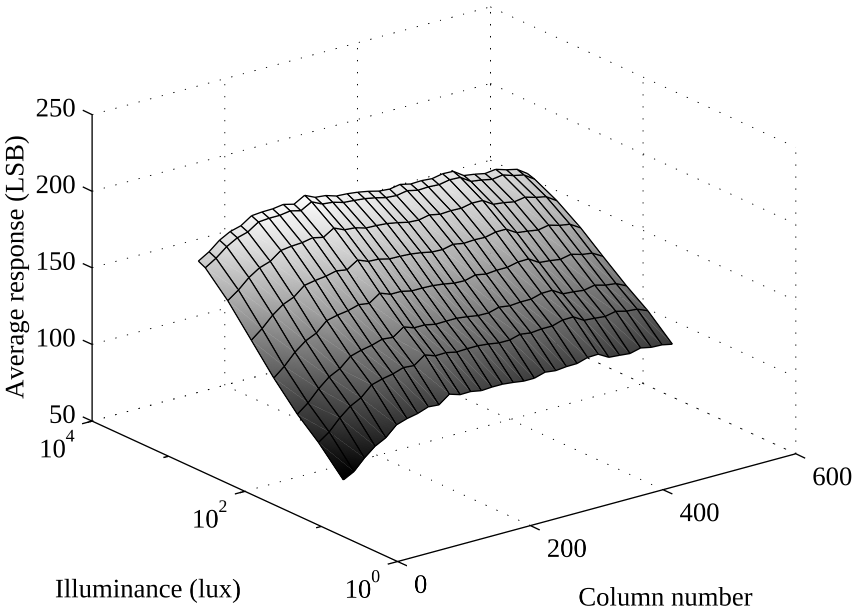


Figure 5.7: The average response $\bar{y}_{ij_2}$ of each column of the Fuga 15RGB versus illuminance x_i and column number j_2. For any column, the average response depends linearly on the logarithm of illuminance. Each column number corresponds to the time the column is digitised from the start of row scanning by the second stage readout.

readout is about two rows or $9\mathrm{ms}$ (calculated using the frame rate and total number of rows in the array). The slow transient response occurs because the Fuga 15RGB was built in a $0.7\mu\mathrm{m}$ $5\mathrm{V}$ process with large transistors, impedences and voltage changes and because a real sensor array has many parasitic effects.

Figure 5.7 plots the average response $\bar{y}_{ij_2}$ of each column versus illuminance and column number j_2, which is proportional to the time the column was read after the start of reading each row. To avoid cluttering the plot with too many lines, as there are 512 columns, responses were averaged columnwise in bins of 16 columns. The figure shows an insufficient settling time for the second stage readout, as illustrated in Figures 5.3 and 5.5, spread out over many columns. Columns are scanned much faster than rows so an equivalent degree of insufficient settling time for the first and second stage readouts would nonetheless affect many more columns than rows. The apparent settling time of 100 columns in Figure 5.7 translates to $0.8\mathrm{ms}$, not much faster than in Figure 5.6. Similar to the first row in Figure 5.6, responses in the first column of Figure 5.7 depend on illuminance because the Fuga 15RGB permits some settling from the initial condition. However, especially since the sensor was operated at the slowest speed, this time is vastly insufficient.

For any row number in Figure 5.6 and any column number in Figure 5.7, both figures show an approximate linear relationship between the average response and the logarithm of illuminance. As with the modelled and simulated results, the gain and offset of this linear dependence varies in an approximately continuous manner, as opposed to a purely random manner if there was only steady state FPN. The lack of surface smoothness in these figures, as compared to Figures 5.3 or 5.5, illustrates the random device parameter variation (V_T, K etc.), reduced by the averaging. Original responses $y_{ij_1j_2}$ were calibrated, according to Chapter 4, for the triple variation model and the experiment was repeated for frame rates of 1.50 and 2.51Hz. The estimated offsets $\hat{a}_{j_1j_2}$, gains $\hat{b}_{j_1j_2}$ and biases $\hat{c}_{j_1j_2}$ for each frame rate were then averaged rowwise and columnwise. These parameter profiles are plotted in Figure 5.8 with the row number on a logarithmic scale as before (but no further averaging across rows or columns).

The offset and gain profiles in Figure 5.8 have similar though inverted trends to the modelled and simulated results in Figure 5.4. Inversion may occur because of precharging rather than discharging of the column and output bus prior to scanning. Higher illuminances actually result in lower voltages, due to the inverting subthreshold load (in Figure 4.1), which means responses are inverted during digitisation for a positive gain, as in Figure 5.8. With a discharged initial condition, digital responses in Figure 5.6 should settle from high to low values for an NMOS source follower, which comprises the first stage readout of the Fuga 15RGB [35]. The advantage of precharging of the column bus in Figure 5.2 (e.g. using a PMOS switch with source at V_{DD} and drain on the bus) is that the load capacitance C is discharged by column transistor $\text{T}4$, which can be made large, rather than charged by pixel transistor $\text{T}2$, which should be small, towards the steady state result. The makers of the Fuga 15RGB do not specify the second stage readout circuit but, as parameter profiles have similar trends rowwise and columnwise in Figure 5.8, it behaves similar to the first stage.

The modelled and simulated results ignored a lot of effects, including bias variation. Calibration of this variation accommodates some of the transient response, as shown in Figure 5.8, which affects the dependence of the estimated offset and gain on row or column number. Note that the bias profile is basin shaped, rowwise and columnwise, which is consistent with vignetting—an effect modelled in Chapter 4. The bias would be higher at the edges because photocurrents would be smaller there, due to vignetting, relative to leakage currents. As the frame rate increases, the parameter profiles change suggesting a transient dependence. The offset profiles have the simplest dependence on frame rate, settling more steeply for slow rates than for fast rates.

All the profile plots change shape with frame rate, particularly the columnwise gain profile, which shows that there is a significant cause of FPN due to transient effects. To assess the impact of gain variation due to transient effects on FPN, responses $y_{ij_1j_2}$ may be calibrated using the double and triple variation models of Chapter 4 with a constraint preventing the gain in either model from varying within a column, though varying as b_{j_2} from column to column, as in models $\hat{y}_{ij_1j_2}$ of Table 5.1. Such a restriction is meaningless with single variation because it assumes the gain does not vary from pixel to pixel. The constrained double variation model may be calibrated efficiently using the raster method of Chapter 3. The constrained triple variation model may be calibrated efficiently with a specific method, following Chapter 4 for the unconstrained triple variation model, involving nonlinear optimisation.

The constrained models in Table 5.1 were calibrated for responses $y_{ij_1j_2}$, recorded

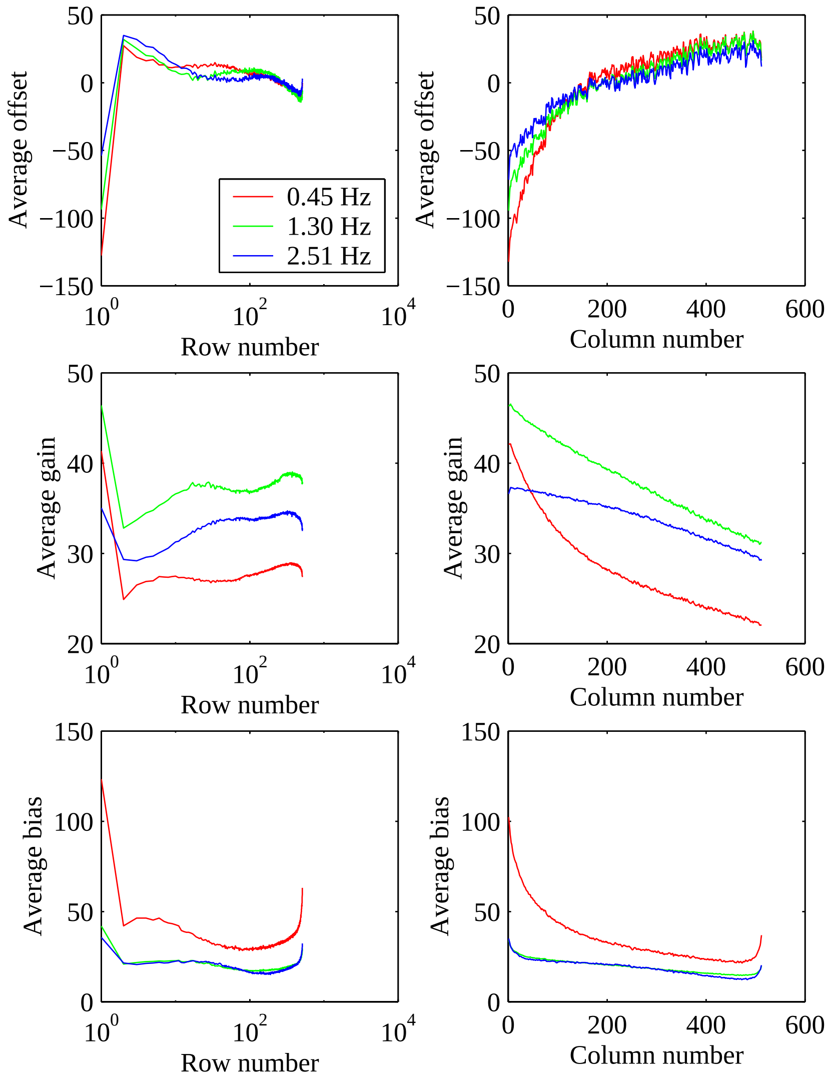


Figure 5.8: The average offset, gain and bias of each row and column of the Fuga 15RGB, after calibration of the triple variation model at frame rates of 0.45, 1.30 and $2.51\mathrm{Hz}$. The row or column number corresponds to the time the row or column is read since the start of reading a frame or row by the first or second stage readout.

Table 5.1: Estimated response $\hat{y}_{ij_1j_2}$ of the $(j_1, j_2)^{\mathrm{th}}$ logarithmic pixel in terms of average response $\bar{y}_i$ or to illuminance x_i, where $l_{ij_1j_2} = \ln(1_i c_{j_1j_2} + 1_{j_1j_2} x_i)$, for the double or triple variation model where the gain b_{j_2} may only vary from column to column. The number of implicit parameters Q is given (assuming x_i is unknown).

Variation	$\hat{y}_{ij_1j_2}$	Q
Constrained double	$1_i a_{j_1j_2} + 1_{j_1} b_{j_2} \bar{y}_i$	$M + N + N_2 - 2$
Constrained triple	$1_i a_{j_1j_2} + b_{\underline{j_2}} l_{ij_1\underline{j_2}}$	$M + 2N + N_2 - 2$

Table 5.2: The residual error $\hat{\sigma}_\epsilon$, average values $\hat{a}$, $\hat{b}$ and $\hat{c}$ of estimated parameters $\hat{a}_{j_1j_2}$, $\hat{b}_{j_2}$ or $\hat{b}_{j_1j_2}$ and $\hat{c}_{j_1j_2}$ and parameter uncertainties $\hat{\sigma}_{\hat{a}}$ and $\hat{\sigma}_{\hat{b}}$ for the double and triple variation models, with unconstrained and constrained gain, where estimated responses $\hat{y}_{ij_1j_2}$, with $\hat{l}_{ij_1j_2} = \ln(1_i \hat{c}_{j_1j_2} + 1_{j_1j_2} \hat{x}_i)$, are fitted to actual responses $y_{ij_1j_2}$.

Variation	$\hat{y}_{ij_1j_2}$	$\hat{\sigma}_\epsilon$	$\hat{a} \pm \hat{\sigma}_{\hat{a}}$	$\hat{b} \pm \hat{\sigma}_{\hat{b}}$	$\hat{c}$
Con. dbl.	$1_i \hat{a}_{j_1j_2} + 1_{j_1} \hat{b}_{j_2} \bar{y}_i$	2.4	$0.0 \pm .87$	$1.0\frac{\mathrm{LSB}}{\mathrm{LSB}} \pm .13\%$	
Unc. dbl.	$1_i \hat{a}_{j_1j_2} + \hat{b}_{j_1j_2} \bar{y}_i$	1.3	0.0 ± 2.8	$1.0\frac{\mathrm{LSB}}{\mathrm{LSB}} \pm 1.7\%$	
Con. tri.	$1_i \hat{a}_{j_1j_2} + \hat{b}_{\underline{j_2}} \hat{l}_{ij_1\underline{j_2}}$	.79	$0.0 \pm .29$	$34\mathrm{LSB} \pm .043\%$	24
Unc. tri.	$1_i \hat{a}_{j_1j_2} + \hat{b}_{\underline{j_1j_2}} \hat{l}_{i\underline{j_1j_2}}$	.59	0.0 ± 1.2	$28\mathrm{LSB} \pm .72\%$	35
In	LSB	LSB	LSB ± LSB		lux

at a frame rate of 0.49Hz, along with the unconstrained double and triple variation models of Chapter 4 (where array index j is decoded into row and column indices j_1 and j_2). Residual errors, average parameter estimates and parameter uncertainties are given in Table 5.2 for these models. Unconstrained double variation has a much lower residual error than constrained double variation whereas unconstrained triple variation has a slightly lower residual error than constrained triple variation. Gain and offset uncertainties show that the constrained estimates are far more certain than the unconstrained ones, meaningful in the triple variation case as the residual errors are comparable. As the stochasticity of bias and illuminance estimates were ignored for both triple variation models, the parameter uncertainties are similarly underestimated.

The residual error versus illuminance is plotted in Figure 5.9 for each calibrated model in Table 5.2. The figure shows that the residual errors for constrained and unconstrained triple variation are similar. Both are relatively independent of illuminance. These results mean that the gain variation observed in the Fuga 15RGB, at least over a two decade dynamic range, may almost entirely be attributed to a columnwise variation introduced by insufficient settling time in the second stage readout. Although insufficient settling time in the first stage readout does introduce a rowwise variation of the gain, as shown in Figure 5.8, this effect may be neglected at low frame rates because it affects only a few rows. Comparing constrained and unconstrained double variation in Figure 5.9 shows that constraining the gain is too restrictive in that case. The reason

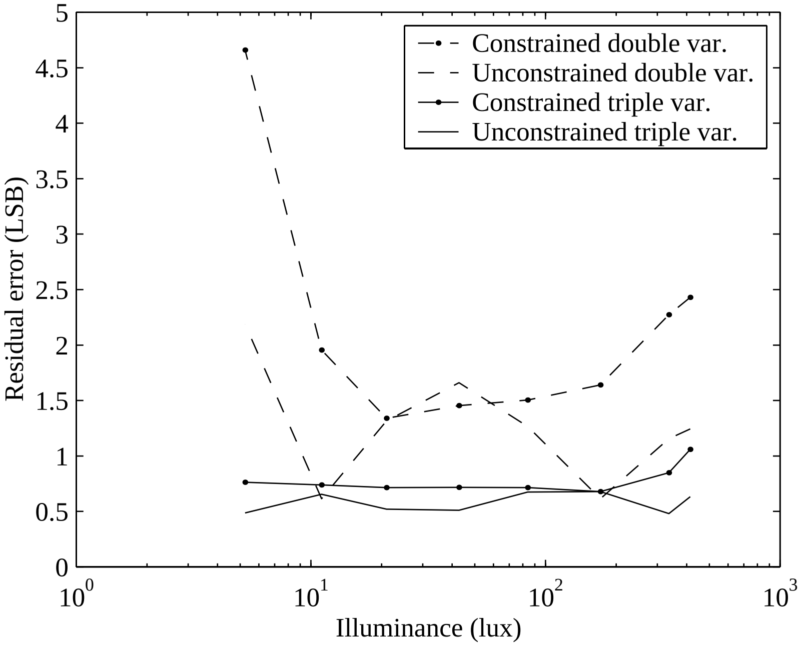


Figure 5.9: The residual error $\hat{\sigma}_{\epsilon_i}$ versus illuminance x_i for calibration of the double and triple variation models, unconstrained and constrained, to Fuga 15RGB responses.

is because gain variation may accommodate some of the ignored bias variation but the latter varies significantly within columns. Thus, pixel-to-pixel bias variation must be permitted for good calibration results.

5.5.2 Switch position

In reality, the first stage readout circuit of the Fuga 15RGB does not precisely match the one given in Figure 5.1, which may account (with precharging) for some of the discrepancies between parameter profiles in Figures 5.4 and 5.8. The positions of the amplifier and switch transistors in each pixel, i.e. $\mathrm{T}2$ and $\mathrm{T}3$, are swapped in the Fuga 15RGB [35], as in Figure 5.10, although IMEC uses the positions in Figure 5.1 for the 2048×2048 logarithmic sensor developed afterwards [24]. Neither the designers at IMEC nor the suppliers at C-Cam Technologies have published the circuitry for the second stage readout of the Fuga 15RGB. Most likely, it is similar to the first.

From the point of view of steady state performance, the switch position in Figure 5.10 is superior to the one in Figure 5.1. When turned on, switch $\mathrm{T}3_{j_1}$ is in saturation for the circuit in Figure 5.10 whereas it is in the triode region for the circuit in Figure 5.1. Though the steady state analysis in Chapter 4 assumed the switch to be ideal, in reality it does affect circuit operation. With $\mathrm{T}2_{j_1}$ and $\mathrm{T}4$ in saturation during normal operation (for either switch position), having the switch also in saturation, as in

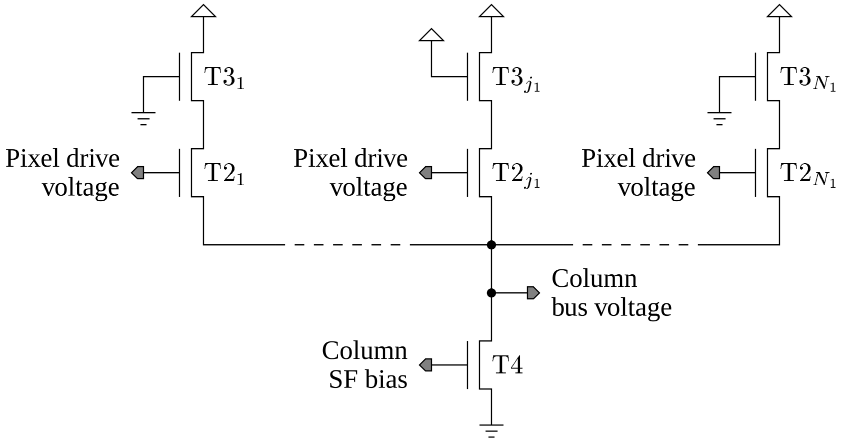


Figure 5.10: The first stage readout of the Fuga 15RGB image sensor consists of N_1 amplifier and switch transistors T2 and T3, one pair in each pixel, and a current source T4, one in each column of pixels. When switch T3_{j_1} is on, where $1 \leq j_1 \leq N_1$, all other switches are off and T2_{j_1} forms a source follower (SF) with T4. Note that the positions of T2 and T3 are reversed compared to the typical circuit of Figure 5.1.

Figure 5.10, leads to a higher gain and a more linear source follower. Consider that the on-resistance of T3_{j_1} in Figure 5.1 depends on its gate-source voltage, which depends on the column bus voltage. However, the column bus voltage in turn depends on the on-resistance of the switch because that resistance determines the drain-source voltage drop across T3_{j_1}. These effects degrade the gain and linearity of the source follower.

However, the switch position in Figure 5.10 leads to a poor transient response. The load impedence that determines the transient response consists primarily of the source-bulk capacitances of the N_1 amplifier transistors, rather than the switch transistors in Figure 5.1. But the amplifier transistors in Figure 5.10 are not in the cutoff region as are the corresponding (in terms of position) switch transistors in Figure 5.1. Since pixel drive voltages maintain their levels irrespective of the switch state, the gate-source voltage of the amplifier transistors may exceed the threshold voltage. When amplifier T2_{j_1} is in saturation with switch T3_{j_1} closed, the amplifier transistors of all other pixels may be in the triode region where they behave as voltage controlled resistors.

When the amplifier transistors of pixels with open switches in Figure 5.10 behave like resistors, the load impedence involved in the transient response depends, in addition to the source-bulk capacitance of each amplifier transistor, on the series connection of each triode resistance with the drain-bulk capacitance of each amplifier transistor and the source-bulk capacitance of each switch transistor, taken in parallel over pixels with open switches. The channel-bulk capacitances of the amplifier transistors also contribute to the load. Thus, the load impedance for the circuit in Figure 5.10 is considerably higher than the load impedence for the circuit in Figure 5.1. Parasitic capacitances are not coupled with the latter because the switch transistor is adjacent

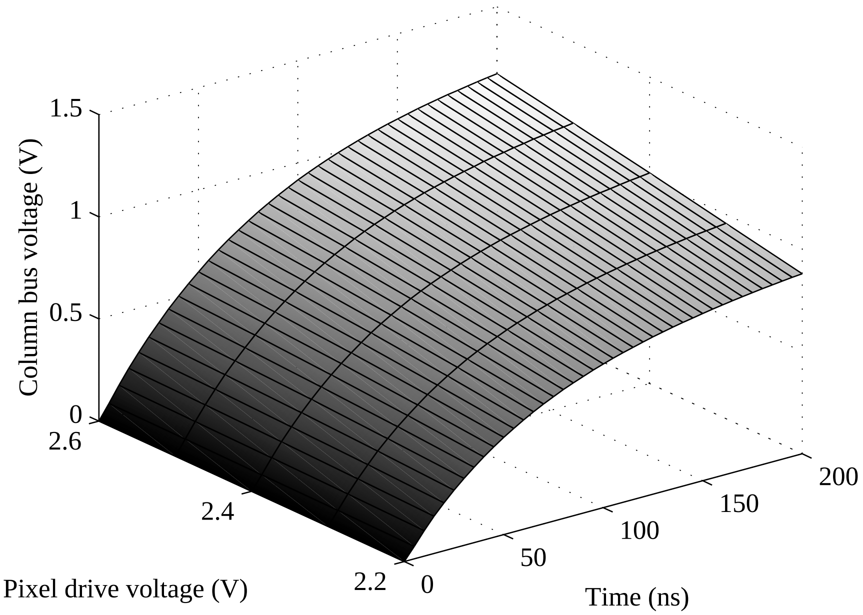


Figure 5.11: The transient response from a discharged state of the first stage readout in Figure 5.10, as simulated for an HDTV example, where the column bus voltage v_G^{T5} is plotted against pixel drive voltage V_G^{T2} and time t. The response does not settle in $200\mathrm{ns}$ whereas the one in Figure 5.5, for the typical circuit, settles in less than $100\mathrm{ns}$.

to the column bus and is always in the cutoff region for deselected pixels. Returning to the HDTV example described in Sections 5.2 and 5.4, the column bus voltage for the switch position in Figure 5.10, as a function of pixel drive voltage and time (since the start of scanning), is given in Figure 5.11. Comparing these simulation results with the ones in Figure 5.5, the switch position of the Fuga 15RGB leads to a much slower settling time than for the switch position in Figure 5.1.

Furthermore, as the drain-source resistance of a transistor in the triode region depends on the gate-source voltage of the transistor, the transient response and settling time of the circuit in Figure 5.10 depends on the gate and source voltage of the amplifiers with open switches, i.e. the pixel drive voltages and the column bus voltage. Triode resistances decrease with increasing gate-source voltage so that coupling of parasitic capacitances becomes more significant. A settling time for a readout circuit that depends in a nonlinear way on drive voltages of deselected pixels, as well as the column bus voltage due to the selected pixel, is highly undesirable. Simulations show that these dependencies complicate the transient response even more with precharging of the column bus prior to scanning. Such nonlinear effects are expected to exacerbate the FPN that appears in an image sensor due to insufficient settling time, especially over a high dynamic range when voltages cover a wide range.

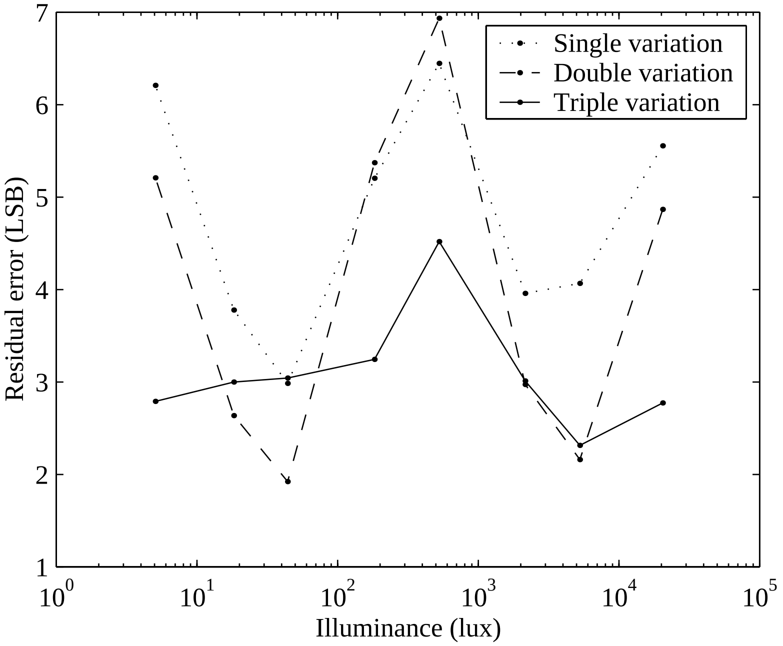


Figure 5.12: The residual error $\hat{\sigma}_{\epsilon_i}$ versus illuminance x_i for calibration of the single, double and triple variation models to Fuga 15RGB responses over a high dynamic range, which shows a performance breakdown especially at bright illuminances.

Indeed, Figure 5.12 shows a breakdown in the calibration methods of Chapter 4 for an experiment with the Fuga 15RGB over a dynamic range of three and a half decades. An 800 Watt tungsten lamp, with dichroic filters to simulate a daylight spectrum, was used to illuminate a sheet of white paper that was imaged eight times, using neutral density filters to simulate intensity variation of the illuminant at half decade intervals. The figure plots the residual error versus illuminance after calibration of the single, double and triple variation models (unconstrained columnwise). All three models give a poorer performance than in Chapter 4, i.e. Figure 4.3. The shapes of the error curves are significantly different, particularly in the mid-range of illuminance. Triple variation, however, still gives the best results and is otherwise nearly flat.

The cause of model breakdown, shown in Figure 5.12, is a transient phenomenon. For the triple variation calibration, Figure 5.13 plots the standard deviation of the residual error for each pixel, taken over the eight illuminances (rather than for each illuminance, taken over the 512^2 pixels, as in Figure 5.12). The high error in the leftmost columns of the image, a band that stretches over all rows, is a second stage transient phenomenon with a highly nonlinear nature, which explains the high error in the triple variation result of Figure 5.12. This band does not appear for the triple variation calibration in Chapter 4, which covered a two decade dynamic range. The error band

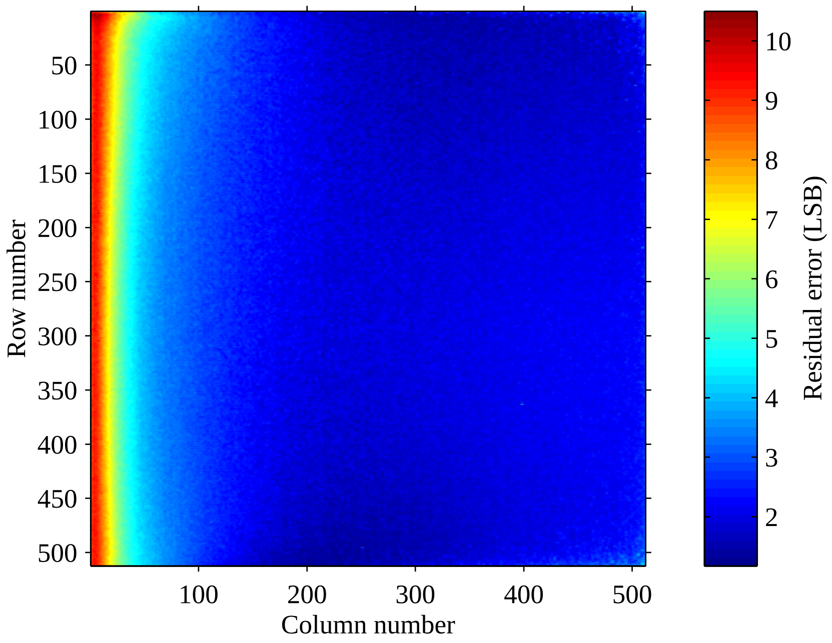


Figure 5.13: The residual error $\hat{\sigma}_{\epsilon_{j_1 j_2}}$ versus row and column numbers j_1 and j_2 for calibration of the triple variation model to Fuga 15RGB responses over a high dynamic range, which shows a performance breakdown especially in the leftmost columns (but also in the topmost row) that suggests a transient cause.

occurs because of insufficient settling time and possibly a poor choice of switch position in the second stage readout of the Fuga 15RGB. A similar high error, not visible in Figure 5.13, appears in the topmost row of the image. Instead of modelling and calibrating this complex phenomenon, the best way to reduce the resulting FPN is to provide more settling time, at the start of reading each frame and row, and to choose the switch position in Figure 5.1 over the switch position in Figure 5.10.

5.6 Conclusion

Whereas the previous chapter considered how parameter variation from pixel to pixel affects the steady state response of a sensor so as to produce FPN, this chapter considered how the transient response of the sensor can lead to FPN regardless of parameter variation. A transient analysis of the photodiode and the logarithmic current-to-voltage converting load was not considered because the bandwidths of these components are sufficient to meet the demands of most applications. On the other hand, because pixels are scanned serially for digitisation by a single ADC, the readout circuit must switch very quickly prior to digitisation, for megapixel sensors operating at video rates, which makes its transient response crucial to image quality. Furthermore, as pixels are scanned in raster fashion using a two stage process, one to copy all pixel voltages in a row to column buffers and the other to copy a column buffer voltage to a single output buffer, each stage has different demands on its transient response. The second stage must operate on the order of a thousand times faster than the first stage and is hence more critical to image quality in terms of transient response.

A model of the transient response of a switched source follower circuit, typical for both the first and second stage readouts, was constructed by solving a differential equation relating the input and output voltage of the readout stage to the designable parameters of the circuit and the initial voltage of the output. The model identifies the load impedence of the circuit to be the parallel combination of the junction capacitances of all the switches. When one row or one column is selected by the first or second stage readout, all other rows or columns have open switches, which are transistors in the cutoff region. The fact that the switches are in the cutoff region when open is important because it reduces the load impedence and makes it independent of the input voltages of all deselected source followers. However, an alternate design of the source follower exists, with the switch and amplifier transistors reversed to improve steady state linearity but it results in a poorer transient response.

The model developed in this chapter was used to show that if a readout circuit for an image sensor does not allow sufficient settling time then digitised responses will vary in a predictable manner from row to row or from column to column, even with a uniform stimulus and no device parameter variation. Furthermore, these effects would appear principally as offset and gain variation correlated to the row or column number, as opposed to purely random offset and gain variation, and therefore could be partly calibrated using steady state methods. The effects would be most noticeable, and hence settling time is most important, for the topmost rows or the leftmost columns as the greatest voltage changes are likely to occur at the start of reading each frame or at the start of reading each row.

Simulations were carried out in an AMS process using Spectre. For an HDTV example, the simulation results agreed with modelled results although there were small discrepancies because the model used simple equations to describe transistor behaviour. These results confirmed that insufficient settling time may be a considerable cause of response variation. Regression analysis showed that, even with insufficient settling time, the input-output relationship is approximately linear but with an offset and gain that vary according to the row or column number. For the readout circuit with the switch in the alternate position, a simulation confirmed that this approach greatly increased the settling time.

Experiments were performed with a Fuga 15RGB sensor. Images were taken of uniform scenes with different aperture settings, to simulate illuminant variation, and with different speed settings of the readout. The results demonstrate transient effects that cause substantial variation of digital responses in a manner similar to the modelled and simulated response. The results were calibrated using the triple variation model of Chapter 4. Plots of the offset, gain and bias, averaged separately over all columns and all rows showed how the offset and gain depended on the transient response. The rowwise and columnwise bias depended on the transient response but also showed signs of vignetting. The transient effects appeared to be more significant from column to column than from row to row of the experimental results, particularly for the gain.

Calibration of the data assuming triple variation but with constraints on the gains so that they do not vary within a column but may vary from column to column gives a residual error almost identical to the case of unconstrained triple variation. The constrained model, however, exhibits a much lower uncertainty in the estimated offsets and gains, which suggests it is a better model to describe FPN in the Fuga 15RGB. Finally, the Fuga 15RGB uses the atypical configuration for the switch transistor in the readout circuit, which may be a cause of complex effects on the response over a high dynamic range. With this readout circuit, the transient response depends on voltages at the inputs of deselected source followers and varies significantly with the initial output voltage of the source follower. Experiments conducted over a dynamic range of three and a half decades, using a tungsten lamp and neutral density filters, demonstrate a breakdown of previous models for calibration because of transient effects.

Like steady state effects due to device parameter variation, transient effects due to insufficient settling time may be a significant cause of FPN in CMOS image sensors. Although much of this effect may be calibrated by assuming offset and gain variation due to the flexibility of these steady state models to accommodate transient effects, the transient effects are inherently more complex and may require digital filtering for proper compensation. The best solution, however, is to test and avoid poor circuit designs and to permit sufficient settling time at the start of reading each frame and row.

Chapter 6

Temperature dependence

6.1 Introduction

Electronic circuits in consumer, industrial and military applications are required to operate in diverse and changing temperatures. Unlike the human eye, image sensors usually do not exist in a homeostatic environment. Due to semiconductor physics, responses to the same stimulus may thus vary with temperature. Whereas previous chapters modelled and calibrated fixed pattern noise (FPN) in logarithmic CMOS image sensors at one temperature, this chapter considers the dependence of FPN on temperature and how to compensate for it. A variation of device parameters, from pixel to pixel or column to column, related to temperature and illuminance sensitivity leads to FPN.

In the study of linear CCD and CMOS image sensors, it is well known that the response of pixels with the aperture of the camera closed, called the *dark response*, is a strong function of temperature. In reality, this dark response also bears upon the response of the pixels with the aperture open (i.e. to a focused image), called the *light response*. As the dark response is only affected by temperature and not illuminance, it may be used to discern and correct unwanted effects of temperature dependence on the light response of the image sensor. Any unwanted effects due to illuminance dependence may be compensated using methods similar to those in Chapter 4.

As in Chapter 4, this chapter considers only the steady state causes of FPN. In reality, the load impedences and settling times described in Chapter 5 are affected by temperature and may lead to temperature-dependent FPN. However, transient effects on sensor responses may be minimised by proper design and timing of the readout circuit, allowing for the worst case load impedence and settling time over the required temperature and illuminance range.

Section 6.2 models the response of logarithmic CMOS image sensors over temperature and illuminance. Section 6.3 describes calibration of the model using images of uniform scenes taken at different temperatures and with different illuminances. As calibration of image sensors may be a costly process, emphasis is placed on reducing the need for temperature and illuminance measurement, reducing the complexity of the model and reducing the number of parameters to be estimated. Sections 6.4 and 6.5

evaluate simplified models and calibrations with simulation and experiment.

6.2 Modelling

To model the response y of a logarithmic pixel to temperature T and illuminance x, the model derived in Chapter 4 for illuminance alone, repeated in (6.1), may be extended by considering the temperature dependence of the physical parameters (of the circuit in Figure 4.1) that make up the offset a, gain b and bias c, repeated in (6.2)–(6.4). The error ϵ, repeated in (6.5), is assumed to be independent of temperature and illuminance.

$$y = a + b\ln(c + x) + \epsilon \tag{6.1}$$

$$\begin{aligned} a = F_{ADC} + G_{ADC}\Bigg(& V_{DD} \\ & + \frac{n^{\mathrm{T1}}kT}{q}\ln\left(\frac{I_{on}^{\mathrm{T1}}}{G_A G_L G_Q A}\right) - V_{on}^{\mathrm{T1}} \\ & - V_T^{\mathrm{T2}} - \sqrt{\frac{K^{\mathrm{T4}}}{K^{\mathrm{T2}}}}\left(V_{GS}^{\mathrm{T4}} - V_T^{\mathrm{T4}}\right) \\ & - V_T^{\mathrm{T5}} - \sqrt{\frac{K^{\mathrm{T7}}}{K^{\mathrm{T5}}}}\left(V_{GS}^{\mathrm{T7}} - V_T^{\mathrm{T7}}\right)\Bigg) \end{aligned} \tag{6.2}$$

$$b = -G_{ADC}\frac{n^{\mathrm{T1}}kT}{q} \tag{6.3}$$

$$c = \frac{I_S}{G_A G_L G_Q A} \tag{6.4}$$

$$\epsilon = \epsilon_Q + \epsilon_N + \epsilon_D \tag{6.5}$$

The offset parameter a in (6.2) is affected by temperature in a number of ways. Threshold voltages V_T have a linear dependence on temperature, as in (6.6), and current gains K depend on temperature by a power law, as in (6.7). These equations are taken from the HSPICE Level 28 model as the simpler Levels 1–3 models used in Chapter 4 do not consider the temperature dependence of V_T or K [43]. T_0 is simply a reference temperature. V_{T0} and K_0 are the threshold voltage and current gain at that temperature. The multiplier TCV and exponent BEX determine how quickly the threshold voltage and current gain vary with temperature.

$$V_T = V_{T0} - TCV(T - T_0) \tag{6.6}$$

$$K = K_0\left(\frac{T}{T_0}\right)^{BEX} \tag{6.7}$$

Returning to the Level 3 model, the parameter V_{on} in (6.2) signifies the gate-source voltage that is the threshold between the weak and strong inversion regions of transistor operation [43]. This threshold depends linearly on temperature, as in (6.8). As I_{on} in (6.2) is the drain-source current at this voltage, its dependence on temperature is given in (6.9) using the Level 1 model of current in the saturation region (ignoring the finite output resistance of transistors in saturation) [43].

$$V_{on} = V_T + \frac{nkT}{q} \tag{6.8}$$

$$I_{on} = K\left(\frac{nkT}{q}\right)^2 \tag{6.9}$$

The gain parameter b depends on temperature in only one way, which is already considered in (6.3). Using the Level 3 model, the slope of the subthreshold response, i.e. voltage versus current on a logarithmic scale, of a diode-connected transistor is a multiple of temperature in Kelvin, as in (6.10) [43].

$$V_{DS} = \frac{nkT}{q}\ln\left(\frac{I_{DS}}{I_{on}}\right) + V_{on} \tag{6.10}$$

Assuming that the optics of a camera are stable with respect to temperature variation, and neglecting any dependence of quantum efficiency on temperature, then the bias parameter c in (6.4) depends on temperature in only one way. In the simplest case, the reverse bias saturation current of the photodiode is an exponential function of temperature (approximately doubling every 10K), as in (6.11) [43].

$$I_S = I_{S0}e^{T/T_\vartriangle} \tag{6.11}$$

Applying the above temperature dependences of physical parameters to (6.2)–(6.4), the response y of a logarithmic pixel to temperature T and illuminance x is modelled in (6.12), with abstract parameters a_k, b_1 and c_1 in (6.13)–(6.17). This model assumes that BEX in (6.7) does not vary from transistor to transistor.

$$y = a_1 + a_2T + a_3T\ln T + b_1T\ln(c_1e^{T/T_\vartriangle} + x) + \epsilon \tag{6.12}$$

$$\begin{aligned} a_1 = F_{ADC} + G_{ADC}\Bigg(& V_{DD} - V_{T0}^{\mathrm{T1}} - 3\,TCV\cdot T_0 \\ & - V_{T0}^{\mathrm{T2}} - \sqrt{\frac{K_0^{\mathrm{T4}}}{K_0^{\mathrm{T2}}}}(V_{GS}^{\mathrm{T4}} - V_{T0}^{\mathrm{T4}} - TCV\cdot T_0) \\ & - V_{T0}^{\mathrm{T5}} - \sqrt{\frac{K_0^{\mathrm{T7}}}{K_0^{\mathrm{T5}}}}(V_{GS}^{\mathrm{T7}} - V_{T0}^{\mathrm{T7}} - TCV\cdot T_0)\Bigg) \end{aligned} \tag{6.13}$$

$$\begin{aligned} a_2 = G_{ADC}\Bigg(& \frac{n^{\mathrm{T1}}k}{q}\ln\left(\frac{K_0^{\mathrm{T1}}T_0^{-BEX}}{G_AG_LG_QA}\left(\frac{n^{\mathrm{T1}}k}{q}\right)^2\frac{1}{e}\right) \\ & + TCV\left(3 - \sqrt{\frac{K_0^{\mathrm{T4}}}{K_0^{\mathrm{T2}}}} - \sqrt{\frac{K_0^{\mathrm{T7}}}{K_0^{\mathrm{T5}}}}\right)\Bigg) \end{aligned} \tag{6.14}$$

$$a_3 = (2 + BEX)G_{ADC}\frac{n^{\mathrm{T1}}k}{q} \tag{6.15}$$

$$b_1 = -G_{ADC}\frac{n^{\mathrm{T1}}k}{q} \tag{6.16}$$

$$c_1 = \frac{I_{S0}}{G_A G_L G_Q A} \tag{6.17}$$

A pixel-to-pixel or column-to-column variation of a_1, a_2, a_3, b_1, c_1 or any combination thereof would cause FPN in an image sensor (T_Δ is not expected to vary). As (6.12) shows, any type of FPN would be temperature-dependent unless only a_1 varied. Strictly speaking, only a_1 and a_2 may vary from column to column as other parameters do not depend on column transistors T4–6 in the circuit of Figure 4.1.

6.3 Calibration

To calibrate a sensor having N pixels over temperature and illuminance, images are taken of a uniform scene at L different temperatures denoted T_h, where $1 \leq h \leq L$, and M different illuminances x_i, where $1 \leq i \leq M$. At the h^{th} temperature and the i^{th} illuminance, the response of the j^{th} pixel, where $1 \leq j \leq N$, is denoted y_{hij}. Due to (6.12), the actual response y_{hij} may be estimated by $\hat{y}_{hij}$ in (6.18), which lacks an error term ϵ_{hij}, with l_{hi} in (6.19). The error is assumed to be independent from sample to sample and to follow a zero-mean Gaussian distribution. Note that (6.18) assumes a variation of offsets a_k and gain b_1 from pixel to pixel, representing $4N$ variables

$$\hat{y}_{hij} = 1_{hi} a_{1j} + 1_i a_{2j} T_h + 1_i a_{3j} T_{\underline{h}} \ln T_{\underline{h}} + b_{1j} T_{\underline{h}} l_{\underline{h}i} \tag{6.18}$$

$$l_{hi} = \ln(1_i c_1 e^{T_h/T_\Delta} + 1_h x_i) \tag{6.19}$$

There are only two variables c_1 and T_Δ in (6.19) as bias variation has not been considered. Chapter 4 shows that including bias variation makes both calibration and correction nonlinear. While including bias variation leads to better results, the method is not practical in cost sensitive applications. Later sections in this chapter shall clarify the limitations of ignoring bias variation. Chapter 4 also showed that nonlinear optimisation may be avoided even when illuminances x_i are assumed to be unknown. Whereas temperatures T_h and illuminances x_i may be known, the cost of calibration may be reduced in terms of computation and measurement by assuming they are unknown, which adds $L + M$ variables. However, if T_h and x_i are unknown then $\hat{y}_{hij}$ in (6.18) is unchanged by the transformations in (6.20), which means there are two fewer variables, in reality, for a total of $L + M + 4N$.

$$(a_{1j}, a_{2j}, a_{3j}, b_{1j}, c_1, x_i, T_h, T_\Delta) \equiv \begin{cases} (a_{1j}, a_{2j} - b_{1j} \ln\gamma, a_{3j}, b_{1j}, \gamma c_1, \gamma x_i, T_h, T_\Delta) \\ (a_{1j}, \frac{a_{2j} - a_{3j}\ln\gamma}{\gamma}, \frac{a_{3j}}{\gamma}, \frac{b_{1j}}{\gamma}, c_1, x_i, \gamma T_h, \gamma T_\Delta) \end{cases} \tag{6.20}$$

Parameters in (6.18) may be estimated by minimising the SSE in (6.21) between the actual responses y_{hij} and estimated responses $\hat{y}_{hij}$. For any choice of c_1, x_i, T_h and T_Δ in (6.19), the raster method of Chapter 3 may be used to estimate a_{kj} and b_{1j} by encoding variables h and i into a single variable that indexes the LM images. Nonetheless, counting the degeneracies in (6.20), nonlinear optimisation is required to estimate the $L + M$ parameters c_1, x_i, T_h and T_Δ. Nonlinear optimisation may be avoided by the *offset cancellation* and *temperature proxy* methods described below.

$$SSE = 1_{hij}(y_{hij} - \hat{y}_{hij})^2 \tag{6.21}$$

6.3.1 Offset cancellation

The nonlinear parameters c_1, x_i, T_h and T_Δ in (6.19) may be reduced by assuming the average of the actual responses over all pixels, denoted $\bar{y}_{hi}$ in (6.22), equals the average of the estimated responses over all pixels, as in (6.23). The assumption is good when N, the number of pixels, is large and the error ϵ_{hij}, between actual and estimated responses, follows a zero-mean Gaussian distribution.

$$\bar{y}_{hi} = \frac{1_j}{N} y_{hij} \tag{6.22}$$

$$\bar{y}_{hi} \approx \frac{1_j}{N} \hat{y}_{hij} \tag{6.23}$$

With the above assumption, $\hat{y}_{hij}$ in (6.18) and $\bar{y}_{hi}$ in (6.23) may be rewritten as (6.24) and (6.25), where a'_{kj}, b'_{1j}, $\bar{a}_k$ and $\bar{b}_1$ are given in (6.26)–(6.29). Note that $\bar{y}_{hi}$ is known from the data in (6.22). The number of implicit parameters in (6.24) equals $LM + L + 4N - 4$, counting $\bar{y}_{hi}$, T_h, a'_{kj} and b'_{1j} and deducting for degeneracies. There are four degeneracies, due to (6.26) and (6.27), whereby the average of a'_{kj} and b'_{1j} over all pixels is zero and one respectively. Thus, the number of implicit parameters have increased by $LM - M - 4$ but the number of nonlinear parameters have been decreased by M, eliminating c_1, x_i, T_Δ and associated degeneracies.

$$\hat{y}_{hij} = 1_{hi} a'_{1j} + 1_i a'_{2j} T_h + 1_i a'_{3j} T_{\underline{h}} \ln T_{\underline{h}} + b'_{1j} \bar{y}_{hi} \tag{6.24}$$

$$\bar{y}_{hi} = 1_{hi} \bar{a}_1 + 1_i \bar{a}_2 T_h + 1_i \bar{a}_3 T_{\underline{h}} \ln T_{\underline{h}} + \bar{b}_1 T_{\underline{h}} l_{\underline{h}i} \tag{6.25}$$

$$a'_{kj} = a_{kj} - b_{1j} \frac{\bar{a}_k}{\bar{b}_1} \tag{6.26}$$

$$b'_{1j} = \frac{b_{1j}}{\bar{b}_1} \tag{6.27}$$

$$\bar{a}_k = \frac{1_j}{N} a_{kj} \tag{6.28}$$

$$\bar{b}_1 = \frac{1_j}{N} b_{1j} \tag{6.29}$$

For any choice of T_h in (6.24), estimates of a'_{kj} and b'_{1j} that minimise the SSE in (6.21) may be found with the raster method. However, nonlinear optimisation is required to estimate the L parameters T_h. Nonlinear optimisation may be avoided altogether if responses y_{h0j} of all pixels are known at temperatures T_h when the illuminance is zero (e.g. by closing the aperture of the camera lens). Following the above derivations and assumptions, these dark responses may be estimated by $\hat{y}_{h0j}$ in (6.30) with $\bar{y}_{h0}$ and l_{h0} in (6.31) and (6.32). Note that y_{h0} is known by (6.33).

$$\hat{y}_{h0j} = 1_h a'_{1j} + a'_{2j} T_h + a'_{3j} T_{\underline{h}} \ln T_{\underline{h}} + b'_{1j} \bar{y}_{h0} \tag{6.30}$$

$$\bar{y}_{h0} = 1_h \bar{a}_1 + \bar{a}_2 T_h + \bar{a}_3 T_{\underline{h}} \ln T_{\underline{h}} + \bar{b}_1 T_{\underline{h}} l_{\underline{h}0} \tag{6.31}$$

$$l_{h0} = \ln(c_1 e^{T_h / T_\Delta}) \tag{6.32}$$

$$\bar{y}_{h0} = \frac{1_j}{N} y_{h0j} \tag{6.33}$$

A comparison of (6.24) and (6.30) suggests that T_h may be eliminated. Subtracting the dark from the light version of actual and estimated responses gives actual and estimated offset-free responses y'_{hij} and $\hat{y}'_{hij}$ in (6.34) and (6.35), where the latter depends on the difference $\bar{y}'_{hi}$ in (6.36) between average light and dark responses with only a gain parameter b'_{1j} per pixel. All offset parameters a'_{kj} in (6.24) are cancelled by subtraction of (6.30). Note that $\bar{y}'_{hi}$ is known because $\bar{y}_{hi}$ and $\bar{y}_{h0}$ in (6.36) are calculated from the actual light and dark responses in (6.22) and (6.33) respectively.

$$y'_{hij} = y_{hij} - 1_i y_{h0j} \tag{6.34}$$

$$\begin{aligned} \hat{y}'_{hij} &= \hat{y}_{hij} - 1_i \hat{y}_{h0j} \\ &= b'_{1j} \bar{y}'_{hi} \end{aligned} \tag{6.35}$$

$$\bar{y}'_{hi} = \bar{y}_{hi} - 1_i \bar{y}_{h0} \tag{6.36}$$

Estimation of the parameters b'_{1j} in (6.35) may be done by minimising the SSE in (6.37) between actual and estimated offset-free responses y'_{hij} and $\hat{y}'_{hij}$. This is easily accomplished with the raster method of Chapter 3. The number of implicit parameters Q for this calibration is given in (6.38), counting offset-free averages $\hat{y}'_{hi}$ and gains b'_{1j} fitted to the LMN offset-free responses and subtracting the degeneracy in (6.27) whereby the average of b'_{1j} over all pixels is one. Assuming N is large then the offset cancellation method involves about $3N$ less parameters than the original calibration.

$$SSE = 1_{hij}(y'_{hij} - \hat{y}'_{hij})^2 \tag{6.37}$$

$$Q = LM + N - 1 \tag{6.38}$$

While the above derivation did not consider the cases where a_1, a_2, a_3 and b_1 in (6.12) do not all vary from pixel to pixel, such a consideration is straightforward with the above results. Any constraints on the offsets a_k do not affect the final model in (6.35) because all offsets are cancelled with or without constraints. Constraining the gain b_1 to remain constant for all pixels results in the model of (6.39) with implicit parameters in (6.40). Because of the degeneracy in (6.27), whereby the average of b'_{1j} is one, there is no need for the estimation of a single parameter. Offset-free averages $\bar{y}'_{hi}$ are still required to determine the residual error. Despite the similarity of nil variation in Chapter 4 to the model in (6.39), the former assumes no offset variation whereas the latter may include offset variation, which is cancelled rather than calibrated.

$$\hat{y}'_{hij} = 1_j \bar{y}'_{hi} \tag{6.39}$$

$$Q = LM \tag{6.40}$$

According to the steady state model in (6.16), the gain b_1 is unlikely to vary from column to column without varying from pixel to pixel because it does not depend on parameters of column transistors. However, this steady state model was simple and did not account for many aspects of transistor behaviour (e.g. finite output resistance in saturation) that may cause a columnwise variation of gain. Furthermore, as Chapter 5 showed, transient effects may lead to a substantial columnwise component in the gain. Inclusion of these effects may be achieved with the raster method. Constraining the gain b_1 so that it may vary only from column to column gives the model in (6.41) with implicit parameters in (6.42). The array index j above has been decoded into row and column indices j_1 and j_2, where $1 \leq j_1 \leq N_1$, $1 \leq j_2 \leq N_2$ and $N = N_1 N_2$.

$$\hat{y}'_{hij_1j_2} = 1_{j_1} b'_{j_2} \bar{y}'_{hi} \tag{6.41}$$

$$Q = LM + N_2 - 1 \tag{6.42}$$

An important feature of the offset cancellation method is that the same gain parameters apply for temperature and/or illuminance changes. While estimates of the gain would be more robust against noise if regressed over multiple temperatures and illuminances simultaneously, it is possible to regress over only illuminance changes (or over only temperature changes) when the model is valid. The advantage of this feature is that there is no need to collect calibration data for more than one temperature (or illuminance), which greatly facilitates calibration.

6.3.2 Temperature proxy

Analysis of (6.31) suggests another way to eliminate unknown parameters T_h in (6.24). Equation (6.31) shows that the average dark response $\bar{y}_{h0}$ is a function of temperature with few parameters. If this function is invertible then T_h in (6.24) may be substituted with a function of $\bar{y}_{h0}$. Unfortunately, the function in (6.31) may not be inverted because of the nonlinearity $T_{\underline{h}} \ln T_{\underline{h}}$. Furthermore, even if it could be inverted, the same nonlinearity in (6.24) means that some, if not all, unknown parameters $\bar{a}_k$ and $\bar{b}_1$ would appear upon substitution as parameters that require nonlinear optimisation.

The only feasible way to eliminate both T_h in (6.24) and nonlinear optimisation of unknowns in (6.31) is to linearise the $T_{\underline{h}} \ln T_{\underline{h}}$ term around an operating point, say the average temperature $\bar{T}$. Analysis of (6.31) and (6.32) reveals that $\bar{y}_{h0}$ is also a function of T_h^2, which must also be linearised to avoid nonlinear optimisation. First order Taylor expansions of these two nonlinear functions, around a reference temperature $\bar{T}$, are given in (6.43) and (6.44) (after simplification).

$$T_{\underline{h}} \ln T_{\underline{h}} \approx T_h (1 + \ln \bar{T}) - 1_h \bar{T} \tag{6.43}$$

$$T_h^2 \approx T_h 2\bar{T} - 1_h \bar{T}^2 \tag{6.44}$$

For a reference temperature of $30°\mathrm{C}$, the worst case error for the linearisations in (6.43) and (6.44) in a 0–$60°\mathrm{C}$ range is 0.1% and 1.2% respectively (note that calculations are done in Kelvins). While these errors suggest that linearisation is worth trying, they do not indicate the relative error introduced into (6.24) or (6.31) as this would also depend on other parameters. The validity of the linearisations may be tested by fitting the average dark response $\bar{y}_{h0}$ to measured temperatures T_h for a linear model and the complete model, as well as the complete model minus either the $T_{\underline{h}} \ln T_{\underline{h}}$ or the

T_h^2 term, and comparing the residual error and parameter uncertainties. Although the relationship between the dark response of *any* pixel and temperature is identical to the relationship between the average dark response of *all* pixels and temperature, taking the average dark response as a proxy for temperature is more robust than taking the dark response of any pixel (or the average dark response of a subset of pixels) because it minimises the effect of stochastic error ϵ_{hij} when the number of pixels N is large.

If the linearisations in (6.43) and (6.44) are valid then solving for T_h as a function of $\bar{y}_{h0}$ in (6.31), substituting the result in (6.24) and simplifying gives the model in (6.45) of estimated responses $\hat{y}_{hij}$ in terms of the average dark and light responses $\bar{y}_{h0}$ and $\bar{y}_{hi}$, where a''_{lj} and b'_{1j} are given by (6.46) and (6.27).

$$\hat{y}_{hij} = 1_{hi} a''_{1j} + 1_i a''_{2j} \bar{y}_{h0} + b'_{1j} \bar{y}_{hi} \tag{6.45}$$

$$a''_{lj} = \frac{d_{kl} a'_{kj}}{d_{11}} \tag{6.46}$$

Coefficients d_{kl} in (6.46) depend on the unknowns $\bar{a}_k$, $\bar{b}_1$ and T_Δ and the operating temperature $\bar{T}$, which may also be considered unknown with no loss of generality. The coefficients are given in (6.47), using the notation of Chapter 2, i.e. Section 2.2.1, to identify indices k and l with the rows and columns of a matrix respectively.

$$d_{kl} ()_{kl} = \begin{pmatrix} \bar{a}_2 + \bar{a}_3(1 + \ln \bar{T}) + \bar{b}_1(\ln c_1 + 2\bar{T}/T_\Delta) & 0 \\ -\bar{a}_1 + \bar{a}_3 \bar{T} + \bar{b}_1 \bar{T}^2/T_\Delta & 1 \\ -\bar{a}_1(1 + \ln \bar{T}) - \bar{a}_2 \bar{T} - \bar{b}_1 \bar{T}(\ln c_1 + (1 - \ln \bar{T})\bar{T}/T_\Delta) & 1 + \ln \bar{T} \end{pmatrix} \tag{6.47}$$

Estimation of the parameters a''_{lj} and b'_{1j} in (6.45) may be accomplished by minimising the SSE in (6.21) between actual and estimated responses y_{hij} and $\hat{y}_{hij}$. A solution may be found using the raster method of Chapter 3. The number of implicit parameters Q for this calibration is given in (6.48), which accounts for the average light responses $\bar{y}_{hi}$, offsets a''_{lj} and gains b'_{1j} minus degeneracies on the offsets and gains. The averages of a''_{1j} and a''_{2j} over all pixels are zero because they are linear functions of a'_{kj} in (6.46), which have zero averages due to (6.26). The average of b'_{1j} over all pixels is one because of (6.27). The average dark responses $\bar{y}_{h0}$ are not counted as implicit parameters in (6.48) because they are not determined from the light responses y_{hij}, which are used to calculate the SSE, residual error and parameter uncertainties.

$$Q = LM + 3N - 3 \tag{6.48}$$

Whereas the above derivation assumed a variation of a_k and b_1 in (6.12) from pixel to pixel, it is not difficult to apply the temperature proxy method for models where parameters do not vary at all or only vary from column to column. Such constraints may be due to steady state effects, e.g. the division of transistors between pixel, column and output circuits in Figure 4.1 of Chapter 4, or due to the accommodation of transient effects, e.g. when the response of the second stage readout does not settle from a discharged or precharged state as in Figure 5.7 of Chapter 5. There are many possible constrained models, which may be divided into two categories—ones where there is

Table 6.1: Estimated response $\hat{y}_{hij}$ of the j^{th} logarithmic pixel, in terms of the average dark and light responses $\bar{y}_{h0}$ and $\bar{y}_{hi}$ of all pixels, to temperature T_h and illuminance x_i for the three feasible constrained models of FPN without columnwise variation. The number of implicit parameters Q is given for each model.

Mod.	$\hat{y}_{hij}$	Q
1	$1_{hi}a''_{1j} + 1_i a''_{2j}\bar{y}_{h0} + 1_j\bar{y}_{hi}$	$LM + 2N - 2$
2	$1_{hi}a''_{1j} + 1_j\bar{y}_{hi}$	$LM + N - 1$
3	$1_j\bar{y}_{hi}$	LM

no columnwise variation, in which case a single variable j suffices to index pixels, and ones where there is some columnwise variation, in which case two variables j_1 and j_2 are necessary to index pixels along rows and columns. The importance of comparing constrained to unconstrained models is that the former, when valid, reduce the number of parameters in the calibration and lead to lower parameter uncertainties. Furthermore, the success or failure of particular constrained models compared to the unconstrained version gives information about the nature of FPN in an image sensor.

Logical considerations limit the number of feasible constrained models. Observe that offsets a''_{lj} of the temperature proxy model depend on intermediate offsets a'_{kj} in (6.46), which in turn depend on original gains b_{1j} in (6.26). As the gains b'_{1j} of the temperature proxy model also depend on b_{1j} in (6.27), it is not possible, in general, for a''_{lj} to vary less than b'_{1j} varies (i.e. from pixel to pixel, column to column or not at all). Similarly, as offsets a''_{1j} depend on a_{1j}, a_{2j} and a_{3j} and offsets a''_{2j} depend on a_{2j} and a_{3j}, because of (6.26), (6.46) and the zero in (6.47), it is not possible, in general, for a''_{1j} to vary less than a''_{2j} varies due to an underlying variation of a_{1j}, a_{2j}, a_{3j} or a combination thereof.

Table 6.1 gives the three feasible constrained models that arise when there is no columnwise variation and Table 6.2 gives the six feasible constrained models that arise when there is columnwise variation. The tables also give the number of implicit parameters Q in each constrained model, accounting for degeneracies. Model 2 in Table 6.1 and Model 6 in Table 6.2 consider cases where a''_2 does not vary from pixel to pixel. Because of the degeneracy that the average of this offset equals zero, there is no need to estimate the parameter and the resulting models do not depend on $\bar{y}_{h0}$. Similarly, Model 3 in Table 6.1 considers the case where a''_1 and a''_2 do not vary, in which case both offsets are zero. Models 1–3 of Table 6.1 and Models 4–6 of Table 6.2 consider cases where b'_1 does not vary, in which case the gain equals one because of a degeneracy. All these models may be calibrated with the raster method.

While there are ten possible models for the temperature proxy method, three of them are the principal ones—the unconstrained model in (6.45) and the first models of Tables 6.1 and 6.2. The first model in each table is the least constrained of the lot. If it is incompatible with the data, determined by comparing the residual error of calibration between constrained and unconstrained versions as in Chapter 3, then all other models in the same table will also be incompatible. If it is compatible then the other models

Table 6.2: Estimated response $\hat{y}_{hij_1j_2}$ of the $(j_1, j_2)^{\mathrm{th}}$ logarithmic pixel, in terms of the average dark and light responses $\bar{y}_{h0}$ and $\bar{y}_{hi}$ of all pixels, to temperature T_h and illuminance x_i for the six feasible constrained models of FPN with columnwise variation. The number of implicit parameters Q is given for each model.

Mod.	$\hat{y}_{hij_1j_2}$	Q
1	$1_{hi}a''_{1j_1j_2} + 1_i a''_{2j_1j_2}\bar{y}_{h0} + 1_{j_1}b'_{1j_2}\bar{y}_{hi}$	$LM + 2N + N_2 - 3$
2	$1_{hi}a''_{1j_1j_2} + 1_{ij_1}a''_{2j_2}\bar{y}_{h0} + 1_{j_1}b'_{1j_2}\bar{y}_{hi}$	$LM + N + 2N_2 - 3$
3	$1_{hij_1}a''_{1j_2} + 1_{ij_1}a''_{2j_2}\bar{y}_{h0} + 1_{j_1}b'_{1j_2}\bar{y}_{hi}$	$LM + 3N_2 - 3$
4	$1_{hi}a''_{1j_1j_2} + 1_{ij_1}a''_{2j_2}\bar{y}_{h0} + 1_{j_1j_2}\bar{y}_{hi}$	$LM + N + N_2 - 2$
5	$1_{hij_1}a''_{1j_2} + 1_{ij_1}a''_{2j_2}\bar{y}_{h0} + 1_{j_1j_2}\bar{y}_{hi}$	$LM + 2N_2 - 2$
6	$1_{hij_1}a''_{1j_2} + 1_{j_1j_2}\bar{y}_{hi}$	$LM + N_2 - 1$

require testing to determine if there is a more specific model that is still compatible.

Lastly, note that Models 2 and 3 of Table 6.1 and Model 6 of Table 6.2 do not have more than one offset or gain term, because the models do not include the average dark response $\bar{y}_{h0}$. Although estimates would be more robust against noise if calibrated over multiple temperatures and illuminances simulataneously, the models may be calibrated with data taken at only one temperature for multiple illuminances. Furthermore, the dark response need not be imaged. Unlike with the offset cancellation method, where a similar situation exists, the simplification discussed here relies on specific constraints on the offsets and gain to hold. As Models 2 and 3 of Table 6.1 may be calibrated at one temperature with no consideration of the dark response, they are analogous to the single and nil variation models of Chapter 4. The only difference is the insight in this chapter that the models are valid over multiple temperatures when specific constraints hold on the physical parameters of Section 6.2.

6.4 Simulations

The circuit in Figure 4.1 of Chapter 4 was simulated using Spectre in Cadence for a $0.35\mu\mathrm{m}$ $3.3\mathrm{V}$ AMS process, described in Chapter 1. All transistors were set to a gate width of $1\mu\mathrm{m}$, which is the minimum width of the drain and source diffusions due to contact design rules [45]. The gate length of all transistors was set to $0.6\mu\mathrm{m}$, which was the minimum length recommended by AMS for transistors in analogue circuits sensitive to threshold voltage variation [45]. Ignoring optical effects, the pixel stimulus x was represented by an ideal current source, in parallel with the reverse biased diode in the pixel. The ADC was not simulated and, therefore, what was the ADC input voltage was taken as the pixel response y. These introduce minor changes to the physical meaning of parameters in (6.13)–(6.17) but not to the abstract model of Section 6.2 or to the calibration methods of Section 6.3.

To collect the data used for calibration, a simultaneous DC, parametric and Monte Carlo analysis was performed. The DC analysis varied the photocurrent in half-decade

Table 6.3: The residual error $\hat{\sigma}_\epsilon$ and parameter uncertainty $\hat{\sigma}_{\hat{b}'_1}$ for calibration of the offset cancellation model to simulated offset-free responses y'_{hij}. Models 1 and 2 consider unconstrained and constrained cases. Model 3 considers an unconstrained calibration at one temperature and Model 4 reports the residual error with extrapolation.

Mod.	y'_{hij} (or y'_{7ij})	$\hat{\sigma}_\epsilon$	$\hat{\sigma}_{\hat{b}'_1}$
1	$\hat{b}'_{1j}\bar{y}'_{hi}$	.42	.25
2	$1_j\bar{y}'_{hi}$	.70	
3	$\hat{b}'_{1j}\bar{y}'_{7i}$	.36	.76
4	$\hat{b}'_{1j}\bar{y}'_{hi}$	.49	
In	mV	mV	$\frac{\mathrm{mV}}{\mathrm{V}}$

steps from $1\mathrm{pA}$ to $1\mu\mathrm{A}$. The parametric analysis varied the temperature from $0°\mathrm{C}$ to $60°\mathrm{C}$ in $5°\mathrm{C}$ steps. The Monte Carlo analysis repeated the simulation 100 times, each time with different device parameters for each transistor (except the common transistor) according to statistical distributions specified by AMS that simulate device mismatch [44]. The results represent the variation in responses over photocurrent and temperature between 100 randomly selected pixels of a potentially larger image sensor. The fact that some pixels may belong to the same column and have physical parameters in common with each other but not with other pixels was not simulated as it was difficult to model this type of variation in Cadence. Furthermore, transient effects were not simulated so all results model the steady state response.

The simulated responses may be denoted y_{hij}, where h indexes the temperatures T_h with $1 \leq h \leq 13$, i indexes the photocurrents x_i with $1 \leq i \leq 13$ and j indexes the pixels with $1 \leq j \leq 100$. By setting the photocurrent to zero and carrying out the parametric and Monte Carlo analysis as before, the dark responses of the pixels, denoted y_{h0j}, was also simulated over temperature and over the same random selection of device parameters that simulated mismatch.

6.4.1 Offset cancellation

Simulated responses were calibrated using the offset cancellation method of Section 6.3.1. Models 1 and 2 of Table 6.3 give the residual error and parameter uncertainty with this method for cases where the gain may vary from pixel to pixel and where the gain may not vary (estimated parameters may differ from pixel to pixel for the raster problem but uncertainties are the same for each parameter, as shown in Chapter 3). For Model 2, there is neither a gain parameter to estimate nor a parameter uncertainty as the gain precisely equals one. Comparison of the residual errors shows that constraining the gain leads to a worse calibration. As offset-free responses y'_{hij} change on average, over all temperatures and pixels, by $40\mathrm{mV}$ per decade of photocurrent change, a residual error of $0.42\mathrm{mV}$ corresponds to a 2.5% contrast sensitivity.

Figure 6.1 plots the residual error versus temperature and photocurrent of the offset cancellation method with unconstrained gain. The error appears to be independent of

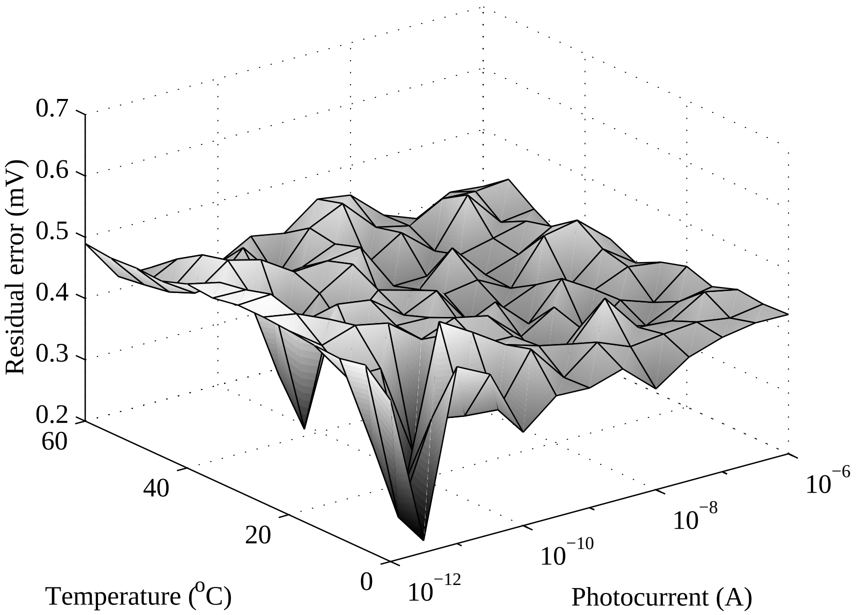


Figure 6.1: The residual error $\hat{\sigma}_{\epsilon_{hi}}$ versus temperature T_h and photocurrent x_i for calibration of the unconstrained offset cancellation model to simulated responses.

temperature and photocurrent over sixty degrees and six decades respectively. Note that the simulation does not include temporal and quantisation noise so these results show how closely the simple model of Section 6.2 matches the complex model of the simulator, in the absence of bias variation.

Section 6.3.1 states that the offset cancellation method may be calibrated at a single temperature because the gain parameters for temperature and illuminance changes are the same as for illuminance changes alone. Model 3 of Table 6.3 gives the residual error and parameter uncertainty when offset-free responses at $30\,^{\circ}\mathrm{C}$, denoted y'_{7ij}, are calibrated. The residual error is slightly less than for Model 1 while the parameter uncertainty is greater, which suggests that the estimated responses overfit the actual responses at the one temperature.

Using the estimated parameters of Model 3 in Table 6.3, the residual error between actual and estimated responses over all temperatures, given in Model 4, is slightly greater than for Model 1. A plot in Figure 6.2 of the residual error versus temperature and photocurrent shows a small degree of overfitting. The error surface drops at $30\,^{\circ}\mathrm{C}$ into a narrow valley whereas at other temperatures the error rises slowly with photocurrent. Nonetheless, the error surface remains relatively flat and random and the residual error of $0.49\mathrm{mV}$ corresponds to a 2.8% contrast sensitivity over all temperatures and illuminances despite calibration at only one temperature.

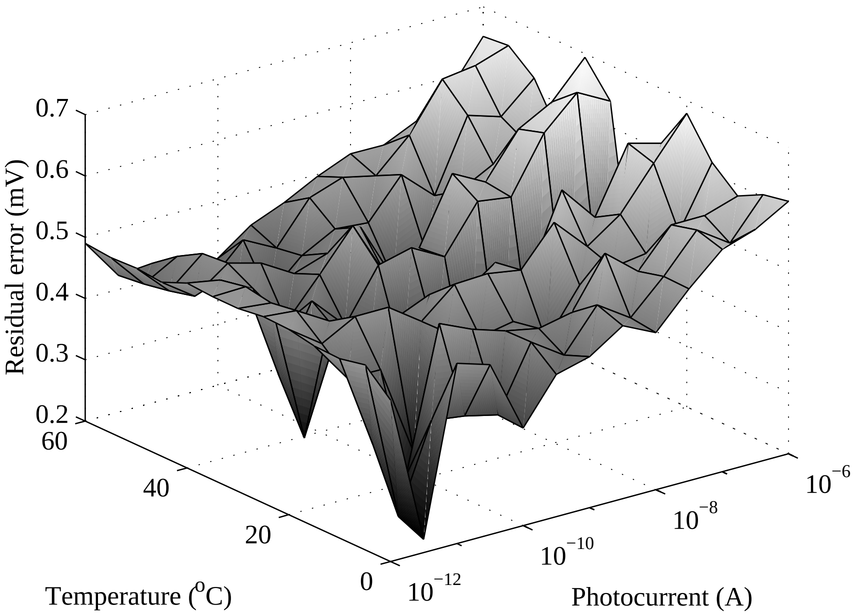


Figure 6.2: The residual error $\hat{\sigma}_{\epsilon_{hi}}$ versus temperature T_h and photocurrent x_i for calibration of the unconstrained offset cancellation model to simulated responses at $30°$C with subsequent extrapolation to all temperatures.

6.4.2 Temperature proxy

The temperature proxy method of Section 6.3.2 requires the average dark response $\bar{y}_{h0}$ to be well modelled by a linear function of temperature T_h, A more complete model includes $T_{\underline{h}} \ln T_{\underline{h}}$ and T_h^2 terms. Table 6.4 gives the residual error for calibrating $\bar{y}_{h0}$ to complete, logarithmic, quadratic and linear models of T_h. The logarithmic and quadratic models equal the complete model without the T_h^2 or $T_{\underline{h}} \ln T_{\underline{h}}$ term respectively. Comparing the residual errors shows that the complete model gives the best result with the linear model having more than double the error. The quadratic model is second best but hardly better than the logarithmic model. However, in the process of calibration, MATLAB warned of an ill-conditioned matrix inversion with the complete model so the quadratic model is the best well-conditioned result.

Table 6.5 gives the estimated parameters and their uncertainties for each model in Table 6.4. The sign inversions for some parameters between the complete and logarithmic models occur because parameters adjust to accommodate a loss of complexity. The nature of parameter adjustment may be deduced with Taylor expansions of the $T_{\underline{h}} \ln T_{\underline{h}}$ and T_h^2 terms over the average temperature $\bar{T}$. A comparison of parameter uncertainties shows that the complete model has high uncertainties for all parameters. The logarithmic and quadratic models have 26% and 25% uncertainties for the $T_{\underline{h}} \ln T_{\underline{h}}$ and T_h^2 coefficients respectively (the logarithmic model also has a high uncertainty for

Table 6.4: The residual error $\hat{\sigma}_\epsilon$ when the simulated average dark response $\bar{y}_{h0}$ is calibrated to complete, logarithmic, quadratic and linear models of temperature T_h.

Mod.	$\bar{y}_{h0}$	$\hat{\sigma}_\epsilon$
1	$1_h\hat{a}_1 + \hat{a}_2 T_h + \hat{a}_3 T_{\underline{h}} \ln T_{\underline{h}} + \hat{b}_1 T_h^2$	.050
2	$1_h\hat{a}_1 + \hat{a}_2 T_h + \hat{a}_3 T_{\underline{h}} \ln T_{\underline{h}}$	.093
3	$1_h\hat{a}_1 + \hat{a}_2 T_h + \hat{b}_1 T_h^2$	.091
4	$1_h\hat{a}_1 + \hat{a}_2 T_h$	.14
In	V	mV

Table 6.5: The parameter values $\hat{a}_k$ and $\hat{b}_1$ and uncertainties $\hat{\sigma}_{\hat{a}_k}$ and $\hat{\sigma}_{\hat{b}_1}$ when the simulated average dark response $\bar{y}_{h0}$ is calibrated to the models of Table 6.4.

Mod.	$\hat{a}_1 \pm \hat{\sigma}_{\hat{a}_1}$	$\hat{a}_2 \pm \hat{\sigma}_{\hat{a}_2}$	$\hat{a}_3 \pm \hat{\sigma}_{\hat{a}_3}$	$\hat{b}_1 \pm \hat{\sigma}_{\hat{b}_1}$
1	3.1 ± 7.4	-42 ± 21	7.7 ± 20	-13 ± 19
2	$1.9 \pm .79$	2.7 ± 13	$-.20 \pm 26$	
3	$2.0 \pm .38$	1.5 ± 3.2		$-.33 \pm 25$
4	$2.0 \pm .032$	$1.3 \pm .16$		
In	$\mathrm{V} \pm \%$	$\frac{\mathrm{mV}}{\mathrm{K}} \pm \%$	$\frac{\mathrm{mV}}{\mathrm{K}} \pm \%$	$\frac{\mu\mathrm{V}}{\mathrm{K}^2} \pm \%$

its T_h coefficient). The linear model is the only one where all parameter estimates are reliable. Compared to the quadratic model, the linear model has order of magnitude lower uncertainties for corresponding parameters despite having only one less degree of freedom. Although the quadratic model has 35% less residual error, linearisation proves to be a robust assumption over a 0–60°C temperature range.

Figure 6.3 plots the residual error as a function of temperature for the quadratic and linear models of the average dark response, which shows that the primary advantage of the quadratic model is at high temperatures. Section 6.3.2 noted that the quadratic term arises from the exponential dependence of the photodiode leakage current on temperature. According to the figure, this dependence becomes significant at about $60^\circ\mathrm{C}$.

Given that linearisation of the $T_{\underline{h}} \ln T_{\underline{h}}$ and T_h^2 terms is reasonable, Table 6.6 goes on to examine the temperature proxy method where light responses y_{hij} are calibrated in terms of the average dark and light responses $\bar{y}_{h0}$ and $\bar{y}_{hi}$. Model 1 of the table gives the residual error and parameter uncertainties for the unconstrained case. Model 2 gives the same for the case where the gain b'_1 is constrained so that it may not vary from pixel to pixel, in which case it equals one according to Section 6.3.2. A comparison of residual errors between Models 1 and 2 shows that the latter overconstrains the calibration. There is no need to test constraints on the other parameters as they may not vary less than the gain varies, according to Section 6.3.2, and Table 6.6 shows that constraining the gain is incorrect. Nonetheless, other possibilities were tested but none of them improve on or compare to Model 1. As actual responses y_{hij} change on

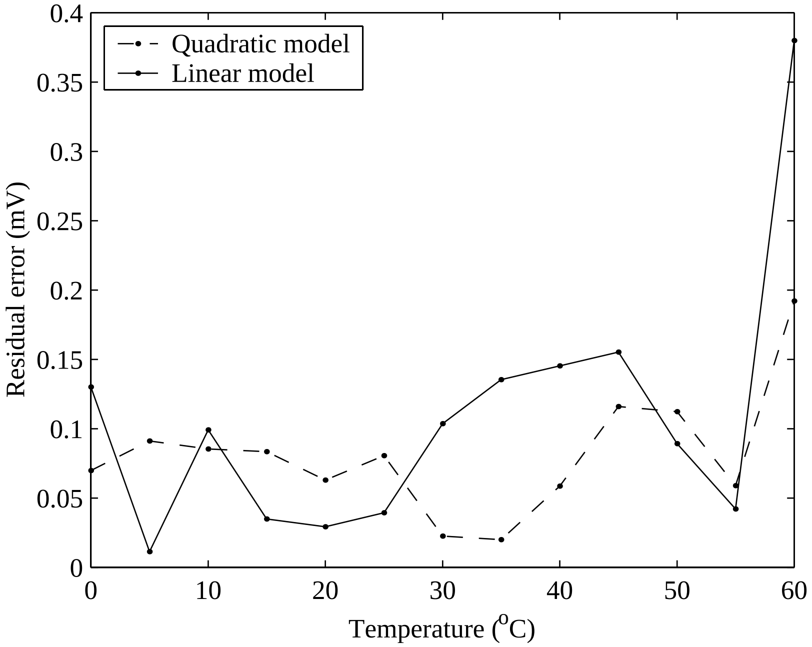


Figure 6.3: The residual error $\hat{\sigma}_{\epsilon_h}$ versus temperature T_h for calibration of the simulated average dark response $\bar{y}_{h0}$ to quadratic and linear models of temperature.

average, over all temperatures and pixels, by $40\mathrm{mV}$ per decade of photocurrent change, a residual error of $0.29\mathrm{mV}$ corresponds to a 1.7% contrast sensitivity.

Figure 6.4 plots the residual error as a function of temperature and illuminance for the unconstrained temperature proxy method. The error appears to be independent of both variables and to vary randomly, as in Figure 6.1. Comparing the offset cancellation method to the temperature proxy method, a natural question arises as to why the residual error of Model 1 in Table 6.3 proves to be 1.4 times greater than the residual error of Model 1 in Table 6.6, especially since offset cancellation requires no linearisation of $T_{\underline{h}} \ln T_{\underline{h}}$ and T_h^2 terms whereas temperature proxy does. The natural answer is that the former method calibrates the difference of two equally noisy measurements—the light and dark responses—whereas the latter method calibrates only one noisy measurement—the light response. The noise in the average dark response, which contributes some error to the temperature proxy method, is small due to averaging. Assuming the stochastic errors in the light and dark responses of each pixel are statistically independent then the stochastic error in the difference should be $\sqrt{2}$ or 1.4 times greater. This explanation accounts for the discrepancy of residual errors between the offset cancellation and temperature proxy methods.

According to Section 6.3.2, constraining the offset a_2'' in the temperature proxy method, which is the coefficient of the average dark response $\bar{y}_{h0}$, so that it may not

Table 6.6: The residual error $\hat{\sigma}_\epsilon$ and parameter uncertainties $\hat{\sigma}_{\hat{a}''_l}$ and $\hat{\sigma}_{\hat{b}'_1}$ for calibration of the temperature proxy model to simulated responses y_{hij}. Models 1 and 2 consider unconstrained and constrained cases. Model 3 considers a constrained calibration at one temperature and Model 4 reports the residual error with extrapolation.

Mod.	y_{hij} (or y_{7ij})	$\hat{\sigma}_\epsilon$	$\hat{\sigma}_{\hat{a}''_1}$	$\hat{\sigma}_{\hat{a}''_2}$	$\hat{\sigma}_{\hat{b}'_1}$
1	$1_{hi}\hat{a}''_{1j} + 1_i\hat{a}''_{2j}\bar{y}_{h0} + \hat{b}'_{1j}\bar{y}_{hi}$	.29	2.2	.93	.29
2	$1_{hi}\hat{a}''_{1j} + 1_i\hat{a}''_{2j}\bar{y}_{h0} + 1_j\bar{y}_{hi}$	.44	3.2	1.4	
3	$1_i\hat{a}''_{1j} + \hat{b}'_{1j}\bar{y}_{7i}$	.29	2.4		1.1
4	$1_{hi}\hat{a}''_{1j} + \hat{b}'_{1j}\bar{y}_{hi}$	.49			
In	V	mV	mV	$\frac{\mathrm{mV}}{\mathrm{V}}$	$\frac{\mathrm{mV}}{\mathrm{V}}$

vary from pixel to pixel means that a''_2 equals zero and responses do not depend on $\bar{y}_{h0}$. When this constraint is valid then the temperature proxy method may be calibrated with responses measured at only one temperature. This constraint is not valid because, according to Section 6.3.2, offset parameters may not vary less than the gain parameter and Table 6.6 showed that constraining the gain is incorrect. However, for the sake of illustration, Model 3 of the table considers the case where the offset a''_2 is constrained to be constant, in which case it equals zero, but the gain b'_1 is unconstrained. Table 6.6 gives the residual error and parameter uncertainties when this model is calibrated with light responses at 30°C only, denoted y_{7ij}. The residual errors of Models 1 and 3 are comparable although parameter uncertainties are greater with the latter, particularly for the gain. Residual errors are comparable because parameters in Model 3 easily accommodate for the loss of complexity when there is only one temperature to consider. Indeed, Model 3 is similar to the double variation model of Chapter 4, in which the single offset parameter included temperature terms.

When parameters estimated at one temperature for Model 3 of Table 6.6 are used to estimate the responses for all temperatures, the residual error between the actual and estimated responses, given in Model 4, is much higher than before. Thus, the best linear model at one temperature does not extrapolate over multiple temperatures. Figure 6.5 plots the residual error for this model as a function of temperature and photocurrent, showing that the error is strongly dependent on temperature, with a minimum at the temperature of calibration, but independent of photocurrent. Nonetheless, there is no difference overall between the offset cancellation and temperature proxy method for the given ranges of temperature and photocurrent, as seen in the residual errors of Model 4 in Tables 6.3 and 6.6. Both methods lead to a 2.8% contrast sensitivity, when extrapolated from a calibration at one temperature. Although Figure 6.2 suggests better extrapolation than does Figure 6.5, the noise floor is higher for offset cancellation.

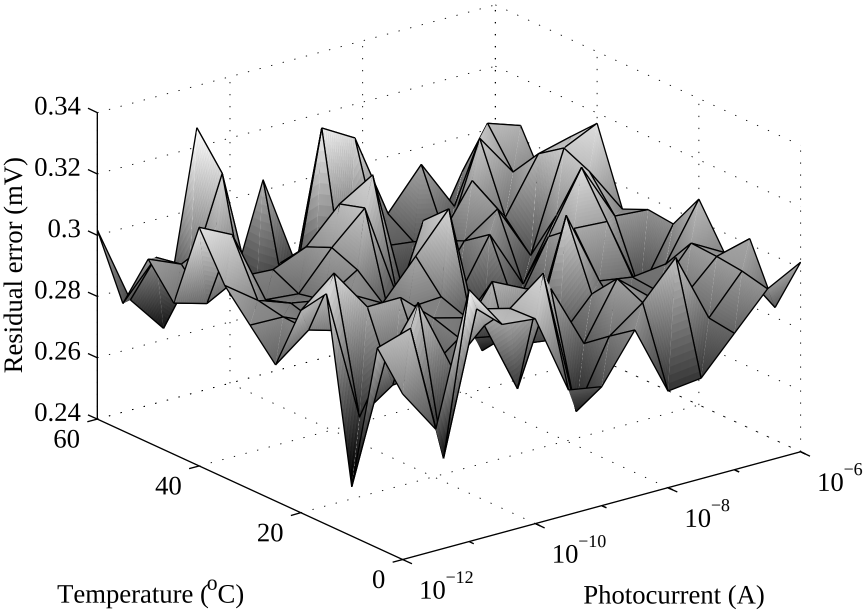


Figure 6.4: The residual error $\hat{\sigma}_{\epsilon_{hi}}$ versus temperature T_h and photocurrent x_i for calibration of the unconstrained temperature proxy model to simulated responses.

6.5 Experiments

To test the proposed calibration techniques on experimental data, images were acquired with a Fuga 15RGB logarithmic sensor. Although the sensor is a colour imager, the results were treated as if they come from a monochromatic camera, as in Chapter 4 (Chapter 7 considers the modelling and calibration of colour logarithmic image sensors). The Fuga 15RGB sensor has a 512×512 array of pixels (i.e. $N = 512^2$). The camera was placed in an oven together with a $2850\mathrm{lm}$ compact fluorescent lamp. Calibration data were collected by imaging a uniformly illuminated sheet of white paper also in the oven. Unfortunately, an oven can only heat the camera and the maximum temperature was limited by the plastic camera housing. The temperature could only be varied from $20°\mathrm{C}$ (room temperature) to $50°\mathrm{C}$.

As the oven's own heating element produced heat too quickly, so that the exterior of the camera would heat up faster than the interior, and as the thermostatic control was unstable at low temperatures, the oven's heating unit was not used. Instead, the insulated interior of the oven was allowed to warm up slowly, at a rate equivalent on average to $3°\mathrm{C}$ per hour, using the $44.5\mathrm{W}$ of power dissipated by the fluorescent lamp. This rate gave plenty of time at each temperature for adjusting the lens aperture to simulate intensity variation of the illuminant. The illuminance of the white paper was measured with a light meter to be $4500\mathrm{lux}$, which did not vary with temperature. The aperture

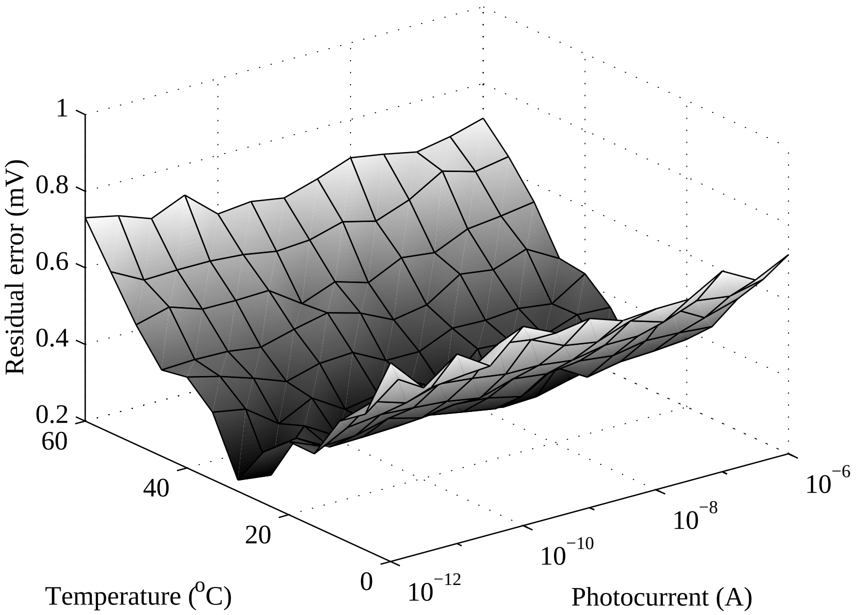


Figure 6.5: The residual error $\hat{\sigma}_{\epsilon_{hi}}$ versus temperature T_h and photocurrent x_i for calibration of a constrained temperature proxy model to simulated responses at $30°\mathrm{C}$ with subsequent extrapolation to all temperatures.

setting was varied from 1.8 to 16 f-stops to simulate seven different illuminances at each temperature. The aperture was also closed and an image was taken of the dark response at each temperature. These images were taken for every $5°\mathrm{C}$ change in the temperature, measured using the oven's digital thermometer. The oven had an internal fan that circulated air to minimise any spatial variation of the interior temperature.

The fluorescent lamp was used to provide ample light without producing too much heat, which would make the oven temperature rise too quickly. However, the light intensity cast by the lamp oscillated at a high frequency, which was recorded by the camera although invisible to the eye. The oscillation manifested as narrow horizontal bands that moved slowly across consecutive images (most likely because the oscillation rate of the lamp and the vertical scan rate of the camera, or their harmonic frequencies, were close). This beating effect, which is a source of error, is reduced by the multiframing process used with the Fuga 15RGB, as described in Chapter 1, because the bands fall in different positions in each frame. The number of captured frames per data frame was nearly doubled, equalling 11, to reduce the beating. The residual error of the multiframing process over the entire range of temperature and illuminance was $3.6\mathrm{LSB}$. Analysis of the residuals showed that the error correlated with the logarithm of illuminance since the band amplitude grew and shrank with aperture variation. Nonetheless, these residuals measure the deviation of captured frames from data frames whereas the

Table 6.7: The residual error $\hat{\sigma}_\epsilon$ and parameter uncertainty $\hat{\sigma}_{\hat{b}'_1}$ for calibration of the offset cancellation model to experimental offset-free responses $y'_{hij_1j_2}$. Models 1–3 consider unconstrained and constrained cases. Model 4 considers an unconstrained calibration at one temperature and Model 5 reports the residual error with extrapolation.

Mod.	$y'_{hij_1j_2}$ (or $y'_{4ij_1j_2}$)	$\hat{\sigma}_\epsilon$	$\hat{\sigma}_{\hat{b}'_1}$
1	$\hat{b}'_{1j_1j_2}\bar{y}'_{hi}$	4.2	6.5
2	$1_{j_1}\hat{b}'_{1j_2}\bar{y}'_{hi}$	12	.85
3	$1_{j_1j_2}\hat{b}'_1\bar{y}'_{hi}$	15	.050
4	$\hat{b}'_{1j_1j_2}\bar{y}'_{4i}$	4.0	17
5	$\hat{b}'_{1j_1j_2}\bar{y}'_{hi}$	4.2	
In	LSB	LSB	$\frac{\mathrm{mLSB}}{\mathrm{LSB}}$

error in the latter is about $\sqrt{11}$ times less due to averaging.

As described in Chapter 5, the Fuga 15RGB exhibits a columnwise pattern in its response due to transient effects. Whereas the simulation in Section 6.4 did not consider such effects, it is impossible to avoid them in the experiment. While transient effects may be calibrated to some degree with linear models, they are fundamentally nonlinear especially with certain readout circuits and conditions. In particular, the Fuga 15RGB exhibits a very nonlinear transient response when stimulated with bright illuminances, as in this experiment (and in Chapter 5). Fortunately, these effects may be reduced substantially by discarding the first 100 columns of each image, as was done. With these considerations, the pixels are indexed here not by a single variable j, where $1 \leq j \leq 512^2$, but by two variables j_1 and j_2, where $1 \leq j_1 \leq 512$ and $1 \leq j_2 \leq 412$, to designate the row and column coordinates respectively so that parameters may be constrained to vary from pixel to pixel, from column to column or not at all. The pixel responses at the seven temperatures T_h, where $1 \leq h \leq 7$, and seven illuminances x_i, where $1 \leq i \leq 7$, are denoted $y_{hij_1j_2}$. Similarly, the dark responses of the pixels at temperatures T_h are denoted $y_{h0j_1j_2}$.

6.5.1 Offset cancellation

Table 6.7 reports the residual errors and parameter uncertainties when the experimental data is calibrated for several versions of the offset cancellation method, described in Section 6.3.1. Models 1–3 consider cases where the gain b'_1 may vary from pixel to pixel, from column to column or not at all. A comparison of the residual errors shows that constraining the gain in any way leads to much worse results, indicating there is substantial parameter variation within and across columns. As offset-free responses $y'_{hij_1j_2}$ change on average, over all temperatures and pixels, by $43\mathrm{LSB}$ per decade of illuminance change, a $4.2\mathrm{LSB}$ residual error corresponds to a 25% contrast sensitivity. This is much worse than the 1–10% contrast sensitivity of the human eye [30].

Figure 6.6 plots the residual error as a function of temperature and illuminance for

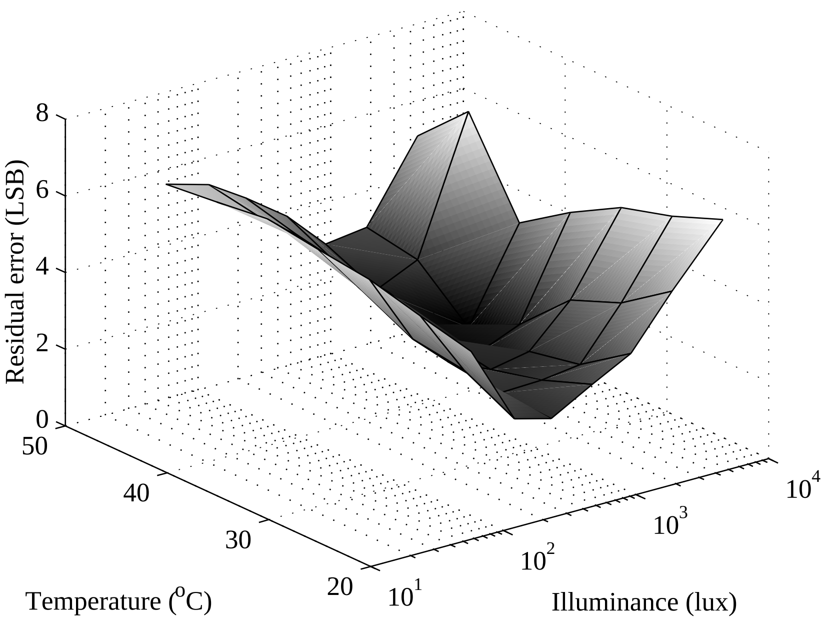


Figure 6.6: The residual error $\hat{\sigma}_{\epsilon_{hi}}$ versus temperature T_h and illuminance x_i for calibration of the unconstrained offset cancellation model to experimental responses.

the unconstrained offset cancellation method, i.e. Model 1 of Table 6.7. Unlike the simulation result in Figure 6.1, the error is strongly dependent on illuminance, with a minimum at the mid-range, and weakly dependent on temperature. The reason for the failure of the offset cancellation method with the experiment is that the Fuga 15RGB exhibits bias variation, as shown in Chapter 4, unlike the simulated circuit. While cancellation of offsets would occur, a subtraction of dark from light responses would contain bias (in addition to gain) variation. Bias variation must be reduced with better process technology and circuit design for offset cancellation to be practical.

Models 4 and 5 of Table 6.7 consider the case when the offset cancellation method is calibrated at one temperature and then tested over all temperatures. Model 4 gives the residual error and parameter uncertainty when offset-free responses at $35\,^\circ\mathrm{C}$, denoted $y'_{4ij_1j_2}$, are calibrated. Model 5 gives the residual error between actual and estimated offset-free responses at all temperatures, using the parameters estimated at the one temperature. Comparing the residual errors of Models 4 and 5 of Table 6.7 to Model 1 shows little improvement for a calibration at one temperature (which involves less data) and no worsening for an extrapolation to all temperatures. Comparing Figure 6.6 to 6.7, which plots the residual error of the extrapolated model versus temperature and illuminance, shows that extrapolation causes no degradation. Thus, temperature dependence does not limit offset cancellation although bias variation does.

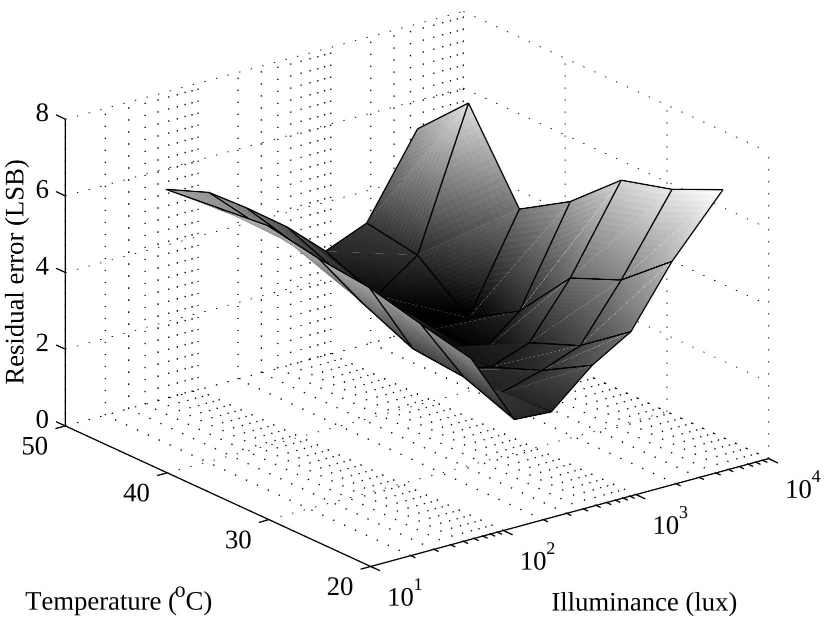


Figure 6.7: The residual error $\hat{\sigma}_{\epsilon_{hi}}$ versus temperature T_h and illuminance x_i for calibration of the unconstrained offset cancellation model to experimental responses at $35°$C with subsequent extrapolation to all temperatures.

6.5.2 Temperature proxy

The temperature proxy method of Section 6.3.2 requires the average dark response $\bar{y}_{h0}$ to be well approximated by a linear function of temperature T_h. As in Section 6.4.2, Table 6.8 considers a calibration of the average dark response to complete, logarithmic, quadratic and linear functions of temperature in Models 1–4 respectively. The residual error of the complete model is about half that of the logarithmic and quadratic models, which have equal results. The residual error of the linear model is about six times that of the complete model and three times that of the other two models. However, MATLAB warned of ill-conditioned matrix inversions in the process of calibrating the complete and quadratic models so the best well-conditioned result is that of the logarithmic model. Ill-conditioned matrix inversions occur when some parameters in a model are nearly redundant with respect to the calibration data.

Table 6.9 gives the estimated parameters and their uncertainties for each model in Table 6.8. The parameter uncertainties corroborate the MATLAB warnings, seeing as the four parameters of the complete model have 26–27$\%$ uncertainties and the three parameters of the quadratic model have 15–65$\%$ uncertainties. The well-conditioned logarithmic model is not much better as its three parameters have 15–24$\%$ uncertainties. With uncertainties of 1.7 and 2.8%, the linear model is the only one with reliable parameters. Having only one less degree of freedom, its parameter uncertainties are an

Table 6.8: The residual error $\hat{\sigma}_\epsilon$ when the experimental average dark response $\bar{y}_{h0}$ is calibrated to complete, logarithmic, quadratic and linear models of temperature T_h.

Mod.	$\bar{y}_{h0}$	$\hat{\sigma}_\epsilon$
1	$1_h\hat{a}_1 + \hat{a}_2T_h + \hat{a}_3T_{\underline{h}} \ln T_{\underline{h}} + \hat{b}_1T_h^2$	.054
2	$1_h\hat{a}_1 + \hat{a}_2T_h + \hat{a}_3T_{\underline{h}} \ln T_{\underline{h}}$	.11
3	$1_h\hat{a}_1 + \hat{a}_2T_h + \hat{b}_1T_h^2$	.11
4	$1_h\hat{a}_1 + \hat{a}_2T_h$	.34
In	LSB	LSB

Table 6.9: The parameter values $\hat{a}_k$ and $\hat{b}_1$ and uncertainties $\hat{\sigma}_{\hat{a}_k}$ and $\hat{\sigma}_{\hat{b}_1}$ when the experimental average dark response $\bar{y}_{h0}$ is calibrated to the models of Table 6.8.

Mod.	$\hat{a}_1 \pm \hat{\sigma}_{\hat{a}_1}$	$\hat{a}_2 \pm \hat{\sigma}_{\hat{a}_2}$	$\hat{a}_3 \pm \hat{\sigma}_{\hat{a}_3}$	$\hat{b}_1 \pm \hat{\sigma}_{\hat{b}_1}$
1	$-10,000 \pm 26$	370 ± 26	-64 ± 26	100 ± 27
2	-370 ± 24	13 ± 16	-2 ± 15	
3	-72 ± 65	1.5 ± 20		-3.2 ± 15
4	230 ± 1.7	$-.46 \pm 2.8$		
In	LSB $\pm$ %	$\frac{\text{LSB}}{\text{K}} \pm \%$	$\frac{\text{LSB}}{\text{K}} \pm \%$	$\frac{\text{mLSB}}{\text{K}} \pm \%$

order of magnitude better than corresponding ones of the logarithmic model. Therefore, linearisation proves to be a reasonable, even compelling, assumption.

Because the simulation and experiment involve different technologies (i.e. $0.35\mu\text{m}$ 3.3V and $0.7\mu\text{m}$ 5V respectively), design choices (e.g. device sizes) and other factors (e.g. optical and ADC effects), parameters in Tables 6.5 and 6.9 may not be readily compared. One exception is the sign of the estimated temperature coefficient $\hat{a}_2$ for the linear model, which is positive in simulation but negative in experiment. Higher photocurrents lead to lower voltages in the simulation, due to the inverting load in Figure 4.1 of Chapter 4, whereas higher illuminances lead to higher integers in the experiment, which means the ADC gain of the Fuga 15RGB is negative.

Figure 6.8 plots the residual error versus temperature for the logarithmic and linear models in Table 6.8 of the average dark response in terms of temperature. The errors of the logarithmic and linear models do not exceed 0.17 and 0.54LSB respectively in this range. Compared to the simulation result in Figure 6.3, there is no marked rise in error for the linear model, which suggests that the exponential dependence of photodiode leakage current on temperature was not dominant in this temperature range. The same reason may explain why the logarithmic model performed better than the quadratic model with the experiment whereas the converse was true with the simulation.

To examine the temperature proxy method further, Table 6.10 reports the residual errors and parameter uncertainties for calibrations of the light responses $y_{hij_1j_2}$ in terms of the average dark and light responses $\bar{y}_{h0}$ and $\bar{y}_{hi}$. Models 1–3 consider the

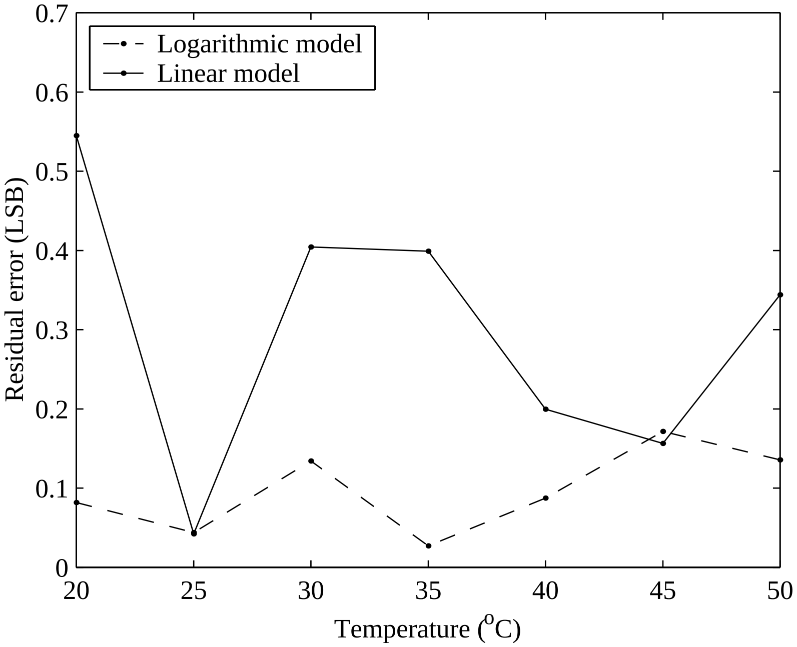


Figure 6.8: The residual error $\hat{\sigma}_{\epsilon_h}$ versus temperature T_h for calibration of the experimental average dark response $\bar{y}_{h0}$ to logarithmic and linear models of temperature.

cases where the gain is permitted to vary from pixel to pixel, column to column or not at all. Examination of the residual errors shows that constraining the gain is incorrect. There is no need to consider cases where offsets a_1'' and/or a_2'' are constrained because these parameters may not vary less than the gain, as argued in Section 6.3.2. Nonetheless, other cases were tested but none improved on or compared to the unconstrained case. As light responses $y_{hij_1j_2}$ changed on average, over all temperatures and pixels, by 43LSB per decade of illuminance change, a residual error of 2.0LSB corresponds to a 12% contrast sensitivity, much better than with offset cancellation.

Figure 6.9 plots the residual error of the unconstrained temperature proxy method, i.e. Model 1 of Table 6.10, versus temperature and illuminance. The error is clearly dependent on illuminance, approximately having a w-shape for any given temperature, and weakly dependent on temperature. This w-shape echoes the shape of the residual error versus illuminance for the double variation model in Chapter 4. Chapter 4 showed that the w-shape arises when responses containing offset, gain and bias variation are calibrated to a model permitting only offset and gain variation. Thus, differences between the simulation result in Figure 6.4 and the experimental result in Figure 6.9 are attributed to the bias variation present in the Fuga 15RGB, as shown in Chapter 4. Another source of deviation is the beating effect of the illumination in the experiment, which causes a tilted w-shape so that the residual error tends to increase with illumi-

Table 6.10: The residual error $\hat{\sigma}_\epsilon$ and parameter uncertainties $\hat{\sigma}_{\hat{a}''_l}$ and $\hat{\sigma}_{\hat{b}'_1}$ for calibration of the temperature proxy model to experimental responses $y_{hij_1j_2}$. Models 1–3 consider unconstrained and constrained cases. Model 4 considers a constrained calibration at one temperature and Model 5 reports the residual error with extrapolation.

Mod.	$y_{hij_1j_2}$ (or $y_{4ij_1j_2}$)	$\hat{\sigma}_\epsilon$	$\hat{\sigma}_{\hat{a}''_1}$	$\hat{\sigma}_{\hat{a}''_2}$	$\hat{\sigma}_{\hat{b}'_1}$
1	$1_{hi}\hat{a}''_{1j_1j_2} + 1_i\hat{a}''_{2j_1j_2}\bar{y}_{h0} + \hat{b}'_{1j_1j_2}\bar{y}_{hi}$	2.0	5.7	79	11
2	$1_{hi}\hat{a}''_{1j_1j_2} + 1_i\hat{a}''_{2j_1j_2}\bar{y}_{h0} + 1_{j_1}\hat{b}'_{1j_2}\bar{y}_{hi}$	2.4	6.2	73	.59
3	$1_{hi}\hat{a}''_{1j_1j_2} + 1_i\hat{a}''_{2j_1j_2}\bar{y}_{h0} + 1_{j_1j_2}\bar{y}_{hi}$	2.6	6.8	80	
4	$1_i\hat{a}''_{1j_1j_2} + \hat{b}'_{1j_1j_2}\bar{y}_{4i}$	2.0	5.3		31
5	$1_{hi}\hat{a}''_{1j_1j_2} + \hat{b}'_{1j_1j_2}\bar{y}_{hi}$	2.4			
In	LSB	LSB	LSB	$\frac{\text{mLSB}}{\text{LSB}}$	$\frac{\text{mLSB}}{\text{LSB}}$

nance in Figure 6.9. A third source of deviation between the simulation and experiment is the nonlinear variation of responses due to the transient response of the Fuga 15RGB, as shown in Chapter 5. Although this has been reduced because the first 100 columns of all images have been discarded, it cannot be eliminated. Following Chapter 5, it is possible to show that the error peak in Figure 6.9 correlates with nonlinearities due to the transient response. In summary, the temperature proxy method accounts for temperature dependence of responses but is limited mainly by bias variation in accounting for illuminance dependence.

A few observations may be made comparing the offset cancellation and temperature proxy methods in terms of the experiment. The residual error with the former method, in Model 1 of Table 6.7, is 2.1 times greater than with the latter method, in Model 1 of Table 6.10. Such a difference may partially be explained, as in Section 6.4.2 for the simulation, by noting that offset cancellation calibrates the difference of two noisy measurements whereas temperature proxy calibrates only one noisy measurement. This explanation accounts for about $\sqrt{2}$ or 1.4 of the error ratio. The remaining discrepancy may be understood in terms of bias variation. Dark and light responses both include bias variation so a difference of the two, as taken with offset cancellation, would exaggerate the nonlinear variation. Furthermore, although the offset cancellation and temperature proxy models do not explicitly consider bias variation, estimated parameters will implicitly accommodate some of the effect. As the latter model involves three times as many parameters per pixel, an accommodation is easier. This observation is supported by the fact that a constraining of the gain in Table 6.7, for the offset cancellation method, causes a much greater increase in residual error than a constraining of the gain in Table 6.10, for the temperature proxy method.

Models 4 and 5 in Table 6.10 consider the case, discussed in Section 6.3.2, when the temperature proxy method is calibrated at one temperature and extrapolated to multiple temperatures. This approach is logical only when the offset a''_2 does not vary from pixel to pixel, in which case it equals zero and responses do not depend on the average dark response, which in turn is logical only when the gain b'_1 does not vary from pixel

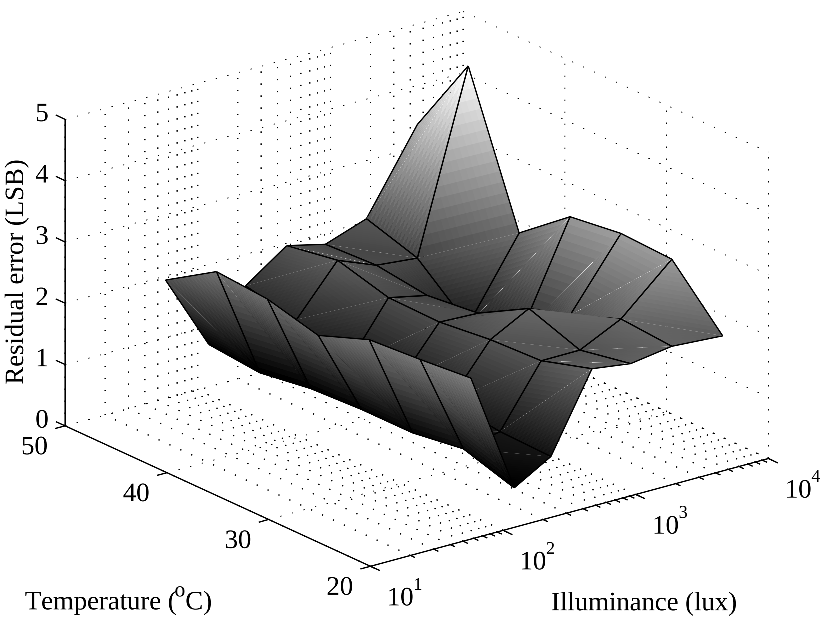


Figure 6.9: The residual error $\hat{\sigma}_{\epsilon_{hi}}$ versus temperature T_h and illuminance x_i for calibration of the unconstrained temperature proxy model to experimental responses.

to pixel. As shown in Models 1 and 3 of Table 6.10, constraining the gain is incorrect. Nonetheless, Model 4 considers the case where a_2'' equals zero but b_1' may vary from pixel to pixel. Because this model corresponds to the double variation model of Chapter 4, offsets a_1'' will accommodate the loss of a_2'' at any one temperature. Indeed, Table 6.10 reports that the residual error and parameter uncertainties for a calibration of light responses at $35°\text{C}$, denoted $y_{4ij_1j_2}$, to Model 4 compares to those results of Model 1. When parameters estimated for Model 4 at one temperature are used to estimate responses at all temperatures, the residual error increases, as given in Model 5.

Figure 6.10 plots the residual error of the temperature proxy method with extrapolation, i.e. Model 5 of Table 6.10, versus temperature and illuminance. Apart from the features in Figure 6.9 that are repeated, Figure 6.10 shows a dependence of residual error on temperature, with a minimum at the temperature of calibration. As the temperature range studied in the experiment is half as much as the range in the simulation, Figure 6.10 does not show the temperature dependence as strongly as does Figure 6.5. The overall residual error of 2.4LSB translates to a 14% contrast sensitivity, which is not much worse than for the temperature proxy method without extrapolation (and the constraint on a_2''). However, the difference would be greater with a wider range because the error in Figure 6.9 is expected to remain independent of temperature whereas the error in Figure 6.10 is expected to exhibit more temperature dependence.

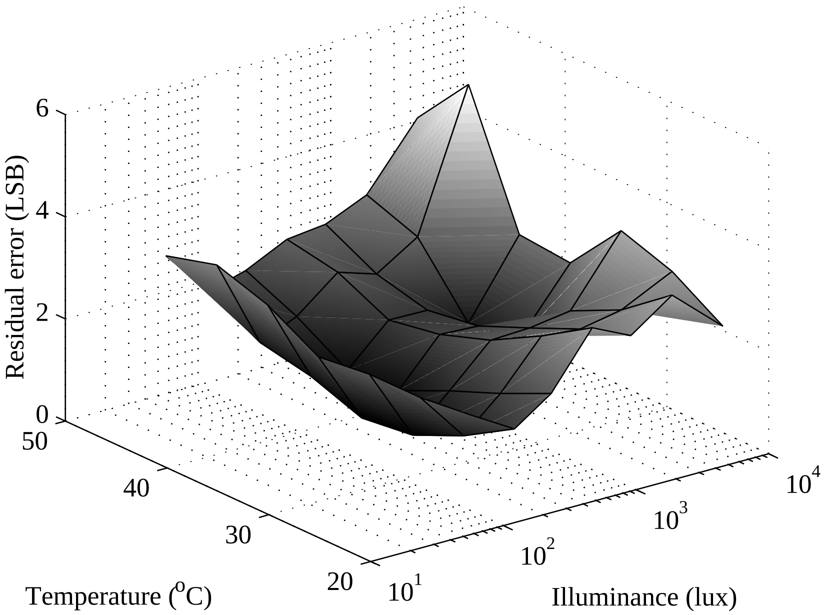


Figure 6.10: The residual error $\hat{\sigma}_{\epsilon_{hi}}$ versus temperature T_h and illuminance x_i for calibration of a constrained temperature proxy model to experimental responses at $35°\mathrm{C}$ with subsequent extrapolation to all temperatures.

6.6 Conclusion

The response of a logarithmic pixel depends on temperature, as well as illuminance, because the threshold voltages, current gains, subthreshold slope, crossover current and leakage current of the circuit depend on temperature. Following semiconductor theory, a model of pixel response y to temperature T and illuminance x is $y = a_1 + a_2T + a_3T\ln T + b_1T\ln(c_1e^{T/T_\Delta} + x) + \epsilon$. A spatial variation of offsets a_k (except a_1), gain b_1, bias c_1 or any combination thereof causes temperature-dependent FPN. However, T_Δ is a process constant and ϵ represents unpredictable error.

This chapter ignored bias variation, which allowed nonlinear optimisation to be avoided. Using the light and dark responses of pixels, i.e. when $x \gg 0$ and $x \approx 0$, models of FPN may be devised that do not require measurement of either temperature or illuminance for calibration. In the offset cancellation method, the difference between the light and dark responses of a pixel is calibrated to the average such difference of all pixels for a uniform scene. In the temperature proxy method, which assumes the average dark response is a linear function of temperature, pixel responses are calibrated as linear functions of the average dark and light response of all pixels to a uniform scene. The raster method is used to calibrate unconstrained and constrained models.

Dark and light responses of logarithmic pixels were simulated for an AMS process from $0°\mathrm{C}$ to $60°\mathrm{C}$ over six decades of photocurrent. The unconstrained offset cancellation model was calibrated with a residual error of $.42\mathrm{mV}$, which corresponds to a 2.5% contrast sensitivity. As the average dark response was a linear model of temperature with a residual error of $.14\mathrm{mV}$, the unconstrained temperature proxy model was calibrated with a residual error of $.29\mathrm{mV}$ or a 1.7% contrast sensitivity. With either method, the residual error was independent of temperature and illuminance but constraining of parameters leads to worse results. While the offset cancellation method is simpler, the temperature proxy method works better because it calibrates a single noisy response rather than the difference of two noisy responses.

Experiments were done with a Fuga $15\mathrm{RGB}$ camera, using an insulated oven, a compact fluorescent lamp and aperture settings to create a controlled temperature variation from $20°\mathrm{C}$ to $50°\mathrm{C}$ and to simulate two decades of illuminance variation. The unconstrained offset cancellation model was calibrated with a residual error of $4.2\mathrm{LSB}$ or a 25% contrast sensitivity. As the average dark response was a linear model of temperature with a residual error of $.34\mathrm{LSB}$, the unconstrained temperature proxy model was calibrated with a residual error of $2.0\mathrm{LSB}$ or a 12% contrast sensitivity. With either method, the residual error was independent of temperature but not illuminance and constraining of parameters leads to worse results. The simulation results were better than the experimental ones mainly because the former did not include bias variation unlike the latter. Offset cancellation suffers more than the temperature proxy method because it has fewer parameters to accommodate bias variation.

A calibration of the offset cancellation and temperature proxy models at one temperature with extrapolation to multiple temperatures was also considered. The residual error with such an approach was independent of temperature only for the offset cancellation method because the same parameters applied equally to temperature and/or illuminance changes. While the simulation results were acceptable, the experimental results suffered from bias variation. Extrapolation works with the temperature proxy method only when specific constraints on the parameters are valid, which was neither the case with simulation or experiment.

Chapter 7

Colour rendition

7.1 Introduction

One problem with logarithmic CMOS image sensors is fixed pattern noise (FPN), considered in Chapters 4–6. FPN is a substantial but predictable error that appears in an image due to a steady state variation from pixel to pixel, or a transient variation from column to column, of parameters that relate stimuli to responses. While FPN correction is necessary to make logarithmic sensors useful, the accurate rendition of scenes on display devices by estimation of real world stimuli from pixel responses is also important. Rendition is more important with colour images because the eye is more sensitive to chromatic errors than to intensity errors [30]. Much has been published about colour rendition in linear sensors but little has been written on rendition in colour logarithmic sensors, the subject of this chapter.

Section 7.2 unites colour theory in linear sensors with monochromatic theory in logarithmic sensors to model colour sensation in logarithmic sensors. Section 7.3 describes a procedure to calibrate this model and Section 7.4 outlines a method to render the response of a calibrated sensor into a standard colour space. Section 7.5 demonstrates calibration and rendition with a Fuga 15RGB logarithmic sensor, a colour version of the Fuga 15d developed at IMEC [32], and compares colour rendition of the sensor to that of conventional digital cameras. For simplicity, transient responses and temperature dependences are not considered in this chapter.

7.2 Modelling

A colour image sensor is made by inserting colour filters in the path of light rays before they form an image on a monochromatic sensor [12]. Corresponding to human colour vision, three filters are needed, selective to the red (R), green (G) and blue (B) regions of the spectrum. Multi-sensor imagers use prisms with special coatings to split and filter an image into three images, which are captured by separate sensors and combined to produce a single image. Single-sensor imagers have a pattern of red, green and blue filters overlaid upon pixels. Though each pixel is selective to only one colour, its neighbours are selective to the other two. By interpolating pixel responses, a red, green and blue response may be estimated for each pixel at a small loss of spatial resolution. As multi-sensor imagers obey a similar theory, the rest of this chapter discusses only single-sensor imagers.

A colour filter on a pixel modifies the spectral composition of incident light prior to absorption by the photodiode in the pixel. The photodiode absorbs the filtered light to varying degrees as a function of wavelength λ. Even attenuation in the lens of the camera is wavelength dependent. Equation (7.1) combines the spectral attenuations of the lens $g_L(\lambda)$, colour filter $g_k(\lambda)$, with $k \in \{R, G, B\}$, and photodiode $g_P(\lambda)$ into one function $f_k(\lambda)$ [12]. Equation (7.2) uses $f_k(\lambda)$ to model the photocurrent I_k induced in a red, green or blue pixel by a spectral irradiance $s(\lambda)$ [7].

$$f_k(\lambda) = g_L(\lambda) g_k(\lambda) g_P(\lambda) \tag{7.1}$$

$$I_k = \int_0^\infty f_k(\lambda) s(\lambda) d\lambda \tag{7.2}$$

A colour image sensor need not estimate $s(\lambda)$ at each pixel to recreate the sensation of colour implied by $s(\lambda)$ on a display device (i.e. a monitor or printer) [12]. In response to a spectral irradiance $s(\lambda)$, human perception of colour may be ascribed to three numbers X, Y and Z [59]. These numbers are inner products, over the visible spectrum, of $s(\lambda)$ and three basis functions $\bar{x}(\lambda)$, $\bar{y}(\lambda)$ and $\bar{z}(\lambda)$, which were standardised by the Commission Internationale de l'Eclairage (CIE) in 1931. Normally, $f_R(\lambda)$, $f_G(\lambda)$ and $f_B(\lambda)$ in (7.1) are designed to approximate linear combinations of $\bar{x}(\lambda)$, $\bar{y}(\lambda)$ and $\bar{z}(\lambda)$ [12]. Therefore, I_R, I_G and I_B in (7.2) may be modelled by linear functions of X, Y and Z, as in (7.3), where $\mathbf{x}$ is a vector of X, Y and Z values and $\mathbf{d}_k$ is a vector array of coefficients, called a *mask*, that relates the photocurrent I_k linearly to $\mathbf{x}$, where $k \in \{R, G, B\}$ as before.

$$I_k = \mathbf{d}_k \bullet \mathbf{x} \tag{7.3}$$

Because the circuits of a colour logarithmic sensor are identical to those of a monochromatic logarithmic sensor, the same equations relate the sensor response to the photocurrent in a pixel. By following the analysis of Chapter 4 for monochromatic sensors, the digital response y of a colour logarithmic pixel to a photocurrent I, which may be for a red, green or blue pixel as in (7.3), may be modelled by (7.4), where a, b, c and ϵ are called the *offset*, *gain*, *bias* and *error* respectively. The offset depends on threshold voltages of the circuit, the gain depends on the subthreshold slope, the bias depends on the photodiode leakage current and the error depends on random noise.

$$y = a + b \ln(c + I) + \epsilon \tag{7.4}$$

7.3 Calibration

The model in (7.4) gives the response of a logarithmic pixel to irradiance focused upon it from a point in a scene. To recreate the scene on a standard display, an image must be rendered from pixel responses. Rendering accuracy depends on calibration of the parameters that relate the response of each pixel to real world stimuli. The calibration divides into two parts, one dealing with FPN and the other with colour.

Table 7.1: Estimated response $\hat{y}_{ij}$ of the j^{th} logarithmic pixel to photocurrent I_{ij}, where i indexes over multiple colour images, for the single, double and triple variation models. The number of implicit parameters Q is given for FPN calibration.

Variation	$\hat{y}_{ij}$	l_{ij}	Q
Single	$1_i a_j + b l_{ij}$	$\ln(1_{ij} c + I_{ij})$	$3M + N - 3$
Double	$1_i a_j + b_{\underline{j}} l_{i\underline{j}}$	$\ln(1_{ij} c + I_{ij})$	$3M + 2N - 6$
Triple	$1_i a_j + b_{\underline{j}} l_{i\underline{j}}$	$\ln(1_i c_j + I_{ij})$	$3M + 3N - 6$

7.3.1 Varying parameters

FPN arises in a logarithmic image sensor, resulting in non-uniform images of uniform surfaces, when a, b, c or a combination thereof vary from pixel to pixel. This distortion is predictable and largely correctable. Chapter 4 identifies three types of FPN of interest. In single variation, only the offset varies with the pixel coordinate j in an array of N pixels, where $1 \leq j \leq N$. Double variation involves offset and gain variation and triple variation involves offset, gain and bias variation. The design and operation of a sensor may favour one of these types so all three are considered in this chapter. Nil variation, where no parameter varies from pixel to pixel, is not considered here as Chapter 4 shows it gives poor results for all levels of illumination.

To correct FPN, the varying parameters are estimated using images of uniform irradiance, preferably white in colour, taken with M different intensities. Indexing these images by i, where $1 \leq i \leq M$, the estimated response $\hat{y}_{ij}$ of the sensor (as opposed to the actual response y_{ij}, which includes an unpredictable error component ϵ_{ij}), is given in Table 7.1 for single, double and triple variation, where I_{ij} is the photocurrent induced for each irradiance at each pixel. Defining the sparse array u_{jk} to equal one when pixel j is of colour k and zero otherwise, I_{ij} is given in (7.5) where I_{ik} is the photocurrent induced for each irradiance by each filter. Note that (7.5) implies an inner product over k with colour values $\{R, G, B\}$ analogous to index values $\{1, 2, 3\}$.

$$I_{ij} = u_{jk} I_{ik} \tag{7.5}$$

There is no need to derive calibrations for the models in Table 7.1 because, when pixels are partitioned by colour, the calibration of FPN in a colour sensor becomes the calibration of FPN in three monochromatic sensors. Following Chapter 4, parameters for each model in Table 7.1 may be estimated by minimising three sum square errors (SSEs) in (7.6) between the actual responses y_{ij} and the estimated responses $\hat{y}_{ij}$ for colours $k = R, G, B$. These calibrations assume that I_{ik} in (7.5) is unknown.

$$SSE_k = 1_i u_{jk} (y_{ij} - \hat{y}_{ij})^2 \tag{7.6}$$

Parameters estimated by minimising the SSEs in (7.6) are not unique. Following Chapter 4, the single and double variation models in Table 7.1 are invariant under transformations (7.7)–(7.9). Similarly, the triple variation model is invariant under transformations (7.7) and (7.9) but (7.8) does not apply because of bias variation.

$$(a, b, c, I) \equiv (a - b \ln \gamma, b, \gamma c, \gamma I) \tag{7.7}$$

$$\equiv (a, b/\gamma, 0, (c + I)^{\gamma}) \tag{7.8}$$

$$\equiv (a, b, c - \gamma, I + \gamma) \tag{7.9}$$

In each colour partition, estimation of parameters for each model in Table 7.1 is limited by (7.7)–(7.9). Only those parameters that vary from pixel to pixel are determinate from images of uniform (but unknown) irradiance. For single variation, the means of the offsets $\bar{a}_k$, one for each partition, the gain b and the bias c are indeterminate but the deviation of the offsets from the means, denoted $\hat{a}_j$ in (7.10), is determinate.

$$\hat{a}_j \approx a_j - u_{jk} \bar{a}_k \tag{7.10}$$

Similarly, for double variation, the means of the offsets $\bar{a}_k$, the means of the gains $\bar{b}_k$ and the bias c are indeterminate. The estimated offsets and gains, denoted $\hat{a}_j$ and $\hat{b}_j$ in (7.11) and (7.12), are linear functions of the actual parameters, with coefficients that depend on the partition.

$$\hat{a}_j \approx a_j - b_{\underline{j}} u_{\underline{j}k} \frac{\bar{a}_k}{\bar{b}_k} \tag{7.11}$$

$$\hat{b}_j \approx \frac{b_{\underline{j}}}{u_{\underline{j}k} \bar{b}_k} \tag{7.12}$$

For triple variation, the means of the offsets $\bar{a}_k$ and the minima of the biases $\breve{c}_k$ are indeterminate. The means of the gains $\bar{b}_k$ are determinate because transformation (7.8) does not apply. The estimated offsets, gains and biases, denoted $\hat{a}_j$, $\hat{b}_j$ and $\hat{c}_j$ in (7.13)–(7.15), are linear functions of the actual parameters, with coefficients that depend on the partition in (7.13) and (7.15).

$$\hat{a}_j \approx a_j - b_{\underline{j}} u_{\underline{j}k} \frac{\bar{a}_k}{\bar{b}_k} \tag{7.13}$$

$$\hat{b}_j \approx b_j \tag{7.14}$$

$$\hat{c}_j \approx u_{\underline{j}k} e^{\bar{a}_k / \bar{b}_k} (c_{\underline{j}} - u_{\underline{j}k} \breve{c}_k) \tag{7.15}$$

Equation (7.16) estimates the error variance $\hat{\sigma}_{\epsilon}^2$ of FPN calibration. This measure, the square root of which is called the residual error, equals the total SSE in (7.6) over the degrees of freedom, which is the number of responses MN minus the number of implicit parameters Q estimated from those responses. Table 7.1 gives the number of implicit parameters, counting estimates of I_{ik}, a_j, b_j and c_j but deducting indeterminate means and minima, for single, double and triple variation.

$$\hat{\sigma}_{\epsilon}^2 = \frac{1_k SSE_k}{MN - Q} \tag{7.16}$$

Table 7.2: Varying parameters a_j, b_j and c_j of the single, double and triple variation models in Table 7.1 are linear functions of estimated parameters $\hat{a}_j$, $\hat{b}_j$ and $\hat{c}_j$ of FPN calibration. However, constant parameters of the same models remain unknown.

Variation	a_j	b_j	c_j	Unknowns
Single	$\hat{a}_j + u_{jk}\bar{a}_k$			$\bar{a}_k, b, c, \mathbf{d}_k$
Double	$\hat{a}_j + \hat{b}_{\underline{j}} u_{\underline{j}k}\bar{a}_k$	$\hat{b}_{\underline{j}} u_{\underline{j}k}\bar{b}_k$		$\bar{a}_k, \bar{b}_k, c, \mathbf{d}_k$
Triple	$\hat{a}_j + \hat{b}_{\underline{j}} u_{\underline{j}k}\frac{\bar{a}_k}{\bar{b}_k}$	$\hat{b}_j$	$u_{\underline{j}k}e^{\bar{a}_k/\bar{b}_k}\hat{c}_{\underline{j}} + u_{jk}\breve{c}_k$	$\bar{a}_k, \breve{c}_k, \mathbf{d}_k$

7.3.2 Constant parameters

Once the offset, gain and bias parameters that vary from pixel to pixel are estimated, the mask parameters of Section 7.2 and indeterminate parameters of Section 7.3.1 need estimation to render an image taken by a colour logarithmic sensor for a standard display. These parameters do not vary from pixel to pixel though they may depend on the pixel colour. As with conventional linear sensors [12], colour calibration is done by imaging a colour chart with patches of known colour and using these ideal values and corresponding image data to estimate parameters of the colour model.

Consider a calibration using M images of a colour chart, indexed by i, taken with different illuminant intensities to cover a wide dynamic range. For the single, double and triple variation models, the estimated response $\hat{y}_{ij}$ of each pixel in each image of the colour chart is given in Table 7.1 with photocurrent I_{ij} in (7.17) instead of (7.5), where $\mathbf{x}_{ij}$ is the ideal colour vector of the j^{th} pixel at the i^{th} illuminant intensity.

$$I_{ij} = u_{\underline{j}k}\mathbf{d}_k \bullet \mathbf{x}_{i\underline{j}} \tag{7.17}$$

Owing to the FPN calibration described in Section 7.3.1, varying parameters of the models in Table 7.1, i.e. a_j, b_j and c_j as appropriate, do not require estimation as they are linear functions, given in Table 7.2, of previous estimates $\hat{a}_j$, $\hat{b}_j$ and $\hat{c}_j$. Nonetheless, several unknowns remain after FPN calibration of the single, double or triple variation models, namely b, c, $\bar{a}_k$, $\bar{b}_k$, $\breve{c}_k$ and $\mathbf{d}_k$ as appropriate. Note that $\bar{b}_k$ is not considered unknown with triple variation because estimated gains $\hat{b}_j$ correspond to actual gains b_j since transformation (7.8) did not apply for this model.

The unknowns listed in Table 7.2 are not all independent because a transformation similar to (7.7) applies. Mean offsets $\bar{a}_k$ may be eliminated, as in Table 7.3, for the single, double and triple variation models by replacing c or $\breve{c}_k$ and $\mathbf{d}_k$ with c'_k and $\mathbf{d}'_k$, where l'_{ij}, c'_j and I'_{ij} are defined in (7.18), the table and (7.19) respectively.

$$l'_{ij} = \ln(1_i c'_j + I'_{ij}) \tag{7.18}$$

$$I'_{ij} = u_{\underline{j}k}\mathbf{d}'_k \bullet \mathbf{x}_{i\underline{j}} \tag{7.19}$$

As some pixels in an image of the colour chart may not belong to any colour patch with known colours $\mathbf{x}_{ij}$, images are segmented to identify the pixels that correspond to a colour patch. So that FPN does not corrupt segmentation, the images are first corrected as in Chapter 4 for monochromatic sensors, using the results of FPN calibration.

Table 7.3: A redundancy of unknowns in Table 7.2 is eliminated by replacing $\bar{a}_k$, c or $\breve{c}_k$ and $\mathbf{d}_k$ with c'_k and $\mathbf{d}'_k$, where l'_{ij} and I'_{ij} are in (7.18) and (7.19), for the single, double and triple variation models. The number of implicit parameters Q is given.

Variation	$\hat{y}_{ij}$	b_j	c'_j	c'_k	$\mathbf{d}'_k$	Q
Single	$1_i\hat{a}_j + bl'_{ij}$	$\hat{b}_{\underline{j}}u_{\underline{j}k}\bar{b}_k$	$u_{jk}c'_k$	$e^{\bar{a}_k/b}c$	$e^{\bar{a}_{\underline{k}}/b}\mathbf{d}_{\underline{k}}$	13
Double	$1_i\hat{a}_j + b_{\underline{j}}l'_{i\underline{j}}$		$u_{jk}c'_k$	$e^{\bar{a}_{\underline{k}}/\bar{b}_{\underline{k}}}c$	$e^{\bar{a}_{\underline{k}}/\bar{b}_{\underline{k}}}\mathbf{d}_{\underline{k}}$	15
Triple	$1_i\hat{a}_j + \hat{b}_{\underline{j}}l'_{i\underline{j}}$		$\hat{c}_j + u_{jk}c'_k$	$e^{\bar{a}_{\underline{k}}/\bar{b}_{\underline{k}}}\breve{c}_{\underline{k}}$	$e^{\bar{a}_{\underline{k}}/\bar{b}_{\underline{k}}}\mathbf{d}_{\underline{k}}$	12

Unknowns b, $\bar{b}_k$, c'_k and $\mathbf{d}'_k$ in Table 7.3 are estimated by minimising the SSE in (7.20) between the actual and estimated responses y_{ij} and $\hat{y}_{ij}$ for segmented pixels, identified by the sparse array v_j that is one for pixels with known colours and zero otherwise.

$$SSE = 1_i v_j (y_{ij} - \hat{y}_{ij})^2 \tag{7.20}$$

Minimising the SSE in (7.20) for any model in Table 7.3 requires nonlinear optimisation as no analytic solution exists for all the unknowns. However, at the minimum of the SSE, b and $\bar{b}_k$ for single and double variation are given by (7.21) and (7.22).

$$b = \frac{(v_j(y_{ij} - 1_i\hat{a}_j)l'_{ij})}{(1_i v_j l'^2_{ij})} \tag{7.21}$$

$$\bar{b}_k = \frac{(u_{j\underline{k}}v_j(y_{ij} - 1_i\hat{a}_j)\hat{b}_j l'_{ij})}{(1_i u_{j\underline{k}} v_j \hat{b}_j^2 l'^2_{ij})} \tag{7.22}$$

Thus, only c'_k and $\mathbf{d}'_k$, which represent 12 variables, require nonlinear optimisation. A suitable optimisation algorithm is the conjugate gradients method [57]. Care must be taken to ensure that guesses of c'_k and $\mathbf{d}'_k$, during the optimisation process, keep the argument of the logarithm in (7.18) positive. This is accomplished by making the SSE in (7.20) return a large value otherwise (∞ in MATLAB) and ensuring that the line minimisation used by the conjugate gradients method copes with such extremes.

Equation (7.23) estimates the error variance $\hat{\sigma}^2_\epsilon$ of colour calibration. This measure, the square root of which is called the residual error, equals the SSE in (7.20) divided by the degrees of freedom, which is M times the number of segmented pixels, in any image of the colour chart, minus the number of parameters Q estimated from the data, as given in Table 7.3 for single, double and triple variation.

$$\hat{\sigma}^2_\epsilon = \frac{SSE}{M(1_j v_j) - Q} \tag{7.23}$$

7.4 Rendition

The purpose of a colour image sensor is to provide an image of a scene that is similar to the real scene when displayed. Therefore, pixel responses must be rendered into a

Table 7.4: Estimated response $\hat{y}_j$ of the j^{th} logarithmic pixel to a colour stimulus $\mathbf{x}_j$, where I'_j is in (7.24), for the single, double and triple variation models using estimates $\hat{a}_j$, $\hat{b}_j$ and $\hat{c}_j$ from FPN calibration and estimates $\hat{b}$ or $\hat{b}_k$ and $\hat{c}_k$ from colour calibration.

Variation	$\hat{y}_j$	l_j	$\hat{b}'_j$	$\hat{c}'_j$
Single	$\hat{a}_j + \hat{b}l_j$	$\ln(\hat{c}'_j + I'_j)$	$\hat{b}_{\underline{j}} u_{\underline{j}k} \hat{b}_k$	$u_{jk}\hat{c}_k$
Double	$\hat{a}_j + \hat{b}'_{\underline{j}} l_{\underline{j}}$	$\ln(\hat{c}'_j + I'_j)$		$u_{jk}\hat{c}_k$
Triple	$\hat{a}_j + \hat{b}_{\underline{j}} l_{\underline{j}}$	$\ln(\hat{c}'_j + I'_j)$		$\hat{c}_j + u_{jk}\hat{c}_k$

Table 7.5: Estimated photocurrent $\hat{I}_j$ of the j^{th} logarithmic pixel to a colour stimulus $\mathbf{x}_j$ for the single, double and triple variation models, which is derived by inverting the models in Table 7.5 using the actual response y_j of the j^{th} logarithmic pixel.

Variation	$\hat{I}_j$	$\hat{l}_j$	$\hat{b}'_j$	$\hat{c}'_j$
Single	$\exp(\hat{l}_j) - \hat{c}'_j$	$(y_j - \hat{a}_j)/\hat{b}$	$\hat{b}_{\underline{j}} u_{\underline{j}k} \hat{b}_k$	$u_{jk}\hat{c}_k$
Double	$\exp(\hat{l}_j) - \hat{c}'_j$	$(y_{\underline{j}} - \hat{a}_{\underline{j}})/\hat{b}'_{\underline{j}}$		$u_{jk}\hat{c}_k$
Triple	$\exp(\hat{l}_j) - \hat{c}'_j$	$(y_{\underline{j}} - \hat{a}_{\underline{j}})/\hat{b}_{\underline{j}}$		$\hat{c}_j + u_{jk}\hat{c}_k$

well-defined colour space, such as CIE XYZ [59], that is understood by display devices. Denoting the offset, gain and bias parameters estimated by FPN calibration in Section 7.3.1 as $\hat{a}_j$, $\hat{b}_j$ and $\hat{c}_j$ and those estimated by colour calibration in Section 7.3.2 as $\hat{b}$ or $\hat{b}_k$ and $\hat{c}_k$, Table 7.4 gives the estimated response $\hat{y}_j$ of a logarithmic pixel to an arbitrary stimulus $\mathbf{x}_j$ for single, double and triple variation, with I'_j given in (7.24).

$$I'_j = u_{\underline{j}k}\mathbf{d}'_k \bullet \mathbf{x}_{\underline{j}} \tag{7.24}$$

Rendering a response y_j into CIE XYZ space involves estimating the corresponding stimulus $\mathbf{x}_j$. First, I'_j in (7.24) is estimated by minimising the SSE in (7.25) between the actual response y_j and estimated response $\hat{y}_j$ of the sensor. Such a minimisation amounts to inversion of the models in Table 7.4, giving estimates $\hat{I}_j$ in Table 7.5.

$$SSE = 1_j(y_j - \hat{y}_j)^2 \tag{7.25}$$

Note that $\hat{I}_j$ estimates (with an unknown gain) the monocolour photocurrent at each pixel. To estimate red, green and blue photocurrents at each pixel, denoted $\hat{I}_{jk}$, linear interpolation over a small neighbourhood suffices as the stimuli of a pixel and its neighbours are highly correlated. Due to (7.24), $\hat{I}_{jk}$ depends linearly on the stimulus $\mathbf{x}_j$. Inversion of this dependence in (7.26), using in matrix form the mask $\hat{\mathbf{d}}_k$ estimated by colour calibration, gives the desired estimate, denoted $\hat{\mathbf{x}}_j$, of the stimulus.

$$\hat{\mathbf{x}}_j = \begin{pmatrix} \hat{d}_{1R} & \hat{d}_{2R} & \hat{d}_{3R} \\ \hat{d}_{1G} & \hat{d}_{2G} & \hat{d}_{3G} \\ \hat{d}_{1B} & \hat{d}_{2B} & \hat{d}_{3B} \end{pmatrix}^{-1} \begin{pmatrix} \hat{I}_{jR} \\ \hat{I}_{jG} \\ \hat{I}_{jB} \end{pmatrix} \tag{7.26}$$

Estimated images $\hat{\mathbf{x}}_j$ in CIE XYZ space may be easily rendered into other useful colour spaces, such as CIE Lab or IEC sRGB [59, 60]. In terms of human vision, Euclidean distances calculated in Lab space correlate with perceptual differences. For computer hardware and software, however, the sRGB space of the International Electrotechnical Commission (IEC) has been accepted internationally as a default standard.

Equation (7.27) estimates the error variance $\hat{\sigma}_E^2$ between ideal and estimated Lab vectors $\mathbf{z}_{ij}$ and $\hat{\mathbf{z}}_{ij}$, rendered from $\mathbf{x}_{ij}$ and $\hat{\mathbf{x}}_{ij}$ respectively, for the segmented pixels in the M images of the colour chart, described in Section 7.3.2. The square root of this measure is called the perceptual error of colour calibration. Note that the denominators in (7.23) and (7.27) are equal, representing the degrees of freedom in the estimation, with Q given in Table 7.3 for the single, double and triple variation models.

$$\hat{\sigma}_E^2 = \frac{1_i v_j \|\mathbf{z}_{ij} - \hat{\mathbf{z}}_{ij}\|^2}{M(1_j v_j) - Q} \tag{7.27}$$

7.5 Experiments

Experiments were done with a Fuga 15RGB logarithmic image sensor, which had a 512×512 pixel array (i.e. $N = 512^2$). Rather than vary the intensity of the overhead fluorescent illuminant, neutral density filters with nominal optical densities of 0.5, 1.0, 1.5 and 2.0 were used to simulate two decades of intensity variation. Effective illuminances were measured with a light meter for each filter and for the case of no filter.

7.5.1 Calibration

A sheet of white paper provided a uniform scene for FPN calibration. Five images were taken (i.e. $M = 5$) using the neutral density filters to span two decades of illuminance. Following Section 7.3.1, spatially varying parameters of the single, double and triple variation models were estimated. The residual error of FPN calibration, formulated in (7.16), was 5.1, 2.2 and 0.6LSB for these models respectively. Thus, triple variation represents FPN well for the Fuga 15RGB. These results are similar to the corresponding results in Chapter 4, where the Fuga 15RGB was treated as a monochromatic sensor. In terms of FPN, treating a colour sensor as monochromatic did not compromise the results of Chapters 4–6.

Next, five images were taken of a Macbeth chart, created by McCamy et al [61], which had 24 painted patches covering a wide gamut of colours. Using the neutral density filters to span two decades of illuminance, the images covered a dynamic range of 3.5 decades as the patches spanned 1.5 decades of reflectance. Following Section 7.3.2, spatially constant parameters of the single, double and triple variation models were estimated. On average, there were 3,839 segmented pixels in each of the 24 patches in

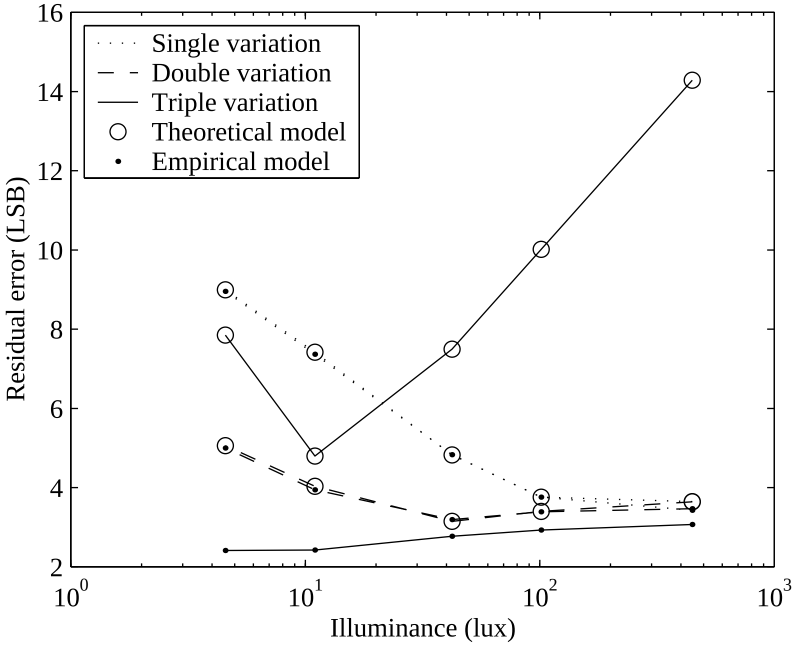


Figure 7.1: The residual error $\hat{\sigma}_{\epsilon_i}$ versus incident illuminance x_i for colour calibration of the single, double and triple variation (theoretical and empirical) models.

each of the five images. The residual error of colour calibration, formulated in (7.23), was 6.1, 3.9 and 9.4LSB for single, double and triple variation respectively, which shows that triple variation performs poorly.

Figure 7.1 plots the residual error versus illuminance of colour calibration, with points marked by circles. That triple variation performs much worse than single or double variation is surprising considering the residual error of FPN calibration is much better for triple variation. Investigation of the colour chart data reveals that, as with the white paper data, triple variation models FPN better than single or double variation. However, the dependence in (7.4) of the digital response y on the photocurrent I proves unsuitable for estimating colour. A comparison of ideal colours with estimated colours suggests a model, given in (7.28), using the function in (7.29).

$$y = a + b \ln(c + f(I)) + \epsilon \tag{7.28}$$

$$f(I) = (\alpha + I)^\beta \tag{7.29}$$

Assuming α and β in (7.29) are constant from pixel to pixel, replacing the theoretical model of (7.4) with the empirical model of (7.28) does not change the results of FPN calibration. The unknowns I_{ik} in Section 7.3.1 are replaced by the unknowns $f(I_{ik})$ with no change to offset, gain and bias estimates. However, colour calibration in Section 7.3.2 must estimate α and β by including them in the conjugate gradients optimisation. As they modify the partial derivatives of the SSE in (7.20), these parameters affect the estimation of other parameters. Furthermore, the degrees of freedom in (7.23) and (7.27) must account for estimation of α and β.

Repeating colour calibration with the empirical model results in a residual error equal to 6.1, 3.9 and 2.7LSB for single, double and triple variation. Figure 7.1 plots the residual error, marked by dots, versus illuminance. Colour calibration with the empirical model improves over the theoretical model substantially for triple variation but negligibly for single and double variation. The latter may be unable to discriminate I in (7.4) from $f(I)$ in (7.28) due to a higher residual error of FPN calibration.

The empirical triple variation model shows a residual error of colour calibration that is nearly flat across 3.5 decades of dynamic range (each point in Figure 7.1 comprises 1.5 decades). However, for single and double variation, the residual error increases with decreasing illuminance with the theoretical or empirical model. This dependence suggests that bias variation, not considered by single and double variation, degrades colour calibration mainly in dim lighting. For triple variation, the slight increase in error with increasing illuminance may be because the neutral density filters, used in taking the dimmer four images, had neither flat nor equal spectral responses and thus modified the colour of transmitted light in addition to the intensity.

7.5.2 Rendition

After FPN and colour calibration, images taken by the Fuga 15RGB may be rendered into a standard colour space such as CIE Lab or IEC sRGB, following Section 7.4 for the theoretical model. For the empirical model, the rendering must include an inversion of (7.29). Using the empirical model, the perceptual error of colour calibration, formulated in (7.27), between the ideal and rendered images of the Macbeth chart was 133, 58 and 20 (CIE Lab units) for single, double and triple variation respectively. Figure 7.2 plots the perceptual error versus illuminance. The figure shows how close the colours of the ideal chart match those of the rendered chart, imaged under varying illuminance, from the perspective of a standard observer (as defined by the CIE).

To put the performance of colour rendition with the Fuga 15RGB in perspective, the perceptual error between an image of the ideal Macbeth chart and images of the chart rendered by conventional digital cameras were calculated from an article by McNamee in *Digital Photographer* [62]. The published images were scanned with an HP Scanjet 5300C and converted from sRGB to Lab space. Table 7.6 lists the perceptual error between pixels of the ideal chart and corresponding pixels of each camera's image.

Comparing Figure 7.2 to Table 7.6 for single and double variation, colour rendition is generally better with conventional cameras than with the Fuga 15RGB. For triple variation, colour rendition of the Fuga 15RGB is comparable to conventional cameras except in dim lighting. Excluding the dimmest image, taken with 5lux of illuminance, the perceptual error is 12 with the Fuga 15RGB for triple variation. This result is comparable to the overall perceptual error in Table 7.6, which equals 15. As the Macbeth chart spans 1.5 decades of reflectance and the Fuga 15RGB images, excluding the dimmest, span 1.5 decades of illuminance, colour rendition of the logarithmic sensor, tested over three decades of dynamic range, competes with colour rendition of conventional cameras, tested over 1.5 decades (McNamee used only one illuminance [62]).

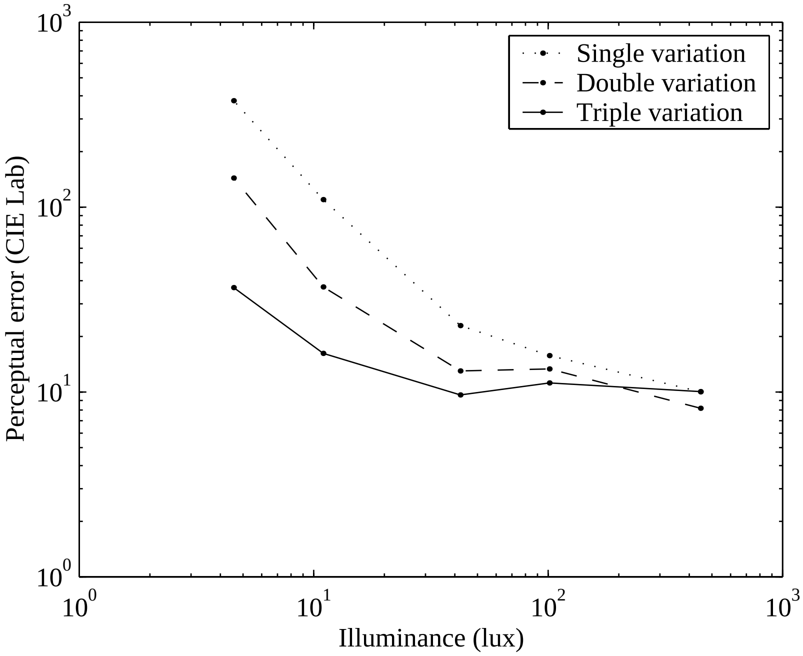


Figure 7.2: The perceptual error $\hat{\sigma}_{E_i}$ versus incident illuminance x_i of rendering a Macbeth chart for the single, double and triple variation (empirical) models.

Note that the perceptual error of colour calibration in Figure 7.2 increases with decreasing illuminance even for triple variation, which has a residual error of colour calibration in Figure 7.1 that decreases with decreasing illuminance. In dim lighting, the bias c dominates the logarithm in (7.28), with α in (7.29), making the photocurrent I difficult to estimate. In other words, the magnitude of the photodiode leakage current reduces the sensitivity of a pixel to a small photocurrent so that the stochastic error ϵ in (7.28) has a greater effect on the response than the stimulus. Decreasing the leakage current, increasing the photocurrent or reducing the stochastic error should lessen this degradation. Decreasing the leakage current would, in theory, also reduce bias variation and improve the performance of double variation relative to triple variation.

Figure 7.3 shows the five Fuga 15RGB images of the Macbeth chart, which were taken with varying illuminance. These images have been rendered into sRGB space for the single, double and triple variation models. The figure also shows an image of the chart with ideal values for the colour patches. Two mechanisms lead to a deviation of the rendered images from the ideal. The first is a residual pixel-to-pixel variation that causes uniform surfaces to appear noisy, especially visible at dimmer illuminances in the single and double variation results of the montage. The second is a colour deviation that causes patches to look different from the ideal, discernible in the triple variation result where some patches have too much or too little brightness, even if the observer's

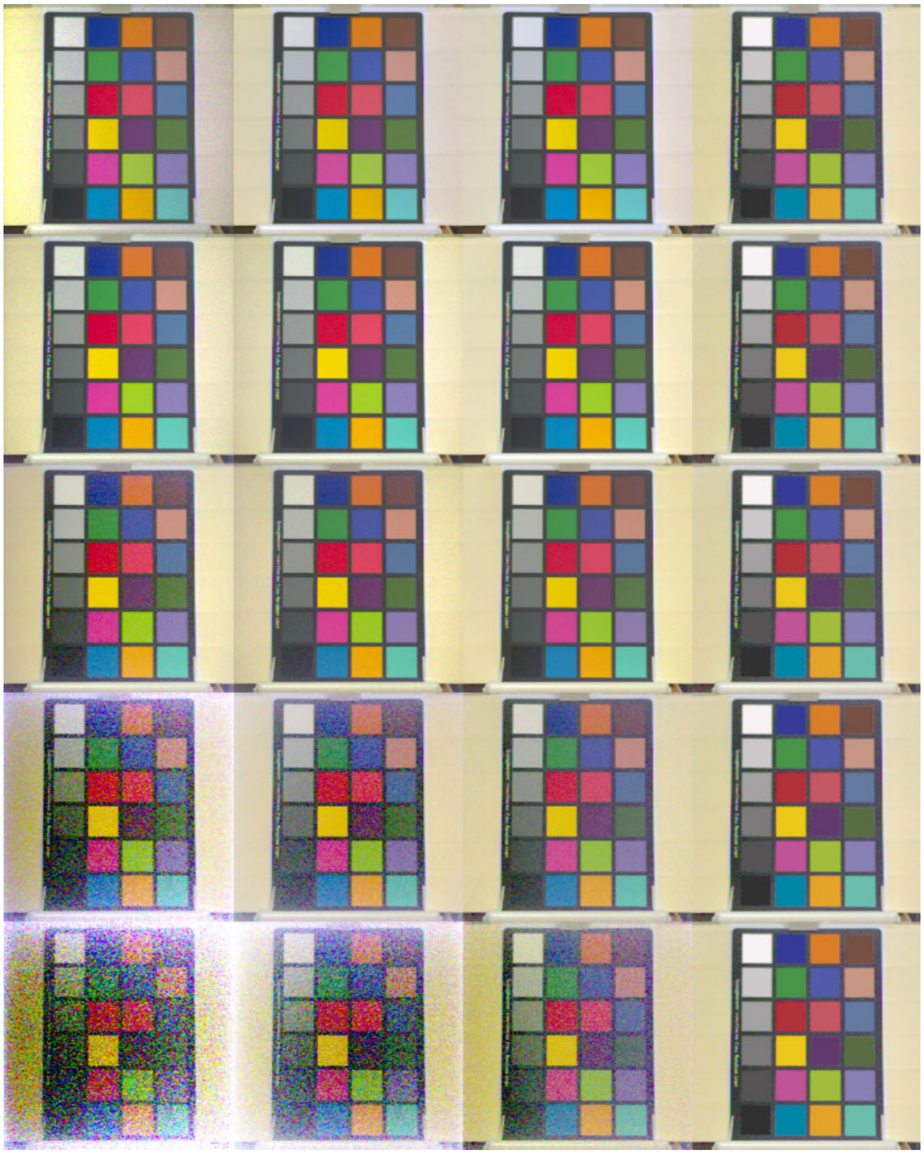


Figure 7.3: Fuga 15RGB images of a Macbeth chart, taken with an incident illuminance of 450, 100, 42, 11 and 4.6lux (top to bottom) and rendered into IEC sRGB space, for the single, double and triple variation empirical models (from left). The far-right images overlay ideal colours of the chart patches on the average triple variation result.

Table 7.6: The perceptual error $\hat{\sigma}_E$ of conventional digital cameras between ideal and actual images of a Macbeth chart, taken at one illuminance only.

Digital camera	Perceptual error
Kodak DCS 265	13
Nikon Coolpix 950	12
Olympus Camedia C-2000 Zoom	16
Canon Powershot Pro 70	17
Ricoh RDC 4200	13
Agfa ePhoto CL 50	15
Fuji MX 2700	15
In	CIE Lab

eye could filter out the residual variation.

One reason for a colour deviation may be that the mechanism relating responses to stimuli is not fully understood in the Fuga 15RGB, evident by the use of an empirical model. Another reason is that, in dim lighting, the dominance of the leakage current over the photocurrent leads to a biased estimate of the stimulus. Indeed, colour matching is better at higher illuminances, as shown in Figures 7.2 and 7.3. The fluorescent illuminant may be another reason as McCamy et al recommended CIE Standard Illuminant C for use with the Macbeth chart [61]. Fluorescent illuminants have spectral irradiance functions with sharp peaks at certain wavelengths that frustrate colour rendition [59]. Furthermore, the neutral density filters were not perfectly neutral.

7.6 Conclusion

Logarithmic CMOS image sensors have a capability to capture scenes bearing a high dynamic range of illuminance and reflectance in a manner that roughly approximates human perception [25]. Permitting high frame rates, they are an attractive technology for motion tracking in outdoor environments [33, 26]. However, research on colour logarithmic sensors has been limited by a lack of theory and results on modelling, calibration and rendition of sensor responses in terms of a standard colour space. This chapter begins to address these problems.

A model for the response of a colour logarithmic sensor to spectral irradiance was constructed by combining the colour model of conventional linear sensors [12] with the monochromatic model of logarithmic sensors. Thus, the digital response y of a logarithmic pixel to a colour stimulus $\mathbf{x}$, given in CIE XYZ space [59], is modelled by $y = a + b \ln(c + \mathbf{d}_k \bullet \mathbf{x}) + \epsilon$ where a, b, c, $\mathbf{d}_k$ and ϵ are called the *offset*, *gain*, *bias*, *mask* and *error* respectively, with k identifying if the pixel is selective to the red, green or blue regions of the spectrum.

Pixel-to-pixel variation of the offset, gain, bias or a combination thereof leads to fixed pattern noise (FPN), which distorts an image in a repeatable and predictable way, most visible with uniform surfaces. Calibration of the image sensor involves estimation of the model parameters. Spatially varying parameters are estimated by partitioning pixels by colour sensitivity and applying the method of monochromatic FPN calibration to each partition. The mask and other spatially constant parameters that remain from FPN calibration are estimated using images of a reference colour chart. Calibrated models may be used to render an image taken with the sensor into CIE XYZ space and then into other useful spaces, such as CIE Lab and IEC sRGB [59, 60].

Using neutral density filters to simulate varying illuminance, experiments were performed with a Fuga 15RGB sensor. A pixel-to-pixel variation of offset, gain and bias modelled FPN well, with a residual error of $0.6\mathrm{LSB}$ for FPN calibration of white paper. Colour calibration of a Macbeth chart [61] showed that the theoretical model did not match the sensor response. An empirical model $y = a + b \ln(c + (\alpha + \mathbf{d}_k \bullet \mathbf{x})^{\beta}) + \epsilon$ worked well, with a residual error of $2.7\mathrm{LSB}$ for colour calibration. The perceptual error with this model was 12, in Lab space, over three decades of dynamic range, comparable to conventional digital cameras over 1.5 decades. The perceptual error increased quickly below five lux of illuminance, possibly because leakage currents reduced the sensitivity of pixels.

Instead of focusing on analogue or digital methods to compensate for offset variation, research in logarithmic sensors should aim to minimise bias variation so that offset variation or offset and gain variation suffices to model FPN, and to minimise bias magnitude, so that colour rendition in dim lighting improves. As the mask depends on spectral responses of photodiodes and overlaid filters and does not seem to vary across pixels, it may be estimated once for a process, a common practice with conventional linear cameras [12], rather than for every sensor. The same may be possible with other spatially constant parameters.

Chapter 8

Conclusion

8.1 Summary

The CCD image sensor, a dominant technology for about three decades, faces tough competition from the CMOS image sensor, a more recent technology. Since their fabrication process is incompatible with conventional electronics, CCD sensors require external circuits to provide bias voltages, clock signals, control logic, analogue-to-digital conversion and signal processing. CMOS technology, however, permits the integration of these circuits on the same die as the sensor to reduce the cost, power consumption, size and weight of the final camera. Fundamentally, CMOS pixels scale well with shrinking process geometries because more electronics can be placed in each pixel to improve the output without affecting sensitivity or resolution. While CCD sensors still dominate the market because of sensitivity, the performance edge of CCD over CMOS is disappearing with shrinking pixel size and increasing video demands. For these and other reasons, such as a higher quantum efficiency, less smear and blooming, better yields and price pressure from more competition, the electronics industry expects CMOS gradually to replace CCD image sensors.

This thesis concerns a subset of CMOS sensor technology, namely logarithmic imagers. A linear pixel (CCD or CMOS) integrates the charge produced by photon absorption over a finite period of time to produce a voltage directly proportional to the light intensity. A logarithmic pixel converts incident photons continuously into a voltage that is proportional, over more than five decades of illuminance, to the logarithm of the light intensity. Logarithmic pixels may be randomly accessed in space and time, since CMOS sensors operate like memory arrays and logarithmic responses are available at any moment, a feature useful in industrial and consumer applications for which frame size and speed may be traded against each other. Studies with pulsed lasers have shown a pixel bandwidth of 100 kHz, at normal light levels, that increases with illumination—the speed of the readout circuit often limits the frame rate. As logarithmic pixels are simple, consisting of three transistors and a diode, sensors have been made with 2048×2048 pixels and acceptable yields.

Light reflected by scenes spans many decades of illuminance, from $10^{-3}\mathrm{lux}$ at night to 1–10^3lux in indoor lighting and up to 10^5lux in bright sunlight. Direct viewing and specularities of bright sources, such as oncoming headlights or the sun, may lead to higher intensities. At any one time, however, the human eye cannot perceive more than five decades. Human perception roughly approximates Weber's law, which says that the threshold to sense a difference between the illuminance of a fixation point and its surroundings is a fraction, about 1–10%, of the surrounding illuminance. When illuminances are encoded by a logarithmic sensor, such a law makes the threshold for sensitivity constant, ideal for quantisation. With a logarithmic sensor, ten bits of quantisation are sufficient to sense illuminance over five decades with 1% accuracy. A linear sensor requires 23 bits to accomplish the same task, which would be costly for still cameras and extremely difficult at video rates. A linear sensor with fewer bits of quantisation could adapt over a high dynamic range by aperture or integration-time control. However, saturated patches would appear in images of scenes that span a high dynamic range. Many non-logarithmic methods have been proposed to extend the dynamic range of image sensors but most result in decreased resolution, sensitivity or frame rate.

Despite nearly a decade of research and development, logarithmic cameras remain of interest mainly to a niche market and largely for the purpose of further research and development. Widespread acceptance is hindered by the substantial fixed pattern noise (FPN) present in images taken by these sensors. Work reported in the literature focuses on analogue and digital techniques to compensate for threshold voltage variation, which is perceived to be the major problem with logarithmic imagers. There is another problem even for an ideal logarithmic imager that is free of FPN. As conventional digital cameras involve well understood mechanisms of colour sensation, acceptable colour rendition is achieved by well defined signal processing. However, this theory has been developed for linear sensors and concerns have been raised in the literature as to the colour rendition capabilities of logarithmic image sensors.

Work reported in this thesis sought the causes of problems with image quality in logarithmic CMOS image sensors and possible solutions, which entailed the modelling and calibration of responses in terms of stimuli. Theoretical work considered the manipulation of image collections, both analytically and numerically, and the physics of integrated circuit devices. Simulation work considered the behaviour of logarithmic pixels, for a popular $0.35\mu\mathrm{m}$ $3.3\mathrm{V}$ process, under controlled and well-defined conditions. Experimental work considered the behaviour of the Fuga 15RGB, a commercially successful logarithmic imager built in a $0.7\mu\mathrm{m}$ $5\mathrm{V}$ process, under laboratory conditions. The rest of this section summarises the main results of this thesis. Section 8.2 considers future work in the field.

8.1.1 Multilinear algebra

If images are considered to be matrices of data then a collection of images may be represented most naturally by a generalisation of the scalar, vector and matrix progression, which is the array. As linear algebra deals with elementary mathematical operations on matrices, so multilinear algebra deals with elementary mathematical operations on arrays. An array of order N is defined to be a functional mapping from a vector of N positive integers, each ranging from one to a specified dimension, to a space of homogenous elements. The elements may themselves be scalars, vectors or matrices.

Differing approaches exist in the literature in terms of the definition of arrays and the formulation of their algebra. The approach taken here follows from tensor calculus but without the customary connection to differential geometry. Multilinear algebra, as defined in this thesis, includes the usual tensor operations of contraction and both inner and outer products but introduces attraction and inter products, which enable elementwise operations. Inner and inter products are shown to be equivalent mathematically, but not computationally, to outer products followed by contraction and attraction respectively. Whereas tensor calculus restricts contraction and inner products, multilinear algebra does not since inter products enable previously impossible associations so that any product of multiple arrays may be rewritten as a sequence of binary products. Application of these ideas to classical linear algebra demonstrates that several elementary array operations, in terms of scalars, vectors and matrices, may not be expressed without new operators, which are therefore introduced.

Any binary product of arrays of arbitrary order is shown to be equivalent to a sequence of matrix multiplications. Consequentially, array multiplication may be efficiently implemented in MATLAB by automatic transformation of the problem, computation of the solution, and transformation of the result. Furthermore, solving multilinear algebraic equations often involves finding the inverse of an array to produce a particular identity upon multiplication. Unlike with matrices, an array may have more than one inverse depending on the required identity. However, if the inverse for a particular identity exists then it is unique and may be found by transforming the problem to a sequence of matrix inversions, computing the solutions and transforming the results back, all of which may be efficiently automated in a MATLAB implementation. Descriptions of tensor calculus found in the literature do not formalise inversion.

Several applications of multilinear algebra were discussed as they prove relevant to the efficient calibration of image sensors. The concept of stochastic arrays, which are random samples of a (potentially infinite) population of arrays, leads to a consideration of statistical variance. The outer, inter and inner variance are defined by applying the usual expectation operator to outer, inter and inner products of a stochastic array, less its mean, with itself. In general, computing the outer variance takes $O(N^2)$ time and space whereas computing the inter and inner variances takes $O(N)$ time and space for a stochastic array with N elements. For problems where N is large, the outer variance should be avoided for complexity and the inner variance should be avoided as it gives very little information. The inter variance, however, gives much information in potentially linear time and space.

The concept of sparse arrays highlighted the savings in processor time and memory space that would be obtained by exploiting the property that some arrays contain a minority of nonzero elements. A simple implementation was described, which stores the sparse array as a native sparse vector in MATLAB and transforms access requests automatically and appropriately. The concept of cell arrays considers a functional mapping from a vector of N positive integers, each ranging from one to a specified dimension, to a space of heterogenous elements. These arrays may be used to formulate and solve systems of equations efficiently although further work on the algebra is required to develop a MATLAB implementation that automates the necessary tasks.

8.1.2 Constrained regression

Consider a sensor in which the output is modelled by a linear function, with unknown coefficients, of the input and a stochastic error. The sensor may be calibrated by estimating the model parameters from observations of the input and corresponding output. If the stochastic error is statistically independent from sample to sample and belongs to a zero-mean Gaussian distribution then maximum likelihood estimation simplifies to the least squares method of multilinear regression. Given an array of N such sensors, where each sensor may provide a different output for the same input, this approach may be applied independently to calibrate each sensor, which takes $O(N)$ time and space if N is much larger than the number of observations and parameters per sensor.

An array of N sensors could simply be modelled as N independent sensors. However, the possibility of relationships between parameters from one sensor to another should be considered for the dual purpose of better understanding and better estimation. Limiting the scope of such relationships to linear equations, the estimation problem becomes one of multilinear regression with linear constraints. Furthermore, the residual error between actual and estimated responses and the uncertainty in estimated parameters are required to assess the calibrated model. Whereas the concept of constrained regression appears in the literature, this thesis applies it to the analysis of sensor arrays, seeking to optimise performance in terms of the processor time and memory space required for computation. Without optimisation, the numerical processing of image collections, in the modelling and calibration of logarithmic sensors, would be impractical.

Two classes of parameter constraints are considered. In the generic problem, parameters across the sensor array may be related by arbitrary linear constraints. In the raster problem, the sensor array is assumed to have a planar structure and each parameter may either vary from sensor to sensor, column to column or not at all. The raster problem is a special case of the generic problem where the constraints are described by a class of sparse arrays.

There are two approaches to constrained regression. The first expresses the constraints explicitly with a Lagrangian. The second expresses the constraints implicitly by equating the parameter space, with a transformation, to a subspace of fewer parameters. Using multilinear algebra, both formulations are investigated to derive a solution to the generic problem. In the worst case, both require $O(N^3)$ time and $O(N^2)$ space. However, when constraints are described by sparse arrays typical of the raster problem, the performance of the implicit formulation improves to $O(N^2)$ time and space. Using Cholesky factorisation to avoid computing full inverse matrices of sparse positive definite matrices, the implicit formulation improves to $O(N)$ time and space, assuming an efficient sparse array implementation. However, an implementation of sparse arrays in MATLAB proved to be inefficient because of internal details of the MATLAB sparse vector and matrix routines. Therefore, an $O(N)$ method involving no sparse arrays was derived to solve the raster problem alone.

These methods were tested on an artificial raster problem, where the output of each sensor is a linear function of a single input, with an offset and gain parameter per sensor, and Gaussian noise. Three scenarios were considered, where the gain varied from sensor to sensor, column to column or not at all. Parameters were estimated for each scenario, using artificially generated data, assuming each possible scenario. When the hypothesis was more specific than the scenario, hence over-constrained, the residual error was higher than that of the correct model. When the hypothesis was more general than the scenario, hence under-constrained, the parameter uncertainties were higher than those of the correct model. Thus, the correct model may be identified and calibrated to minimise the residual error first and the parameter uncertainties second.

The time and space requirements of the various formulations agreed with the predictions, for various sizes of the artificial raster problem, with the raster formulation giving the best performance, even when internal details of sparse vectors and matrices in MATLAB are discounted. Methods to find a solution using only classical linear algebra were found to take at least $O(N^2)$ time and space, which may be traced to a lack of the attraction and inter product operations that are available in multilinear algebra.

8.1.3 Fixed pattern noise

A detailed model was derived to describe the operation of a logarithmic image sensor, from the incidence of light on a pixel to the digital response of the sensor when that pixel is addressed. This derivation contains numerous physical parameters but may be abstracted by the model $y = a + b\ln(c + x) + \epsilon$, where x and y are the incident illuminance and corresponding response and a, b, c and ϵ are named the offset, gain, bias and error respectively. The offset consists of threshold voltages and current gain ratios of various transistors in the signal path. The gain consists of subthreshold slope parameters of the load transistor in the pixel. The bias consists of photodiode leakage currents and optical gain parameters. The error consists of quantisation and temporal noise as well as uncertainty in the underlying device models, i.e. higher order effects.

A variation of the offset, gain, bias or a combination thereof from pixel to pixel causes a nonuniform image of a uniform surface, which is FPN. Although it is most noticeable in images of uniform surfaces, FPN is always present. Knowing precisely which parameters are varying from pixel to pixel improves understanding of the sensor array and permits better calibration with lower residual error and parameter uncertainty. Four models of parameter variation were considered. Nil variation assumes the ideal case where no parameters vary. In single variation, the assumption generally found in the literature, only the offset varies from pixel to pixel. In double variation, the offset and gain vary, and in triple variation, all three parameters vary. These models may be calibrated using images of a uniform surface taken with different illuminances.

Initially, it appears that calibrating the response y of all pixels in terms of the illuminance x requires measurement of the latter. This measurement may be avoided by making x a parameter, which introduces a small degree of redundancy so not all parameters may be estimated for each type of variation. However, the component that varies from pixel to pixel, which is responsible for FPN, may be estimated. Furthermore, nonlinear optimisation of the bias c may be avoided, when it does not vary, by calibrating the response of a pixel in terms of the average response of all pixels to the same illuminance. The parameter values, residual error and parameter uncertainties may be estimated for nil, single and double variation using the raster method. For triple variation, multilinear regression reduces the number of unknowns by two thirds, with the rest estimated by nonlinear optimisation. The residual error of triple variation may be calculated as before and, by ignoring the stochasity of the nonlinear part, parameter uncertainties may also be calculated. For all types of variation, methods are derived to correct FPN in subsequent images using the calibrated models.

Calibration was demonstrated using simulated and experimental data. For the simulated data, which covered six decades of photocurrent, the residual error was 20, $.44$, $.29$ and $.28\text{mV}$ for nil, single, double and triple variation respectively. In addition to the residual errors, the parameter uncertainties were comparable for double and triple variation. Because uncertainties were underestimated with triple variation, since the stochasticity of the nonlinear part was ignored, double variation proved to be the best model of FPN. The simulation did not consider the variation of photodiode leakage current or aperture effects, which explains the resulting absence of bias variation.

For the experimental data, which covered two decades of illuminance, the residual error was 20, 5.2, 2.3 and $.68\text{LSB}$ for nil, single, double and triple variation respectively. Therefore, triple variation proved to be the best model of FPN. With the experimental data, a plot of the residual error versus illuminance was relatively flat for triple variation but was highly dependent on illuminance for single and double variation (with the simulation data, the plot was flat for double and triple variation but was curved for single variation). A good model would have a residual error relatively independent of illuminance, which suggests that FPN would not increase dramatically when the model is extrapolated to illuminances outside of the calibration range.

Lastly, FPN correction was demonstrated over almost three and a half decades of dynamic range using images taken by the Fuga 15RGB. Triple variation gave the best results, especially in dim lighting, followed by double variation. Single variation gave good results only in bright lighting, for a small range of illuminance, when the effect of bias and gain variation may be ignored. Nil variation always gave poor results.

8.1.4 Transient response

In addition to the steady state model summarised above, this thesis modelled the transient response of logarithmic CMOS image sensors. The transient response of the pixel circuit is ignored as continuous pixel operation permits high bandwidths that typically exceed the rate of pixel access. For example, a 30Hz frame rate satisfies the motion sensitivity of human perception. Greater demands are placed on the readout circuit, which may be divided into two stages. For each column, in parallel over all columns, the first stage drives the column bus for the pixel in a selected row. The second stage drives the output bus for the buffer in a selected column. Rows and columns are scanned in raster fashion. The first and second stages switch at frequencies of about 100kHz and 100MHz respectively, for megapixel sensors operating at video rates. Typically, the stages are source follower circuits, where the first tends to be NMOS and the second PMOS. The transient response of the first stage is derived and an analogous response may be derived for the second stage.

Since the column bus of the first stage is connected to the source terminals of many switches, the source-bulk junction capacitances of these transistors form the principal load. Relating the column bus voltage to the pixel drive voltage with a differential equation, the transient response may be derived from when the switch of the selected pixel is turned on given the initial voltage of the column bus. The response is similar to the step response of a first order low pass filter and a time constant may be calculated. If sufficient time is not given for the response to settle before digitisation then the steady state will not be reached and the transient response will define the digital response. This condition is likely to occur in the first few rows of an image as the column buses of the parallel first stages may have discharged or been precharged at the start of reading a frame. This condition is *most* likely to occur in the first several columns of an image as the output bus of the second stage may have discharged or been precharged at the start of reading each row in a frame.

Premature digitisation will cause a repeatable and predictable nonuniformity in images of uniform surfaces, which is FPN, even if there is no device parameter variation. This nonuniformity will in general be a convolution of the signal, where the premature response of a pixel in the first few rows or several columns will depend on the premature responses of previously read pixels, as they determine the initial condition of the buses. However, much of this nonuniformity may be expressed as an offset and gain variation of the first and second stage response to illuminance. Therefore, a good part of FPN due to the transient response of an image sensor may be calibrated and corrected with previously described methods for the steady state. Estimated parameters will vary mostly in the first few rows and several columns of the sensor array and will then settle into the variation caused by steady state nonuniformity alone, which should not correlate with row or column number.

These predictions were verified by simulation and experiment. Simulation results for the first stage, which did not include steady state variation, agreed closely with theoretical results. Estimated offset and gain parameters had a variation in the first few rows that eventually settles to constant values. A similar variation may be shown in the first several columns for the second stage. The experimental results showed a settling of pixel responses over the first few rows and several columns of images of a uniform surface, taken with varying illuminance. Such images were taken for three different frame rates and were calibrated using the triple variation model. Plots of the average offset, gain and bias of each row or column showed a substantial variation in the first few rows and several columns of the sensor array and the shape of each plot changes with increasing frame rate.

Calibrating the experimental data, over two decades of illuminance, to a model in which the gain may vary from column to column but not within a column, although the offset and bias may vary from pixel to pixel, gives a residual error comparable to that of unconstrained triple variation. However, parameter uncertainties are much lower with the constrained model, which means it is a better description of reality. These results suggest that gain variation in the Fuga 15RGB is a columnwise effect that is caused predominantly by transient effects. A variation of the subthreshold slope parameters that define the steady state gain is insignificant over two decades of dynamic range.

Experimental results did not correspond fully to simulation results because, firstly, the experiment includes steady state variation, especially of the bias. Secondly, while the first stage of the Fuga 15RGB is an NMOS source follower, details of the second stage were not available. Thirdly, the position of the switch transistor in the readout circuit of the Fuga 15RGB is different from that of typical readout circuits. Whereas the atypical position is somewhat better for the steady state response, it is much worse for the transient response as it increases the settling time substantially and makes the load impedence depend on illuminance. In consequence, FPN introduced by premature digitisation cannot be modelled by an offset and gain variation over a high dynamic range. Testing the Fuga 15RGB over a high dynamic range shows a breakdown of the triple variation model due to a complex response in the first several columns of the image sensor (as well as the first few rows).

The best way to correct transient-induced FPN is to avoid circuit designs with poor or complex transient responses, to permit enough time for the readout circuit to settle, especially at the start of reading each frame and of reading each row in a frame, and to fix the timing of the readout circuit so it may not be changed after calibration. Indeed, as logarithmic sensors operate continuously and involve no exposure control by integration time, there is no reason to provide more time for settling than the worst case settling time and, as shown, good reason not to provide less time.

8.1.5 Temperature dependence

As threshold voltages, current gains, subthreshold parameters and leakage currents all depend on temperature, the response y of a logarithmic pixel and hence FPN depends on temperature T as well as illuminance x. Although there are numerous physical parameters that describe the temperature and illuminance dependence, they may be abstracted by the model $y = a_1 + a_2 T + a_3 T \ln T + b_1 T \ln(c_1 e^{T/T_\Delta} + x) + \epsilon$, where a_k, b_1 and c_1 are offset, gain and bias parameters, T_Δ is a process constant and ϵ is the stochastic error. A pixel-to-pixel or column-to-column variation of any parameter other than a_1 will cause temperature-dependent FPN.

Since calibration of bias variation over multiple temperatures and illuminances requires substantial nonlinear optimisation, only models where the bias is constant spatially are considered. The most obvious method of calibration estimates parameters to fit responses y to measured temperatures T and illuminances x, for images of a uniform surface taken at multiple temperatures and illuminances. However, these measurements and nonlinear optimisation may be avoided by making T and x parameters. Such an approach introduces a small degree of redundancy into the model so that not all parameters may be estimated from the data. Nonetheless, the parameters that vary from pixel to pixel or column to column, which are responsible for FPN, may be estimated with the raster method, when images of the dark response, where $x \approx 0$, are available at the same temperatures of the light response, where $x \gg 0$.

There are two approaches to calibration using dark and light responses. In the offset cancellation method, the dark response is subtracted from the light response to eliminate all offset parameters. The offset-free response of each pixel is then calibrated against the average such response of all pixels. Such a calibration involves the estimation of only gain parameters. In the temperature proxy method, the $T \ln T$ term in models of the dark and light response and a T^2 term in the model of the dark response are linearised. When these linearisations are reasonable, the average dark response is a linear function of temperature and the light response of each pixel may be written as a linear function of the average dark and light responses of all pixels. Calibration involves the estimation of three parameters per pixel. Cases where parameters of the offset cancellation or temperature proxy methods are constrained to vary from column to column or not at all were also considered.

The offset cancellation and temperature proxy methods were tested on simulation and experimental data. The simulation covered $60\,^\circ\mathrm{C}$ of temperature and six decades of photocurrent but did not include bias variation. The experiment covered $30\,^\circ\mathrm{C}$ of temperature and two decades of illuminance and included bias variation. Calibration of simulation data with the unconstrained offset cancellation method gave a residual error of $.42\mathrm{mV}$, which corresponds to 2.5% contrast sensitivity. For the temperature proxy method, calibration of the average dark response to a linear model of temperature gave a residual error of $.14\mathrm{mV}$ and proved to be more robust than a model that included $T\ln T$ and T^2 terms. Calibrating light responses for the unconstrained temperature proxy method gave a residual error of $.29\mathrm{mV}$, which corresponds to a 1.7% contrast sensitivity. Constraining parameters of either the offset cancellation or temperature proxy method gives worse results. Whereas the residual error versus temperature and illuminance is flat for both the offset cancellation and temperature proxy methods, the latter gives a better result as it involves the calibration of a single noisy signal rather than the difference of two noisy signals.

Calibration of the experimental data with the unconstrained offset cancellation method gave a residual error of $4.2\mathrm{LSB}$. This figure corresponds to a 25% contrast sensitivity, much worse than human perception, and the residual error depends strongly on illuminance although weakly on temperature. The failure of the method is attributed to bias variation, which is not cancelled but exacerbated by a subtraction of dark from light responses. For the temperature proxy method, the average dark response may be represented by a linear model of temperature with $.34\mathrm{LSB}$ residual error. Linearisation proves to be robust as calibration of models that include $T\ln T$ and/or T^2 terms are hardly better in terms of residual error but substantially worse in terms of parameter uncertainties. Calibration of the light response with the unconstrained temperature proxy method gives a residual error of $2.0\mathrm{LSB}$, which corresponds to a 12% contrast sensitivity. The residual error is relatively independent of temperature but depends on illuminance with a w-shaped curve. As shown previously with double variation, this shape is a consequence of underlying bias variation. The temperature proxy method performs better than the offset cancellation method because the former calibrates a single noisy signal whereas the latter calibrates the difference of two noisy signals and because the former has more parameters per pixel to accommodate the underlying bias variation. Constraining any parameter of either the offset cancellation or temperature proxy method gives worse results.

A calibration of the offset cancellation and temperature proxy models at one temperature was considered. However, performance degrades when calibrated models are extrapolated to all temperatures except for the offset cancellation method and simulated responses. Degradation occurs because of bias variation with experimental responses and because certain constraints do not hold for the temperature proxy method with either simulated or experimental responses. Experimental results are worse than simulation results because of, apart from bias variation in the former, an invisible oscillation in the illuminance of the experimental light source and a nonlinear modulation of responses due to the transient behaviour of the Fuga 15RGB. Measures were taken to reduce both effects but they could not be perfectly eliminated.

8.1.6 Colour rendition

FPN characterisation and correction is principally concerned with the nonuniformity present in images and this distortion may be corrected by modelling and calibrating pixel responses relative to other pixel responses with little concern for the absolute stimuli. Colour rendition, however, requires not only FPN correction but the reproduction of colour stimuli. By combining colour theory of linear image sensors with FPN theory of logarithmic image sensors, a model of colour logarithmic image sensors was constructed and a process derived, as has been done with conventional digital cameras, to calibrate the model and achieve good colour rendition. In this manner, the response y of a colour logarithmic pixel to a stimulus $\mathbf{x}$, which is a vector in the standard CIE XYZ colour space, is modelled by $y = a + b\ln(c + \mathbf{d} \bullet \mathbf{x}) + \epsilon$, where a, b, c, $\mathbf{d}$ and ϵ are the offset, gain, bias, mask and error respectively. The mask is a vector of coefficients describing the colour filter placed over the pixel in question. For simplicity, transient and temperature effects are ignored.

As before, a variation of device parameters from pixel to pixel (or column to column) leads to FPN. Three types are considered—single, double and triple variation—which are the models most likely to be used depending on the circuit design and desired complexity of FPN correction. By partitioning pixels in the sensor array according to the type of overlaid colour filter (red, green or blue), FPN calibration of a colour sensor reduces to FPN calibration of three monochromatic sensors, and methods previously summarised may be used to estimate the spatially varying parameters from images of a uniform surface. However, a second calibration is required to estimate the spatially constant parameters that remain unknown but which are necessary to describe digital responses in terms of colour stimuli. This calibration requires nonlinear optimisation. Parameters are estimated using segmented images of a colour chart having patches of known colour, with some unknowns reduced by analytical manipulation. Once the colour calibration is completed, the estimated parameters may be used to correct FPN and render arbitrary images into a standard colour space.

These methods were tested on experimental data collected with the Fuga 15RGB. A comparison of residual errors showed that triple variation outperformed single and double variation for calibrating FPN, as before, with an error of $0.6\mathrm{LSB}$ over two decades of illuminance. The residual error of colour calibration, however, was 6.1, 3.9 and $9.4\mathrm{LSB}$ respectively for single, double and triple variation. The larger error with triple variation is because the given model of pixel response is unsuitable for describing absolute dependence on stimuli rather than the relative dependence on stimuli from pixel to pixel, which compensates for incompleteness of the underlying device models, that is sufficient for FPN calibration. As triple variation calibrates FPN caused by bias variation, the limitation is more apparent with it than with single or double variation. Empirical analysis led to the model $y = a + b\ln(c + (\alpha + \mathbf{d} \bullet \mathbf{x})^{\beta}) + \epsilon$. Using this empirical model instead of the theoretical model does not affect the results of FPN calibration but affects those of colour calibration, where the residual error of single, double and triple variation changes to 6.1, 3.9 and $2.7\mathrm{LSB}$ respectively.

The calibrated empirical model was subsequently used to render images of a standard colour chart into the CIE Lab space. Euclidean distances in this space correspond to perceptual differences and the perceptual error of the single, double and triple variation models, over a dynamic range spanning 3.5 decades of illuminance and reflectance, was 133, 58 and 20 respectively. The triple variation empirical model gave the best rendition, especially in dim lighting. The images were also rendered into the IEC sRGB format for display purposes, which validated the perceptual error comparison. Performance deteriorates with the logarithmic sensor in dim lighting for all models because the bias, irrespective of variation, limits sensitivity. Excluding the dimmest image, which reduced the dynamic range to three decades, the perceptual error improved to 12 for triple variation. Computing the perceptual error between ideal and actual images of the same chart, taken from *Digital Photographer*, for several conventional digital cameras leads to an overall perceptual error of 15 over 1.5 decades of dynamic range. Thus, using the triple variation empirical model, the colour rendition of a logarithmic image sensor competes with that of linear image sensors.

8.2 Future work

Developers of digital cameras have sought to render images with a maximum of perceptual accuracy and a minimum of effort. By deriving a model of the logarithmic CMOS image sensor, supported by semiconductor theory, and deriving a method of calibration, validated with simulation and experiment, the work reported in this thesis has shown how these digital cameras fall short of rendering an image with a maximum of perceptual accuracy. Although this work has successfully derived digital methods to improve the image quality, these methods do not always involve a minimum of effort, especially when a maximum of perceptual accuracy is required. However, an understanding of the main results of this thesis will help developers to design, model and calibrate a better logarithmic CMOS image sensor—one which comes closer to matching the performance of the human eye.

8.2.1 Pixel circuit

The bias in the logarithmic response of a pixel, which is due to the photodiode leakage current and optical vignetting, is a major cause of problems. The presence of bias variation, as opposed to only offset variation or offset and gain variation, means that nonlinear optimisation is required for effective FPN calibration and correction. Without bias variation, FPN calibration and correction over a wide range of temperature and illuminance would be vastly simplified using the offset cancellation method with extrapolation or vastly improved using the temperature proxy method without extrapolation. In addition, the method of colour calibration, although still requiring nonlinear optimisation, would be simpler. Apart from the problem of bias variation, there is the problem of bias magnitude. The relative magnitude of leakage current to photocurrent, even with no leakage current variation, means that the sensitivity of logarithmic pixels diminishes at low illuminances, leading to poor colour rendition in dim lighting. Problems with bias variation and magnitude may be addressed to some degree by optical considerations. Better lens and aperture designs or smaller sensor dimensions would reduce the bias variation caused by vignetting. Similarly, optical designs with lower

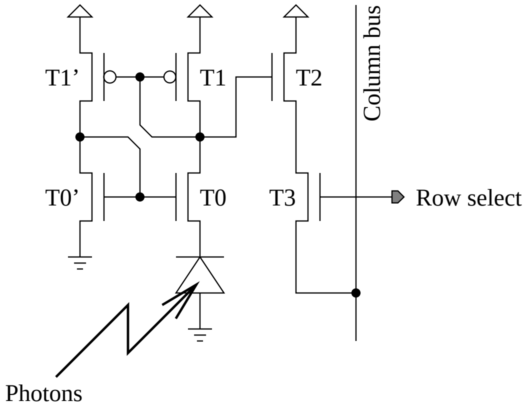


Figure 8.1: A double current mirror pixel, where $\mathrm{T0}$ & $\mathrm{T0}'$ and $\mathrm{T1}$ & $\mathrm{T1}'$ are subthreshold NMOS and PMOS mirrors while $\mathrm{T2}$ & $\mathrm{T3}$ belong to the first stage readout. Negative feedback keeps the reverse bias voltage of the photodiode close to zero.

minimum f-stop numbers and the deposition of microlenses on logarithmic sensors will boost the photocurrent relative to the leakage current.

In addition to optical methods, there may be electronic methods to reduce bias variation and magnitude by considering the reverse bias leakage current of the pixel photodiode. Advancements and tailoring of fabrication processes may offer some relief but so may novel circuit designs. The reverse bias leakage current is a function of reverse bias voltage, although the photocurrent is not. Therefore, keeping this voltage as close to zero as possible would help. Figure 8.1 shows a pixel circuit that may achieve this. It is composed of a double current mirror (DCM), where $\mathrm{T0}$ and $\mathrm{T0}'$ form an NMOS current mirror, $\mathrm{T1}$ and $\mathrm{T1}'$ form a PMOS current mirror and transistors $\mathrm{T2}$ and $\mathrm{T3}$ belong to the first stage readout. Both current mirrors operate in weak inversion. For the current in the left side of the circuit i_L to equal the current in the right side i_R, as required approximately by the PMOS current mirror, the gate-source voltage of $\mathrm{T0}$ must approximately equal the gate-source voltage of $\mathrm{T0}'$. Thus, the source voltage of $\mathrm{T0}$, which is the reverse bias voltage of the photodiode, will be kept approximately at zero by feedback. The current i_R, therefore, will consist of photocurrent with a minimal amount of leakage current. Note that the diode connected PMOS transistor, i.e. $\mathrm{T1}$, replaces the diode connected NMOS transistor in the conventional logarithmic pixel circuit of Figure 4.1. As before, this transistor is designed to operate in weak inversion over the expected range of photocurrent.

The pixel circuits in Figures 4.1 and 8.1 were simulated, as well as a pixel circuit similar to Figure 4.1 but with a PMOS instead of an NMOS load. The simulation covered six decades of photocurrent for a $0.35\mu\mathrm{m}$ $3.3\mathrm{V}$ AMS process, where the widths and lengths of all transistors were set to $1\mu\mathrm{m}$ and $0.6\mu\mathrm{m}$ respectively. Figure 8.2 plots the pixel drive voltage, i.e. at the gate of $\mathrm{T2}$, versus photocurrent. The figure shows that the use of a PMOS instead of an NMOS load for logarithmic conversion results

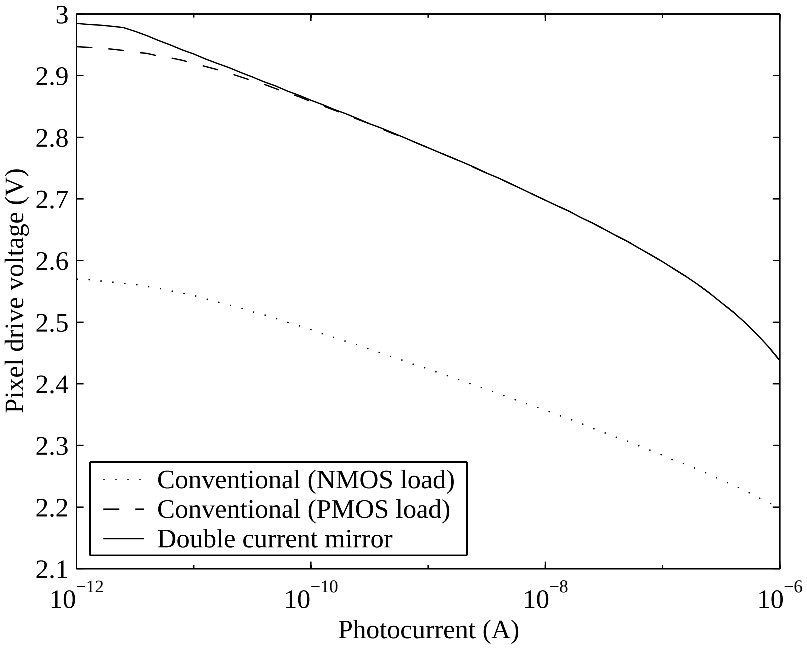


Figure 8.2: Simulated pixel drive voltage V_G^{T2} with respect to photocurrent I_P for the conventional pixel circuit of Figure 4.1, with an NMOS or PMOS load, and the DCM pixel circuit of Figure 8.1 (with a PMOS load).

in a higher gain—a subthreshold slope of 82 instead of $65\mathrm{mV}$ per decade. As may be shown with simulation, the reason is because the PMOS load has source and bulk nodes at the same potential (the bulk node is not shown in Figure 8.1) whereas the NMOS load does not (in the p-sub process). A higher gain in the pixel means a higher signal relative to subsequent noise introduced by the readout and ADC circuit. Figure 8.2 also shows that the subthreshold slope decreases at low photocurrents, starting at about $100\mathrm{pA}$ for the conventional pixels. This is more obvious in the figure with the PMOS load as its response deviates from that of the DCM pixel. However, the response with the NMOS load has the same shape. The subthreshold slope of the DCM pixel begins to decrease at about $3\mathrm{pA}$, which means it has better sensitivity in dim lighting. Note that the responses of the two pixels with PMOS loads exhibits a strong inversion effect for photocurrents greater than $0.1\mu\mathrm{A}$, unlike that of the NMOS load. The reason is because of a lower mobility of holes compared to electrons, which means the on-current of a PMOS transistor is lower than that of an equally sized NMOS transistor. Extension of the logarithmic range may be achieved using wider or shorter PMOS devices but note that the DCM pixel makes up for the loss at low photocurrents.

Lastly, Figure 8.3 plots the reverse bias voltage of the photodiode for each of the pixel simulations. As can be seen, only the DCM pixel has a reverse bias voltage close

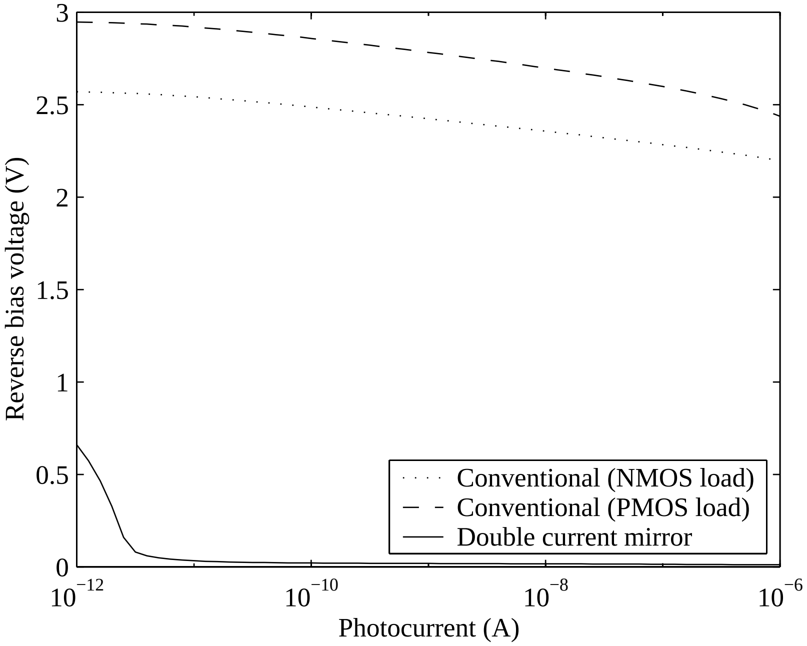


Figure 8.3: Simulated reverse bias voltage V_P of the photodiode with respect to photocurrent I_P for the conventional pixel circuit of Figure 4.1, with an NMOS or PMOS load, and the DCM pixel circuit of Figure 8.1 (with a PMOS load).

to zero over the wide photocurrent range (the conventional pixels do not even have constant reverse bias voltages). The increase in reverse bias voltage at low photocurrents may indicate that some current is always needed for correct operation of the feedback mechanism. When the photocurrent is too low for the circuit in Figure 8.1 to work, the reverse bias voltage is allowed to increase. However, the simulation may not be reliable for small currents. The effect of device parameter variation on the feedback needs consideration but it may not be reliably assessed with a Monte Carlo simulation due to the small currents involved and the lack of stochastic variation of leakage currents as well as distance and layout considerations in the mismatch model. Transistors in the current mirrors may be laid out next to each other with good alignment. Therefore, the performance of the DCM pixel may best be judged with experiment. The obvious disadvantage of the DCM pixel, as compared to the conventional NMOS pixel, is that it requires three additional transistors and that two of them are PMOS, which requires an n-well in each pixel. However, a layout with a fill factor of 28% is possible for a $10\mu\text{m} \times 10\mu\text{m}$ pixel in the AMS process, which is reasonable.

8.2.2 Readout circuit

As shown in this thesis, the readout circuit may contribute significantly to FPN, especially with a poor choice of switch position and with premature digitisation. These transient issues dominate the gain variation of the Fuga 15RGB. However, the transient response of the readout circuit is not the only contribution to gain variation, as shown in Chapter 4 with the simulation results, where double variation proved to be the best model of steady state variation. Once bias variation is sufficiently reduced, a reduction of steady state gain variation would lead to further simplification of FPN calibration and correction and colour calibration and rendition. In theory, if both bias and gain variation were sufficiently minimised, no FPN calibration would be necessary as offset variation would be adequately corrected by subtracting the dark response of a pixel from the light response, which would also compensate for temperature and aging effects. Steady state gain variation may be attributed to two sources: a variation of the subthreshold slope of the pixel load, which was considered in Chapter 4, and a variation of the small signal gain of the readout stages, which was not considered in this thesis. Preliminary simulations suggest the latter is more significant.

Optimisation of the readout stages will reduce their small signal gain variation. Preliminary analysis indicates that the switch transistor plays a significant role in any optimisation. When the width of the switch transistor is small (or the length is large), its drain-source resistance in the on-state is high, which has a nonlinear effect on the response as follows. For the first or second stage, the bus voltage of the source follower depends on the voltage across the switch but the voltage across the switch itself depends on the bus voltage because the on-resistance of the switch is set by the gate-source voltage of the transistor. Furthermore, the on-resistance and nonlinear response will vary with the threshold voltage of the switch, causing a small signal gain variation. Increasing the width of all switch transistors to achieve smaller on-resistances also increases the source-bulk junction capacitances, which increases the settling time of the readout stage. Therefore, optimisation must balance the desire to minimise gain variation with the desire to have small transistors and short settling times. Performance may improve with shrinking feature sizes, as decreasing the length of the switch reduces the on-resistance and transistor size without increasing the junction capacitance.

Another contribution to steady state gain variation is a column-to-column variation of the current source that biases the source follower of the parallel first stage readouts. The magnitude of the current is determined by the gate-source voltage and size of transistor $\mathrm{T}4$ in the circuit of Figure 4.1. A variation of the threshold voltage of $\mathrm{T}4$ from column to column causes the current to vary, which in turn causes the small signal gain to vary. Such variation may be minimised by designing the circuit to operate with a large value of the gate-source voltage so that the degree of current variation relative to threshold voltage variation is reduced.

Readout circuits that do not involve source followers may be considered to see if they provide a faster response and/or a lower gain variation. One circuit that may have a fast response and a near unity gain (the source follower gain is less than unity because of the body effect) is the differential amplifier with feedback [13], given in Figure 8.4. In this readout circuit, as drawn for the first stage, $\mathrm{T}2$ and $\mathrm{T}3$ are part of a pixel circuit as before. The remaining transistors are found at the end of each column. The

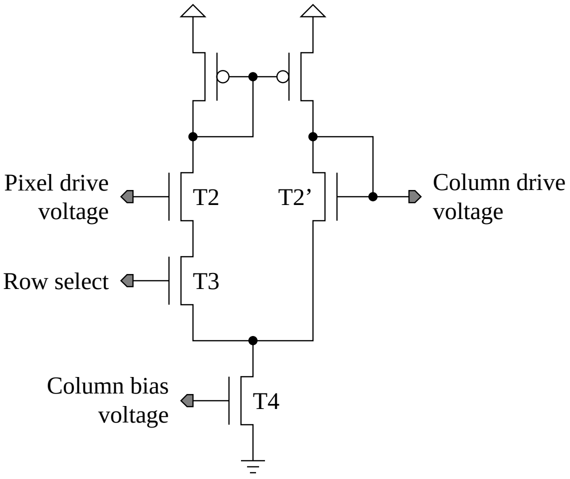


Figure 8.4: The first stage readout implemented with a unity gain differential amplifier instead of a source follower. Transistors T2 & T3 belong to the pixel circuit. The remaining transistors, including the PMOS current mirror, belong to the column circuit.

differential amplifier is biased by the current source T4 and its output, i.e. the drain of T2′, is connected to one of its inputs, i.e. the gate of T2′, to achieve negative feedback. As the current in the left half of the circuit approximately equals the current in the right half of the circuit, due to the PMOS current mirror, the gate-source voltage of T2′ approximately equals the gate-source voltage of T2, assuming the on-resistance of the switch T3 is negligible. Since the source voltages of T2 and T2′ are also approximately equal, when there is a negligible voltage across T3, the gate voltage of T2′ follows the gate voltage of T2 with near unity gain. In practice, the circuit may exhibit gain variation from pixel to pixel and column to column due to nonidealities of the switch T3 and because, with many transistors, there are many device parameters that may vary from column to column. However, the differential amplifier with feedback may be ideal for the second stage readout as it may be optimised to achieve a better transient response than the simple source follower with less concern for gain variation. In the second stage, which must operate at a much faster rate than the first stage, there is only one such amplifier driving the output bus and so all transistors, except for the input and switch, are common to the response of all pixels.

A careful theoretical analysis supplemented by Monte Carlo, DC and transient simulations will illustrate how to design the readout circuit so as to minimise steady state gain variation without spoiling the transient response. The best design will ultimately need confirmation with experiment.

8.2.3 Tone mapping

Once a high dynamic range image is captured, corrected for FPN and rendered into a standard colour space, a problem remains in terms of display. Standard displays are incapable of rendering more than two decades of illuminance [5]. A solution to this problem does not necessarily require a high dynamic range display although the development of such a display would enhance the sense of virtual reality. Rather, the high dynamic range image must be mapped to the low dynamic range display with a minimal loss of perceptual information in the process. This challenging task, called *tone mapping*, involves making bright objects darker and dark objects brighter while preserving the relative brightness of objects in the scene and the sensation of colour (for colour images). There is a biological precedent for tone mapping as the optic nerve does not carry as much dynamic range information as humans can perceive and therefore the eye itself accomplishes some of the tasks described above by dilation and constriction of the pupil, as the eye moves from fixation point to fixation point in a scene, and by adaptation of the retinal cells. However, these nonlinear aspects of scene perception are not understood as well as the linear aspects of light sensation.

Different approaches exist in the literature for tone mapping, such as homomorphic filtering, retinex filtering and histogram modification. Homomorphic and retinex filtering perform a two dimensional convolution operation on images to imitate lateral inhibition in the retina. While this operation may be useful for machine vision, it may be argued that spatial filtering is uneccessary for images displayed to human observers as such processing, which includes edge enhancement, would occur during observation of the display. Therefore, any tone mapping must consider the subsequent processing likely to occur with human perception. Only the processing that would fail to occur, because of the limited dynamic range of the display, needs to be reproduced artificially.

Histogram based approaches show a lot of promise in the mapping of high dynamic range images to low dynamic range displays. Figure 8.5 shows a high dynamic range image, displayed with only two decades of illuminance, using four types of histogram processing. The first mapping shows only the central two decade range of illuminances, with saturated patches for other illuminances. The second mapping compresses the high dynamic range to two decades with a gamma function so that the minimum and maximum illuminances of the recorded image correspond to the minimum and maximum illuminances of the displayed image. This approach tends to obscure perceptible detail when most illuminances cover a narrow range as it chooses a mapping based on extreme illuminances. A third approach is histogram equalisation, which applies a monotonic function to pixel responses so that the displayed image uses available illuminances equally. The disadvantage of this approach is that it exaggerates contrast when most illuminances cover a narrow range and it may increase the visibility of noise.

The best approach shown in Figure 8.5 is taken from Larson et al's work on rendering of computer generated images for the visualisation of architectural designs [5]. This application involves a simulation of the light field encountered at a viewpoint in a virtual world, using ray tracing with models of illuminant sources and object reflectances. The light field is computed in a standard colour space and often contains a high dynamic range of illuminances, which are impossible to display. In essence, this is the same problem encountered when displaying an image taken with a high dynamic

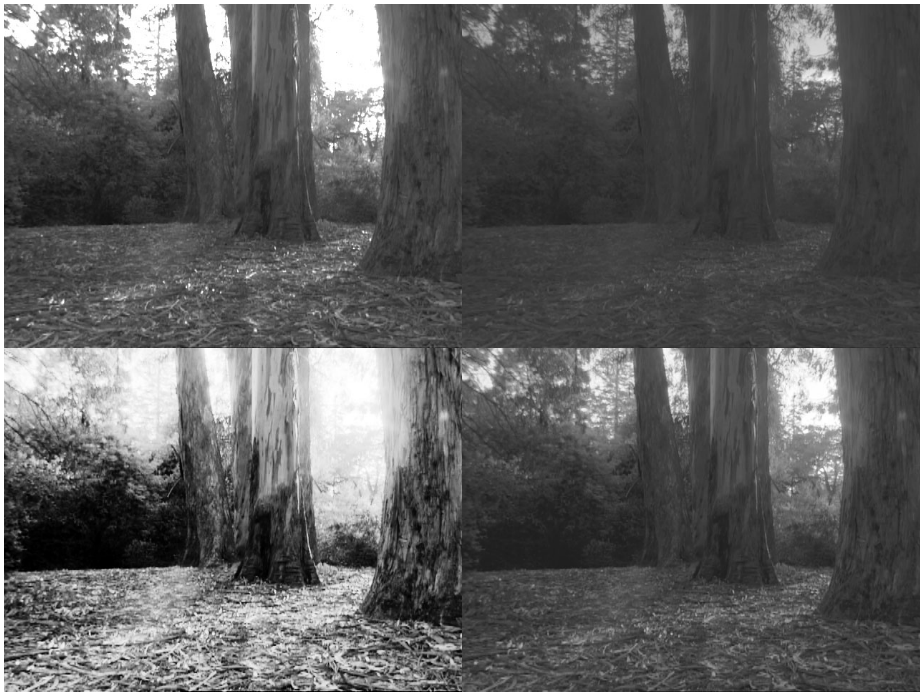


Figure 8.5: Tone mapping of a high dynamic range image using histogram clipping (top left), gamma compression (top right), histogram equalisation (bottom left) and Larson et al's method (bottom right) [5].

range sensor that is calibrated to a standard colour space. Larson et al developed an algorithm, based on human vision, to map such images to an image for a standard display, in a manner that simulates a direct observation of the scene. This algorithm is a histogram method because the same monotonic function is applied to the response of every pixel. In one sense, it is like histogram equalisation because it tries to equalise the available display illuminances. However, the algorithm prevents the displayed contrast from exceeding the contrast in the original image. Future work will investigate this and other approaches to map high dynamic range images, taken with logarithmic CMOS image sensors, to low dynamic range displays.

Bibliography

[1] Dileepan Joseph, Lionel Tarassenko, and Steve Collins, “Analysis and simulation of a cascaded delta delta-sigma modulator,” *Computer Standards & Interfaces*, vol. 23, no. 2, pp. 103–10, May 2001.

[2] Dileepan Joseph and Steve Collins, “Modelling, calibration and correction of nonlinear illumination-dependent fixed pattern noise in logarithmic CMOS image sensors,” in *Proceedings of the 18th IEEE Instrumentation and Measurement Technology Conference*, May 2001, vol. 2, pp. 1296–301, Rediscovering Measurement in the Age of Informatics.

[3] Dileepan Joseph and Steve Collins, “Modelling, calibration and rendition of colour logarithmic CMOS image sensors,” in *Proceedings of the 19th IEEE Instrumentation and Measurement Technology Conference*, May 2002, vol. 1, pp. 49–54, The Frontier of Instrumentation and Measurement.

[4] IMS Chips, “HDRC VGA Imager and Camera Data and Features,” Tech. Rep., Institute for Microelectronics Stuttgart, Sept. 2000.

[5] Gregory Ward Larson, Holly Rushmeier, and Christine Piatko, “A Visibility Matching Tone Reproduction Operator for High Dynamic Range Scenes,” *IEEE Transactions on Visualization and Computer Graphics*, vol. 3, no. 4, pp. 291–306, Oct.–Dec. 1997.

[6] Bart Dierickx, “The Human eye versus Silicon,” Tech. Rep., Interuniversity MicroElectronics Center, Aug. 1999, Presented at the 1997 IEEE Workshop on CCD & AIS.

[7] Tarek Lulé, Stephan Benthien, Holger Keller, Frank Mütze, Peter Rieve, Konstantin Seibel, Michael Sommer, and Markus Böhm, “Sensitivity of CMOS Based Imagers and Scaling Perspectives,” *IEEE Transactions on Electron Devices*, vol. 47, no. 11, pp. 2110–22, Nov. 2000.

[8] Joseph J. Atick and A. Norman Redlich, “Towards a Theory of Early Visual Processing,” *Neural Computation*, vol. 2, pp. 308–20, 1990.

[9] Carver Mead, *Analog VLSI and Neural Systems*, Addison-Wesley Publishing Company, USA, 1989.

[10] Gillian F. Marshall and Steve Collins, "A High Dynamic Range Front End for Automatic Image Processing Applications," in *Proceedings of the SPIE*, May 1998, vol. 3410, pp. 176–85, Advanced Focal Plane Arrays and Electronic Cameras II.

[11] Jim Giles, "Think like a bee," *Nature*, vol. 410, pp. 510–2, 29 March 2001.

[12] Arch C. Luther, *Video Camera Technology*, Artech House, Boston, 1998.

[13] Terry Zarnowski, Tom Vogelsong, and Jeff Zarnowski, "Inexpensive Image Sensors Challenge CCD Supremacy," *Photonics Spectra*, pp. 188–92, May 2000.

[14] Keith Diefendorff, "CMOS Image Sensors Challenge CCDs," *Microprocessor Report*, pp. 1–5, 22 June 1998, MicroDesign Resources.

[15] Albert J. P. Theuwissen, "CCD or CMOS Image Sensors for Consumer Digital Still Photography," in *2001 International Symposium on VLSI Technology, Systems, and Applications*, Apr. 2001, pp. 168–71.

[16] Nicolas Mokhoff, "CMOS image chips beg fab questions," *EE Times*, 6 Feb. 2001.

[17] Chappell Brown, "CMOS design challenges high-end sensor market," *EE Times*, 3 April 2001.

[18] Hui Tian, Boyd Fowler, and Abbas El Gamal, "Analysis of Temporal Noise in CMOS Photodiode Active Pixel Sensor," *IEEE Journal of Solid-State Circuits*, vol. 36, no. 1, pp. 92–101, Jan. 2001.

[19] Hon-Sum Wong, "Technology and Device Scaling Considerations for CMOS Imagers," *IEEE Transactions on Electron Devices*, vol. 43, no. 12, pp. 2131–42, Dec. 1996.

[20] Sunetra K. Mendis, Sabrina E. Kemeny, Russell C. Gee, Bedabrata Pain, Craig O. Staller, Quiesup Kim, and Eric R. Fossum, "CMOS Active Pixel Image Sensors for Highly Integrated Imaging Systems," *IEEE Journal of Solid-State Circuits*, vol. 32, no. 2, pp. 187–97, Feb. 1997.

[21] Markus Loose, Karlheinz Meier, and Johannes Schemmel, "CMOS image sensor with logarithmic response and self calibrating fixed pattern noise correction," in *Proceedings of the SPIE*, May 1998, vol. 3410, pp. 117–27, Advanced Focal Plane Arrays and Electronic Cameras II.

[22] Markus Loose, Karlheinz Meier, and Johannes Schemmel, "A Self-Calibrating Single-Chip CMOS Camera with Logarithmic Response," *IEEE Journal of Solid-State Circuits*, vol. 36, no. 4, pp. 586–96, Apr. 2001.

[23] Nico Ricquier and Bart Dierickx, "Active Pixel CMOS Image Sensor with On-Chip Non-Uniformity Correction," in *IEEE Workshop on Charge-Coupled Devices and Advanced Image Sensors*, Apr. 1995.

[24] Danny Scheffer, Bart Dierickx, and Guy Meynants, “Random Addressable 2048×2048 Active Pixel Image Sensor,” *IEEE Transactions on Electron Devices*, vol. 44, no. 10, pp. 1716–20, Oct. 1997.

[25] Orly Yadid-Pecht, “Wide-dynamic-range sensors,” *Optical Engineering*, vol. 38, no. 10, pp. 1650–60, Oct. 1999.

[26] Muahel Tabet, Nick Tu, and Richard Hornsey, “Modeling and characterization of logarithmic complementary metal-oxide-semiconductor active pixel sensors,” *Journal of Vacuum Science & Technology A*, vol. 18, no. 3, pp. 1006–9, May–June 2000.

[27] Spyros Kavadias, Bart Dierickx, and Danny Scheffer, “On-chip offset calibrated logarithmic response image sensor,” in *IEEE Workshop on Charge-Coupled Devices and Advanced Image Sensors*, June 1999, pp. 68–71.

[28] Spyros Kavadias, Bart Dierickx, Danny Scheffer, Andre Alaerts, Dirk Uwaerts, and Jan Bogaerts, “A Logarithmic Response CMOS Image Sensor with On-Chip Calibration,” *IEEE Journal of Solid-State Circuits*, vol. 35, no. 8, pp. 1146–52, Aug. 2000.

[29] Daniel J. Jobson, Zia ur Rahman, and Glenn A. Woodell, “A Multiscale Retinex for Bridging the Gap Between Color Images and the Human Observation of Scenes,” *IEEE Transactions on Image Processing*, vol. 6, no. 7, pp. 965–76, July 1997.

[30] Robert M. Boynton, *Human Color Vision*, University of California, San Diego, 1979.

[31] David X. D. Yang, Abbas El Gamal, Goyd Fowler, and Hui Tian, “A 640×512 CMOS Image Sensor with Ultrawide Dynamic Range Floating-Point Pixel-Level ADC,” *IEEE Journal of Solid-State Circuits*, vol. 34, no. 12, pp. 1821–34, Dec. 1999.

[32] Bart Dierickx, Danny Scheffer, Guy Meynants, Werner Ogiers, and Jan Vlummens, “Random addressable active pixel image sensors,” in *Proceedings of the SPIE*, Oct. 1996, vol. 2950, pp. 2–7, Advanced Focal Plane Arrays and Electronic Cameras.

[33] B. Hoefflinger, H.-G. Graf, U. Seger, and A. Siggelkow, “Imager for Robust High-Speed Vision,” in *Proceedings for the Dedicated Conference on Robotics, Motion and Machine Vision in the Automotive Industries*, Sept. 1995, pp. 289–93, 28th International Symposium on Automotive Technology and Automation.

[34] C-Cam Technologies, *Introduction software for Fuga RGB*, Vector International, 30 April 1998.

[35] C-Cam Technologies, “Fuga Data Sheets,” Tech. Rep., Vector International, 3 April 1998.

[36] Bart Dierickx, “RE: Fuga 15d query,” Electronic mail, 22 July 2002.

[37] Daniel G. Antzoulatos and Alexander A. Sawchuk, “Hypermatrix Algebra: Theory,” *CVGIP: Image Understanding*, vol. 57, no. 1, pp. 24–41, Jan. 1993.

[38] G. Blaha, “A few basic principles and techniques of array algebra,” *Bulletin Geodesique*, vol. 51, no. 3, pp. 177–202, 1977.

[39] Richard A. Snay, “Applicability of Array Algebra,” *Reviews of Geophysics and Space Physics*, vol. 16, no. 3, pp. 459–64, Aug. 1978.

[40] Richard A. Strelitz, “Moment tensor inversions and source models,” *Geophysical Journal of the Royal Astronomical Society*, vol. 52, no. 2, pp. 359–64, Feb. 1978.

[41] Mark S. Ghiorso, “LSEQIEQ: A FORTRAN IV subroutine package for the analysis of multiple linear regression problems with possibly deficient pseudorank and linear equality and inequality constraints,” *Computers & Geosciences*, vol. 9, no. 3, pp. 391–416, 1983.

[42] Jan R. Magnus and Heinz Neudecker, *Matrix Differential Calculus with Applications in Statistics and Econometrics*, John Wiley & Sons, Chichester, 1988.

[43] Daniel P. Foty, *MOSFET Modeling with SPICE: Principles and Practise*, Prentice Hall, Upper Saddle River, NJ, 1997.

[44] Austria Micro Systems, $0.35\mu\mathrm{m}$ *CMOS Process Parameters*, Document 9933016.

[45] Austria Micro Systems, $0.35\mu\mathrm{m}$ *CMOS Design Rules*, Document 9931032.

[46] Leonid Libkin, Rona Machlin, and Limsoon Wong, “A Query Language for Multidimensional Arrays: Design, Implementation, and Optimization Techniques,” *SIGMOD Record*, vol. 25, no. 2, pp. 228–39, June 1996.

[47] L. T. Milov, “Multidimensional matrix derivatives and sensitivity analysis of control systems,” *Automation and Remote Control*, vol. 40, no. 9.1, pp. 1269–77, Sept. 1979.

[48] Akimichi Takemura, “Tensor Analysis of ANOVA Decomposition,” *Journal of the American Statistical Association*, vol. 78, no. 384, pp. 894–900, Dec. 1983.

[49] Masakazu Suzuki and Kiyotaka Shimizu, “Analysis of distributed systems by array algebra,” *International Journal of Systems Science*, vol. 21, no. 1, pp. 129–55, Jan. 1990.

[50] J. H. Heinbockel, *Introduction to Tensor Calculus and Continuum Mechanics*, Old Dominion University, Norfolk, VA, 1996.

[51] J. L. Synge and A. Schild, *Tensor Calculus*, Dover Publications, New York, 1949.

[52] Peter Baumann, “A Database Array Algebra for Spatio-Temporal Data and Beyond,” in *Lecture Notes in Computer Science*, July 1999, vol. 1649, pp. 76–93, 4th International Workshop on Next Generation Information Technologies and Systems.

[53] John R. Gilbert, Cleve Moler, and Robert Schreiber, “Sparse Matrices in MATLAB: Design and Implementation,” *SIAM Journal on Matrix Analysis and Applications*, vol. 13, no. 1, pp. 333–56, Jan. 1992.

[54] Richard L. Scheaffer and James T. McClave, *Probability and Statistics for Engineers*, Wadsworth Publishing Company, Belmont, CA, 1995.

[55] Hong Wang and Wansoo T. Rhee, “An algorithm for estimating the parameters in multiple linear regression model with linear constraints,” *Computers & Industral Engineering*, vol. 28, no. 4, pp. 813–21, Oct. 1995.

[56] G. von Fuchs, J. R. Roy, and E. Schrem, “Hypermatrix solution of large sets of symmetric positive-definite linear equations,” *Computer Methods in Applied Mechanics and Engineering*, vol. 1, no. 2, pp. 197–216, Aug. 1972.

[57] Christopher M. Bishop, *Neural Networks for Pattern Recognition*, Oxford University Press, Oxford, 1995.

[58] Kenneth R. Laker and Willy M. C. Sansen, *Design of analog integrated circuits and systems*, McGraw-Hill, Singapore, 1994.

[59] Asim Kumar Roy Choudhury, *Modern Concepts of Colour and Appearance*, Science Publishers, Enfield, NH, 2000.

[60] International Electrotechnical Commision, *Default RGB colour space—sRGB*, Oct. 1999, Document 61966.

[61] C. S. McCamy, H. Marcus, and J. G. Davidson, “A Color-Rendition Chart,” *Journal of Applied Photographic Engineering*, vol. 2, no. 3, pp. 95–9, Summer 1976.

[62] Mike McNamee, “A Snapshot in Time,” *Digital Photographer*, vol. 13, pp. 32–8, July 1999.