VIDEO PROCESSING APPLICATIONS OF HIGH SPEED CMOS IMAGE SENSORS

A DISSERTATION
SUBMITTED TO THE DEPARTMENT OF ELECTRICAL ENGINEERING
AND THE COMMITTEE ON GRADUATE STUDIES
OF STANFORD UNIVERSITY
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

Suk Hwan Lim
March 2003

I certify that I have read this dissertation and that in my opinion it is fully adequate, in scope and quality, as a dissertation for the degree of Doctor of Philosophy.

[illegible]

Abbas El Gamal
(Principal Adviser)

I certify that I have read this dissertation and that in my opinion it is fully adequate, in scope and quality, as a dissertation for the degree of Doctor of Philosophy.

[illegible]

Teresa Meng

I certify that I have read this dissertation and that in my opinion it is fully adequate, in scope and quality, as a dissertation for the degree of Doctor of Philosophy.

[illegible]

Brian Wandell

I certify that I have read this dissertation and that in my opinion it is fully adequate, in scope and quality, as a dissertation for the degree of Doctor of Philosophy.

[illegible]

John Apostolopoulos

Approved for the University Committee on Graduate Studies:

Abstract

An important trend in the design of digital cameras is the integration of capture and processing onto a single CMOS chip. Although integrating the components of a digital camera system onto a single chip significantly reduces system size and power, it does not fully exploit the potential advantages of integration. We argue that a key advantage of integration is the ability to exploit the high speed imaging capability of CMOS image sensors to enable new applications and to improve the performance of existing still and video processing applications. The idea is to capture frames at much higher frame rates than the standard frame rate, process the high frame rate data on chip, and output the video sequence and the application specific data at standard frame rate.

In the first part of the dissertation we discuss two applications of this idea. The first is optical flow estimation, which is the basis for many video applications. We present a method for obtaining high accuracy optical flow estimates at a standard frame rate by capturing and processing a high frame rate version of the video, and compare its performance to methods that only use standard frame rate sequences. We then present a method that uses a video sequence and accurate optical flow estimates to correct sensor gain Fixed Pattern Noise (FPN). Simulation and experimental results demonstrate that significant reduction in gain FPN can be achieved using our method.

In the second part of the dissertation we discuss hardware implementation issues of high speed CMOS imaging systems. We designed, fabricated and tested a 352×288 pixel CMOS Digital Pixel Sensor chip with per-pixel single-slope ADC and 8-bit dynamic memory in a standard digital 0.18μm CMOS process. The chip performs “snap-shot” image acquisition at continuous rate of $10,000$ frames/s or 1 Gpixels/s. We then discuss the projected limits of integrating memory and processing with a CMOS image sensor in 0.18μm process and below. We show that the integration of an entire video camera system on a chip is not only feasible at 0.18μm process, but in fact underutilizes the possible on-chip processing power. Further, we show that the projected available on-chip processing power and memory are sufficient to perform applications such as optical flow estimation.

Acknowledgments

There is always too little to space when it comes to thanking all the people whom I spent considerable time together. Although some times were a little tough, I had wonderful time during my graduate studies at Stanford University and I am deeply indebted to many people.

First, I would like to thank my adviser Prof. Abbas El Gamal. It has been a great pleasure and honor to be one of his students. All this work would not have been possible without without his guidance and support. Especially, his broad knowledge in many areas of electrical engineering enabled me to obtain great academic advice from a wide range of fields. It also inspired me to conduct research in a broader way.

I would also like to thank professors who served on my orals and dissertation committee. I am grateful to Prof. Teresa Meng, my associate adviser, for her generous support and guidance especially during early part of my graduate studies. I want to thank Prof. Brian Wandell who led the Programmable Digital Camera (PDC) project together with Prof. El Gamal. It was almost like having another principal adviser. I am very grateful for that and it was such a great pleasure to work with him. I also would like to thank Dr. John Apostolopoulos. Part of my optical flow estimation research conducted done while I was taking his video processing class. Not only did he give me great advice for my research, but has been and is a great mentor for me. I truly appreciate it.

Also, I want to thank Prof. Tomasi, Dr. Fleet, Prof. Heeger and Prof. Godfrey for valuable discussions and comments. I would also like to thank my former and current group members. Dr. David Yang, Dr. Boyd Fowler, Dr. Hui Tian, Dr. Xinqiao Liu, Ting Chen, Khaled Salama, Helmy Eltoukhy, Sam Kavusi, Ali Ozer, Sina Zahedi and

Hossein Kakavand. I am reluctant to call them as group members since they are more like friends. They made my life at Stanford pleasing and enjoyable. I would also like to thank Charlotte Coe, Kelly Yilmaz and Denise Cuevas for their assistance.

I also like to thank the sponsors of PDC project for funding major part of my graduate studies. I would like to thank DARPA for partially supporting me during my graduate studies in Stanford.

I wish to thank all my friends at Stanford with whom I had lots of fun with. To name a few, Eun Sun Park, Haktae Lee, SoYoung Kim, Gwang Hyun Gho, Paul Wang Lee, JiHi Jung, Jaewook Kim, Nahmsuk Oh, Eui-Young Chung, Sunghee Yun, Mario Mendoza, Mai-Sie Chan, Xin Tong, Wonjong Rhee and Sangeun Han. They all have been great in helping me out and I was able to have lots of fun with them. I would also like to thank all my friends both in U.S. and Korea for their company and support, especially my Seoul Science High School alumni friends and HwaHyunHoi friends.

Last, but definitely not the least, I would like to thank my family. I would never be where I am if it were not for their unconditional love and support. I dedicate my thesis to my family.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Digital cameras and image sensors

Digital still and video cameras are rapidly becoming ubiquitous, due to reduced costs and increasing demands of multimedia applications. Digital still cameras are especially becoming very popular and are rapidly replacing analog and film cameras. Although replacing film cameras is one of many approaches one can take for digital imaging, it does not fully exploit the capabilities of digital imaging. Especially with the emergence of CMOS image sensors, digital still and video cameras enable many new imaging applications such as machine vision, biometrics and image-based rendering. Moreover, with miniaturization and cost reduction, image sensors can be embedded in virtually every multimedia system such as PC-based web cameras, cell phones, PDAs, games and toys.

Every digital imaging system employs an image sensor which converts light signals into electrical signals. The image sensor plays a pivotal role in the final image quality. Most of digital cameras today use the charge-coupled devices (CCDs) to implement the image sensors [1]-[8]. In the CCD image sensors, incident photons are converted to charge which are then accumulated by the photodetectors during exposure time. During the following readout, the accumulated charge in the array is sequentially transferred into the vertical and horizontal CCDs and finally shifted out to chip level output amplifier where it is converted to voltage signal as is shown in Figure 1.1. CCDs generally consume lots of power because of high capacitance and high switching frequency. Since CCD image sensor is fabricated using specialized process with optimized photodetectors, it has very low noise and good uniformity but cannot be integrated with memory and processing which are typically implemented in CMOS technology. Thus, a typical digital camera system today (see Figure 1.2) employs a CCD image sensor and several other chips for analog signal generation, A/D conversion, digital image processing and compression, control, interface, and storage.

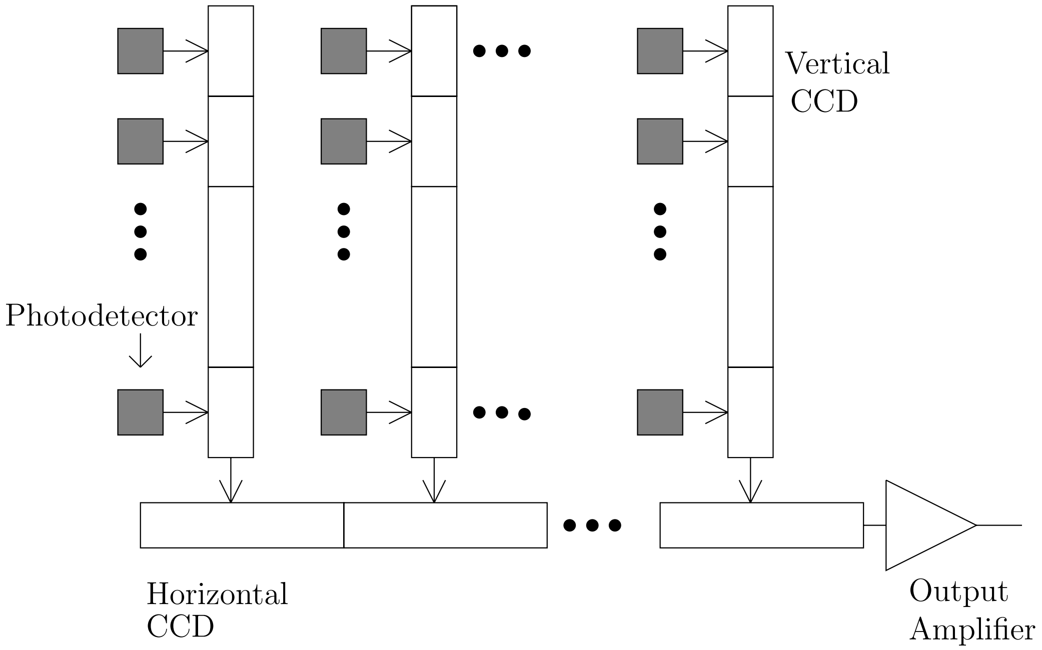


Figure 1.1: Block diagram of a typical CCD image sensors.

In contrast, recently developed CMOS image sensors are fabricated using standard CMOS process with no or minor modification [9]-[11]. Similar to CCDs, incident photons are converted to charge which are then accumulated by the photodetectors during exposure time. Unlike CCDs, however, charge (or voltage) in the pixel array are read out using the row decoders and column amplifiers and multiplexers. This readout scheme is similar to a memory structure. Currently, there are three pixel architectures for CMOS image sensors: Passive Pixel Sensor (PPS), Active Pixel

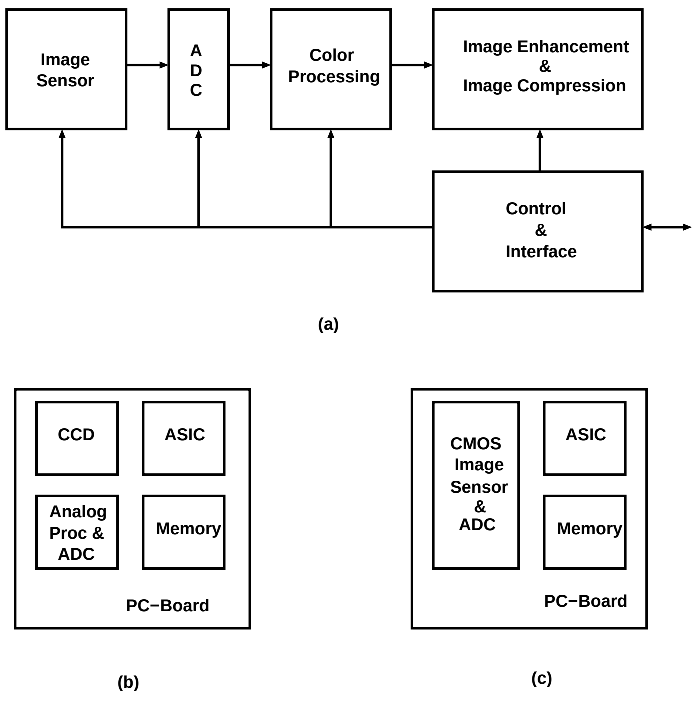


Figure 1.2: Digital Camera System: (a) functional block diagram, (b) implementation using CCD, and (c) implementation using CMOS image sensor.

Sensor (APS) and Digital Pixel Sensor (DPS). PPS [12]-[18] has only one transistor per pixel, as shown in Figure 1.3. The charge in each pixel is read out via a column charge amplifier located outside of the pixel array. Although PPS has small pixel size and large fill factor, it suffers from slow readout speed and low SNR. APS [19]-[33] tries to solve these problems by having a buffer in each pixel, which is normally implemented with three or four transistors (see Figure 1.4). In comparison to PPS, APS has larger pixel size and lower fill factor, but its readout is faster and has higher SNR. In DPS, each pixel has an ADC and all ADCs operate in parallel as shown in Figure 1.5. With an ADC per pixel, massively parallel A/D conversion and high speed digital readout become practical, eliminating analog A/D conversion and readout bottlenecks. The main drawback of DPS is its large pixel size due to the increased number of transistors per pixel, which is less problematic as CMOS technology scales down to 0.18μm and below.

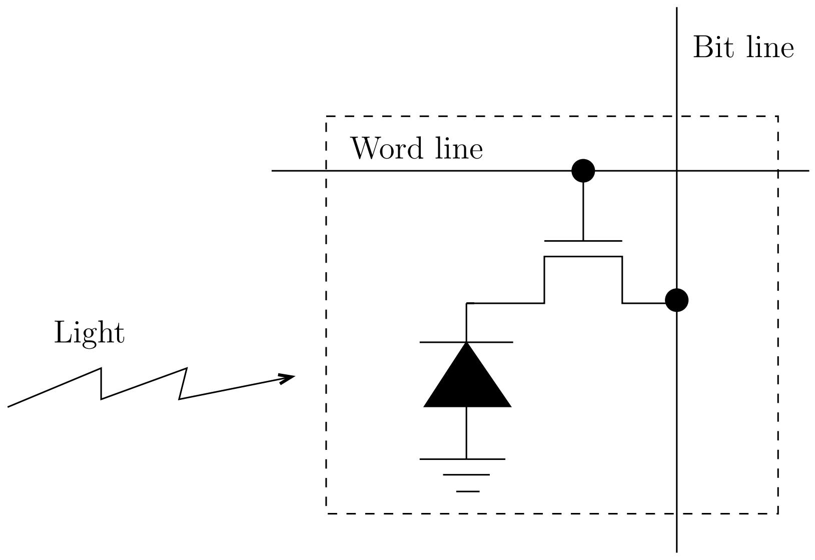


Figure 1.3: Passive pixel sensor (PPS)

Regardless of the architecture, current CMOS image sensors typically have lower image quality and higher noise level than CCD image sensor, mainly because the fabrication process cannot be optimized for image sensing. Moreover, it has higher fixed pattern noise since image data are read out through different chains of buffers

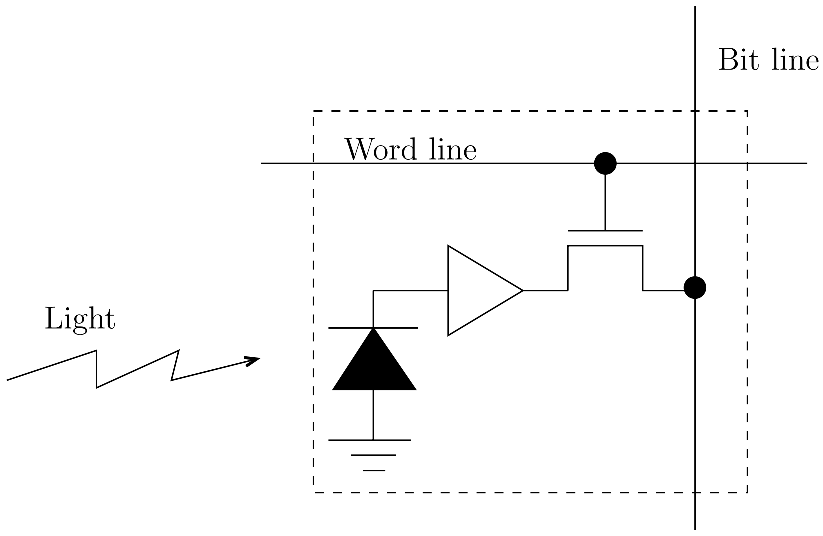


Figure 1.4: Active Pixel Sensor (APS)

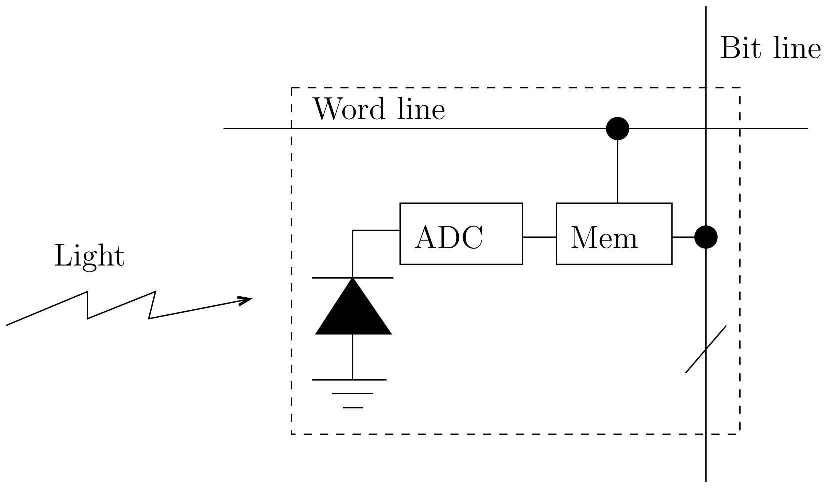


Figure 1.5: Digital Pixel Sensor (DPS)

and amplifiers. CMOS image sensors, however, have other unique advantages over CCDs. First, CMOS image sensors consume much less power than CCD image sensor due to lower voltage swing, switching frequency and capacitance. Second, integrating image sensing with A/D conversion, memory and processing on a single chip is possible for CMOS image sensor. Several researchers [34, 35, 36, 37] have exploited these advantages and have demonstrated low power consumption and reduction in chip count for a digital camera system by integrating the analog signal generation, A/D conversion, and some of the control and image processing with the sensor on the same chip. Loinaz *et al.* [35] describe a PC-based single chip digital color camera, which performs image capturing using a photogate APS, automatic gain control, an 8-bit full flash ADC, and all the computationally intensive pixel-rate tasks such as color interpolation, color correction, and image statistics computation. Smith *et al.* [36] describe a single chip CMOS NTSC video camera that integrates an APS, a half-flash sub-ranging ADC, and all the processing necessary to produce color NTSC video with only an external power supply and a crystal oscillator.

1.2 High frame rate capture – standard frame rate output

Commercially available PC camera chips now routinely integrate A/D conversion, gamma correction, exposure and gain control, color correction and white balance with a CMOS CIF and VGA size image sensor. As CMOS image sensors scale to 0.18μm processes and below, integration of the rest of the camera system becomes feasible resulting in true "camera-on-chip". Although integrating the camera system shown in Figure 1.2 onto a single chip can significantly reduce system size and power, it does not fully exploit the potential advantages of integration. In this dissertation we argue that a key advantage of integration is the ability to exploit the high speed imaging capability of CMOS image sensors. Several recent papers have demonstrated the high speed imaging capability of CMOS image sensors [38, 39, 40, 41]. Krymski *et al.* [38] describe a 1024×1024 Active Pixel Sensor (APS) with column level ADC achieving frame rate of 500 frames/s. Stevanovic *et al.* [39] describe 256 × 256 APS with 64 analog outputs achieving frame rate of 1000 frames/s. Kleinfelder *et al.* [40] describe a 352 × 288 Digital Pixel Sensor(DPS) with per pixel bit parallel ADC achieving 10,000 frames/s or 1 Giga-pixels/s.

The high speed imaging capability of CMOS image sensors can benefit conventional camera systems by enabling more efficient implementations of several applications such as motion estimation [42], video stabilization, and video compression, and of new applications such as multiple capture for enhancing dynamic range [43, 44, 45] and motion blur-free capture [46]. Digital still and video cameras, however, operate at low frame rates and it would be too costly, if not infeasible, to operate them at high frame rates due to the high output data rate requirements of the sensor, the memory, and the processing chips. Integrating the memory and processing with the sensor on the same chip solves the high output data rate problem and provides an economical way to exploit the high speed capability of a CMOS image sensor. The basic idea, which will be explored in this dissertation (see Figure 1.6 and Handoko *et al.* [42, 47]), is to (i) operate the sensor at a much higher frame rate than the standard frame rate, (ii) exploit the high on-chip bandwidth between the sensor, the memory and the processors to process the high frame rate data, and (iii) only output the images with any application specific data at the standard frame rate. Thus, off-chip data rate which is very important for the system cost is not increased although high frame rate sequences are used.

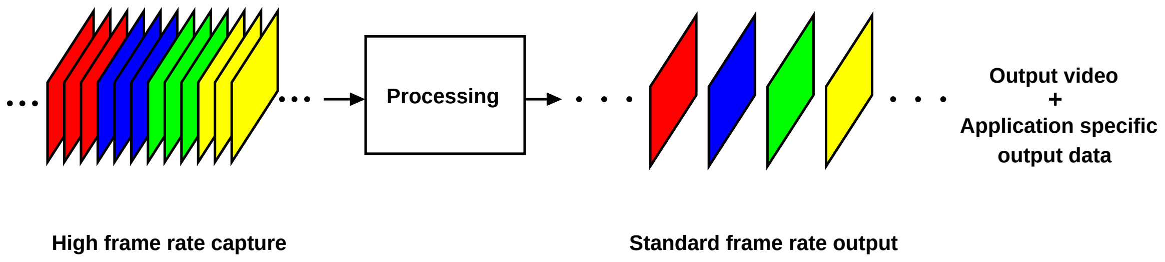


Figure 1.6: High frame rate capture – standard frame rate output.

Extending dynamic range and capturing motion blur-free images with this approach have been explored by several researchers [43, 44, 45, 46]. In those applications, video data at each pixel are processed temporally, independent of the neighboring pixels. This has limitations because the spatial information is not exploited. In this dissertation, we extend our high-frame-rate-capture/standard-frame-rate-output approach from 1D temporal processing to 3D spatio-temporal processing. This extension will enable more efficient implementations of several applications in video processing, computer vision and even image-based rendering. Moreover, it opens door to many new applications in those fields.

1.3 Thesis organization

The dissertation discusses both the hardware and algorithmic aspects of video processing applications using high speed imagers and can thus be divided into two main parts. The first part, which includes Chapter 2 and 3, describes video processing applications enabled by high speed imaging capability of CMOS image sensors. The applications described follow the basic approach described in the previous section. The second part, which is Chapter 4, is on hardware and implementation issues. We present a DPS chip that demonstrates the high speed imaging capability of CMOS image sensor and then show that implementing a high speed imaging system on a single chip is feasible. We next briefly summarize of each chapter.

Chapter 2 describes a method for obtaining high accuracy optical flow estimates at a standard frame rate by capturing and processing a high frame rate version of the video and compare its performance to methods that only use standard frame rate sequences. We demonstrate significant performance improvements over conventional optical flow estimation that use standard frame rate image sequences.

Chapter 3 describes a method that uses a video sequence and accurate optical flow estimates to correct sensor gain Fixed Pattern Noise (FPN). The captured sequence and its optical flow are used to estimate gain FPN. Assuming brightness constancy along the motion trajectories, the pixels are grouped in blocks and each block's pixel gains are estimated by iteratively minimizing the sum of the squared brightness variations along the motion trajectories. Significant reductions in gain FPN are demonstrated using both real and synthetically generated video sequences with modest computational requirements.

Chapter 4 discusses the hardware implementation issues of the high-speed imaging system. On the sensor side, a 352×288 pixel CMOS Digital Pixel Sensor chip with per-pixel single-slope ADC and 8-bit dynamic memory in a standard digital 0.18μm CMOS process is described. The chip performs "snap-shot" image acquisition at continuous rate of 10,000 frames/s or 1 Gpixels/s. The chip demonstrates the high speed imaging capability of CMOS image sensors. We then discuss the limits on memory size and processing power that can be integrated with a CMOS image sensor in 0.18μm process and below. We show that the integration of an entire video camera system on a chip is not only feasible at 0.18μm process, but in fact underutilizes the possible on-chip processing power. Further, we argue that the on-chip processing power and memory are sufficient to perform applications such as optical flow estimation by operating the sensor at high frame rate. As technology scales, applications that require even more processing power and memory such as tracking, pattern recognition, and 3D structure estimation may be implemented on a single chip.

Finally Chapter 5 concludes the thesis and discusses the most likely directions for future related research.

Chapter 2

Optical Flow Estimation

2.1 Introduction

A key problem in the processing of video sequences is estimating the motion between video frames, often referred to as optical flow estimation (OFE). Once estimated, optical flow can be used in performing a wide variety of tasks such as video compression, 3-D surface structure estimation, super-resolution, motion-based segmentation and image registration. Optical flow estimation based on standard frame rate video sequences, such as 30 frames/s, has been extensively researched with several classes of methods developed including gradient-based, region-based matching, energy-based, Bayesian, and phase-based. Excellent survey papers that briefly describe several classes of methods and compare the performance of the methods can be found in [48, 49, 50].

There are several benefits of using high frame rate sequences for OFE. First, as frame rate increases, the intensity values along the motion trajectories vary less between consecutive frames when illumination level changes or occlusion occurs. Since many optical flow estimation methods explicitly or implicitly assume that intensity along motion trajectories stay constant [48, 49, 50], it is expected that using high frame rate sequences can enhance the estimation accuracy of these algorithms. Another important benefit is that as frame rate is increased the captured sequence exhibits less motion aliasing. Indeed large errors due to motion aliasing can occur even when using the best optical flow estimators. For example, when motion aliasing occurs a wagon wheel might appear to rotate backward even to a human observer when seen through devices such as movie screen and TV. This specific example is discussed in more detail in Section 2.3. There are many instances when the standard frame rate of 30 frames/s is not sufficient to avoid motion aliasing and thus incorrect optical flow estimates. Note that motion aliasing not only depends on the velocities but also on the spatial bandwidths. Thus, capturing sequences at a high frame rate not only helps when velocities are large but also for complex images with low velocities but high spatial bandwidths.

This chapter is organized as follows. In Section 2.2 we present a method [51, 52] for accurate optical flow estimation at a standard frame rate from a high frame rate version of the video sequence. This method is based on the well-known Lucas-Kanade algorithm [53]. Using synthetic input sequences generated by image warping of a still image, we also show significant improvements in accuracy attained using the proposed method. We then examine the memory and computational requirements of the proposed method. In Section 2.3 we give a brief review of 3-D spatio-temporal sampling theory and the analyze the effects of temporal sampling rate and motion aliasing on OFE. We present simulation results using sinusoidal input sequences showing that the minimum frame rate needed to achieve high accuracy is largely determined by the minimum frame rate necessary to avoid motion aliasing. In Section 2.4 we discuss how the proposed method can be used with OFE algorithms other than the Lucas-Kanade algorithm. In particular, we extend the Haussecker algorithm [64] to work with high frame rate sequences and show that with this extension high accuracy optical flow estimates can be obtained even when brightness varies with time.

2.2 Optical flow estimation using high frame rate sequences

2.2.1 Proposed method

In this subsection we present a method for obtaining high accuracy optical flow estimates at a standard frame rate by capturing and processing a high frame rate version of the video. The idea is to estimate optical flow at a high frame rate and then carefully integrate it temporally to estimate the optical flow between frames at the slower standard frame rate. Temporal integration, however, must be performed without losing the accuracy gained by using the high frame rate sequence. Obviously, if the temporal integration does not preserve the accuracy provided by the high frame rate sequence, then this approach would lose many of its benefits.

The block diagram of our proposed method is shown in Figure 2.1 for the case when the frame rate is 3 times the standard frame rate. We define OV as the *oversampling factor* (i.e., the ratio of the capture frame rate to the standard frame rate) and thus $OV = 3$ in the block diagram. Consider the sequence of high-speed frames beginning with a standard-speed frame (shaded frame in the figure) and ending with the following standard-speed frame. We first obtain high accuracy optical flow estimates between consecutive high-speed frames. These estimates are then used to obtain a good estimate of the optical flow between the two standard-speed frames.

We first describe how optical flow at a high frame rate is estimated. Although virtually any OFE method can be employed for this stage, we decided to use a gradient-based method since higher frame rate leads to reduced motion aliasing and better estimation of temporal derivatives, which directly improve the performance of such methods. In addition, because of the smaller displacements between consecutive frames in a high-speed sequence, smaller kernel sizes for smoothing and computing gradients can be used, which reduces the memory and computational requirements of the method.

Of the gradient-based methods, we chose the well known Lucas-Kanade's algorithm [53], which was shown to be among the most accurate and computationally

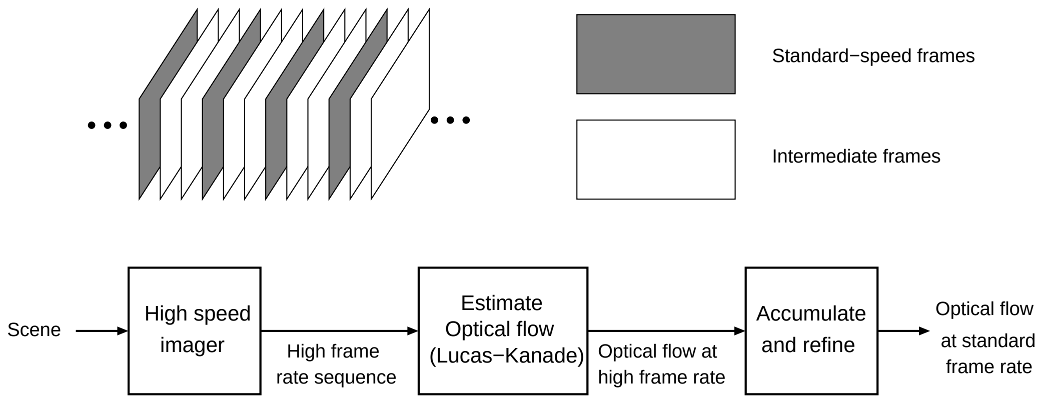


Figure 2.1: The block diagram of the proposed method ($OV = 3$).

efficient methods for optical flow estimation [48]. A block diagram of the Lucas-Kanade OFE method is shown in Figure 2.2. Each frame is first pre-filtered using a spatio-temporal low pass filter to reduce aliasing and systematic error in the gradient estimates. The gradients i_x, i_y, and i_t are typically computed using a 5-tap filter [48]. The velocity vector is then computed for each pixel (x, y) by solving the 2×2 linear equation

$$\begin{bmatrix} \sum wi_x^2 & \sum wi_xi_y \\ \sum wi_xi_y & \sum wi_y^2 \end{bmatrix} \begin{bmatrix} v_x \\ v_y \end{bmatrix} = - \begin{bmatrix} \sum wi_xi_t \\ \sum wi_yi_t \end{bmatrix}.$$

Note that we have not included the spatial parameters (x, y) in the formulation to simplify notation. Here $w(x, y)$ is a window function that assigns higher weight to the center of the neighborhood around (x, y) and the sums are typically over 5×5 pixel neighborhoods.

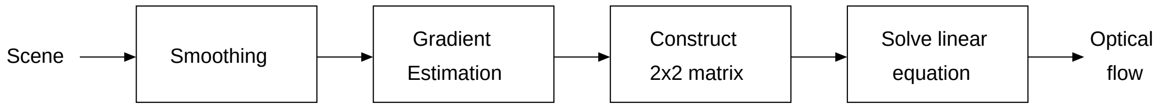


Figure 2.2: Block diagram of Lucas-Kanade method. Note that the last three blocks are performed for each pixel of each frame.

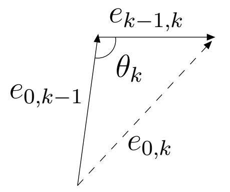
After optical flow has been estimated at the high frame rate, we use it to estimate the optical flow at the standard frame rate. This is the third block of the block diagram in Figure 2.1. The key in this stage is to integrate optical flow temporally without losing the accuracy gained using the high frame rate sequences. A straightforward approach would be to simply accumulate the optical flow estimates between consecutive high-speed frames along the motion trajectories. The problem with this approach is that errors can accumulate with the accumulation of the optical flow estimates. To understand how errors can accumulate for a pixel, consider the diagram in Figure 2.3, where $e_{k,l}$ is the magnitude of the OFE error vector between frames k and l. Assuming that θ_k, the angles between the error vectors in the figure are random and uniformly distributed and that the mean squared magnitude of the OFE error between consecutive high-speed frames are equal, i.e., $E(e^2_{j-1,j}) = E(e^2_{0,1})$ for $j = 1, \ldots, k$, the total mean-squared error is given by

$$\begin{aligned} E(e^2_{0,k}) &= E(e^2_{0,k-1}) + E(e^2_{k-1,k}) - 2E(e_{k-1,k}e_{0,k-1}\cos\theta_k) \\ &= \sum_{j=1}^{k} E(e^2_{j-1,j}) - 2\sum_{j=1}^{k} E(e_{j-1,j}e_{0,j-1}\cos\theta_j) \\ &= \sum_{j=1}^{k} E(e^2_{j-1,j}) = kE(e^2_{0,1}), \end{aligned}$$

which grows linearly with k. On the other hand, if the optical flow estimation errors are systematic, i.e., line up from one frame to the next, and their magnitudes are temporally independent, which yields $E(e_{j-1,j}e_{l-1,l}) = E(e_{j-1,j})E(e_{l-1,l})$, then the total mean-squared error is given by

$$\begin{aligned} E[e^2_{0,k}] &= E[e^2_{0,k-1} + e^2_{k-1,k} + 2e_{k-1,k}e_{0,k}] = E[(e_{0,k-1} + e_{k-1,k})^2] \\ &= E[(\sum_{j=1}^{k} e_{j-1,j})^2] = k^2E[e^2_{0,1}], \end{aligned}$$

which grows quadratically with k. In practice, the optical flow estimation error was shown to have a random component and a non-zero systematic component by several



$$e_{0,k}^2 = e_{0,k-1}^2 + e_{k-1,k}^2 - 2e_{k-1,k}e_{0,k}\cos\theta_k$$

Figure 2.3: The accumulation of error vectors when accumulating optical flow without using refinement.

researchers [54, 55, 56, 57], and as a result, the mean-squared error $E[e_{0,k}^2]$ is expected to grow faster than linear but slower than quadratic in k.

To prevent this error accumulation, we add a refinement (or correction) stage after each iteration (see Figure 2.4). We obtain frame $\hat{k}$ by warping frame 0 according to our accumulated optical flow estimate $\tilde{d}$, and assume that frame k is obtained by warping frame 0 according to the true motion between the two frames, (which we do not know). By estimating the displacement between frames k and $\hat{k}$, we can estimate the error between the true flow and the initial estimate $\tilde{d}$. In the refinement stage, we estimate this error and add it to the accumulated optical flow estimate. Although the estimation of the error is not perfect, we found that it significantly reduces error accumulation.

A description of the proposed method is given below. Consider $OV+1$ high-speed frames beginning with a standard-speed output frame and ending with the following one. Number the frames $0, 1, \ldots, OV$ and let $\hat{d}_{k,l}$ be the estimated optical flow (displacement) from frame k to frame l, where $0 \leq k \leq l \leq OV$. The end goal is to estimate the optical flow between frames 0 and OV, i.e, $\hat{d}_{0,OV}$.

Proposed method:

1. Capture a standard-speed frame, set $k = 0$.

2. Capture the next high-speed frame and set $k = k + 1$.

3. Estimate $\hat{d}_{k-1,k}$ using Lucas-Kanade method.

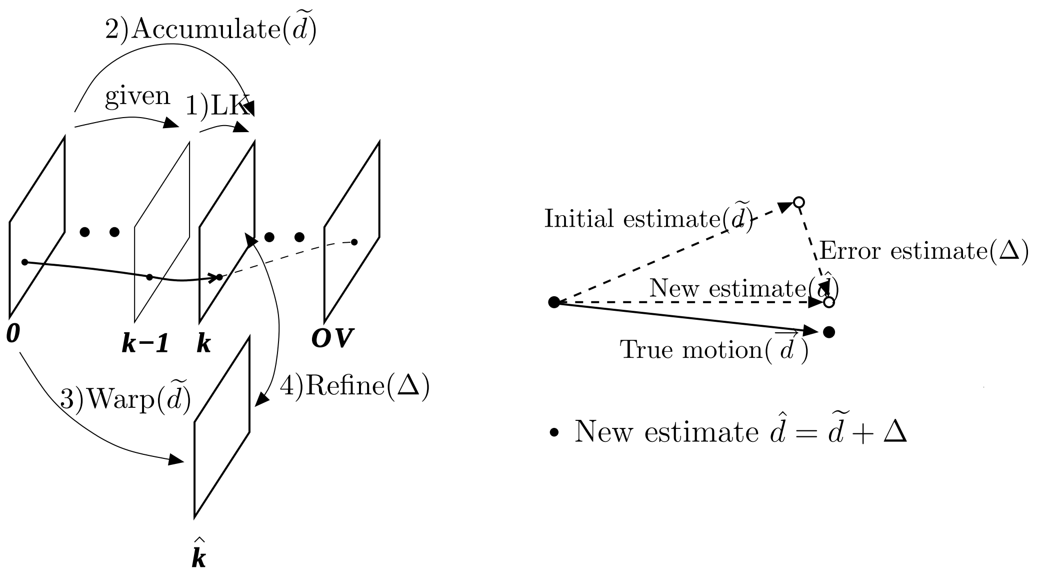


Figure 2.4: Accumulate and refine stage.

4. $\tilde{d}_{0,k} = \hat{d}_{0,k-1} + \hat{d}_{k-1,k}$ where addition of optical flow estimates are along the motion trajectories.
5. Estimate Δ_k, the displacement between frame k and $\hat{k}$.
6. Set refined estimate $\hat{d}_{0,k} = \tilde{d}_{0,k} + \Delta_k$.
7. Repeat steps 2 through 6 until $k = OV$
8. Output $\hat{d}_{0,OV}$ the final estimate of optical flow at the standard frame rate

Since the proposed algorithm is iterative, its memory requirement is independent of frame rate. Furthermore, since it uses 2-tap temporal filter for smoothing and estimating temporal gradients, its memory requirement is less than that of the conventional Lucas-Kanade method, which typically uses a 5-tap temporal filter. Assuming an $M \times N$ image, our method requires approximately $190MN(OV)$ operations per frame and $12MN$ bytes of frame memory. By comparison the standard Lucas-Kanade method as implemented by Barron *et al.* [48] requires $105MN$ operations per frame and $16MN$ bytes of frame memory.

2.2.2 Simulation and results

In this subsection, we describe the simulations we performed using synthetically generated natural image sequences to test our optical flow estimation method. To evaluate the performance of the proposed method and compare with methods using standard frame rate sequences, we need to compute the optical flow using both the standard and high frame rate versions of the same sequence, and then compare the estimated optical flow in each case to the true optical flow. We use synthetically generated video sequences obtained by warping of a natural image. The reason for using synthetic sequences, instead of real video sequences, is that the amount of displacement between consecutive frames can be controlled and the true optical flow can be easily computed from the warping parameters.

We use a realistic image sensor model [60] that incorporates motion blur and noise in the generation of the synthetic sequences, since these effects can vary significantly as a function of frame rate, and can thus affect the performance of optical flow estimation. In particular, high frame rate sequences have less motion blur but suffer from lower SNR, which adversely affects the accuracy of optical flow estimation. The image sensor in a digital camera comprises a 2-D array of pixels. During capture, each pixel converts incident photon flux into photocurrent. Since the photocurrent density $j(x, y, t)$ A/cm^2 is too small to measure directly, it is spatially and temporally integrated onto a capacitor in each pixel and the charge $q(m, n)$ is read out at the end of exposure time T. Ignoring dark current, the output charge from a pixel can be expressed as

$$q(m,n) = \int_0^T \int_{ny_0}^{ny_0+Y} \int_{mx_0}^{mx_0+X} j(x,y,t)dxdydt + N(m,n), \qquad (2.1)$$

where x_0 and y_0 are the pixel dimensions, X and Y are the photodiode dimensions, (m, n) is the pixel index, and $N(m, n)$ is the noise charge. The noise is the sum of two independent components, shot noise and readout noise. The spatial and temporal integration results in low pass filtering that can cause motion blur. Note that the pixel intensity $i(m, n)$ commonly used in image processing literature is directly proportional to the charge $q(m, n)$.

The steps of generating a synthetic sequence are as follows.

1. Warp a high resolution (1312×2000) image using perspective warping to create a high resolution sequence.

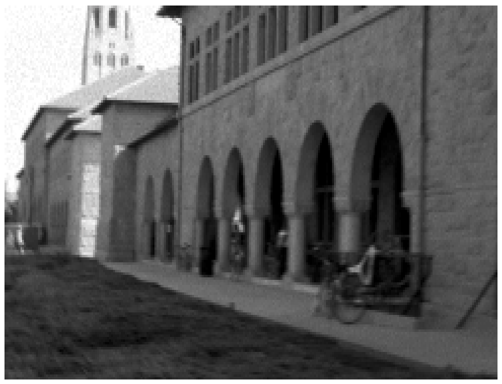
2. Spatially and temporally integrate (according to Equation (1)) and subsample the high resolution sequence to obtain a low resolution sequence. In our example, we subsampled by factors of 4×4 spatially and 10 temporally to obtain each high-speed frame.

3. Add readout noise and shot noise according to the model.

4. Quantize the sequence to 8 bits/pixel.

Three different scenes derived from a natural image (Figure 2.5) were used to generate the synthetic sequences. For each scene, two versions of each video, one captured at a standard frame rate ($OV = 1$) and the other captured at four times the standard frame rate ($OV = 4$), are generated as described above. The maximum displacements were between 3 and 4 pixels/frame at the standard frame rate. We performed optical flow estimation on the ($OV = 1$) sequences using the standard Lucas-Kanade method as implemented by Barron *et al.* [48] and on the ($OV = 4$) sequences using the proposed method. Both methods generate optical flow estimates at a standard frame rate of 30 frames/s. Note that the standard Lucas-Kanade method was implemented using 5-tap temporal filters for smoothing and estimating temporal gradients while the proposed method used 2-tap temporal filters. The resulting average angular errors between the true and the estimated optical flows are given in Table 2.1. The densities of all estimated optical flows are close to 50%.

The results demonstrate that using the proposed method in conjunction with the high frame rate sequence can achieve higher accuracy. Note that the displacements were kept relatively small (as measured at the standard frame rate) to make comparison between the two methods more fair. As displacements increase, the accuracy of the standard Lucas-Kanade method deteriorates rapidly and hierarchical methods should be used in the comparison instead. On the other hand, the proposed method is much more robust to large displacements because of the higher sampling rate.


(a)

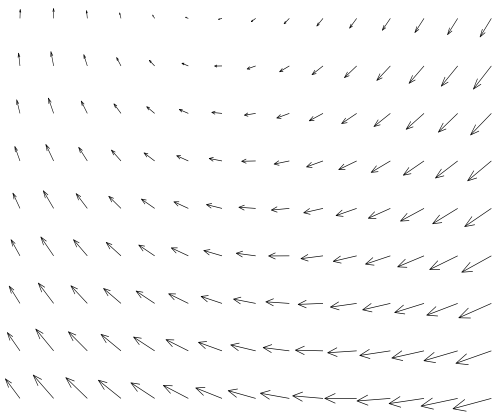

(b)

Figure 2.5: (a) One frame of a test sequence and (b) its known optical flow.

Scene	Lucas-Kanade method at standard frame rate ($OV = 1$)		Proposed method using high frame rate sequence ($OV = 4$)	
	Angular error	Magnitude error	Angular error	Magnitude error
1	4.43°	0.24	3.43°	0.14
2	3.94°	0.24	2.91°	0.17
3	4.56°	0.32	2.67°	0.17

Table 2.1: Average angular error and magnitude error using Lucas-Kanade method with standard frame rate sequences versus the proposed method using high frame rate sequences.

To investigate the gain in accuracy of the proposed method for large displacements, we applied the Lucas-Kanade method, our proposed method with $OV = 10$, and the hierarchical matching-based method by Anandan [61] as implemented by Barron [48] to a synthetic sequence. The maximum displacement was 10 pixels/frame at the standard frame rate. The average angular errors and magnitude errors of the estimated optical flows are given in Table 2.2. For comparison, we calculated average errors for Anandan's method at locations where Lucas-Kanade method gave valid optical flow, although Anandan's method can provide 100% density. Thus, values in the table were calculated where the densities of all estimated optical flows are close to 50%.

	Angular error	Magnitude error
Lucas-Kanade method	$9.18°$	1.49
Anandan's method	$4.72°$	0.53
Proposed method ($OV = 10$)	$1.82°$	0.21

Table 2.2: Average angular and magnitude error using Lucas-Kanade, Anandan's and proposed method.

2.3 Effect of motion aliasing on optical flow estimation

This section reviews 3-D spatio-temporal sampling theory and investigates the effect of motion aliasing on the accuracy of optical flow estimation. We hypothesize that the minimum frame rate necessary to achieve good performance is largely determined by the minimum frame rate necessary to prevent motion aliasing in the sequence. This is supported in Subsection 2.3.2 through simulation results using the proposed method.

2.3.1 Review of spatio-temporal sampling theory

A simplified but highly insightful example of motion is that of global motion with constant velocity in the image plane. Assuming that intensity values are constant along the motion trajectories without any occlusion, the pixel intensity is given by

$$\begin{aligned} i(x, y, t) &= i(x - v_x, y - v_y, 0) \\ &= i_0(x - v_x, y - v_y), \end{aligned}$$

where $i_0(x, y)$ denotes the 2-D pixel intensity for $t = 0$ and v_x and v_y are the global velocities in the x and y directions, respectively. This is commonly assumed either globally or locally in many applications such as motion-compensated standards conversion and video compression. After taking the Fourier transform, we obtain

$$I(f_x, f_y, f_t) = I_0(f_x, f_y) \cdot \delta(f_x v_x + f_y v_y + f_t),$$

where $I_0(f_x, f_y)$ is the 2-D Fourier transform of $i_0(x, y)$ and $\delta(\cdot)$ is the 1-D Dirac delta function. Thus, it is clear that the energy of $I(f_x, f_y, f_t)$ is confined to a plane given by $f_x v_x + f_y v_y + f_t = 0$. If we assume that $i_0(x, y)$ is bandlimited such that $I(f_x, f_y) = 0$ for $|f_x| > B_x$ and $|f_y| > B_y$, then $i(x, y, t)$ is bandlimited temporally as well, i.e, $I(f_x, f_y, f_t) = 0$ for $|f_t| > B_t$ where $B_t = B_x v_x + B_y v_y$. Note that the temporal bandwidth depends on *both the spatial bandwidths and the spatial velocities.*

To simplify our discussion, we assume in the following that sampling is performed only along the temporal direction and that the spatial variables are taken as continuous variables (no sampling along the spatial directions). While this may initially seem somewhat strange, it greatly simplifies the analysis, and interestingly is not entirely unrealistic, since it is analogous to the shooting of motion picture film, where each film frame corresponds to a temporal sample of the video. Figure 2.6 shows the spatio-temporal spectrum of video when sampled only in the temporal direction. For simplicity of illustration, we consider its projection onto the (f_x, f_t)-plane, where the support can be simplified to $f_x v_x + f_t = 0$. Each line represents the spatio-temporal support of the sampled video sequence.

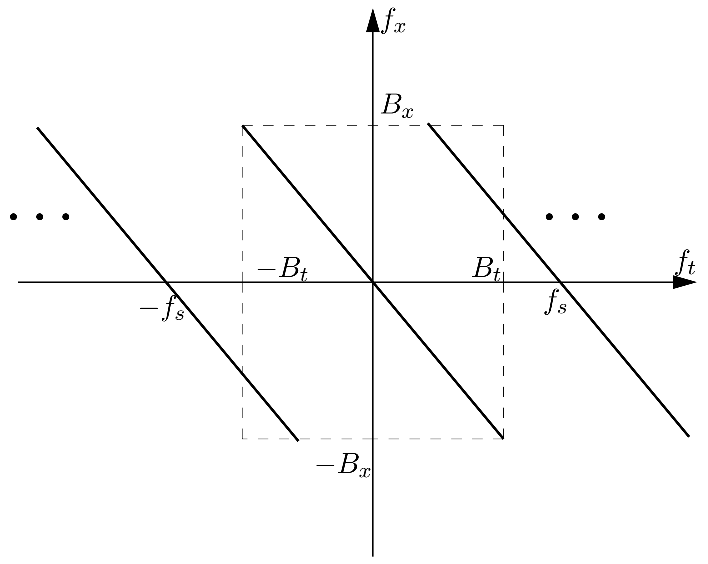


Figure 2.6: Spatio-temporal spectrum of a temporally sampled video.

Let us consider the problem of how fast we should sample the original continuous video signal along the temporal dimension such that it can be perfectly recovered from its samples. Assume that an ideal lowpass filter with rectangular support in the 3-D frequency domain is used for reconstruction, although in certain ideal cases, a sub-Nyquist sampled signal can also be reconstructed by an ideal motion-compensated reconstruction filter assuming the replicated spectra do not overlap (see [50] for details). To recover the original continuous spatio-temporal video signal from its temporally sampled version, it is clear from the figure that the temporal sampling frequency (or frame rate) f_s must be greater than $2B_t$ in order to avoid aliasing in the temporal direction. If we assume global motion with constant velocity v_x and v_y (in pixels per standard-speed frame) and spatially bandlimited image with B_x and B_y as the horizontal and vertical spatial bandwidths (in cycles per pixel), the minimum temporal sampling frequency $f_{\mathrm{s,Nyq}}$ to avoid motion aliasing is given by

$$f_{\mathrm{s,Nyq}} = 2B_t = 2B_x v_x + 2B_y v_y, \tag{2.2}$$

where $f_{\mathrm{s,Nyq}}$ is in cycles per standard-speed frame. Note that the temporal sampling frequency in cycles per standard-speed frame is the oversampling factor OV. Moreover, since OV is an integer in our framework to ensure that standard-speed frames correspond to a captured high-speed frame (see Figure 2.1), the minimum oversampling factor to avoid motion aliasing, OV_{theo}, can be represented as

$$\begin{aligned} OV_{\mathrm{theo}} &= \lceil f_{\mathrm{s,Nyq}} \rceil \\ &= \lceil 2B_x v_x + 2B_y v_y \rceil . \end{aligned}$$

To illustrate this relationship consider the simple case of a sequence with only global motion in the horizontal direction (i.e., with $v_y = 0$). Figure 2.7 plots $OV_{\mathrm{theo}} = \lceil 2B_x v_x \rceil$ versus horizontal velocity and spatial bandwidth for this case.

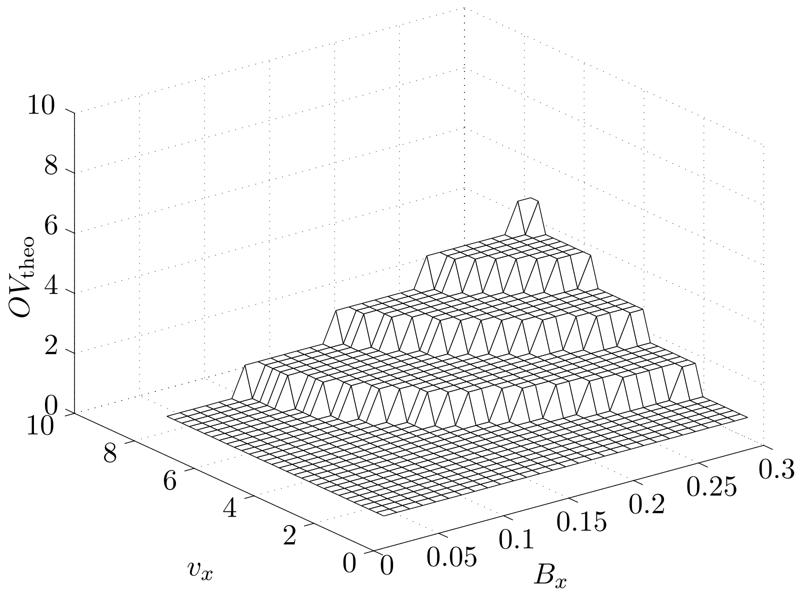


Figure 2.7: Minimum OV to avoid motion aliasing, as a function of horizontal velocity v_x and horizontal spatial bandwidth B_x.

Motion aliasing adversely affects the performance of optical flow estimation even as perceived by the human visual system. This is illustrated by the classic example of a rotating wagon wheel (see Figure 2.8). In this example the wagon wheel is rotating counter-clockwise and we wish to estimate its motion from two frames captured at times $t = 0$ and $t = 1$. The solid lines represent positions of the wheel and spokes at $t = 0$ and the dashed lines represent the positions at $t = 1$. Optical flow is locally estimated for the two shaded regions of the image in Figure 2.8. As can be seen, the optical flow estimates are in the *opposite* direction of the true motion (as often experienced by a human observer watching through display devices such as TVs and projectors). The wheel is rotating counter-clockwise, while the optical flow estimates from the local image regions would suggest that it is rotating clockwise. This ambiguity is caused by insufficient temporal sampling and the fact that optical flow estimation (and the human visual system) implicitly assume the smallest possible displacements (corresponding to a lowpass filtering of the possible motions).

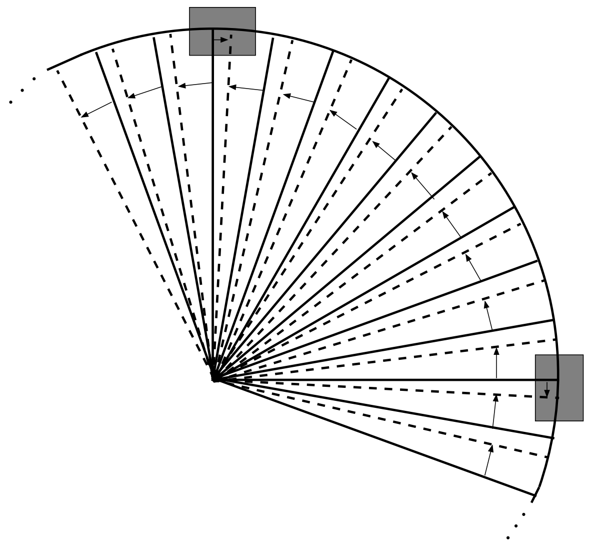


Figure 2.8: Wagon wheel rotating counter-clockwise illustrating motion aliasing from insufficient temporal sampling: the local image regions (gray boxes) appear to move clockwise.

Let us consider the spatio-temporal frequency content of the local image regions in Figure 2.8. Since each shaded region has a dominant spatial frequency component and the assumption of global velocity for each small image region [48] holds, its spatio-temporal frequency diagram can be plotted as shown in Figure 2.9 (A). The circles represent the frequency content of a sinusoid and the dashed lines represent the plane where most of the energy resides. Note that the slope of the plane is inversely proportional to the negative of the velocity. The spatio-temporal frequency content of the baseband signal after reconstruction by the OFE algorithm is plotted in Figure 2.9 (B). As can be seen aliasing causes the slope at which most of the energy resides to not only be different in magnitude, but also to have a different sign, corresponding to motion in the opposite direction. This example shows that motion aliasing can cause incorrect motion estimates for any OFE algorithm. To overcome motion aliasing, one must either sample sufficiently fast, or have prior information about the possible motions as in the case of the moving wagon wheel, where the human observer makes use of the direction of motion of the wagon itself to correct the misperception about the rotation direction of the wheel.

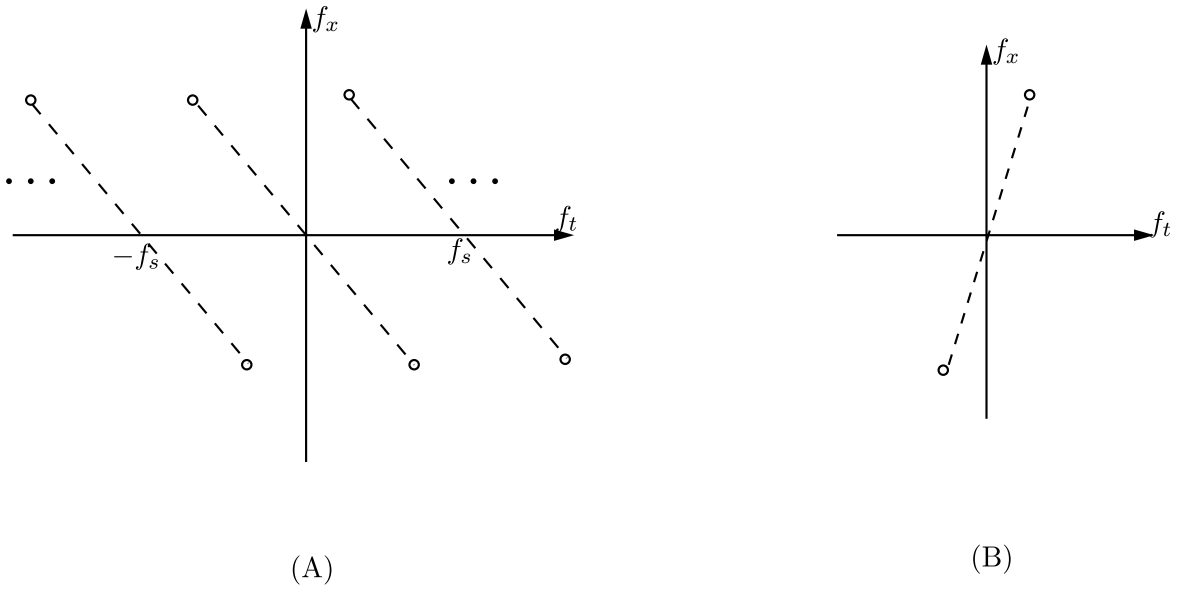


Figure 2.9: Spatio-temporal diagrams of, (A) the shaded region in Figure 2.8 and (B) its baseband signal.

2.3.2 Simulation and results

In this subsection we discuss simulation results using sinusoidal test sequences and the synthetically generated natural image sequence used in Subsection 2.2.2. The reason for using sinusoidal sequences is to assess the performance of the proposed method as spatial frequency and velocity are varied in a controlled manner. As discussed in the previous subsection, motion aliasing depends on both the spatial frequency and the velocity and can have a detrimental effect on optical flow estimation. Using a natural sequence, it would be difficult to understand the behavior of the proposed method with respect to spatial frequency, since in such a sequence, each local region is likely to have different spatial frequency content and the Lucas-Kanade method estimates optical flow by performing spatially local operations. In addition, typical figures of merit, such as average angular error and average magnitude error, would be averaged out across the frame. The use of sinusoidal test sequences can overcome these problems and can enable us to find the minimum OV needed to obtain a desired accuracy, which can then be used to select the minimum high-speed frame rate for a natural scene.

We considered a family of 2-D sinusoidal sequences with equal horizontal and vertical frequencies $f_x = f_y$ moving only in the horizontal direction at speed v_x (i.e., $v_y = 0$). For each f_x and v_x, we generated a sequence with $OV = 1$ and performed optical flow estimation using the proposed method. We then incremented OV by 1 and repeated the simulation. We noticed that the average error drops rapidly beyond a certain value of OV and that it remained relatively constant for OVs higher than that value. Based on this observation we defined the minimum oversampling ratio OV_{exp} as the OV value at which the magnitude error drops below a certain threshold. In particular, we chose the threshold to be 0.1 pixels/frame. Once we found the minimum value of OV, we repeated the experiment for different spatial frequencies and velocities. The results are plotted in Figure 2.10.

Recall the discussion in the previous subsection (including Figure 2.7) on the minimum oversampling factor as a function of spatial bandwidth and velocity needed to avoid motion aliasing. Note the similarity between the theoretical results in Figure 2.7 and their experimental counterpart in Figure 2.10. This is further illustrated by the

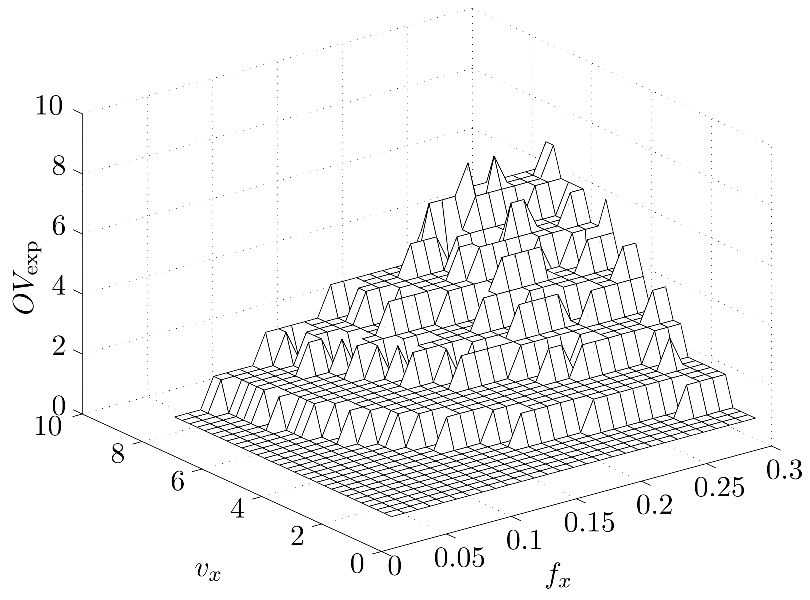


Figure 2.10: Minimum OV as a function of horizontal velocity v_x and horizontal spatial frequency f_x.

plot of their difference and its histogram in Figure 2.11. This similarity supports our hypothesis that reduction in motion aliasing is one of the most important benefits of using high frame rate sequences. The difference in Figure 2.11 can be further reduced by sampling at a higher rate than $\lceil f_{\mathrm{s,Nyq}} \rceil$ to better approximate brightness constancy and improve the estimation of temporal gradients. It has been shown that gradient estimators using a small number of taps suffer from poor accuracy when high frequency content is present [58, 59]. In our implementation, we used a 2-tap temporal gradient estimator, which performs accurately for temporal frequencies $f_t < \frac{1}{3}$ as suggested in [58]. Thus we need to sample at a rate higher than 1.5 times the Nyquist temporal sampling rate. Choosing an OV curve that is 1.55 times the Nyquist rate (i.e., $\lceil 1.55 f_{\mathrm{s,Nyq}} \rceil$), in Figure 2.12 we plot the difference between the OV_{exp} curve in Figure 2.10 and the OV curve. Note the reduction in the difference achieved by the increase in frame rate.

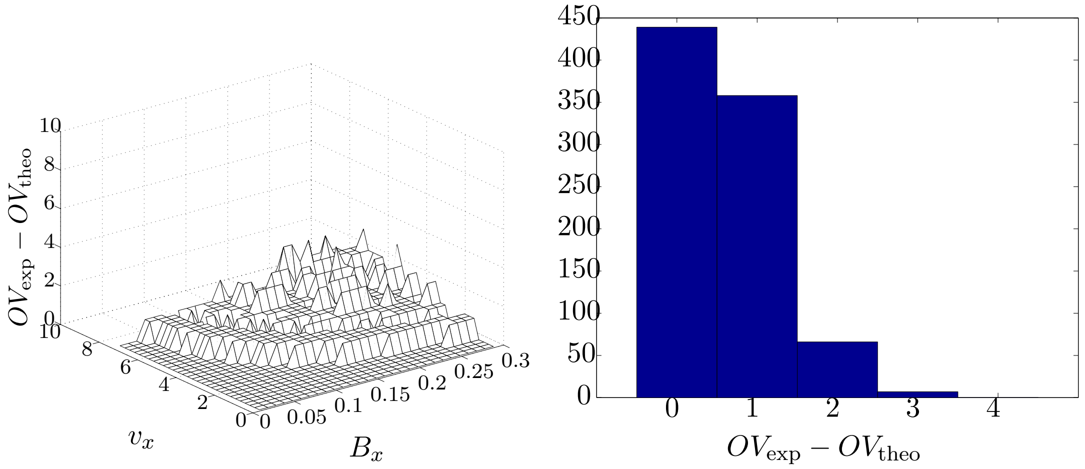


Figure 2.11: Difference between the empirical minimum OV and OV corresponding to the Nyquist rate.

We also investigated the effect of varying OV and motion aliasing on accuracy using the synthetically generated image sequences presented in Subsection 2.2.2. Figure 2.13 plots the average angular error of the optical flow estimates using the proposed method for OV between 1 and 14. The synthetic test sequence had a global displacement of 5 pixels/frame at $OV = 1$. As OV was increased, motion aliasing

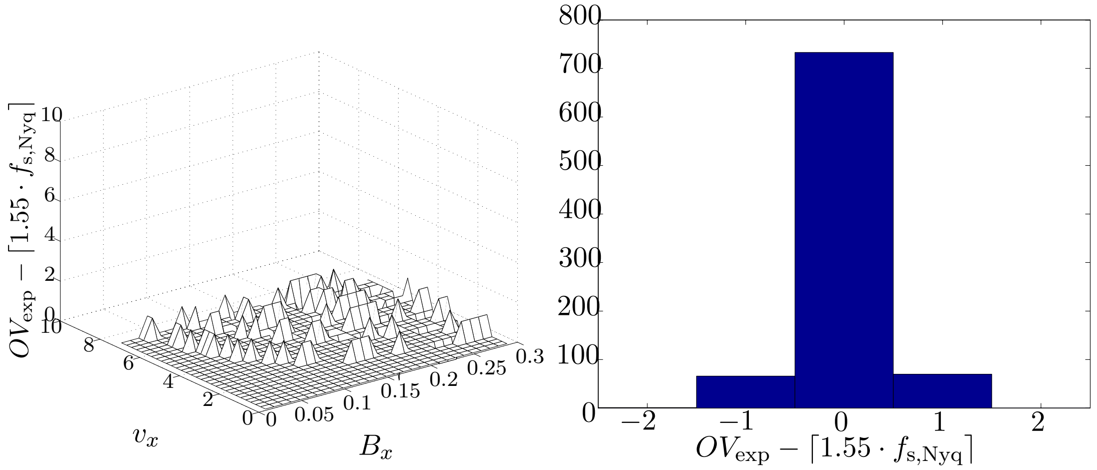


Figure 2.12: Difference between the empirical minimum OV and OV corresponding to the 1.55 times the Nyquist rate.

and the error due to temporal gradient estimation decreased, leading to higher accuracy. The accuracy gain resulting from increasing OV, however, levels off as OV is further increased. This is caused by the decrease in sensor SNR due to the decrease in exposure time and the leveling off of the reduction in motion aliasing. For this example sequence, the minimum error is achieved at $OV = 6$, where displacements between consecutive high-speed frames are approximately 1 pixel/frame.

To investigate the effect of motion aliasing, we also estimated the energy in the image that leads to motion aliasing. Note that since the sequence has global motion with constant velocity, the temporal bandwidth of the sequence can be estimated as $B_t = 5B_x + 5B_y$ by assuming the knowledge of initial estimates of $v_x = v_y = 5$ pixels/frame. Thus, motion aliasing occurs for spatial frequencies $\{f_x, f_y\}$ that satisfy the constraint $f_x + f_y > OV/10$. By using 2D-DFT of the first frame and this constraint, we calculated the energy in the sequence that is motion aliased for different OVs. Figure 2.14 plots the average angular error versus the energy that is motion aliased. Each point corresponds to an OV value and it is clear that the performance of the proposed OFE method is largely influenced by the presence of motion aliasing.

This confirms our hypothesis that motion aliasing significantly affects the performance of optical flow estimation and that a key advantage of high frame rate is the

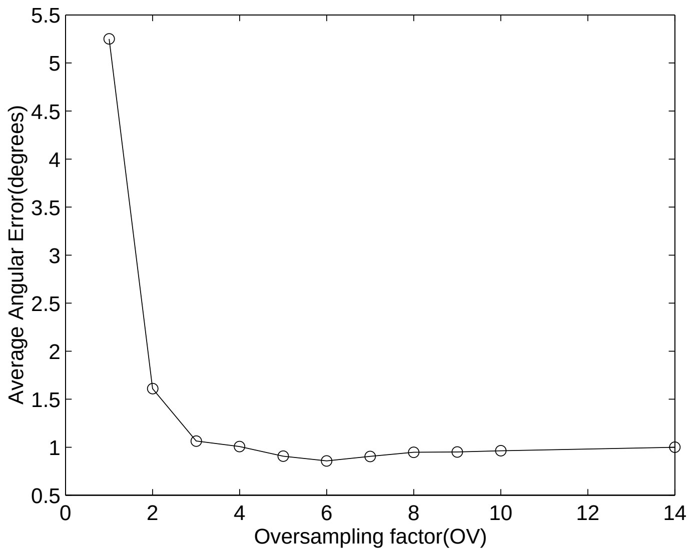


Figure 2.13: Average angular error versus oversampling factor (OV).

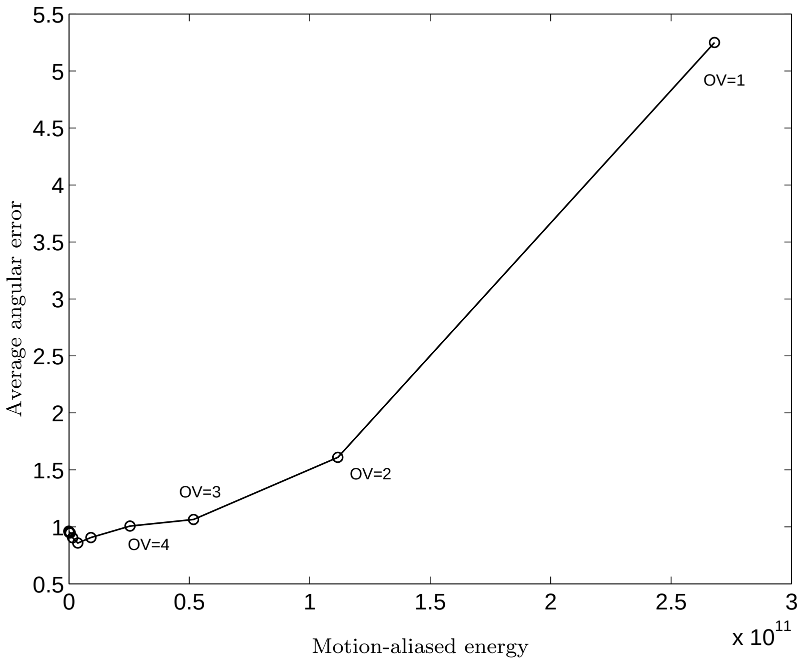


Figure 2.14: Average angular error versus energy in the image that leads to motion aliasing.

reduction of motion aliasing. Also, this example shows that with initial estimates of velocities, we can predict the amount of energy in the image that will be aliased. This can be used to identify the necessary frame rate to achieve high accuracy optical flow estimation for a specific scene.

2.4 Extension to handle brightness variation

In the previous sections we described and tested a method for obtaining high accuracy optical flow at a standard frame rate using a high frame rate sequence. We used the Lucas-Kanade method to estimate optical flow at high frame rate and then accumulated and refined the estimates to obtain optical flow at standard frame rate. The Lucas-Kanade method assumes brightness constancy, and although high frame rate makes this assumption more valid, in this section we show that brightness variations can be handled more effectively using other estimation methods. Specifically, we show that by using an extension of the Haussecker [64] method, temporal oversampling can benefit optical flow estimation even when brightness constancy assumption does not hold.

There have been many proposals of how to handle the case when the brightness constancy assumption does not hold [64, 62, 63, 65, 66, 67, 68]. It has been shown that a linear model with offset is sufficient to model brightness variation in most cases [62, 63, 68]. For example, Negahdaripour *et al.* developed an OFE algorithm based on this assumption and demonstrated good performance [62, 63]. Haussecker *et al.* developed models for several cases of brightness variation and described a method for coping with them [64]. We will use Haussecker's framework with the assumption of linear brightness variation for estimating optical flow at high frame rate.

2.4.1 Review of models for brightness variation

We begin with a brief summary of the framework described in [64]. The brightness change is modeled as a parameterized function h, i.e.,

$$i(\mathbf{x}(t), t) = h(i_0, t, \mathbf{a}),$$

where $\mathbf{x}(t)$ denotes the path along which brightness varies, $i_0 = i(\mathbf{x}(0), 0)$ denotes the image at time 0, and $\mathbf{a}$ denotes a Q-dimensional parameter vector for the brightness change model. The total derivative of both sides of this equation yields

$$(\nabla i)^T \mathbf{v} + i_t = f(i_0, t, \mathbf{a}), \tag{2.3}$$

where f is defined as

$$f(i_0, t, \mathbf{a}) = \frac{\mathrm{d}}{\mathrm{d}t}[h(i_0, t, \mathbf{a})].$$

Note that when brightness is constant, $f = 0$ and Equation 2.3 simplifies to the conventional brightness constancy constraint. The goal is to estimate the parameters of the optical flow field $\mathbf{v}$ and the parameter vector $\mathbf{a}$ of the model f. Remembering that $h(i_0, t, \mathbf{a} = \mathbf{0}) = i_0$, we can expand h using the Taylor series around $\mathbf{a} = \mathbf{0}$ to obtain

$$h(i_0, t, \mathbf{a}) \approx i_0 + \sum_{k=1}^{Q} a_k \frac{\partial h}{\partial a_k}.$$

Thus, f can be written as a scalar product of the parameter vector $\mathbf{a}$ and a vector containing the partial derivatives of f with respect to the parameters a_k, i.e.,

$$f(i_0, t, \mathbf{a}) = \sum_{k=1}^{Q} a_k \frac{\partial f}{\partial a_k} = (\nabla_{\mathbf{a}} f)^T \mathbf{a}. \tag{2.4}$$

Using Equation 2.4, Equation 2.3 can be expressed as

$$\mathbf{c}^T \mathbf{p}_h = 0,$$

where

$$\begin{aligned} \mathbf{c} &= [(\nabla_{\mathbf{a}} f)^T, (\nabla i)^T, i_t]^T \\ \mathbf{p}_h &= [-\mathbf{a}^T, \mathbf{v}^T, 1]^T. \end{aligned}$$

Here, the $(Q + 3)$-dimensional vector $\mathbf{p}_h$ contains the flow field parameters and the brightness parameters of h. The vector $\mathbf{c}$ combines the image derivative measurements and the gradient of f with respect to $\mathbf{a}$. To solve for $\mathbf{p}_h$, we assume that $\mathbf{p}_h$ remains constant within a local space-time neighborhood of N pixels. The constraints from the N pixels in the neighborhood can be expressed as

$$\mathbf{G}\mathbf{p}_h = 0,$$

where $\mathbf{G} = [\mathbf{c}_1, ..., \mathbf{c}_N]^T$. The estimate of $\mathbf{p}_h$ can be obtained by a total least squares (TLS) solution.

2.4.2 Using Haussecker method with high frame rate

We assume a linear model with offset for brightness variation which yields $f = a_1 + a_2 i_0$. We use Haussecker's method to estimate v_x, v_y, a_1 and a_2 for every high-speed frame. We then accumulate and refine v_x, v_y, a_1 and a_2 in a similar manner to the method described in Section 2.2 to obtain optical flow estimates at a standard frame rate.

The parameters v_x and v_y are accumulated and refined exactly as before, and we now describe how to accumulate and refine a_1 and a_2 along the motion trajectories. To accumulate a_1 and a_2, we first define $\hat{a}_{1(k,l)}$ and $\hat{a}_{2(k,l)}$ to be the estimated brightness variation parameters between frames k and l along the motion trajectory. We estimate $\hat{a}_{1(k-1,k)}$ and $\hat{a}_{2(k-1,k)}$ and assume that $\hat{a}_{1(0,k-1)}$ and $\hat{a}_{2(0,k-1)}$ are available from the previous iteration. Since $f = a_1 + a_2 i_0$, we model the brightness variation such that

$$\begin{aligned} i_{k-1} - i_0 &= \hat{a}_{1(0,k-1)} + \hat{a}_{2(0,k-1)} i_0 \\ i_k - i_{k-1} &= \hat{a}_{1(k-1,k)} + \hat{a}_{2(k-1,k)} i_{k-1}, \end{aligned}$$

for each pixel in frame 0, where i_k is the intensity value for frame k along the motion trajectory. By arranging the terms and eliminating i_{k-1}, we can express i_k in terms of i_0 such that

$$i_k = \hat{a}_{1(k-1,k)} + (1 + \hat{a}_{2(k-1,k)})(\hat{a}_{1(0,k-1)} + (1 + \hat{a}_{2(0,k-1)}) i_0). \tag{2.5}$$

Let $\tilde{a}_{1(0,k)}$ and $\tilde{a}_{2(0,k)}$ denote the accumulated brightness variation parameters between frames 0 and k along the motion trajectory. Therefore, by definition, $i_k = \tilde{a}_{1(0,k)} + (1 + \tilde{a}_{2(0,k)})i_0$ and by comparing this equation with Equation 2.5, accumulated brightness variation parameters are obtained by

$$\begin{aligned} \tilde{a}_{1(0,k)} &= \hat{a}_{1(k-1,k)} + (1 + \hat{a}_{2(k-1,k)})\hat{a}_{1(0,k-1)} \\ \tilde{a}_{2(0,k)} &= \hat{a}_{2(k-1,k)} + (1 + \hat{a}_{2(k-1,k)})\hat{a}_{2(0,k-1)}. \end{aligned}$$

Frame $\hat{k}$ is obtained by warping frame 0 according to our initial estimate of optical flow between frames 0 and k and changing the brightness according to $\tilde{a}_{1(0,k)}$ and $\tilde{a}_{2(0,k)}$, i.e.,

$$\text{Frame } \hat{k} = (1 + \tilde{a}_{2(0,k)})i_k(x - \tilde{v}_{x(0,k)}, y - \tilde{v}_{y(0,k)}) + \tilde{a}_{1(0,k)},$$

where $\tilde{v}_{x(0,k)}$ and $\tilde{v}_{y(0,k)}$ are the accumulated optical flow estimates between frames 0 and k. By estimating the optical flow and brightness variation parameters between original frame k and motion-compensated frame $\hat{k}$, we can estimate the error between the true values and the initial estimates obtained by accumulating. For the optical flow, we estimate the error and add it to our initial estimate, whereas for the brightness variation parameters, we perform the refinement as

$$\begin{aligned} \hat{a}_{1(0,k)} &= a_{1\Delta} + (1 + a_{2\Delta})\tilde{a}_{1(0,k)} \\ \hat{a}_{2(0,k)} &= a_{2\Delta} + (1 + a_{2\Delta})\tilde{a}_{2(0,k)}, \end{aligned}$$

where $a_{1\Delta}$ and $a_{2\Delta}$ are the brightness variation parameters between frames k and $\hat{k}$. The accumulation and refinement stage is repeated until we have the parameters between frames 0 and OV.

We tested this method using the sequences described in Subsection 2.2.2 but with global brightness variations. In these sequences, however, the global brightness changed with $a_{1(0,OV)} = 5$ and $a_{2(0,OV)} = 0.1$. We performed optical flow estimation on the $OV = 1$ sequences using the Haussecker's method and on the $OV = 4$ sequences using our extended method. The resulting average angular errors and magnitude errors between the true and the estimated optical flows are given in Table 2.3.

Scene	Haussecker's method ($OV = 1$)		The proposed method ($OV = 4$)	
	Angular error	Magnitude error	Angular error	Magnitude error
1	5.12°	0.25	3.33°	0.15
2	6.10°	0.32	2.99°	0.18
3	7.72°	0.54	2.82°	0.18

Table 2.3: Average angular error and magnitude error using Haussecker's method with $OV = 1$ sequences versus proposed extended method with $OV = 4$ sequences.

These results demonstrate that using high frame rate, high accuracy optical flow estimates can be obtained even when brightness varies with time, i.e., when brightness constancy assumption does not hold. Furthermore, with this extension, we have also demonstrated that our proposed method can be used with OFE algorithms other than the Lucas-Kanade algorithm.

2.5 Summary

In this chapter, we described a method for improving the optical flow estimation accuracy for video at a conventional standard frame rate, by initially capturing and processing the video at a higher frame rate. The method begins by estimating the optical flow between frames at the high frame rate, and then accumulates and refines these estimates to produce accurate estimates of the optical flow at the desired standard frame rate. The method was tested on synthetically generated video sequences and the results demonstrate significant improvements in OFE accuracy. Also, with sinusoidal input sequences, we showed that reduction of motion aliasing is an important potential benefit of using high frame rate sequences. We also described methods to estimate the required oversampling rate to improve the optical flow accuracy, as a function of the velocity and spatial bandwidth of the scene. The proposed method can be used with other OFE algorithms besides the Lucas-Kanade algorithm. For example, we began with the Haussecker algorithm, designed specifically for optical flow estimation when the brightness varies with time, and extended it with the proposed method to work on high frame rate sequences. Furthermore, we demonstrated that our extended version provides improved accuracy in optical flow estimation as compared to the original Haussecker algorithm operating on video captured at the standard frame rate.

Chapter 3

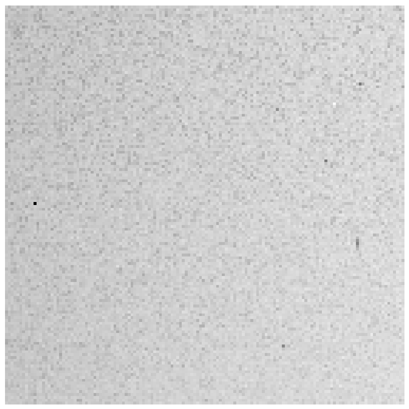
Gain Fixed Pattern Noise Correction

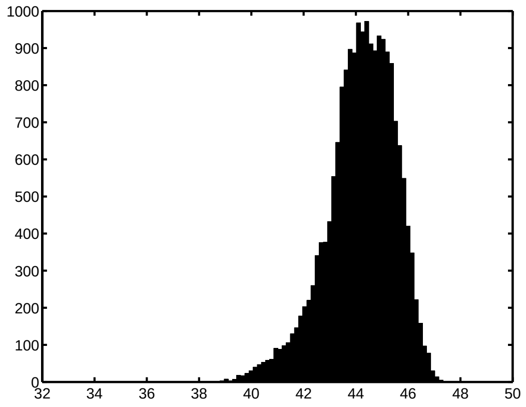
3.1 Introduction

Most image sensors have linear transfer function such that the pixel intensity value i as a function of its input signal s, *e.g.*, photocurrent density [60], can be expressed as

$$i = hs + i_{os}, \tag{3.1}$$

where h is the gain factor and i_{os} is the offset, which includes the dark signal as well as the offset due to the amplifiers and buffers. Since all the pixels do not have the same gain h and offset i_{os}, image data read out of the image sensor pixel array are not uniform even under uniform illumination. Figure 3.1 illustrates an image (with its histogram) obtained by capturing 100 frames under uniform illumination and averaging them to significantly reduce the temporal noise. Note that the image has spatial variation even when there is little temporal noise. Fixed pattern noise (FPN) is this spatial variation of output pixel values under uniform illumination. FPN is caused by variations in pixel gains and offsets due to device mismatches and process parameter variations across an image sensor. It is a major source of image quality degradation especially in CMOS image sensors [72, 73]. In a CCD sensor, since all pixels share the same output amplifier, FPN is mainly due to variations in photodetector area and dark current. In a CMOS image sensor, however, pixels are read out over different chains of buffers and amplifiers each with different gain and offset, resulting in relatively high FPN.


Uniform light temporally averaged



Histogram

Figure 3.1: An image and its histogram of uniform illumination illustrating FPN

FPN can be divided into offset FPN and gain FPN. Offset FPN is due to pixel to pixel variations in i_{os} and can be significantly reduced by correlated-double sampling (CDS). CDS first captures a frame with no exposure time (immediately after pixel reset) and then subtracts it off the desired frame with proper exposure time. Gain FPN is caused by variations in the gain factor h. While offset FPN can be significantly reduced using correlated double sampling (CDS), no method exists for effectively reducing gain FPN. In [74] a method is proposed for reducing gain FPN by characterizing the sensor's pixel gains after manufacture and storing the gains in a lookup table that is subsequently used to perform the correction. A problem with this method is that gain FPN changes with temperature and aging, making a "static" gain lookup table approach inaccurate. Another method would be to characterize the sensor's pixel gains before each capture. This is not feasible since characterizing gain FPN requires many captures at different uniform illuminations.

In this chapter, we present a method to estimate and correct gain FPN using a video sequence and its optical flow. This method can be used in digital video or still cameras without requiring multiple captures of uniformly illuminated scenes at different intensities [70, 71]. The key idea of the method is to assume brightness constancy along the motion trajectories and use this information to estimate the gains for each pixel. For example, when a light intensity patch falls on a pixel at $t = 0$ and on another pixel at $t = 1$, we can estimate the ratio of the gains at these two pixels. By gathering the ratio of gains for all the pixels in the image sensor and for multiple frames, we can estimate the gain for all the pixels in the image sensor. Since this method tracks the intensity variations along the motion trajectories due to gain FPN, it requires global motion between frame which needs to be estimated before this method is applied. Note that the required motion in the scene can be provided by simply panning the camera during capture.

In the following section, we describe the image and FPN model used throughout the chapter. In Section 3.3, we describe our algorithm for estimating and correcting gain FPN and illustrate its operation via simple 1D examples. In Section 3.4, we show simulation results using a synthetically generated sequence and its optical flow. We then show experimental results using a real video sequence taken with our experimental imaging system [75].

3.2 Image and fixed pattern noise model

In this chapter, we only consider gain FPN and assume that offset FPN has been canceled with CDS. After eliminating the offset term in Equation 3.1 and including gain variations, we obtain

$$\begin{aligned} i(x,y,t) &= i_0(x,y,t) + \Delta i(x,y,t) \\ &= (h_0 + \Delta h(x,y))s(x,y,t) \\ &= (1 + \frac{\Delta h(x,y)}{h_0})i_0(x,y,t) \\ &= a(x,y)i_0(x,y,t), \end{aligned}$$

where $i_0(x,y,t)$ is the ideal intensity value at pixel (x,y) and time (frame) t, h_0 is the nominal gain factor, and $\Delta h(x,y)$ is the deviation in gain for pixel (x,y). Gain

FPN can be represented as the pixel to pixel variation of $a(x, y)$ and its magnitude is $\sigma_{\Delta h}/h_0$. Although gain FPN can slowly vary with temperature and aging, we assume here that $a(x, y)$ is constant while capturing several frames with the imager. Note that $a(x, y) = 1$ for all (x, y) in an ideal sensor having no gain FPN.

To quantify the effect of different device parameters on the gain FPN, we define parameter values $Z_1, Z_2, ..., Z_k$ and express $Z_i = z_i + \Delta Z_i$ where z_i the nominal value of the device parameter and ΔZ_i is the variation of Z_i. Thus the variation of gain ΔH can be represented as

$$\Delta H = \sum_{i=1}^{k} \frac{\partial h}{\partial z_i} \cdot \Delta Z_i. \tag{3.2}$$

For Passive Pixel Sensor (PPS), Z_is are photodiode area A_D and the feedback capacitance C_f in the column opamp. For Active Pixel Sensor (APS), Z_is are A_D, photodiode capacitance C_D, the gain of the source followers A_{sf} and the gain of ADC A_{ADC} (if there are more than 1 ADC). For Digital Pixel Sensor (DPS), Z_is are A_D, C_D and A_{ADC}.

Some device parameters contribute to individual pixel gain non-uniformity whereas some others contribute to row or column gain non-uniformity. Row or column component appears as stripes in the image and can result in significant image quality degradation. Thus, we can divide ΔH in Equation 3.2 into pixel gain FPN component ΔH_X, column gain FPN component ΔH_Y and row gain FPN component ΔH_Z. This model will later be used in Section 3.4 to synthetically generate video sequence corrupted by gain FPN.

We assume brightness constancy, which implies that brightness is constant along each motion trajectory. Brightness constancy is commonly assumed in the development of many video processing and computer vision algorithms [50, 48]. Thus, if $F+1$ frames are captured using an $M \times N$ pixel image sensor, the ideal pixel intensity value at t, $i_0(x, y, t)$, can be expressed in terms of the ideal pixel intensity at $t = 0$,

$j(x, y) = i_0(x, y, 0)$, as

$$i_0(x+d_x(x,y,t), y+d_y(x,y,t), t) = j(x,y), \text{ for } x = 1,\dots,M,\ y = 1,\dots,N, \text{ and } t = 0,\dots,F, \tag{3.3}$$

where $d_x(x, y, t)$ and $d_y(x, y, t)$ are the displacements (optical flow) between frame 0 and t for pixel (x, y) in frame 0. Note that by definition $d_x(x, y, 0) = d_y(x, y, 0) = 0$. This model is illustrated in Figure 3.2, which depicts the pixel locations of a moving patch of constant intensity in each frame. Under this ideal model the pixel output values within the patch in all frames are equal.

When gain FPN and temporal noise are added to the ideal model, the pixel intensity value $i(x, y, t)$ becomes

$$i(x + d_x, y + d_y, t) = a(x + d_x, y + d_y)j(x, y) + N(x + d_x, y + d_y, t), \tag{3.4}$$

where $N(x, y, t)$ is the additive temporal noise for pixel (x, y) at time t. For notational simplicity, we omitted the index (x, y, t) in d_x and d_y. Note that the gain FPN component $a(x, y)$ is constant over time t. Thus in the example in Figure 3.2, the pixel values within the patch would be different. However, note that if we ignore temporal noise, the ratio of the pixel output values within the patch equals the ratio of the gains at those tracked pixel locations. These ratios can then be used to correct for gain FPN.

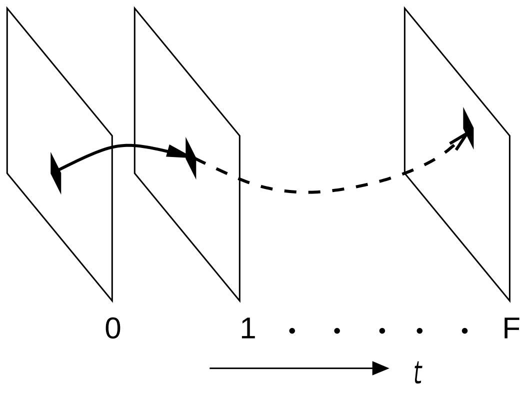


Figure 3.2: Sequence of frames with brightness constancy.

3.3 Description of the algorithm

The goal here is to estimate $j(x, y)$ from $i(x, y, 0), \ldots, i(x, y, F)$ in the presence of temporal noise and gain FPN. To do so, we formulate the problem as follows. Let $\hat{j}(x, y \mid t)$ be a linear estimate of $j(x, y)$ obtained from $i(x, y, t)$ of the form

$$\hat{j}(x, y \mid t) = k(x + d_x, y + d_y) i(x + d_x, y + d_y, t), \tag{3.5}$$

where $k(x, y)$ is a coefficient function that we need to estimate. Because of the brightness constancy assumption, $j(x, y)$ is constant over time, and hence $\hat{j}(x, y \mid t)$ does not depend on time. Using this fact, we find $k(x, y)$ that minimizes the mean square error (MSE) between $\hat{j}(x, y \mid 0)$ and $\hat{j}(x, y \mid t)$. To reduce the computational complexity of estimating $k(x, y)$, we divide the image into non-overlapping blocks and independently estimate $k(x, y)$ for each block. Thus to estimate $k(x, y)$ for pixels in block B, we minimize the MSE function

$$\begin{aligned} E_B &= \sum_{t=1}^{F} \sum_{(x,y)\in B} (\hat{j}(x, y \mid 0) - \hat{j}(x, y \mid t))^2 & (3.6) \\ &= \sum_{t=1}^{F} \sum_{(x,y)\in B} (k(x, y) i(x, y, 0) - k(x + d_x, y + d_y) i(x + d_x, y + d_y, t))^2. & (3.7) \end{aligned}$$

In the following subsection, we describe how the estimate is found for the case when the displacements are integer valued. In Subsection 3.3.2, we extend the discussion to the non-integer case.

3.3.1 Integer displacements

Let R be the set of pixel locations $(x + d_x, y + d_y)$ along the motion trajectories for $(x, y) \in B$, and n_B and n_R be the number of pixels in B and R, respectively. We define the n_R-vector $\mathbf{k}$ to consist of the elements $k(x, y)$ in R beginning with the elements in the block B. Warping $k(x, y)$ to form $k(x + d_x, y + d_y)$ can be represented by multiplying the vector $\mathbf{k}$ with an $n_B \times n_R$ matrix $T(t)$, which is formed as follows.

When brightness at the pixel location i in frame 0 moves to pixel location j in frame t, the ith row of $T(t)$ is assigned a 1 to its jth element and 0 to all its other elements. Let $I(t)$ be the $n_B \times n_B$ diagonal matrix whose diagonal elements are $i(x+d_x, y+d_y, t)$ for $(x,y) \in B$. We can now rewrite Equation 3.7 in a matrix form as

$$E_B = \sum_{t=1}^{F} \| \begin{bmatrix} I(0) & 0_{n_B \times (n_R - n_B)} \end{bmatrix} \mathbf{k} - I(t)T(t)\mathbf{k} \|^2, \tag{3.8}$$

where

$$T(t)_{ij} = \begin{cases} 1, & \text{when } i\text{th pixel moves to } j\text{th pixel, } 1 \leq i \leq n_B \text{ and } 1 \leq j \leq n_R \\ 0, & \text{otherwise.} \end{cases} \tag{3.9}$$

To obtain an unbiased estimate of $\hat{j}(x, y \mid t)$, we require that $\mathbf{1}^T\mathbf{k} = n_R$, where $\mathbf{1} = \begin{bmatrix} 1 & 1 & \dots & 1 \end{bmatrix}^T$. Thus, we wish to minimize

$$E_B = \sum_{t=1}^{F} \|(\begin{bmatrix} I(0) & 0_{n_B \times (n_R - n_B)} \end{bmatrix} - I(t)T(t))\mathbf{k} \|^2, \tag{3.10}$$

subject to $\mathbf{1}^T\mathbf{k} = n_R$.

This is a quadratic optimization problem with a linear constraint and thus has a unique global optimum, which can be found using standard methods, *e.g.*, steepest descent or conjugate gradient [76]. Optionally, to make use of the fact that the elements of $\mathbf{k}$ are close to 1, a regularization term $\lambda\|\mathbf{k} - \mathbf{1}\|^2$ may be added to E_B. This becomes useful when temporal noise is high.

After estimating $k(x,y)$ for the entire image, one block at a time, the gain FPN corrected value for each pixel (x,y) can be computed as

$$\begin{aligned} \hat{j}(x,y,0) &= k(x,y)i(x,y,0) \\ &= a(x,y)k(x,y)i_0(x,y,0) + k(x,y)N(x,y,0). \end{aligned}$$

We use $\hat{j}(x,y|0)$ over other $\hat{j}(x,y|t)$s because it does not suffer from interpolation error when the displacements are non-integers (as discussed in the following subsection).

Note also that this method does not result in higher temporal noise since the average value of the $k(x, y)$s is 1.

We illustrate our method via the simple 1-D example in Figure 3.3. In this example, we track brightness at 4 pixel locations in frame 0 and correct gain FPN for 6 pixel locations. Thus, we have $n_B = 4$, $F = 2$ and $n_R = 6$. The numbers on the left side are the ideal pixel intensities and the ones on the right side are the pixel intensities when gain FPN is included. Each arrow represents the motion for each pixel between consecutive frames. We assume pixel gains of $0.95, 1.02, 1.08, 0.97, 0.95, 1.03$ and ignore temporal noise. Thus,

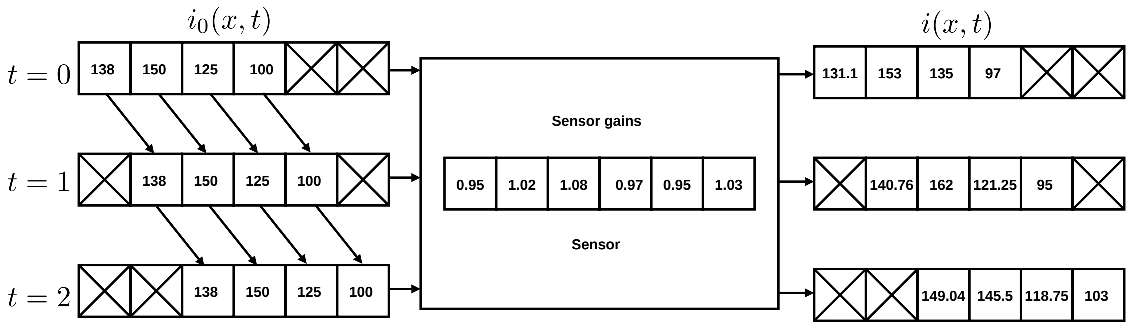


Figure 3.3: 1-D case simple example when the displacements are integers.

$$
\begin{aligned}
I(0) &= \begin{bmatrix} 131.1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 153 & 0 & 0 & 0 & 0 \\ 0 & 0 & 135 & 0 & 0 & 0 \\ 0 & 0 & 0 & 97 & 0 & 0 \end{bmatrix}, \quad I(1) = \begin{bmatrix} 140.76 & 0 & 0 & 0 \\ 0 & 162 & 0 & 0 \\ 0 & 0 & 121.25 & 0 \\ 0 & 0 & 0 & 95 \end{bmatrix}, \\
I(2) &= \begin{bmatrix} 149.04 & 0 & 0 & 0 \\ 0 & 145.5 & 0 & 0 \\ 0 & 0 & 118.75 & 0 \\ 0 & 0 & 0 & 103 \end{bmatrix}, \quad T(1) = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}, \\
T(2) &= \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}.
\end{aligned}
$$

Solving Equation 3.10, we obtain

$$
\mathbf{k}^* = \begin{bmatrix} 1.0503 & 0.9782 & 0.9239 & 1.0286 & 1.0503 & 0.9687 \end{bmatrix}^T .
$$

We can now correct gain FPN using Equation 3.11 and we obtain

$$
I(0)\mathbf{k}^* = \begin{bmatrix} 137.68 & 149.66 & 124.72 & 99.77 \end{bmatrix}^T = 0.998 \begin{bmatrix} 138 & 150 & 125 & 100 \end{bmatrix}^T .
$$

Note that the gain FPN have been completely removed.

3.3.2 Non-integer displacements

The method described in the previous section can be extended to the more realistic non-integer displacement case depicted in Figure 3.4. The solid lines in the figure represent the pixel grid. The shaded area A is a brightness patch that covers a pixel in frame 0 and moves to location B in frame t. Equation 3.5 cannot be directly used in this case since location B overlaps with several pixels due to the non-integer displacement of A and $k(x+d_x, y+d_y)$ and $i(x+d_x, y+d_y, t)$ were only defined for integer valued $x+d_x$ and $y+d_y$. To extend our method to non-integer displacements we use spatial interpolation to estimate the intensity values at the non-integer patch locations and define $I(t)$ matrices using the interpolated values $\tilde{i}(x+d_x, y+d_y, t)$ instead of pixel intensities.

To define the gain FPN correction coefficient $k(x+d_x, y+d_y)$ at the non-integer location B note that B partially overlaps with 4 pixels with possibly different gain FPN values. We define the gain FPN of B as the weighted average of the four pixel gain FPN values, where the weight for a pixel is the fraction of its overlap area with B. The gain FPN correction coefficient $k(x+d_x, y+d_y)$ can be similarly defined as

$$k(x+d_x, y+d_y) = \sum_{i=0}^{1}\sum_{j=0}^{1} C_{ij} k(x+\lfloor d_x \rfloor + i, y + \lfloor d_y \rfloor + j), \tag{3.11}$$

where

$$\begin{aligned} C &= \begin{bmatrix} (1-\alpha)(1-\beta) & (1-\alpha)\beta \\ \alpha(1-\beta) & \alpha\beta \end{bmatrix} \\ \alpha &= d_x - \lfloor d_x \rfloor, \; \beta = d_y - \lfloor d_y \rfloor. \end{aligned}$$

The index (x, y, t) in C, $\lfloor d_x \rfloor$, $\lfloor d_y \rfloor$, α and β are omitted for notational simplicity.

With these modifications, we use Equation 3.5 to obtain the estimate of the ideal pixel intensity. The MSE function to be minimized is expressed in the matrix form of Equation 3.10 using the modified definitions of $\tilde{i}(x+d_x, y+d_y, t)$ and $k(x+d_x, y+d_y)$. In this case $I(t)$ is the $n_B \times n_B$ diagonal matrix whose diagonal elements are $\tilde{i}(x+d_x, y+d_y, t)$. When brightness at pixel location (x, y) in frame 0 moves to the location $(x+d_x, y+d_y)$ in frame t, the row of the $T(t)$ matrix that corresponds to pixel location (x, y), is formed by assigning C_{ij}s to the elements corresponding to pixel locations $(x+\lfloor d_x \rfloor + i, y + \lfloor d_y \rfloor + j)$ and 0s otherwise.

We illustrate the non-integer displacement case with the simple 1D example in Figure 3.5. In this example, we track brightness at 4 pixel locations and correct gain FPN for 5 pixels. Thus, $n_B = 4$, $F = 2$ and $n_R = 5$. Note that the magnitude of

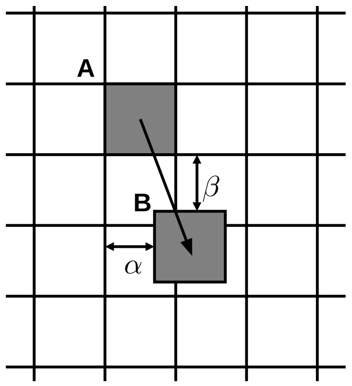


Figure 3.4: Non-integer d_x and d_y displacements.

the displacements between frames 0 and 1 is 0.5 pixel and thus $T(1)$ is no longer a permutation matrix. Since half of the brightness patch at the first (left most) pixel moves to the first pixel and the other half moves to the second pixel, we set $T(1)_{11} = T(1)_{12} = 0.5$. Other rows of $T(1)$ can be defined similarly. Also, since $i(x + d_x, y + d_y, 1)$ is not defined for non-integer d_x and d_y, the diagonal elements of $I(1)$ can be obtained by interpolating $i(x, y, 1)$ to find $\tilde{i}(x + d_x, y + d_y, 1)$. In this example, $I(1)$ is obtained by performing bilinear interpolation and temporal noise is ignored.

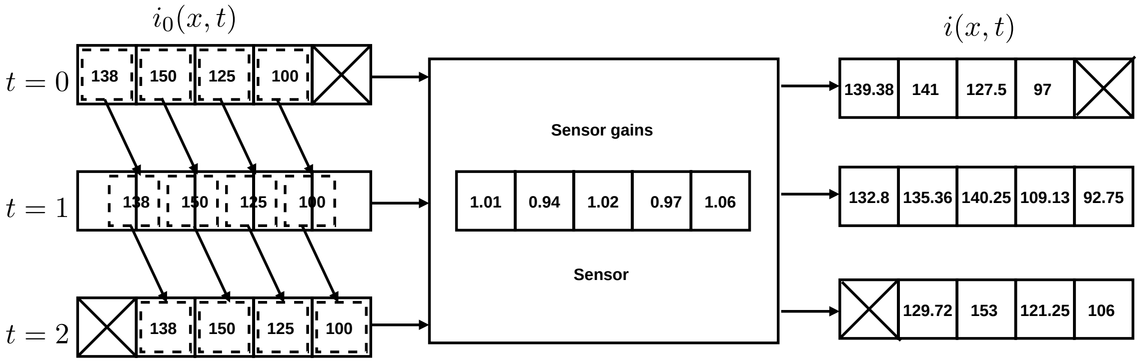


Figure 3.5: 1-D case simple example when the displacements are non-integers.

$$I(0) = \begin{bmatrix} 139.38 & 0 & 0 & 0 & 0 \\ 0 & 141 & 0 & 0 & 0 \\ 0 & 0 & 127.5 & 0 & 0 \\ 0 & 0 & 0 & 97 & 0 \end{bmatrix}, \quad I(1) = \begin{bmatrix} 134.1 & 0 & 0 & 0 \\ 0 & 137.8 & 0 & 0 \\ 0 & 0 & 124.7 & 0 \\ 0 & 0 & 0 & 100.9 \end{bmatrix},$$

$$I(2) = \begin{bmatrix} 129.72 & 0 & 0 & 0 \\ 0 & 153 & 0 & 0 \\ 0 & 0 & 121.25 & 0 \\ 0 & 0 & 0 & 106 \end{bmatrix}, \quad T(1) = \begin{bmatrix} 0.5 & 0.5 & 0 & 0 & 0 \\ 0 & 0.5 & 0.5 & 0 & 0 \\ 0 & 0 & 0.5 & 0.5 & 0 \\ 0 & 0 & 0 & 0.5 & 0.5 \end{bmatrix},$$

$$T(2) = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}.$$

Solving Equation 3.10, we obtain

$$\mathbf{k}^* = \begin{bmatrix} 0.9737 & 1.0465 & 0.9879 & 1.0389 & 0.9530 \end{bmatrix}^T.$$

We can now correct gain FPN using Equation 3.11 and we obtain

$$I(0)\mathbf{k}^* = \begin{bmatrix} 135.73 & 147.56 & 125.96 & 100.77 \end{bmatrix}^T.$$

Note that unlike in the integer displacement example, where gain FPN was completely corrected for, in this example gain FPN cannot be completely corrected due to interpolation errors.

3.4 Results

To test our method, we applied it to synthetically generated video sequences so that the amount of gain FPN and displacement between frames can be controlled, and the performance of gain FPN correction can be measured. We generated the sequences using a realistic image sensor model, which included motion blur, read noise, shot noise, and gain FPN. The following image sensor parameters are used: conversion gain of $32.5\mu V/e^-$, read noise of 50 electrons, voltage swing of 1V, 8-bit ADC, pixel gain variation of 5%, *i.e.*, $\frac{\sigma_{H_{pix}}}{h_0} = 0.05$. Detailed description of how these sequences are generated is provided in Chapter 2. We measure performance by computing the mean square error (MSE) between the noise-free image and the gain FPN corrected image and comparing it to the MSE between the noise-free image and the input image before gain FPN correction (see Figure 3.6).

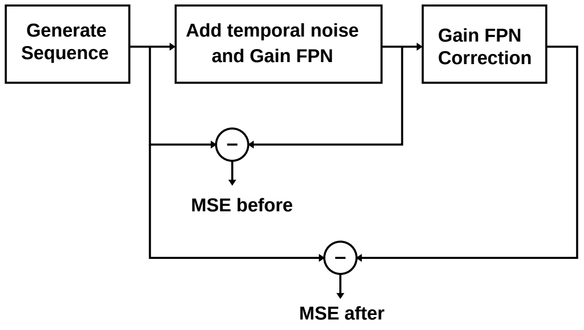


Figure 3.6: Simulation setup.

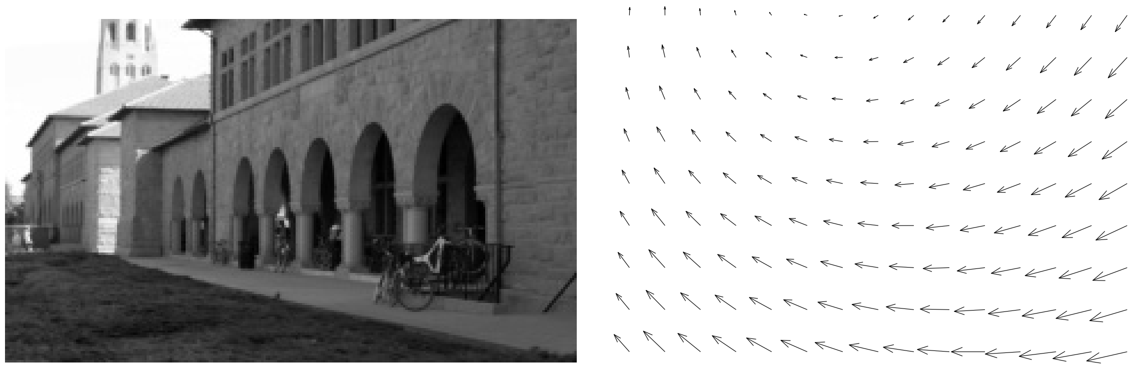
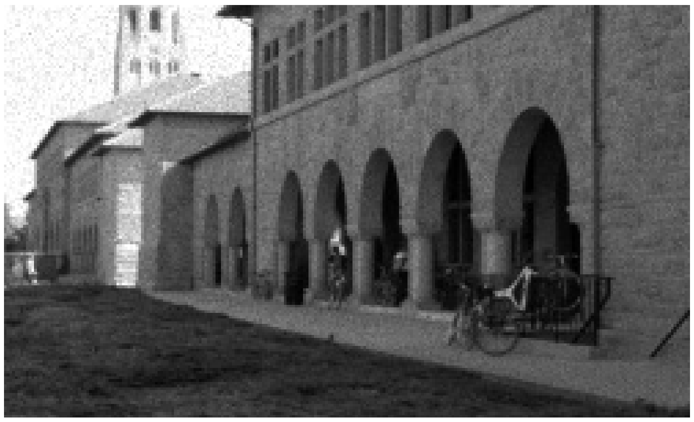
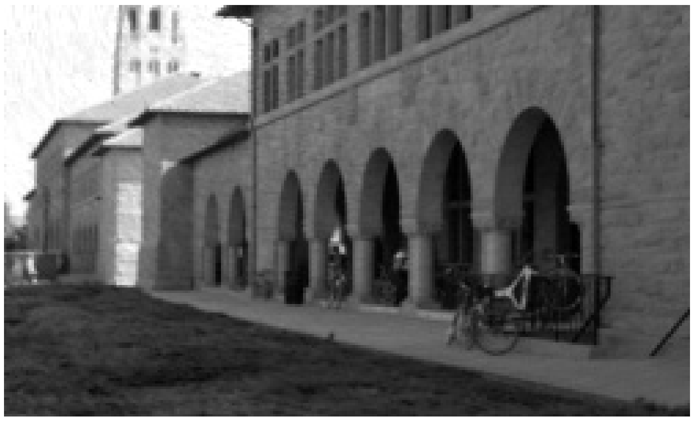


Figure 3.7: Original scene and its optical flow.

The original image with no FPN or temporal noise is shown in Figure 3.7 together with its optical flow. Figure 3.8 shows one frame of each sequence before and after gain FPN correction using block size of 5×5. After gain FPN correction, the MSE is reduced from 32.38 to 11.54 (a reduction of 4.48dB). Although gain FPN is not totally removed due to temporal noise, interpolation error and motion blur, its effect is far less visible as can be seen from the figure. The slight blockiness in the gain FPN corrected image can be reduced by using larger block size at the cost of higher computational complexity.



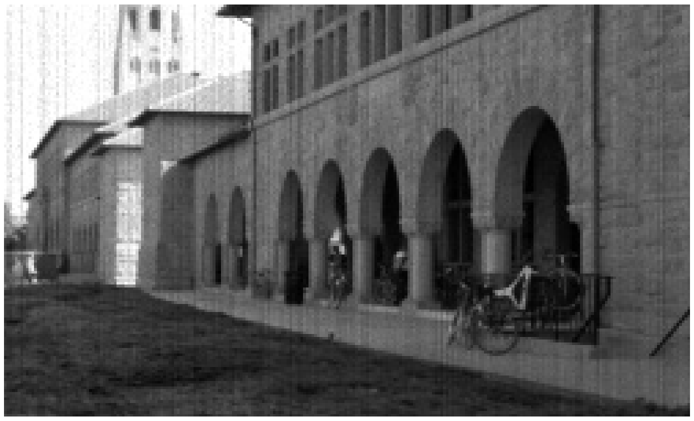
Before correction (MSE=32.38)



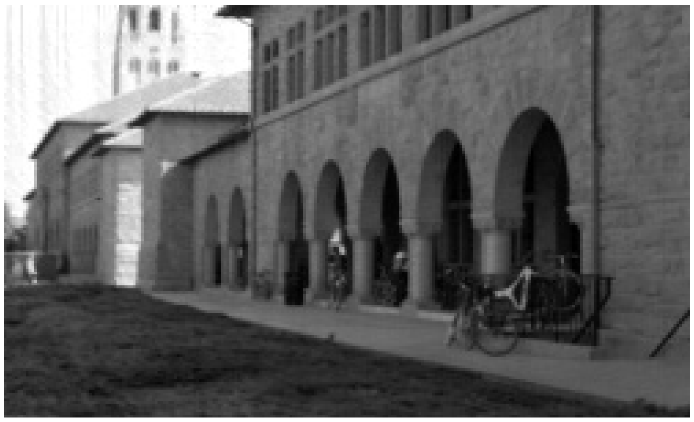
After correction (MSE=11.54)

Figure 3.8: Images before and after correction with 5% of pixel gain variation.

Most CMOS image sensors suffer from column FPN due to column-wise readout circuits [60] in addition to pixel FPN. We tested our method assuming both are present with pixel gain variations of 3%, *i.e.*, $\frac{\sigma_{H_{pix}}}{h_0} = 0.03$, and column gain variations of 4%, *i.e.*, $\frac{\sigma_{H_{col}}}{h_0} = 0.04$. We assume that column and pixel components are uncorrelated [73]. Figure 3.9 shows one frame of each sequence before and after gain FPN correction again using block size of 5×5. After gain FPN correction, the MSE is reduced from 31.17 to 13.49 (a reduction of 3.64dB).



Before correction ($MSE = 31.17$)



After correction ($MSE = 13.49$)

Figure 3.9: Images before and after correction with 3% of pixel and 4% of column gain variation.

We also applied our method to real video sequences captured using an experimental high speed imaging system [75]. The system is based on the DPS chip described in [40] and can operate at frame rates of up to 1400 frames/s. The sequence was obtained by shaking an eye chart in front of the camera and capturing 5 frames at 200 frames/s. We estimated optical flow of the sequence using the method described in Chapter 2.

Figure 3.10 shows one frame of each sequence before and after gain FPN correction. Gain FPN correction was again performed with block size of 5×5. Note that the horizontal lines in the image before correction, which are caused by a systematic difference between the gain of pixels in even and odd lines due to vertical mirroring in pixel layout [40], are removed after FPN correction. Also note that FPN correction did not result in any image distortion or blurring. To clearly illustrate the effect of gain FPN correction, zoomed in images are shown in Figure 3.11.

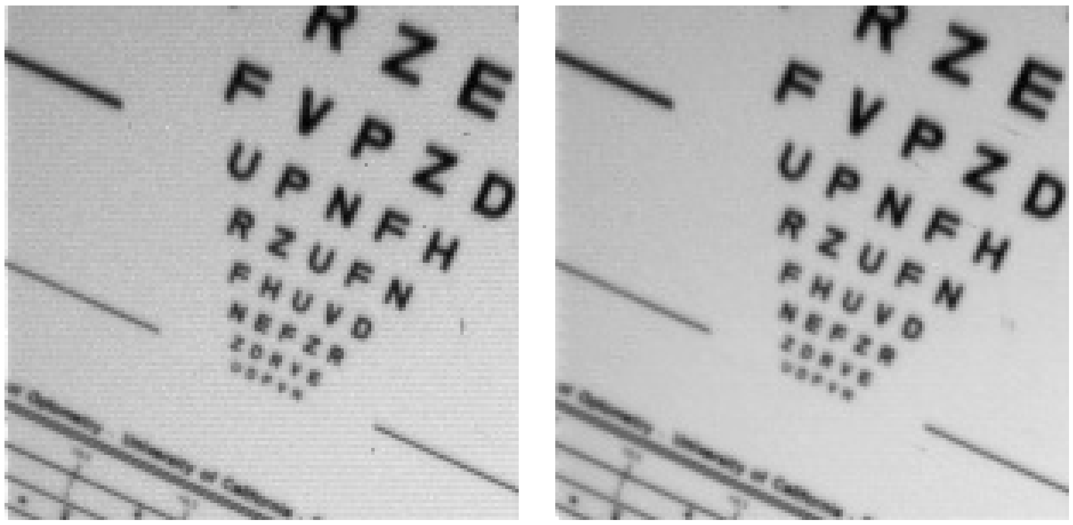
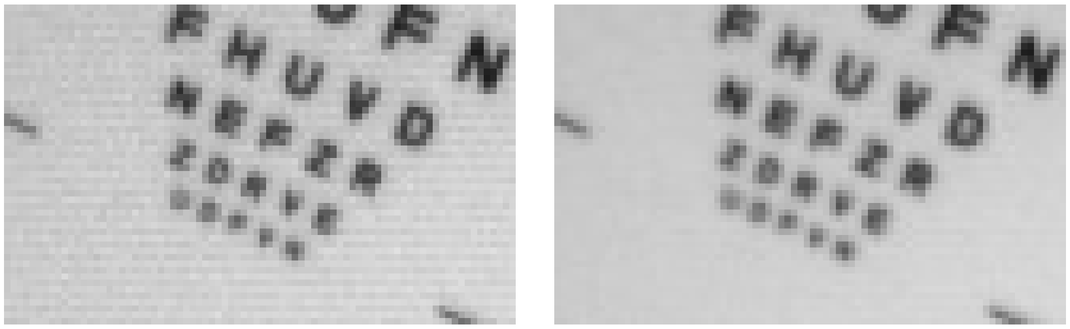

Before correction After correction

Figure 3.10: Images before and after correction for real sequence.



Before correction After correction

Figure 3.11: Zoomed in images before and after correction for real sequence.

3.5 Complexity

In this section, we discuss the computational complexity issues when correcting for gain FPN from a video sequence and its optical flow. We assume that the optical flow has already been estimated and we will not discuss the complexity of obtaining optical flow. Also, as can be seen in Equation 3.11, once we have calculated gain FPN correction coefficient, it is straight-forward to obtain the gain FPN corrected image. Thus, we will primarily discuss the complexity of calculating the coefficients which are obtained by solving Equation 3.10.

One can directly solve Equation 3.10 using the Lagrange multiplier method. This, however, requires calculating the inverse of an $(n_R+1)\times(n_R+1)$ matrix, which has high computational complexity and can be numerically unstable. For example, even when $n_R = 16$, we have to calculate the inverse of a 17×17 matrix which is not trivial. To alleviate the computational burden, we used an iterative method to obtain $\mathbf{k}^*$. We examined the steepest descent and conjugate gradient method, which gave results that are very close to the global optimum with less than 10 iterations.

The derivation in Section 3.3 was for a block in the image and we did not assume anything about the size of the block. The block can be as large as the image itself, but that would require too many computations. To investigate the effect of the block size on the complexity, we calculate the number of computations per pixel required to obtain $k(x,y)$. For each block with n_B pixels, calculation of $k(x,y)$ requires $n_R(2n_R^2+$

$n_B F n_R + 2(p-1)n_R + 7p)$ operations, where n_R is the number of elements in the set of pixel locations along the motion trajectories for the pixels in the block and p is the number of iterations. Note that the number of operations depends on n_R which depends on the optical flow of the sequence. With the assumption that pixels do not move in or out of the block *i.e.,* $n_B = n_R$, the number of operations/pixel to obtain gain FPN correction coefficients versus n_B is plotted in Figure 3.12 for $F = 4$. We can see that the number of computations required increases as the block size increases. Although smaller block sizes result in lower complexity, it results in poorer performance. This is because smaller blocksizes cannot take advantage of more information provided by neighboring pixels that are outside of the block. We have found that using 5×5 (*i.e.,* $n_B = 25$) size blocks results in a good trade-off between the computational complexity and the performance of the algorithm.

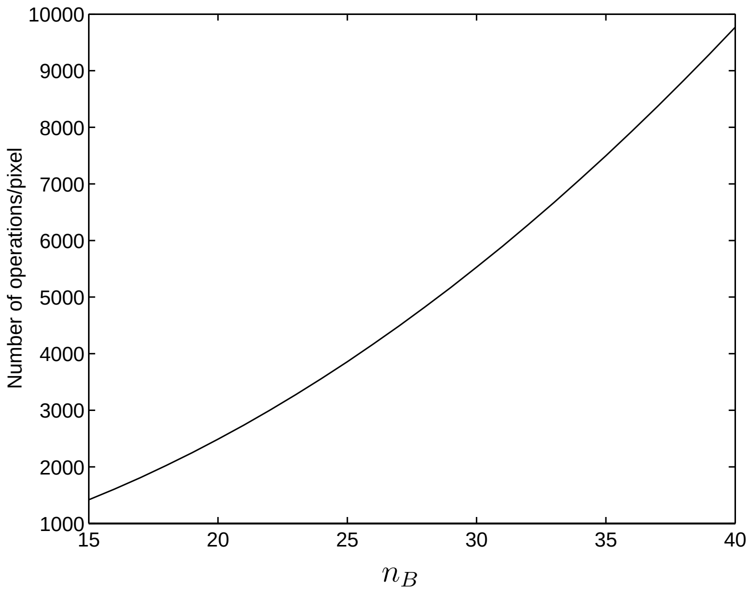


Figure 3.12: Number of operations/pixel versus n_B when 5 frames are used.

3.6 Summary

This chapter presented a method for gain FPN correction using a video sequence and its estimated optical flow. Conceptually, the method can be thought of as digital CDS that cancels gain FPN rather than offset FPN. It can be used in a digital video or still camera by taking a video sequence with motion prior to capture and using it to estimate gain FPN. The sequence is then used to estimate pixel gains by iteratively minimizing the sum of the squared brightness variations along the motion trajectories. To reduce complexity we divide the pixel array into block's, as is commonly done in video coding, and perform the estimation separately for each block. We found that block size of 5×5 provides a reasonable trade-off between complexity and performance. Using iterative minimization methods also allows us to lower computational requirements and reduce gain FPN incrementally by utilizing previously computed $\mathbf{k}^*$s as initial estimates instead of starting from $\mathbf{k} = \mathbf{1}$. We tested our gain FPN correction method on synthetically generated sequences and demonstrated significant gain FPN reduction even in the presence of motion blur and temporal noise.

Chapter 4

Hardware and Implementation Issues

In this chapter, we discuss hardware implementation issues of high speed CMOS imaging systems. We describe a 352×288 pixel CMOS Digital Pixel Sensor chip with per-pixel single-slope ADC and 8-bit dynamic memory in a standard digital 0.18μm CMOS process. The chip performs "snap-shot" image acquisition at continuous rate of $10,000$ frames/s or 1 Gpixels/s. We then explore the limits of integrating memory and processing with a CMOS image sensor in 0.18μm process and below. We show that the integration of an entire video camera system on a chip is not only feasible at 0.18μm process, but in fact underutilizes the possible on-chip processing power. Further, we show that the on-chip processing power and memory are sufficient to perform applications such as optical flow estimation.

4.1 A 10,000 frames/s Digital Pixel Sensor (DPS)

In this section, a 352×288 CMOS DPS with per-pixel ADC and digital memory which we fabricated in a standard digital 0.18μm CMOS technology is presented. The goals of our design are: (i) to demonstrate a DPS with bit-parallel ADC and memory per pixel (our earlier implementations [77, 78, 43] employed a shared bit-serial ADC and a 1-bit latch per 2×2 block of pixels), (ii) to evaluate the scalability and performance of image sensors implemented in a standard digital 0.18 μm CMOS process, (iii) to demonstrate the speed potential of DPS, in particular, to reach or exceed continuous $10,000$ frames/s operation and sustain 1 Gpixels/s throughput, and (iv) to provide a platform for experimenting with algorithms and circuits that exploit high speed imaging and embedded pixel-level digital memory.

The DPS architecture described in this chapter fulfills the requirements of high speed imaging with practically no limit on array size. It performs fully pixel-parallel image acquisition. Pixel reset is performed in parallel for all pixels and the reset duration is completely programmable, permitting higher shutter speeds that are independent of frame rates. The massively-parallel per-pixel A/D conversion scheme demonstrated here results in a high digitization rate that is independent of array size. This is a key advantage of DPS over APS employing column-level, chip-level, or off-chip ADCs where digitization rates do not scale linearly with the number of pixels in the array.

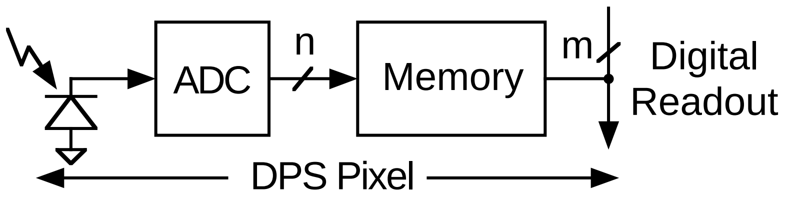


Figure 4.1: Simple DPS pixel block diagram.

In this section, we first describe the DPS chip architecture and main characteristics. In Section 4.1.2, we describe the details of the pixel design. In Section 4.1.3, we discuss the chip operation including the different imaging modes. Finally, in Section 4.1.4, we present the chip characterization results including ADC performance, QE, dark current, noise, digital noise coupling, and sample images.

4.1.1 DPS chip overview

A photomicrograph of the DPS chip is shown in Figure 4.2 and the main chip characteristics are listed in Table 4.1. The chip contains 3.8 million transistors on a 5×5mm die. The sensor array is 352×288 pixels in size, conforming to the CIF format. Each pixel is 9.4μm on a side and contains 37 transistors including a photogate, transfer gate, reset transistor, a storage capacitor, and an 8-bit single-slope ADC with an 8-bit 3T-DRAM. The chip also contains test structures that we used for detailed characterization of APS and DPS pixels [79]. The test structures can be seen in upper center area of the chip.

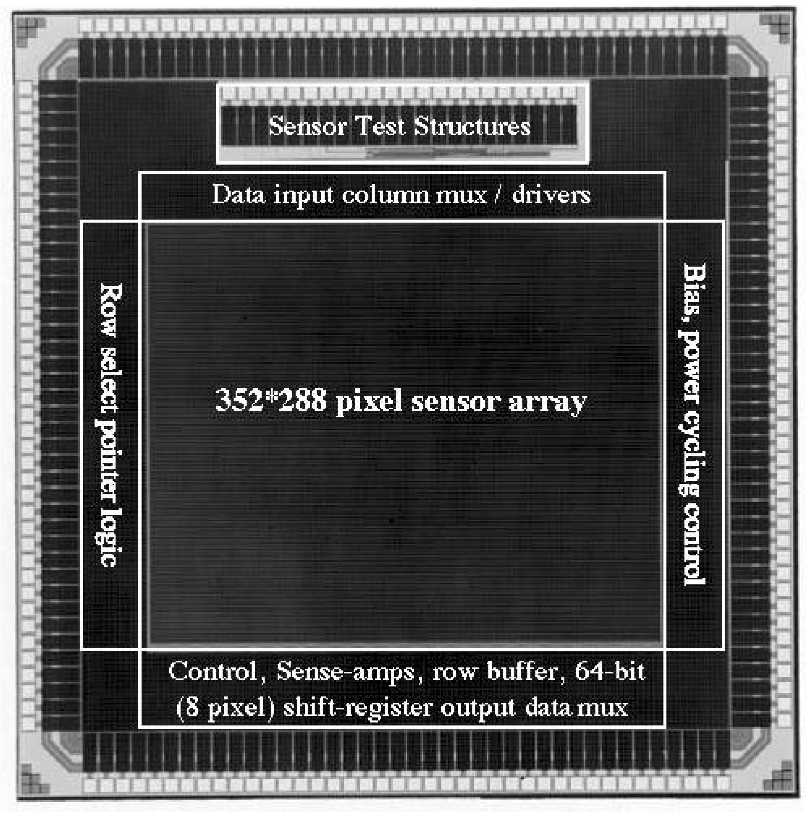

Sensor Test Structures
Data input column mux / drivers
Row select pointer logic
352*288 pixel sensor array
Bias, power cycling control
Control, Sense-amps, row buffer, 64-bit
(8 pixel) shift-register output data mux

Figure 4.2: DPS Chip photomicrograph. The chip size is 5×5mm.

Figure 4.3 shows a block diagram of the DPS chip. At the center is the sensor array. The periphery above the sensor core contains an 8-bit gray code counter, an auxiliary code input, and multiplexers and tri-state column data drivers that are used to write data into the memory within the pixel array. The column multiplexers can be used to substitute arbitrary patterns for the standard gray code during data conversion. This facilitates the use of nonlinear ADC transfer functions, for example,

Technology	0.18μm 5-metal CMOS
Die size	5×5 mm
Array size	352×288 pixels
Number of transistors	3.8 million
Readout architecture	64-bit (167 MHz)
Max output data rate	> 1.33 GB/s
Max continuous frame rate	$> 10,000$ frames/s
Max continuous pixel rate	> 1 Gpixels/s
Pixel size	$9.4 \times 9.4\mu$m
Photodetector type	nMOS Photogate
Number of transistors/pixel	37
Sensor fill factor	15%

Table 4.1: Chip characteristics.

for compression of dynamic range and contrast stretching. To the left of the sensor array is the readout control periphery that includes a row select pointer for addressing the pixel-level memory during readout. To the right of the sensor array is the bias generation and power-down circuits, which are used to digitally control the per-pixel ADC and memory sense-amp biases. The analog ramp signal input to the array needed for the per-pixel ADCs is supplied by an off-chip DAC.

Below the sensor core is the digital readout circuits that include column sense-amps for reading the pixel-level memory and an output multiplexing shift register. The pixel values are read out of the memory one row at a time using the row select pointer and column sense-amps. Each row is then buffered and pipelined so that as one row is being shifted out of the chip the following row is read out of the memory. A 64-bit wide parallel-in, serial-out shift-register bank was used instead of a large multiplexer since in a shift register data moves in small increments, reducing local capacitance and drive circuit performance requirements. With each clock cycle, eight 8-bit pixel values are read out in a continuous stream with no waits or gaps between rows. An entirely closed-loop clocking system is used to assure clock and data integrity. The 64-bit output bus is clocked at 167 MHz for a 1.33 GB/s readout rate.

In the lower left corner of the chip is the readout control block. Since the chip is

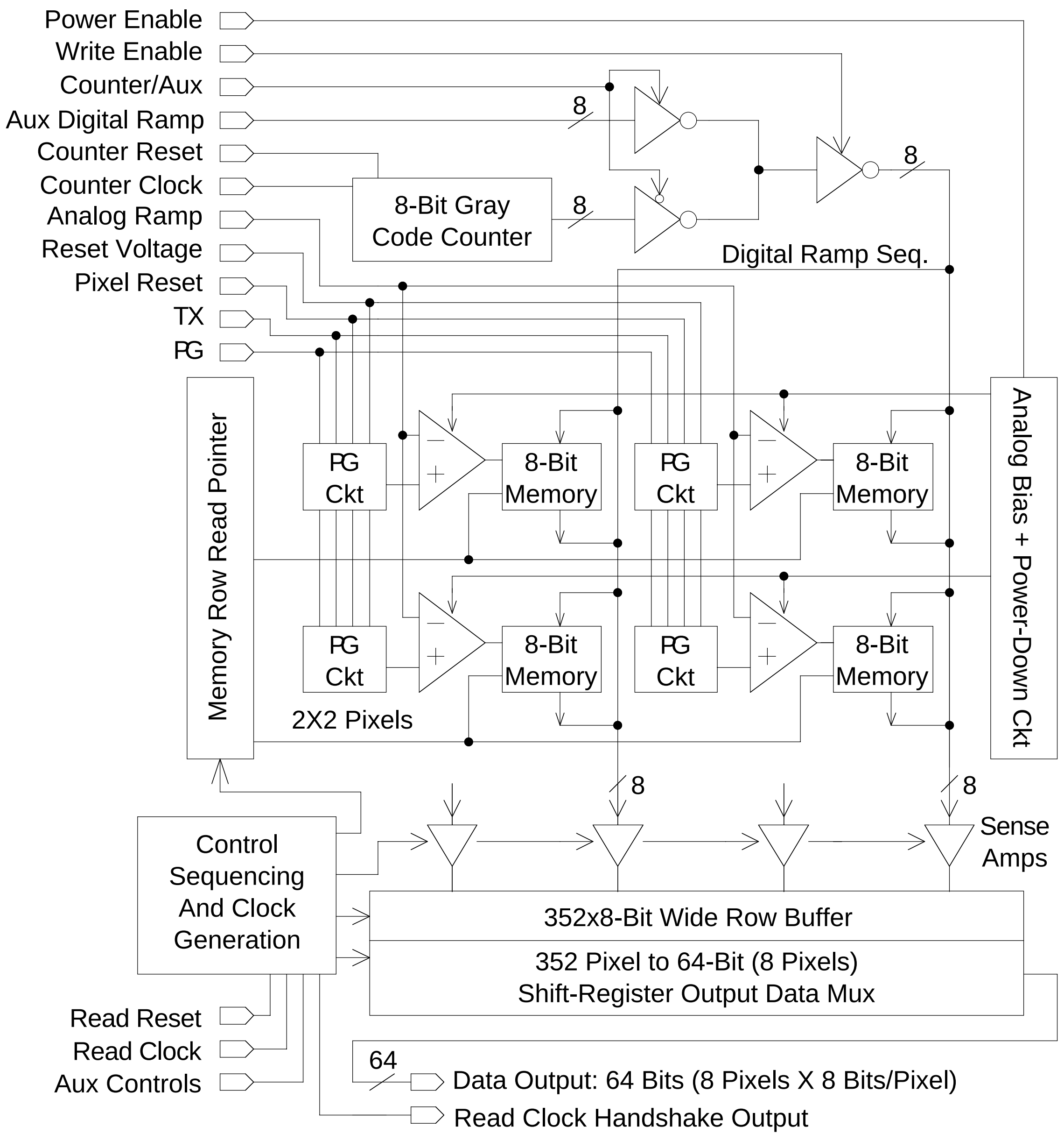


Figure 4.3: DPS block diagram.

to be clocked at upwards of 167 MHz, it was important to keep off-chip high speed controls to a minimum. The control block provides all the signals needed for readout from a single frame reset followed by a single continuous clock burst. A 6-phase clock generator using feedback to ensure correct margins is used to drive the shift registers. During chip testing or experimental operation, the control block can be bypassed and a set of auxiliary input controls used. Almost all digital circuitry in the periphery of the chip was designed using static logic to permit arbitrarily low clock rates.

4.1.2 Pixel design

The pixel circuit is shown in Figure 4.4. It consists of a photogate circuit, a comparator and an 8-bit memory. The photogate circuit consists of an nMOS photogate, a transfer gate, a reset transistor and a sample capacitor. We decided to use a photogate to achieve high conversion gain and because preliminary process data indicated that native photodiodes have unacceptably high leakage. We implemented the photogate circuit using the standard thick oxide (3.3V) transistors that normally used in I/O circuits, to avoid the high gate and subthreshold leakage currents of the thin oxide (1.8V) transistors. Implementing the photogate and reset transistor using thick oxide transistors also makes it possible to use higher gate voltages than the nominal 1.8V supply to increase voltage swing.

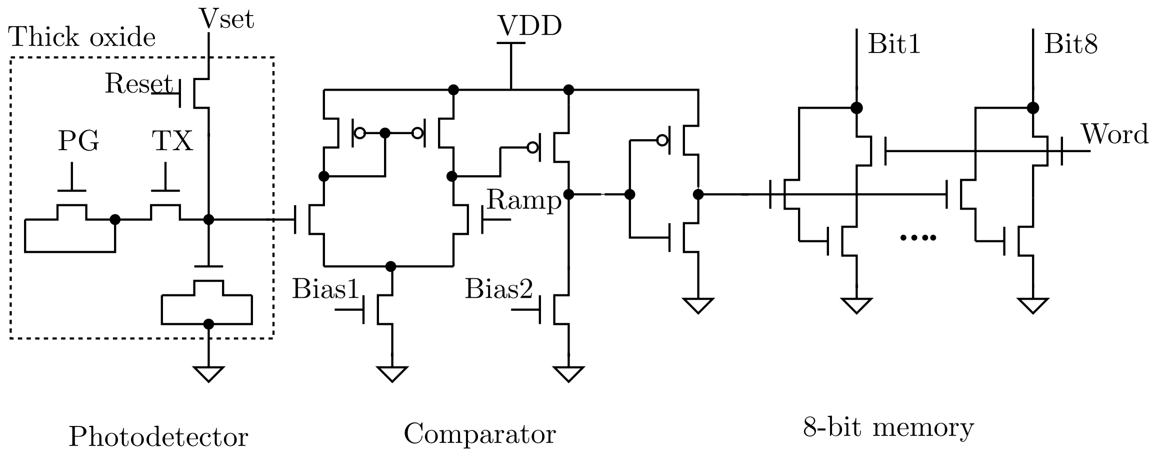


Figure 4.4: Pixel schematic.

The comparator consists of a differential gain stage, a single-ended gain stage, followed by a CMOS inverter. It is designed to provide gain sufficient for 10-bits of resolution with an input swing of 1V and a worst case settling time of 80ns. This provides the flexibility to perform 8-bit A/D conversion over a 0.25V range in under 25μs, which is desirable for high speed and/or low light operation.

The pixel-level memory was implemented using 3T dynamic memory cells with a single read/write port to achieve small area and high speed readout. The memory was designed for a maximum data hold time of 10ms. This required the use of larger than minimum gate length access transistors and holding the bit lines at around $V_{dd}/2$ to combat high transistor off-currents. Writing into the memory is locally controlled by the comparator. During readout, single-ended charge-redistribution column sense-amps, located in the periphery and not shown in the figure, are used for robustness against the effects of capacitive coupling between the closely spaced bit lines.

The comparator and pixel-level memory circuits can be electrically tested by applying analog signals to the sense node through the V_{set} signal, performing A/D conversion using the normal input ramp and the on-chip gray-code generator, and then reading out the digitized values. In this way, except for the photodetectors, the DPS chip can be electrically tested and characterized without the need for light or optics.

Figure 4.5 shows the layout of a 2×2 pixel block. The four large squares are the photogates, which are sized and spaced equally in the horizontal and vertical dimensions. The fill factor of this pixel is 15%. The silicide layer, which is opaque, was blocked from the photogates. The 3-stage comparators are seen near the top and bottom of the pixel quad. The digital memory is located in the two sections near the center of the quad. The smaller squares are the capacitors, with the transfer and reset transistors near by.

The pixels are mirrored about the horizontal axis in order to share the n-well and some of the power and bias lines. With digital CDS as discussed in Section 4.1.3, we did not observe any offset FPN due to mirroring. A small layout asymmetry, however, has resulted in odd/even row gain FPN. Memory bitlines (metal 3) and digital ground (metal 1) run vertically over the memory, while analog signal (metal 2) and power

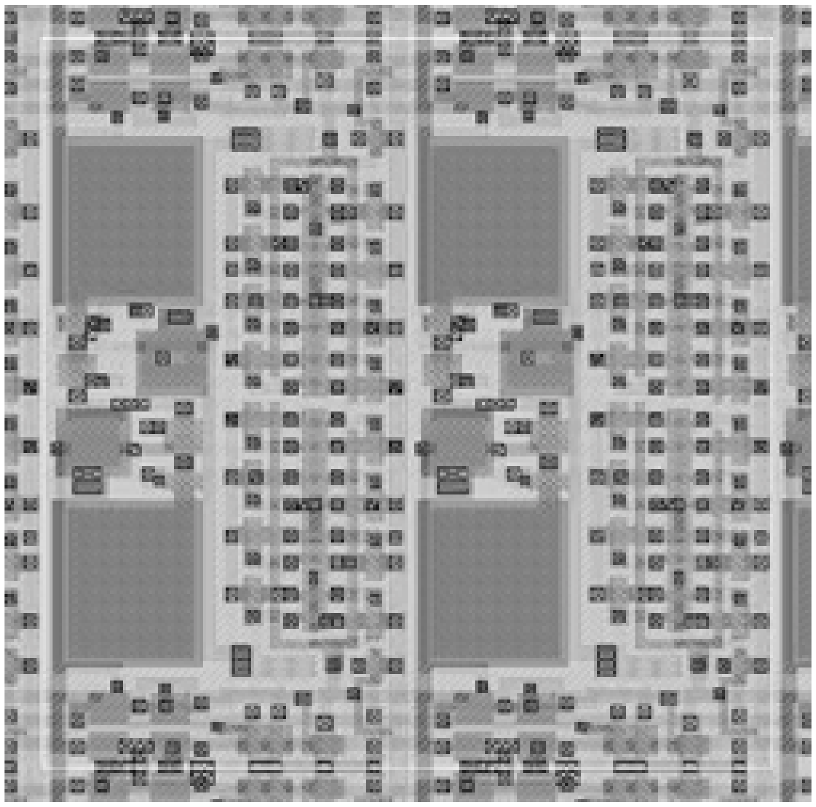


Figure 4.5: DPS pixel layout (2×2 pixel block shown). Pixel size is $9.4 \times 9.4\mu$m.

distribution (metal 4) run horizontally on top of the comparators. Metal 5 covers most of the array and acts as a light shield. Pixel array analog and digital grounds are kept separate in order to reduce noise coupling from the digital memory into the sensitive analog components.

4.1.3 Sensor operation

In this section we describe the details of the DPS chip operation. First we describe the A/D conversion operation. Next we discuss the basic imaging modes of operation including single frame capture, digital correlated double sampling, high speed operation, and multiple image capture.

A/D conversion operation

Figure 4.6 illustrates the per-pixel single-slope A/D conversion technique used in our chip. The globally distributed voltage ramp is connected to each pixel's comparator inverting ("–") input. The non-inverting ("+") input on each comparator is directly connected to the sense node. The globally distributed gray coded counter values, shown as a stepped "digital ramp," are simultaneously applied to the per-pixel memory bit lines.

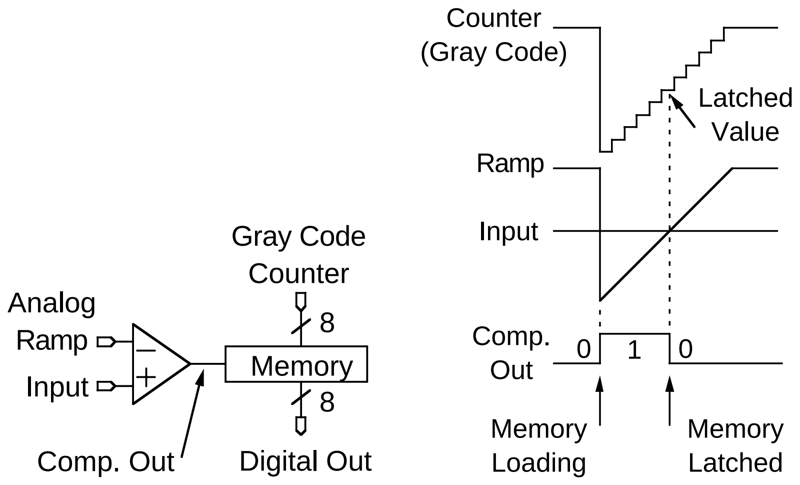


Figure 4.6: Single-slope ADC operation.

At the beginning of conversion, the ramp voltage is lowered to just below the lowest expected sense node voltage, which sets the comparator output to high. This enables the per-pixel memory to begin loading the gray code values. The ramp is then swept linearly until it exceeds the reset voltage. Simultaneously, the gray code counter sweeps across an equivalent set of values (256 for 8 bits). As the ramp crosses each pixel's sense node voltage, its comparator output switches low, and the gray code value present at that moment is latched in the pixel's memory. At the end of conversion, each pixel's memory contains an 8-bit gray coded value that is a digital representation of its input voltage.

Although using a linear ramp is the typical approach, it is possible to use alternative ramp profiles such as piecewise linear or exponential curves that compress or expand different illumination ranges. It is also possible to change the gain of the A/D conversion by changing the voltage range of the analog ramp. One may also use alternate sequences for the digital inputs using the auxiliary inputs.

Imaging modes

Figure 4.7 depicts a simplified timing diagram for the DPS chip. Operation can be divided into four main phases: reset, integration, A/D conversion, and readout. The reset, integration and A/D conversion phases occur completely in parallel over the entire array, *i.e.*, in "snap-shot" mode, thus avoiding image distortion due to the row by row reset and readout of APS. To minimize charge injection into the sense node, which causes high FPN, a shallow reset signal falling edge is used. Sensor integration is limited by dark current or signal saturation on the long end and by internal time constants on the short end. Practical lower and upper bounds on integration time were found to be under 10μs to well over 100ms.

After integration, per-pixel single-slope A/D conversion is simultaneously performed for all pixels, as discussed in the previous subsection. Typical conversion time is 25μs, and can be as low as 20μs at the highest frame rates. After conversion, readout commences. The readout of one frame is completed in around 75μs.

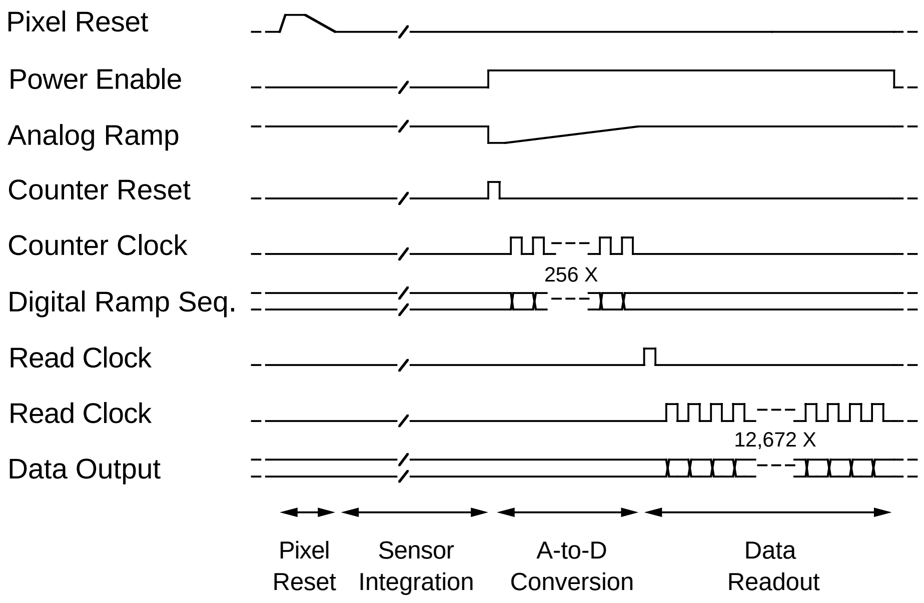


Figure 4.7: Simplified DPS timing diagram.

4.1.4 Testing and characterization

The DPS chip has been tested and characterized and shown to be fully functional. In the following subsections, we present the electrical, optical, and noise characterization results, and show results demonstrating that digital readout noise has little or no effect on the imaging performance of the chip.

Table 4.2 summarizes the DPS characterization results. Of particular interest is the measured average power consumption of only 50mW at 10,000 frames/s. The pixel-level comparators consume around 30mW of static power, while the digital readout circuits consume around 20mW of dynamic power. The poor imaging performance of the standard 0.18μm CMOS process resulted in high dark signal of 130mV/s and low QE of 13.6%. The major reason for the low QE is the high recombination rate in the highly doped substrate. With conversion gain of 13.1μV/e$^-$, sensitivity was just over 0.1V/lux.s. Dark current and QE can be significantly improved with minor process modifications that should not significantly affect pixel area or chip performance. The ADC integral nonlinearity (INL) was measured over the maximum useful input range of 1V, at a typical 1,000 frames/s, without correlated double sampling, and averaged for all pixels. It was found to be 0.22% or 0.56 LSB. We also found that reducing the swing to 0.9V improves INL to 0.1% or 0.25 LSB.

Power used at 10K fps	50 mW, typical
ADC architecture	Per-pixel single-slope
ADC resolution	8-bits
ADC conversion time, typical	$\sim 25\mu$s, ($\sim 20\mu$s, min.)
ADC range, typical	1 V
ADC integral nonlinearity	<0.22% (0.56 LSB)
Dark current	130 mV/s, 10 nA/cm^2
Quantum efficiency	13.6%
Conversion gain	13.1 μV/e$^-$
Sensitivity	0.107 V/lux.s
FPN, dark w/CDS	0.027% (0.069 LSB)
Temporal noise, dark w/CDS	0.15% (0.38 LSB)

Table 4.2: DPS chip characterization summary. All numbers, except for power consumption are at 1000 frames/s.

To determine dark current, conversion gain, and QE of the DPS pixel, our chip included single pixel APS and DPS test structures that can be individually accessed and whose sense node voltages can be directly readout. The test structures are described in detail in [79]. For completeness, we provide the results that are relevant to the DPS chip.

Sample sequence

Figure 4.8 shows 12 frames from a milk drop splashing sequence. The sensor was operated at 1400 frames/s where the odd frames were the dark frames captured immediately after reset. We performed digital CDS, resulting in a sequence with equivalent frame rate of 700 frames/s. In the figure, only every 10th frames are shown. It is interesting to see the details of milk drop splashing captured through the high speed camera. The overall image quality appears to be satisfactory for high speed motion analysis and other high speed video applications.

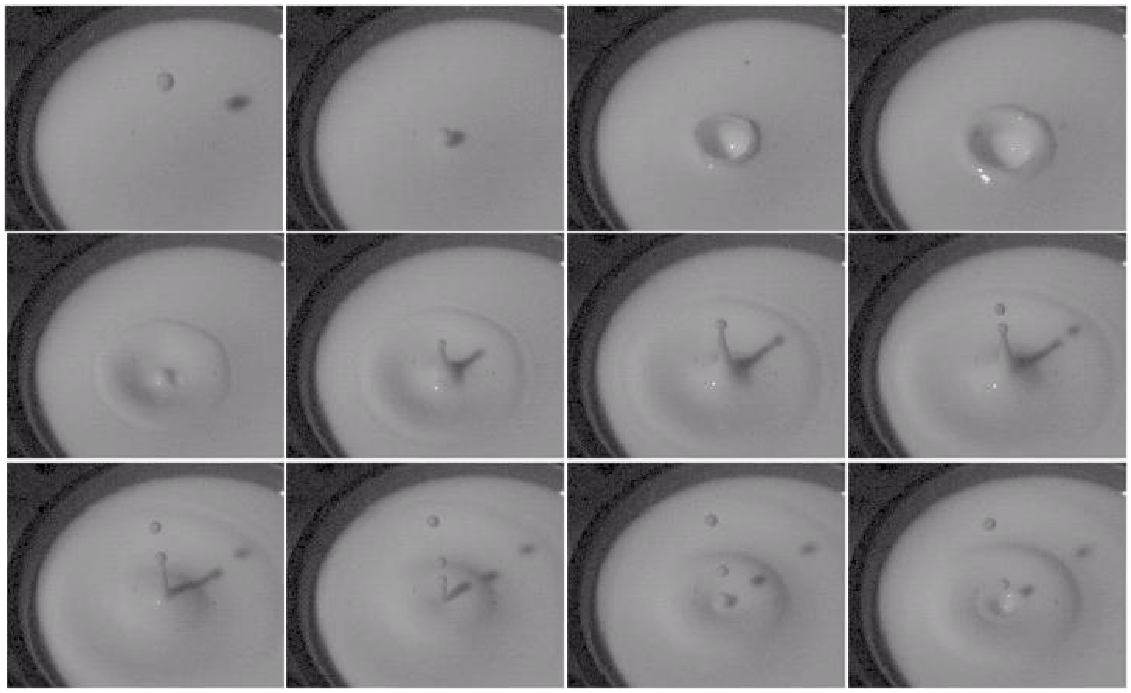


Figure 4.8: A 700 frames/s video sequence (frames 100, 110,. . . , 210 are shown)

4.1.5 Summary

A Digital Pixel Sensor implemented in a standard digital CMOS 0.18μm process was described. The 3.8 million transistor chip has 352×288 pixels. Each $9.4 \times 9.4\mu$m pixel contains 37 transistors implementing a photogate circuit, an 8-bit single-slope ADC, and 8 3T DRAM cells. Pixel reset, integration and A/D conversion occur in full frame parallel “snap-shot” fashion. Data is read out via a 64 bit wide bus at 167 MHz for a peak data bandwidth of 1.34GB/s. The DPS chip achieved continuous 10,000 frames/s operation and sustained 1 Gpixels/s throughput, while using only 50 mW of power. With further scaling, significant additional per-pixel memory, processing power and speed will inevitably become practical, further enhancing the capabilities of the DPS approach.

4.2 Memory and processing integration limits

In this section we explore the limits of integrating memory and processing with a CMOS image sensor in 0.18μm process and below. Our purpose is to demonstrate that only integrating the camera system in Figure 1.2 underutilizes the possible on chip processing power and to show that applications such as optical flow estimation can be performed on a single chip imaging system.

The single chip imaging system architecture we consider is shown in Figure 4.9. It comprises an APS with column level ADC or a DPS (with pixel level ADC), frame memory with wide write bus from the sensor, a SIMD processor array, and a controller. This is a natural choice since most image processing algorithms are spatially local and shift invariant. Several researchers have investigated implementations of this generic architecture [86, 87, 88, 89]. Forchheimer *et al.* [86] describe a 1.2μm CMOS chip with an APS with 8bit ADC, 8bit bi-directional shift register, 128bits of memory, and a bit-serial processor per column. Hong *et al.* [87] describe an array of SIMD processors performing video compression using vector quantization. Each processor performs 87 operations/pixel·frame at 30 frames/s and handles 16 columns. Hsieh *et al.* [89] discuss a video compression architecture for single-chip digital CMOS camera. Each processor handles 16 columns and is designed for MPEG2 encoder and DV encoder, which require 1.8 billion operations per second.

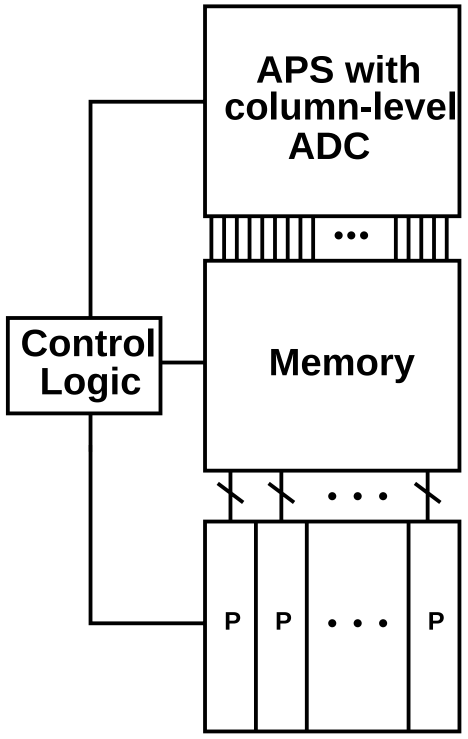


Figure 4.9: Single chip imaging system architecture.

To explore the memory size and processing that can be integrated with an image sensor at 0.18μm process and below, we make the following assumptions:

- We assume that the sensor captures video sequence at constant frame rate of f_s frames/s, and outputs video sequence at a standard frame rate of 30 frames/s. We denote the oversampling factor by $OV = \frac{f_s}{30}$.
- We assume a fixed die core area of 1cm$\times$1cm.
- We assume a 640×480 image sensor array size with 5μm$\times$5μm pixel size. Thus it occupies an area of 3.2$\times$2.4 mm^2. We assume that the row readout speed of the sensor is fast enough and is not the limiting factor in determining the throughput.

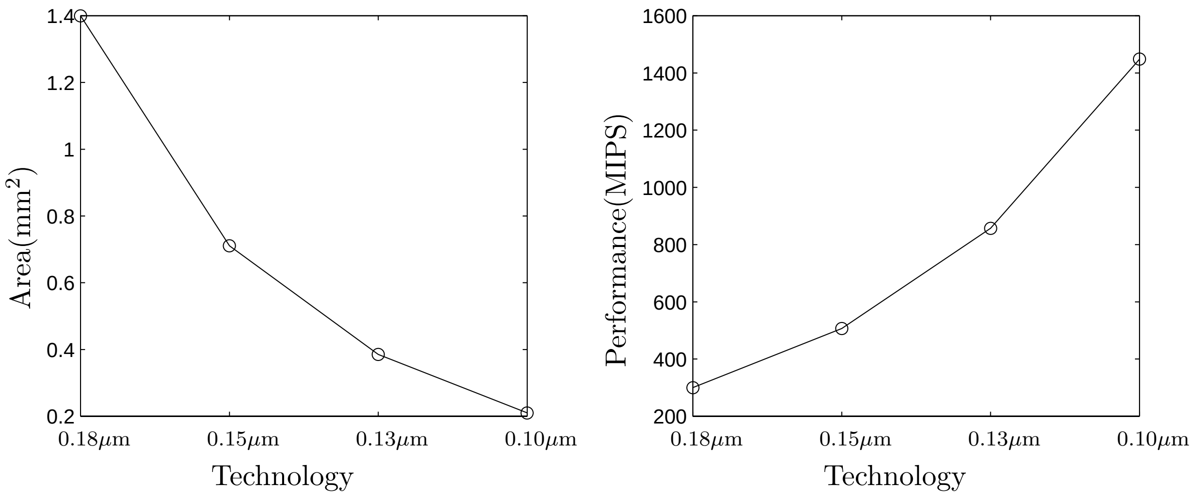


Figure 4.10: Area and performance of embedded processor as a function of process generation.

- We assume SIMD processor array with each processor comprising a 32-bit RISC core with a dedicated hardware MAC and occupying 1.4mm^2 [90]. The area and performance of the processor as a function of process generation is given in Figure 4.10 assuming no change in processor architecture [91, 92].
- We assume that the memory uses embedded DRAM, which typically lags commodity DRAM by one process generation. The embedded DRAM density (Gbit/cm^2) including overhead for 0.18μm technology and below is provided [91] in Figure 4.11.
- We assume that 68mm^2 of chip core area is available for frame memory and SIMD processor array after subtracting off the image sensor area and an estimate of 24.32mm^2 of ADC, control logic, and routing overhead.

Figure 4.12 shows the integration limits for 0.18μm technology using the above assumptions. The lines represent the maximum number of operations/pixel·frame versus the maximum number of bytes/pixel possible for OV values of 1, 4 and 10.

Table 2 lists the memory and processing requirements for performing the applications in a conventional digital video camera system (see Figure 1.2) operating at the

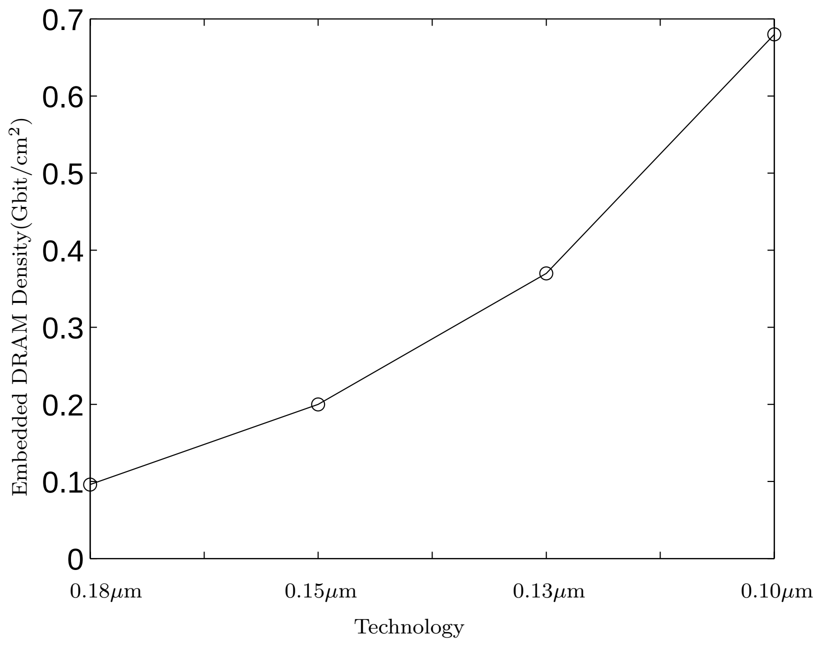


Figure 4.11: Embedded DRAM density as a function of process generation.

Application	Operations/pixel·frame	Number of bytes/pixel
Color processing	32	3
JPEG	68	3
MPEG2	220	7

Table 4.3: Processing and memory required to implement digital video camera system.

standard 30 frames/s rate. The estimated numbers provided in the table assume the following:

- Color processing includes color interpolation, white balancing, color correction and gamma correction.
- Color interpolation is performed using bilinear interpolation with kernel size of 3×3.
- White balancing is performed using the simple Gray world algorithm.

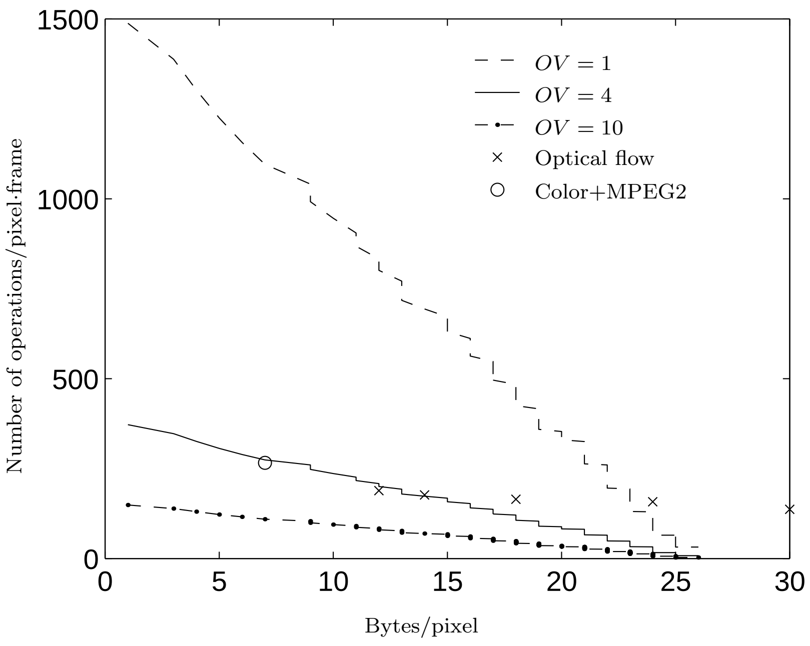


Figure 4.12: Maximum number of operations/pixel·frame vs. maximum number of bytes/pixel in 0.18μm CMOS process.

- Color correction is performed by multiplying a 3×3 matrix to a vector formed by R,G and B values in each pixel.
- MPEG2 is assumed to have 50% of B-frames.
- Motion estimation assumed the 3-step logarithmic search algorithm [93].

The memory and processing requirements for implementing color processing and MPEG2 encoding functions with the image sensor on a single chip are 252 operations/pixel per frame and 7 bytes of memory per pixel, respectively. This is plotted in Figure 4.12 for 0.18μm technology. Note that the requirements are not only easily satisfied, but that the available on-chip processing power is not fully utilized.

The processing and memory requirements for the optical flow estimation algorithm described in section 2 are also plotted in Figure 4.12. Different points represent the trade-offs between processing and memory requirements. It is clear from the plot that we can perform optical flow estimation at 120 frame/s with $OV = 4$ in 0.18μm technology.

Figure 4.13 shows the integration limits for 0.15μm down to 0.1μm technologies. The lines represent the maximum number of operations/pixel·frame versus the maximum number of bytes/pixel possible for the different technology generations at $OV = 10$. The figure clearly demonstrates a key assertion of our work – that merely integrating the functions of a conventional digital camera does not fully exploit the potential advantages of integration. As technology scales more compute intensive applications that can take advantage of high speed imaging such as optical flow estimation, tracking, gain FPN correction, motion segmentation, and 3D structure estimation can be implemented on a single chip imaging system.

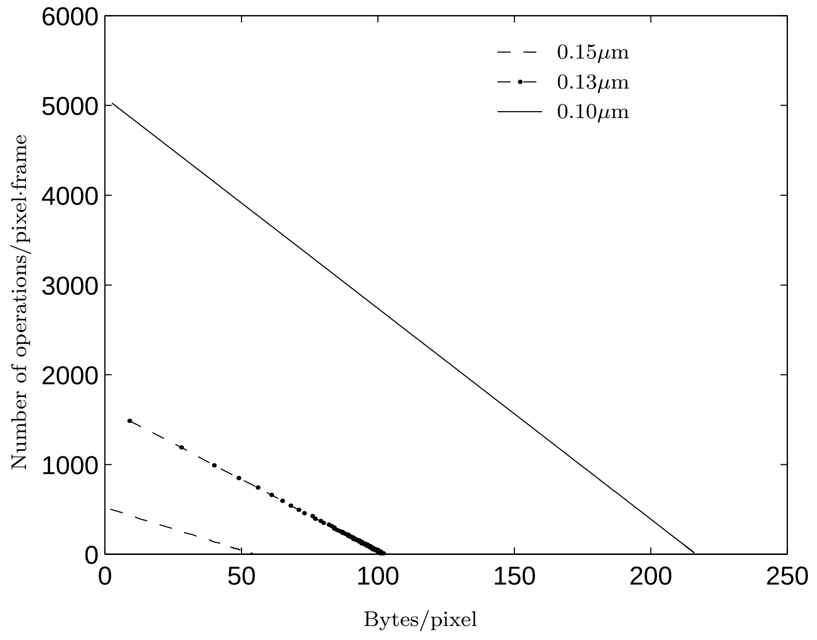


Figure 4.13: Maximum number of operations/pixel·frame vs. maximum number of bytes/pixel in 0.15μm, 0.13μm and 0.10μm technologies at $OV = 10$.

Chapter 5

Summary and Future Work

5.1 Summary

An important trend in the design of digital cameras is the integration of capture and processing onto a single CMOS chip. Although integrating the components of a digital camera system onto a single chip significantly reduces system size and power, it does not fully exploit the potential advantages of integration. We argue that a key advantage of integration is the ability to exploit the high speed imaging capability of CMOS image sensors to enable new applications and to improve the performance of existing applications such as optical flow estimation. By integrating the memory and the processing with the CMOS image sensor on the same chip, the availability of high on-chip bandwidth can be used to alleviate the high data rate problem. In this thesis, we explored the idea of capturing images at much higher frame rates than the standard frame rate, processing the high frame rate data on chip, and outputing the video sequence and the application specific data at standard frame rate.

This idea has been previously applied to enhancing the image quality such as dynamic range extension and motion blur-free image capture. In these applications, the video data at each pixel were processed temporally while spatially neighboring pixels were not utilized in the processing. Since many important operations in video processing applications can benefit from *or require* the spatial information of neighboring pixels, it is important to extend this idea to 3D spatio-temporal processing. In this dissertation, we extended this idea from 1D temporal processing to 3D spatio-temporal processing. This led us to make three main contributions. First, we developed an optical flow estimation method that uses high frame rate sequences to estimate high accuracy optical flow. Second, we developed a method for correcting gain FPN using an arbitrary video sequence and its estimated optical flow. Third, we showed that it is feasible to implement such a high-speed imaging system by designing and building a high-speed CMOS image sensor and also calculating the projected limits of integrating memory and processing with the sensor. We next provide more detailed summaries of each contribution.

In the first part of the dissertation, we describe a method for providing improved optical flow estimation accuracy for video at a conventional standard frame rate, by initially capturing and processing the video at a higher frame rate. The method begins by estimating the optical flow between frames at the high frame rate using the well known Lucas-Kanade method, and then accumulates and refines these estimates to produce accurate estimates of the optical flow at the desired standard frame rate. We presented simulation results using sinusoidal input sequences showing that the minimum frame rate needed to achieve high accuracy is largely determined by the minimum frame rate necessary to avoid motion aliasing. Using synthetic input sequences generated by image warping of a still image, we also show the significant improvements in accuracy achieved using the proposed method. We also showed how the proposed method can be used with optical flow estimation algorithms other than the Lucas-Kanade algorithm. In particular, we extended the Haussecker algorithm to work with high frame rate sequences and showed that with this extension high accuracy optical flow estimates can be obtained even when brightness varies with time.

In the second part of the dissertation, we described a method for gain FPN correction using a video sequence and its estimated optical flow. This part serves as an illustration of how spatio-temporal processing can be used to enhance image quality. Conceptually, this method can be thought of as digital CDS that cancels gain FPN rather than offset FPN. It can be used in a digital video or still camera by taking a video sequence with motion prior to capture and using it to estimate gain FPN. The sequence is then used to estimate pixel gains by iteratively minimizing the sum of the squared brightness variations along the motion trajectories. To reduce complexity we divide the pixel array into blocks, as is commonly done in video coding, and perform the estimation separately for each block. We found that a block size of 5×5 pixels provides a reasonable trade-off between complexity and performance. Using iterative minimization methods also allows us to lower computational requirements and reduce gain FPN incrementally. We tested our gain FPN correction method on synthetically generated sequences and demonstrated significant gain FPN reduction even in the presence of motion blur and temporal noise.

In the third part of the dissertation, we discuss the hardware implementation issues of high speed CMOS imaging systems. To show that high frame rate capture is possible, we first presented a 352×288 CMOS Digital Pixel Sensor (DPS) chip. Fabricated in a standard 0.18μm process, this chip is the first ever published that has a single slope ADC and 8-bit digital memory per pixel. It achieves an ultra high frame rate of $10,000$ frames/s, at what we believe to be lower cost than commercially available high speed CCD image sensors. The chip has 3.8 million transistors while each $9.4 \times 9.4\mu$m pixel contains 37 transistors implementing a photogate circuit, an 8-bit single-slope ADC, and 8 3T DRAM cells. Pixel reset, integration and A/D conversion occur in full frame parallel “snap-shot” fashion. Data is read out via a 64 bit wide bus at 167 MHz for a peak data bandwidth of 1.34 GB/s. The DPS chip achieved sustained 1 Gpixels/s throughput, while using only 50 mW of power.

We then explore the limits of integrating memory and processing with a CMOS image sensor in 0.18μm process and below. We show that the integration of an entire video camera system on a chip is not only feasible at 0.18μm process, but in fact underutilizes the possible on-chip processing power. Further, we show that the on-chip processing power and memory are sufficient to perform applications such as optical flow estimation, and that as technology scales applications that may benefit from high speed imaging and require even more processing power and memory than optical flow estimation, such as tracking, pattern recognition, and 3D structure estimation, can be performed on a single chip digital imaging system.

5.2 Recommendation for future work

In this thesis, we discussed hardware and algorithmic aspects of high-speed imaging. Specifically, we developed an optical flow estimation algorithm and a gain fixed pattern noise reduction method that can benefit from high frame rate and showed that it is feasible to implement such a system on a single chip. One of the biggest contributions of this thesis is extending our high frame rate capture – standard frame rate output approach from 1D temporal processing to 3D spatio-temporal processing. Estimating optical flow and motion between frames is the first and a very important step for many video processing and computer vision applications and thus opens the door to exploiting high frame rate imaging capability for those applications. We believe that identifying and developing new video processing and computer vision applications that benefit from high frame rate sequences and their optical flow estimates is worth exploring.

Gain FPN correction, described in Chapter 3, served as an example of using high frame rate sequences to improve image quality. Many similar applications may be worth exploring, and we believe superresolution is a particularly promising application. Superresolution reconstruction algorithms produce a high resolution (HR) image from a sequence of low resolution (LR) images and their estimated optical flow. Higher frame rate can benefit superresolution reconstruction algorithm since the amount of motion blur decreases and optical flow can be estimated more accurately. Having less motion blur is advantageous since its effect is spatio-temporal low-pass filtering, which is detrimental for superresolution. In addition, more LR frames can be used to estimate the HR image, which can result in higher robustness and better fidelity. However, many existing superresolution algorithms can not fully utilize accurate optical flow since the optical flow is rounded to the grid size of the HR image and interpolation error occurs when warping is performed. Also, many such algorithms have very high computational complexity and memory requirements. Thus, a superresolution algorithm that can fully utilize the accurate optical flow with low complexity would be very beneficial. One way to lower computational complexity might be to divide the high resolution image into non-overlapping blocks and perform super-resolution on the blocks independently. Although this will make it difficult to utilize the spatio-temporal information near the boundary of the blocks, it will greatly reduce computational complexity and allow implementation of algorithms that employ more accurate modeling of the motion and the sensor.

There are other possible applications of high speed image sensors in computer vision and image-based rendering. For example, a high frame rate sequence and its optical flow estimates can be used to accurately track and estimate 3D structure and motion of objects. Current algorithms are based on standard frame rate cameras and better performance can potentially be achieved by using high-speed cameras. Also, high-speed cameras can potentially replace or complement arrays of cameras used in many computer vision and image-based rendering applications. Specifically, the use of high-speed cameras with proper motion coupled with development of proper algorithms can perform tasks that were only possible by costly array of cameras.

Bibliography

[1] I. Akiyama et al. “A 1280×980 Pixel CCD Image Sensor”. *Proceedings of the 1986 International Solid State Circuits Conference*, San Fransico, CA, February 1986.

[2] T. Nobusada et al. “Frame Interline CCD Sensor for HDTV Camera”. *Proceedings of the 1989 International Solid State Circuits Conference*, San Francisco, CA, USA, February 1989.

[3] H. Tseng, B.T. Nguyen, and M. Fattahi. “A high performance 1200*400 CCD array for astronomy applications”. *Proceeding of SPIE Electronic Imaging Conference*, volume 1071, pages 170–185, 1989.

[4] J. T. Bosiers et al. “An S-VHS Compatible 1/3” Color FT-CCD Imager with Low Dark Current by Surface Pinning”. *IEEE Transactions on Electron Devices*, 42(8):1449–1460, August 1995.

[5] T. Yamada et al. “A 1/2 inch 1.3M–Pixel Progressive–Scan IT–CCD for Still and Motion Picture Applications”. *Proceedings of the 1998 International Solid State Circuits Conference*, pages 178–179, San Francisco, CA, USA, February 1998.

[6] K. Itakura et al. “A 1mm 50k–Pixel IT CCD Image Sensor for Miniature Camera System”. *Proceedings of the 1998 International Solid State Circuits Conference*, pages 180–181, San Francisco, CA, USA, February 1998.

[7] Tetsuo Yamada, et al. “A Progressive Scan CCD Imager for DSC Applications”. *Proceedings of the 2000 International Solid State Circuits Conference*, Volume 43. pages 110–111, 2000

[8] Keisuke Hatano, et al. "A 1/3inch 1.3Mpixel Signle-Layer Electrode CCD with High-Frame-Rate Skip Mode". *Proceedings of the 2000 International Solid State Circuits Conference*, Volume 43. pages 112–113, 2000

[9] H.-S. Wong, "Technology and Device Scaling Considerations for CMOS Imagers", *IEEE Transactions on Electron Devices*, Vol. 43, No. 12, pp. 2131–2141, Dec. 1996.

[10] E.R. Fossum. "CMOS image sensors: electronic camera on a chip". *Proceedings of International Electron Devices Meeting*, pages 17–25, Washington, DC, USA, December 1995.

[11] S.K Mendis et al. "Progress in CMOS active pixel image sensors". *Charge-Coupled Devices and Solid State Optical Sensors IV–Proceedings of the SPIE Electronic Imaging Conference*, volume 2172, pages 19–29, San Jose, CA, USA, February 1994.

[12] P.B. Denyer et al. "Intelligent CMOS imaging". *Charge-Coupled Devices and Solid State Optical Sensors IV–Proceedings of the SPIE Electronic Imaging Conference*, volume 2415, pages 285–91, San Jose, CA, USA, February 1995.

[13] P.B. Denyer et al. "CMOS image sensors for multimedia applications". *Proceedings of IEEE Custom Integrated Circuits Conference*, volume 2415, pages 11.15.1–11.15.4, San Diego, CA, USA, May 1993.

[14] P.Denyer, D. Renshaw, G. Wang, M. Lu, and S.Anderson. "On-Chip CMOS Sensors for VLSI Imaging Systems". *VLSI-91*, 1991.

[15] P.Denyer, D. Renshaw, G. Wang, and M. Lu. "A Single-Chip Video Camera with On-Chip Automatic Exposure Control". *ISIC-91*, 1991.

[16] M. Gottardi, A. Sartori, and A. Simoni. "POLIFEMO: An Addressable CMOS 128×128 - Pixel Image Sensor with Digital Interface". Technical report, Istituto Per La Ricerca Scientifica e Tecnologica, 1993.

[17] W. Hoekstra et al. "A memory read–out approach for 0.5μm CMOS image sensor". *Proceedings of the SPIE Electronic Imaging Conference*, volume 3301, San Jose, CA, January 1998.

[18] Lliana Fujimori, et al. "A 256x256 CMOS Differential Passive Pixel Imager with FPN Reduction Techniques". *Proceedings of the 2000 International Solid State Circuits Conference*, Volume 43. pages 106–107, February 2000.

[19] H.S. Wong. "CMOS active pixel image sensors fabricated using a 1.8V 0.25um CMOS technology". *Proceedings of International Electron Devices Meeting*, pages 915–918, San Francisco, USA, December 1996.

[20] B. Dierickx. "Random addressable active pixel image sensors". *Advanced Focal Plane Arrays and Electronic Cameras – Proceedings of the SPIE Electronic Imaging Conference*, volume 2950, pages 2–7, Berlin, FRG, October 1996.

[21] F. Pardo et al. "Response properties of a foveated space-variant CMOS image sensor". *IEEE International Symposium on Circuits and Systems Circuits and Systems Connecting the World – ISCAS 96*, Atlanta, GA, US, May 1996.

[22] J. Solhusvik. "Recent experimental results from a CMOS active pixel image sensor with photodiode and photogate pixels". *Advanced Focal Plane Arrays and Electronic Cameras – Proceedings of the SPIE Electronic Imaging Conference*, volume 2950, pages 18–24, Berlin, FRG, October 1996.

[23] R.A. Panicacci et al. "128 Mb/s multiport CMOS binary active-pixel image sensor". *Proceedings of the 1996 International Solid State Circuits Conference*, pages 100–101, San Francisco, CA, USA, February 1996.

[24] R.H. Nixon et al. "256×256 CMOS active pixel sensor camera-on-a-chip". *Proceedings of the 1996 International Solid State Circuits Conference*, pages 178–179, San Francisco, CA, USA, February 1996.

[25] M. Yamawaki et al. "A pixel size shrinkage of amplified MOS imager with two-line mixing". *IEEE Transactions on Electron Devices*, 43(5):713–19, May 1996.

[26] E.R. Fossum. “Ultra low power imaging systems using CMOS image sensor technology”. *Advanced Microdevices and Space Science Sensors – Proceedings of the SPIE Electronic Imaging Conference*, volume 2267, pages 107–111, San Diego, CA, USA, July 1994.

[27] E.R. Fossum. “Active Pixel Sensors: are CCD’s dinosaurs”. *Proceedings of SPIE Electronic Imaging Conference*, volume 1900, pages 2–14, San Jose, CA, February 1993.

[28] A. Dickinson, S. Mendis, D. Inglis, K. Azadet, and E. Fossum. “CMOS Digital Camera With Parallel Analog-to-Digital Conversion Architecture”. *1995 IEEE Workshop on Charge Coupled Devices and Advanced Image Sensors*, April 1995.

[29] J.E.D Hurwitz et al. “800–thousand–pixel color CMOS sensor for consumer still cameras”. *Proceedings of the SPIE Electronic Imaging Conference*, volume 3019, pages 115–124, San Jose, CA, January 1997.

[30] O. Yadid-Pecht et al. “Optimization of noise and responsivity in CMOS active pixel sensors for detection of ultra low–light levels”. *Proceedings of the SPIE Electronic Imaging Conference*, volume 3019, pages 125–136, San Jose, CA, January 1997.

[31] Kazuya Yonemoto, et al. “A CMOS Image Sensor with a Simple FPN-Reduction Technology and a Hole-Accumulated Diode”. *ISSCC2000 Technical Digest*, Volume 43. pages 102–103, 2000

[32] Tadashi Sugiki, et al. “A 60mW 10b CMOS Image Sensor with Column-to-Column FPN Reduction”. *Proceedings of the 2000 IEEE International Solid State Circuits Conference*, Volume 43. pages 108–109, 2000

[33] Kwang-Bo Cho, et al. “A 1.2V Micropower CMOS Active Pixel Image Sensor for Portable Applications”. *Proceedings of the 2000 IEEE International Solid State Circuits Conference*, Volume 43. pages 114–115, 2000

[34] Eric R. Fossum, "Digital camera system on a chip," *IEEE Micro*, vol. 18, no. 3, pp. 8–15, May-June 1998.

[35] Marc J. Loinaz, Kanwar Jit Singh, Andrew J. Blanksby, David A. Inglis, Kamran Azadet, and Bryan D. Ackland, "A 200mW 3.3V CMOS color camera IC producing 352×288 24-b video at 30 frames/s," *IEEE Journal of Solid-State Circuits*, vol. 33, no. 12, pp. 2092–2103, December 1998.

[36] Stewart Smith, J.E.D. Hurwitz, Mairi J. Torrie, Donald J. Baxter, Andrew A. Murray, Paul Likoudis, Andrew J. Holmes, Mark J. Panaghiston, Robert K. Henderson, Stuart Anderson, Peter B. Denyer, and David Renshaw, "Single-chip 306×244 pixel CMOS NTSC video camera and a descenedant coprocessor device," *Proceedings of the 1998 IEEE International Solid State Circuits Conference*, vol. 33, no. 12, pp. 2104–2111, December 1998.

[37] B. Ackland and Alex Dickinson, "Camera on a chip," *Proceedings of the 1996 IEEE International Solid-State Circuits Conference*, vol. 39, pp. 22–25, February 1996.

[38] A. Krymski, D. Van Blerkom, A. Andersson, N. Block, B. Mansoorian, and E.R. Fossum, "A High Speed, 500 Frames/s, 1024×1024 CMOS Active Pixel Sensor," *Symposium on VLSI Circuits*, pp. 137–138, 1999.

[39] N. Stevanovic, M. Hillegrand, B.J. Hostica, and A. Teuner, "A CMOS Image Sensor for High Speed Imaging," *Proceedings of the 2000 International Solid State Circuits Conference*, no. 104-105, February 2000.

[40] Stuart Kleinfelder, Suk Hwan Lim, Xinqiao Liu, and Abbas El Gamal, "A $10,000$ Frame/s 0.18μm CMOS Digital Pixel Sensor with Pixel-Level Memory," *Proceedings of the 2000 International Solid State Circuits Conference*, pp. 88–89, February 2001.

[41] Stuart Kleinfelder, Suk Hwan Lim, Xinqiao Liu, and Abbas El Gamal, "A $10,000$ Frame/s 0.18μm CMOS Digital Pixel Sensor with Pixel-Level Memory,"

IEEE Journal of Solid-State Circuits, vol. 36, no. 12, pp. 2049–2059, December 2001.

[42] D. Handoko, S. Kawahito, Y. Takokoro, M. Kumahara, and A. Matsuzawa, "A CMOS Image Sensor for Focal-plane Low-power Motion Vector Estimation," *Symposium of VLSI Circuits*, pp. 28–29, June 2000.

[43] David Yang, Abbas El Gamal, Boyd Fowler, and Hui Tian, "A 640×512 CMOS Image Sensor with Ultra Wide Dynamic Range Floating-Point Pixel-Level ADC," *Proceedings of the 1999 IEEE International Solid State Circuits Conference*, pp. 308–309, February 1999.

[44] O. Yadid-Pecht and E.R. Fossum, "CMOS APS with autoscaling and customized wide dynamic range," *IEEE Workshop on Charge-Coupled Devices and Advanced Image Sensors*, pp. 48–51, June 1999.

[45] X. Liu and A. El Gamal, "Photocurrent estimation from multiple nondestructive samples in CMOS image sensors," *Proceedings of the SPIE Electronic Imaging Conference*, vol. 4306, January 2001.

[46] X. Liu and A. El Gamal, "Simultaneous Image Formation and Motion Blur Restoration via Multiple Capture," *IEEE International conference on Acoustics, Speech, and Signal Processing*, vol. 3, pp. 1841–1844, May 2001.

[47] S.H. Lim and A. El Gamal, "Integration of Image Capture and Processing – Beyond Single Chip Digital Camera," *Proceedings of the SPIE Electronic Imaging Conference*, vol. 4306, pp. 219–226, January 2001.

[48] J.L. Barron, D.J. Fleet, and S.S. Beauchemin, "Performance of Optical Flow Techniques," *International Journal of Computer Vision*, vol. 12(1), pp. 43–77, February 1994.

[49] C. Stiller and J. Konrad, "Estimating Motion in Image Sequences," *IEEE Signal Processing Magazine*, pp. 70–91, July 1999.

[50] A. Murat Tekalp, *Digital Video Processing*, Prentice-Hall, New Jersey, USA, 1995.

[51] S.H. Lim and A. El Gamal, "Optical Flow Estimation Using High Frame Rate Sequences," *Proceedings of the 2001 International Conference on Image Processing*, vol. 2, pp. 925–928, October 2001.

[52] S.H. Lim, J. Apostolopoulos and A. El Gamal, "Optical Flow Estimation Using High Frame Rate Sequences," *submitted to IEEE Transactions on Image Processing*

[53] B. D. Lucas and T. Kanade, "An iterative image registration technique with an application to stereo vision," *Proceedings of DARPA Image Understanding*, pp. 121–130, 1981.

[54] J.K. Kearney, W.B. Thompson, and D.L. Boley, "Optical flow estimation: An error analysis of gradient-based methods with local optimization," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 9(2), pp. 229–244, March 1987.

[55] R. Battiti, E. Amaldi, and C. Koch, "Computing Optical Flow Across Multiple Scales: An Adaptive Coarse-to-Fine Strategy," *International Journal of Computer Vision*, vol. 6(2), pp. 133–145, 1991.

[56] Joseph Weber and Jitendra Malik, "Robust Computation of Optical Flow in Multi-Scale Differential Framework," *International Journal of Computer Vision*, vol. 14, no. 1, pp. 67–81, 1995.

[57] Jonathan W. Brandt, "Analysis of Bias in Gradient-Based Optical Flow Estimation," in *Twenty-Eighth Asilomar Conference on Signals, Systems and Computers*, 1994, vol. 1, pp. 721–725.

[58] Eero P. Simoncelli, "Design of Multi-dimensional Derivative Filters," *IEEE International Conference on Image Processing*, vol. 1, pp. 790–794, 1994.

[59] Alan V. Oppenheim and Ronald W. Schafer, *Discrete-time Signal Processing*, Prentice-Hall, New Jersey, USA, 1999.

[60] Abbas El Gamal, *EE392B Classnotes: Introduction to Image Sensors and Digital Cameras*, http://www.stanford.edu/class/ee392b, Stanford University, 2001.

[61] P. Anandan, "A computational framework and an algorithm for the measurement of visual motion," *International Journal of Computer Vision*, vol. 2, no. 3, pp. 283–310, January 1989.

[62] Shahriar Negahdaripour and Chih-Ho Yu, "A Generalized Brightness Change Model for Computing Optical Flow," *Fourth International Conference on Computer Vision*, pp. 2–11, 1993.

[63] Shahriar Negahdaripour, "Revised Definition of Optical Flow: Integration of Radiometric and Geometric Cues for Dynamic Scene Analysis," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 20, no. 9, pp. 961–979, 1998.

[64] Horst W. Haussecker and David J. Fleet, "Computing Optical Flow with Physical Models of Brightness Variation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 23, no. 6, pp. 661–673, 2001.

[65] G.D. Hager and P.N. Belhumeur, "Efficient Region Tracking with Parametric Models of Geometry and Illumination," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 20, no. 10, pp. 1025–1039, 1998.

[66] D.W. Murray and B.F. Buxton, *Experiments in the Machine Interpretation of Visual Motion*, MIT Press, Cambridge, Mass., 1990.

[67] A. Verri and T. Poggio, "Motion Field and Optical Flow: Qualitative Properties," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 11, no. 5, pp. 490–498, 1989.

[68] Marco Mattavelli and Andre Nicoulin, "Motion estimation relaxing the constancy brightness constraint," *IEEE International Conference on Image Processing*, vol. 2, pp. 770–774, 1994.

[69] A. Singh, *Optical flow computation: a unified perspective*, IEEE Computer Society Press, Los Alamitos, CA, 1991.

[70] S.H. Lim and A. El Gamal, "Gain Fixed Pattern Noise Correction via Optical Flow Estimation," *Proceedings of the SPIE Electronic Imaging Conference*, vol. 4669, pp. 384–391, January 2002.

[71] S.H. Lim and A. El Gamal, "Gain Fixed Pattern Noise Correction via Optical Flow Estimation," *to be published in IEEE Transactions on Circuits and Systems*

[72] Andrew Blanksby and Marc J. Loinaz, "Performance Analysis of a Color CMOS Photogate Image Sensor," in *IEEE Transactions on Electron Devices*, January 2000, vol. 47, pp. 55–64.

[73] A. El Gamal, B. Fowler, H. Min, and X. Liu, "Modeling and Estimation of FPN Components in CMOS Image Sensors," *Proceedings of the SPIE Electronic Imaging Conference*, vol. 3301, pp. 168–177, April 1998.

[74] Hans S. Bloss, Juergen D. Ernst, Heidrun Firla, Sybille C. Schmoelz, Stephan K. Gick, and Stefan Lauxtermann, "High-speed Camera based on a CMOS Active Pixel Sensor," *Proceedings of SPIE Electronic Imaging Conference*, vol. 3968, pp. 31–38, February 2000.

[75] Ali Ercan, Feng Xiao, Xinqiao Liu, SukHwan Lim, Abbas El Gamal, and Brian Wandell, "Experimental High Speed CMOS Image Sensor System and Applications," *Proceedings of the First IEEE International Conference on Sensors*, June 2002.

[76] Gene H. Golub and Charles F. Van Loan, *Matrix Computations*, John Hopkins University Press, Maryland, USA, 1996.

[77] D. Yang, B. Fowler, and A. El Gamal. "A Nyquist Rate Pixel Level ADC for CMOS Image Sensors". *Proc. IEEE 1998 Custom Integrated Circuits Conference*, pages 237–240, May 1998.

[78] D. Yang, B. Fowler and A. El Gamal, "A Nyquist Rate Pixel Level ADC for CMOS Image Sensors", *IEEE Journal of Solid State Circuits*, Vol. 34, No. 3, pp. 348-356, Mar. 1999.

[79] Hui Tian, X.Q. Liu, S.H. Lim, S. Kleinfelder and A. El Gamal, "Active Pixel Sensors Fabricated in a Standard 0.18μ CMOS Technology", in *Proceedings of the SPIE Electronic Imaging Conference*, Vol. 4306, Jan. 2001.

[80] A. El Gamal, D. Yang and B. Fowler, "Pixel level Processing - Why, What and How?", *in Proceedings of SPIE Electronic Imaging Conference*, Vol. 3650, Jan. 1999, pp. 2-13.

[81] B.M Coaker, N.S. Xu, R.V. Latham and F.J. Jones, "High-speed imaging of the pulsed-field flashover of an alumina ceramic in vacuum", *IEEE Transactions on Dielectrics and Electrical Insulation*, Vol. 2, No. 2, pp. 210-217, Apr. 1995.

[82] B.T. Turko and M. Fardo, "High speed imaging with a tapped solid state sensor", *IEEE Transactions on Nuclear Science*, Vol. 37, No. 2, pp. 320-325, Apr. 1990.

[83] Robert G. Root and Paul Falkos, "Repetitively Pulsed Plasma illumination Source Improvements", in *Proceedings of the SPIE Electronic Imaging Conference*, Vol. 3642, pp. 104-115, Feb. 1999.

[84] O. Takahashi, et. al., "A 1 GHz Fully Pipelined 3.7 ns Address Time 8kx1024 Embedded DRAM Macro", in *Proceedings of the 2000 International Solid State Circuits Conference*, pp. 396-397, Feb. 2000.

[85] M. Fukaishi, et. al., "A 20 Gb/s CMOS Multi-Channel Transmitter and Receiver Chip Set for Ultra-High Resolution Digital Display", in *Proceedings of the 2000 International Solid State Circuits Conference*, pp. 260-261, Feb. 2000.

[86] R. Forchheimer, K. Chen, C. Svensson, and A. Odmark, "Single-Chip Image Sensors With a Digital Processor Array," in *Journal of VLSI Signal Processing*, vol. 5, pp. 121–131, 1993.

[87] S. H. Hong and W. Yang, "An Embeddable Low Power SIMD Processor Bank," *Proceedings of the 2000 International Solid State Circuits Conference* , pp. 192–193, February 2000.

[88] J. Gealow and C. Sodini, "A Pixel-Parallel Image Processor Using Logic Pitch-Matched to Dynamic Memory," in *IEEE Journal of Solid-State Circuits*, vol. 34, pp. 831–839, June 1999.

[89] J. Hsieh and T. Meng, "Low-Power Parallel Video Compression Architecture for a Single-Chip Digital CMOS Camera," in *Journal of VLSI Signal Processing*, vol. 21, pp. 195–207, 1999.

[90] http://www.mips.com, "Mips technologies, inc.,"

[91] http://public.itrs.net, "International Technology Roadmap for Semiconductors 1999 Edition,"

[92] M. J. Flynn, P. Hung, and K. W. Rudd, "Deep-submicron Microprocessor Design Issues," in *IEEE Micro*, vol. 19, pp. 11–22, July-August 1999.

[93] Vasuder Bhaskaran and Konstantinos Konstantinides, *Image and video compression standards, algorithms and architectures*, Kluwer Academic Publishers, USA, 1997.