LOW POWER FEATURE-EXTRACTION SMART CMOS IMAGE SENSOR DESIGN

ZHANG XIANGYU

School of Electrical and Electronic Engineering

A thesis submitted to the Nanyang Technological University
in partial fulfillment of the requirement for the degree of
Doctor of Philosophy

2017

Acknowledgements

I would like to express my sincere gratitude to my supervisor assistant professor CHEN Shoushun and my co-supervisor associate professor LAM Ying Hung, for their scientific guidance throughout my PhD study. I also express my thanks to School of Electrical and Electronic Engineering in Nanyang Technological University for awarding scholarship as financial support during my PhD study.

Appreciation is also extended to my fellow classmates, including Mr. Guo Menghan, Dr. Zhao Bo, Dr. Yuan Chao, Dr. Yu Hang and Dr. Qian Xinyuan, for their assistance and encouragement.

Last but not the least, I would like to express my sincerely thanks to my family members for their encouragement throughout my PhD study.

Contents

List of Figures

List of Tables

List of Abbreviations

CCD	Charge Coupled Device
CMOS	Complementary Metal Oxide Semiconductor
DSC	Digital Still Camera
VLSI	Very Large Scale Integration
APS	Active Pixel Sensor
AER	Address Event Representation
DVS	Dynamic Vision Sensor
FF	Fill Factor
3DIC	3D Integrated Circuit
FD	Floating Diffusion
IC	Integrated Circuit
CDS	Correlated Double Sampling
FPN	Fixed Pattern Noise
FIS	Front Illuminated Sensor
BIS	Back Illuminated Sensor
QE	Quantum Efficiency
PPS	Passive Pixel Sensor
DPS	Digital Pixel Sensor
SNR	Signal to Noise Ratio
TSL	Transversal Signal Line

DR	Dynamic Range
LoG	Laplacian of Gaussian
DoG	Difference of Gaussian
RGB	Red Green Blue
HSV	Hue Saturation Value
BJT	Bipolar Junction Transistor
PGA	Programmable Gain Amplifier
ADC	Analog to Digital Convertor
ROI	Region Of Interest
HDL	Hardware Description Language
CPLD	Complex Programmable Logic Device
OTA	Operational Transconductance Amplifier
PWM	Pulse Width Modulation
TFS	Time to First Spike
MIM	Metal Isolator Metal
DFF	D-type Flip Flop
UGB	Unity Gain Bandwidth
FPGA	Field Programmable Gate Array
SDRAM	Synchronous Dynamic Random Access Memory
USB	Universal Serial Bus
PCB	Printed Circuit Board
SPI	Serial Peripheral Interface
GUI	Graphical User Interface
SOI	Silicon On Insulator
RC	Resistor Capacitor
FOV	Field of View

UWB	Ultra Wide Band
WTA	Winner Take All
MAE	Mean Absolute Error

Abstract

Digital image sensors have been utilized in various applications including, but not limited to, consumer electronics, machine vision, and security surveillance. In the early days of digital cameras, charge-coupled-device (CCD) technology was dominant because of its superior imaging quality. In recent years, with the advancement of complementary-metal-oxide-semiconductor (CMOS) technology, CMOS image sensors have prevailed due to their lower power consumption and inexpensive fabrication cost. The design of commercial image sensors focuses on the improvement of imaging quality by increasing spatial resolution with smaller pixel size. When sensors without an on-chip computation capability are applied to the computer vision field, they produce massive amounts of raw image data. To extract valuable image features, such redundant raw data has to be transmitted and then processed by the vision processing system. This process wastes tremendous hardware resources and causes significant power consumption. Therefore, there is a growing demand for the smart image sensor that can perform on-chip image processing to directly export valuable image features such as spatial contrast or temporal motion.

In many computer vision applications, motion features are widely used to identify active objects from stationary backgrounds. Once active regions with motion activities are extracted, more advanced processing such as image compression, object segmentation, and pattern recognition can be implemented on the acquired images. Visual motion detection can be achieved by measuring light intensity changes along with

time on every pixel. Light intensities can be transformed into electrical signals in the form of currents, charges, or voltages. By detecting temporal variations in these signals with dedicated circuits, a motion-detection function can be implemented on image sensors. Various smart image sensors with on-chip motion feature-extraction capabilities have been reported in recent years. They can be classified into two main categories: “discrete mode” and “continuous mode”. In the first mode, an analog intensity image is acquired by integrating incident light within a certain period. A comparison of two consecutive frames can realize the motion-detection function. However, this algorithm suffers from a severe aliasing effect due to the discrete differentiation between two separate frames. In the “continuous mode”, incident light is directly converted into photo currents or intensity voltages without the integrating operation. By applying continuous differentiation on these signals, motion features can be extracted more efficiently. In order to report motion activities in real time, researchers innovatively developed an asynchronous address-event-representation (AER) strategy to export motion address events from smart image sensors to post processors. Unfortunately, image processing on address events is different with the conventional frame-based strategy because of their asynchronous nature, and specific event-based algorithms have to be customized based on applications.

The aim of this work is to design a smart CMOS image sensor that can extract and process motion features on the same chip so as to simplify the entire visual system with minimal computation burden. In order to achieve this ultimate goal, several smart motion-detection image sensors have been developed. The first image sensor was designed based on the temporal difference algorithm to on-chip export motion features, and this sensor is a compact visual system that integrates motion detection and feature processing on the same chip. In order to maximize photon sensitivity in every pixel, a feature-extraction image sensor based on an emerging 3D integration

technology was developed. The sensor can extract either temporal motion or spatial contour in real time. Also, an event-clustering image sensor specialized for object tracking applications was proposed. This sensor continuously detects motion events and on-chip processes them in parallel to export specific “event-flow” features. Simulation results show that this image sensor significantly simplifies and accelerates visual object tracking systems.

Chapter 1

Introduction

1.1 Motivation and Objective

1.1.1 Challenges of Digital Image Sensors

In the past three decades, there has been a significant advancement of digital image sensors from experimental prototypes in the lab into commercial products on the market. Nowadays, digital cameras have been utilized in various applications including consumer electronics, medical instruments, security surveillance, industrial manufacture and so on. In practice, these applications require image sensors with excellent performance in some specific functions. For example, in consumer electronics fields, such as the digital-still-camera (DSC), imaging quality is the dominant criteria for evaluating sensor performance. In medical instrument applications, such as the gastroscope, image sensors should be capable of operating in extreme conditions from time to time. In security surveillance fields, a wider dynamic range is preferable as image sensors can be required to work continuously over day and night. In industrial manufacture applications, robustness becomes the key factor to ensure an uninterrupted operation for a long time.

For many years, digital image sensors have been dominated by the charge-coupled-device (CCD) technology invented in Bell Labs. The main advantage of this technology

is that it can provide superior imaging quality in resolution and uniformity. However, convetional image sensors solely report raw intensity images with a huge amount of primitive information. In order to extract valuable image features, such as spatial contour or temporal motion, a massive amount of redundant raw data has to be transmitted and further processed by the entire visual system. This process wastes substantial bandwidth resources and causes tremendous power consumption. With the increase of the temporal and spatial resolution in digital image sensors, such limitation becomes even worse than ever before. For the sake of simplicity, the digital image sensor in iPhone 6 is chosen as an example for analysis. With a resolution of 3120 $\times$ 2340 and in a speed of 60 frames/s without image compression, the required data rate for the image sensor is over 1.34 GBytes/s. In this case, image storage will overflow within a few seconds and it is unfeasible for any existing visual processing system to render such huge amount of image data in real time. If basic feature-extraction functions are implemented on the focal plane of the image sensor, then system-level visual processing can be accelerated accordingly. How to efficiently integrate feature extraction onto image sensors has become an important research topic in recent years.

1.1.2 Design of Smart Image Sensors

Complementary-metal-oxide-semiconductor (CMOS) technology provides an alternative approach to develop image sensors. In contrast to CCD technology, CMOS process requires less power consumption and lower manufacturing cost. Moreover, CMOS image sensors are fabricated based on standard very-large-scale-integration (VLSI) process, which enables designers to integrate image acquisition and feature extraction on the same chip. Therefore, CMOS image sensors become versatile by implementing diverse image processing algorithms on the focal plane. The feature-extraction vision sensor is categorized as a smart image sensor [16].

In computer vision fields, temporal motion is one of the most important features to distinguish active regions from stationary backgrounds. When motion features are extracted from an intensity image, more advanced processing, such as image compression, pattern recognition, and object tracking, can be implemented on it [17]. Conventional computer-based motion extraction is realized by dedicated algorithms, which include but are not limited to, optical flow techniques [18], temporal difference methods [19], and background subtraction algorithms [20]. Among them, the temporal difference algorithm is the simplest one, which compares the difference between two consecutive frame images, and it can be implemented on conventional CMOS active-pixel-sensors (APS) by embedding an additional capacitor inside every pixel to buffer prior frame information [21]. These image sensors can simultaneously report two consecutive frame images to extract visual motions on the focal plane. However, image acquisition in this approach requires a long integration period, and there is a severe "aliasing effect", in which the temporal difference algorithm has an optimal response only when the sensor's frame rate matches the motion speed [22]. Furthermore, because of a scanning readout strategy, all pixels in a temporal-difference image sensor are sequentially reported to a common output channel, and the active pixels that detect visual motion activities cannot be attended immediately until they are accessed by the readout circuit with certain temporal delays. All above limitations constrain frame-based motion-detection image sensors to be utilized in high-speed applications [12].

In comparison to frame-based image sensors, biological visual systems from living beings work more efficiently by using neuron spikes to encode visual information in the viewing field. There is no artificial frame signal embedded in the stream of neuron spikes, and they are continuously transmitted in neural networks. In recent two decades, researchers in neuromorphic engineering have developed an event-driven strategy called address-event-representation (AER) as a universal protocol to communicate neuron spikes within artificial neural chips [23]. Many asynchronous image

sensors based on the AER protocol have been proposed in the literature [24]. Instead of sequentially scanning every pixel, an AER-based image sensor selectively allocates its output channel to active pixels in demand, which significantly increases readout efficiency with less redundancy. It is a fact that some vertebrate eyes are inherently embedded with basic feature-extraction capabilities, such as contrast extraction or motion detection [25]. In recent years, a variety of event-based smart image sensors that integrate feature-extraction functions and the AER communication strategy are developed to mimic biological eyes. A dynamic-vision-sensor (DVS) [12] is the ultimate one, and it is often categorized as a silicon retina which continuously quantizes relative intensity variations into a stream of asynchronous binary motion events to identify the vision changes in the viewing field. Compared to the frame-based readout strategy, data redundancy in event-based smart image sensors is significantly suppressed, which makes them quite appealing for high-speed vision applications.

Despite the high efficiency of AER image sensors, image processing on address events remains a challenge due to the fact that existing computer-based processing algorithms are dominated by frame-based approaches. Hence, address events from silicon retinas have to be preprocessed for adaption to frame-based algorithms. It is common to separate unordered address events based on a constant time slice to reconstruct pseudo frames. However, in this approach, it is highly probable that address events from the same region are assigned into separate frames, which causes inevitable confusion in subsequent vision processing procedures [26]. An elegant solution is to customize event-based image processing algorithms for specific applications, and it has become a new research trend as evidenced by numerous publications in recent years. An event-driven visual system that integrates motion sensing, event processing, and object recognition is reported in [27]. The proposed system is designed to constantly recognize and track a rotating dot in a certain size. However, the design is

quite complex with massive hardware cost, which prevents it from being used in practical applications. To simplify an event-driven vision system, it is feasible to implement motion event processing algorithms on smart silicon retinas. Hence, the smart image sensor not only extracts basic visual motion but also on-chip processes it to export some customized features for specific applications so that the entire vision system becomes more compact and efficient accordingly.

1.1.3 Objective and Contribution

The ultimate goal of this work is to design a smart image sensor that integrates image acquisition, feature extraction, and event processing on the same chip so that it can independently work as a complete visual system. The major contribution of this work resides in the implementation of image processing functions on the focal plane of the image sensor. Since image features are directly exported from front-end smart image sensors, post vision processors become much more compact with less computation burden. In order to achieve this ultimate goal, several smart image sensors implemented with different feature-extraction algorithms have been developed.

A frame-based motion-detection image sensor for kinetic object localization applications is proposed. The motion-detection function in this image sensor is developed by the common temporal difference algorithm. Binary motion events are periodically detected by comparing the difference between two consecutive frame images. There is an object localization module embedded on this image sensor, in which motion events are grouped into respective objects based on a distance criterion. When the main object of interest is localized, this sensor reports an intensity image on the object region. Since motion detection and event processing are implemented on the same chip, there is no additional computation or storage required in external processors. Another important feature of this image sensor is that it only consumes less than

0.4 mW under a recording speed of 100 frames/s, which makes it quite appealing for sensor network applications.

The implementation of the temporal difference algorithm in the first smart image sensor requires an additional capacitor inside every pixel, which occupies a certain silicon area and decreases the effective fill-factor (FF) for image acquisition. An emerging 3D integrated-circuit (3DIC) technology provides an ideal solution to this problem by placing image acquisition and processing circuit in separate tiers. A smart feature-extraction image sensor developed in the 3D integration technology is proposed. In this sensor, every pixel is embedded with a capacitor to store the acquired intensity information. Hence, the sensor simultaneously exports two consecutive frame images, and temporal difference computation in the post processor is achieved accordingly. Also, there is a programable resistor network connecting to the pixel array, which implements Gaussian filtering to the intensity image on capacitors. By comparing the difference between the original image with the Gaussian smoothed one, spatial contours in the acquired image are extracted accordingly. Since photodiodes in this sensor are separately allocated in the top tier which is fully exposed to incident light, this image sensor has an extreme fill factor over 95%.

Asynchronous event-based silicon retinas surpass frame-based counterparts with much higher efficiency by using an address-event-representation (AER) protocol. However, post image processing based on silicon retinas is a challenging work due to their asynchronous nature. Rather than direct exporting raw address events, it would be better to on-chip process them in silicon retinas so as to export some customized image features for specific applications. A smart image sensor that integrates motion detection and event processing on the same chip is proposed. In this sensor, a specialized event-clustering algorithm is implemented by dedicated circuits on every pixel to continuously process motion events in parallel. The transient response of the event-clustering algorithm is presented as analog voltages on the pixel array, and they are

termed as “event-flow” features whose amplitude indicates the density and frequency of motion events on the focal plane. A hybrid readout strategy in the proposed image sensor selectively exports those active pixels with sufficient event-flow features to an off-chip post processor where a pseudo frame of event-flow features is reconstructed. Hence, this image sensor is compatible with both frame-based and event-based image processing algorithms. Also, the event-clustering algorithm is customized for object tracking applications, and the sensor has been evaluated by comprehensive simulations in different object tracking frameworks.

1.2 Report Organization

The rest of this thesis is organized as follows:

Chapter 2 exhaustively reviews the related work in the literature. This chapter consists of three sections: conventional digital image sensors, computer-based image processing and smart motion-detection image sensors. The first section covers the fundamental principles of digital image sensors. After that, there is a discussion about the existing challenges in computer-based image processing, followed by an introduction on smart image sensors. Next, a variety of smart motion-detection image sensors reported in the literature are discussed in details. Their performance is summarized and compared at the end of this chapter.

Chapter 3 presents two frame-based smart image sensors. This chapter starts with deep analyses of the temporal difference algorithm. Then, the first frame-based motion-detection image sensor is proposed. There are detailed descriptions on the circuit implementation and experimental results of the image sensor. Next, a feature-extraction image sensor based on 3D integration technology is proposed. Also, the implementation of the feature-extraction algorithm in this image sensor is presented in

details. At the end of this chapter, there is a comparison between the proposed smart image sensors with those reported in the literature.

Chapter 4 reports an event-based motion-detection image sensor for object tracking applications. This chapter initially reviews the status of event-based image processing and highlights its existing challenges. Then, an event-clustering algorithm is proposed to process motion events on the focal plane. There is a discussion on the implementation of the event-clustering algorithm. Next, the proposed image sensor is evaluated by system-level simulations in different object tracking frameworks. The last section of this chapter draws a conclusion by comparing the proposed event-clustering image sensor with those reported in the literature.

Chapter 5 concludes this thesis and makes suggestions for future work.

Chapter 2

Literature Review

This chapter presents exhaustive reviews on the related work in the literature. The organization of this chapter follows the evolution of smart image sensors in the past three decades. Two major image acquisition technologies are introduced: charge-coupled-device (CCD) and complementary-metal-oxide-semiconductor (CMOS). This chapter also outlines the inevitable challenges when conventional image sensors are exploited in high-speed vision processing applications. The analyses indicate that the smart image sensor that integrates feature-extraction capabilities is an elegant solution to alleviate the problems outlined. Inspired by biological retinas, a variety of smart image sensors are proposed in the literature. They can be classified into two main categories based on the image acquisition strategy: "discrete mode" and "continuous mode". Both of them are deeply discussed and analyzed in this chapter. The smart image sensors elaborated in the literature are summarized and compared at the end of this chapter.

2.1 Fundamentals of CCD Image Sensors

The solid state imaging device is a special component that can convert photons into charges, when exposed to incident light. Since its invention by George Smith and Willard Boyle in Bell Labs [28], CCD technology has dominated the design of digital image

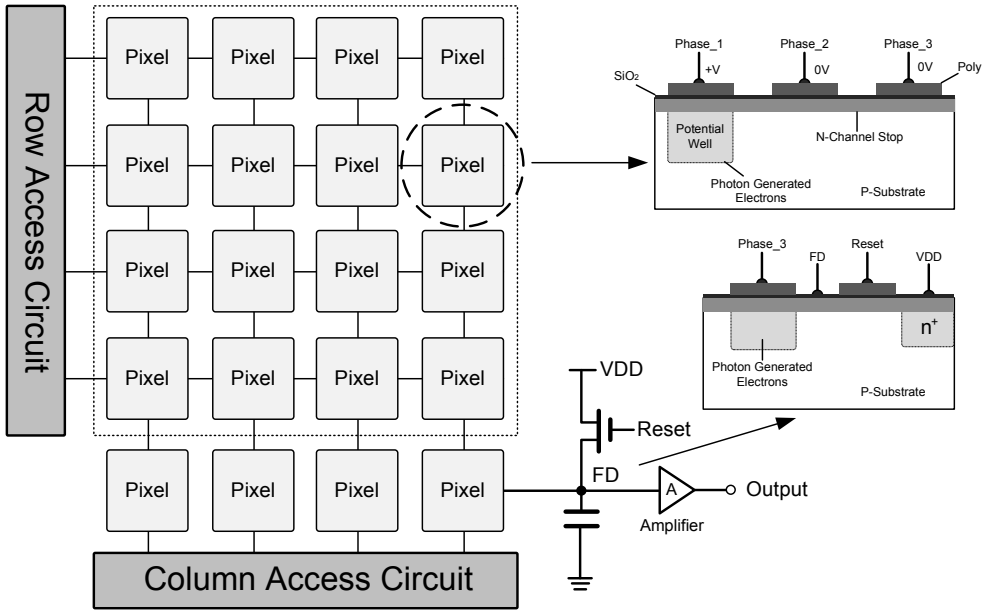


Figure 2.1: System architecture of a common CCD image sensor with the cross-sectional view of the pixel structure and readout path.

sensors for nearly 40 years up to the 21st century. A CCD imaging device is formed by thousands of light sensing pixels on a thin silicon substrate. The fundamental light sensing cell of the CCD sensor is a metal oxide silicon capacitor (MOSCAP), which serves as a photodiode and a storage capacitor. When a reverse biasing voltage is applied on the MOS capacitor, a potential well exists near to the interface between the silicon and oxide. Electrons generated by incident photons are migrated and stored in this deep potential well. After an integration period, the amount of electrons inside this well is linearly proportional to the photons it absorbs. When multiples of such structures are assembled in a single line or as a two dimensional array, a typical CCD image sensor is physically implemented.

Fig. 2.1 shows the system architecture of the typical CCD image sensor with main building blocks drawn in a cross-sectional view. In a two dimensional array, pixels are physically isolated by the barriers in the substrate. The image acquisition in a CCD

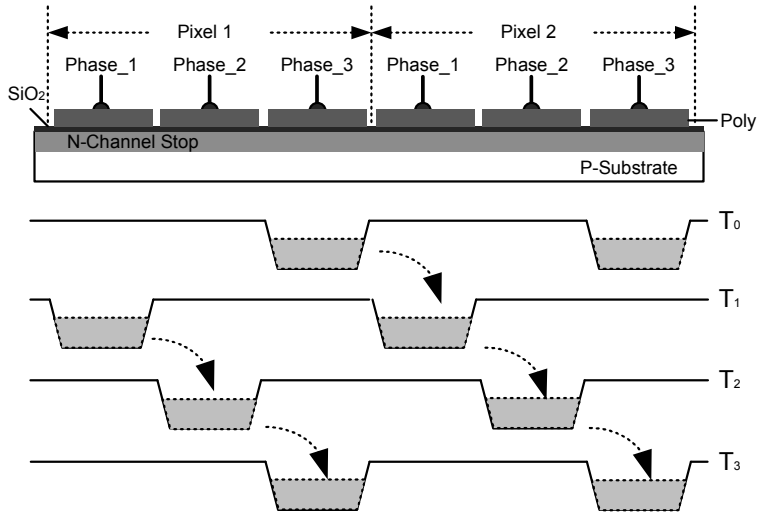


Figure 2.2: Three-phase pixel structure for CCD image sensors.

device involves three stages: photon collection, charge transfer and signal conversion. All these operations are controlled precisely by the biasing voltages on the pixels. Photon collection is accomplished by the aforementioned MOS capacitor. The most common charge transfer configuration is a three-phase CCD pixel design as shown in Fig. 2.2. Every pixel is divided into three individual phases with the parallel potential wells defined by the voltages applied on the electrodes, termed as "Phase_1", "Phase_2" and "Phase_3". During the integration period, shown as T_0 in the timing diagram, "Phase_3" is biased with a high voltage, while "Phase_1" and "Phase_2" are biased with low voltages. Hence, a potential well is created to collect photon-generated electrons. At the end of the integrating operation, during T_1 period, "Phase_1" is brought into a higher voltage while a low voltage is applied on "Phase_3". Hence, the integrated charges in the potential well are shifted from "Pixel_1" to "Pixel_2". Similarly, once "Phase_2" is changed into a high potential, charges are moved to the "Phase_2" region accordingly.

Such three-phase operations repeat one-by-one or row-by-row to achieve the parallel and serial charge transfer, as illustrated in Fig. 2.3 (a). In this figure, the bottom row works as a serial shift register, while other rows are shifted in parallel to this row.

The charge packages in the bottom row are sequentially shifted into a global sensing node which transforms photon charges into analog voltages for readout via an amplification buffer. Generally, the sensing node can be simply implemented using a floating-diffusion (FD) region with a reset transistor, shown in Fig. 2.1. The reset transistor is only turned on to clear FD for a new charge package when the readout on current pixel is completed by the external receiver. When the contents in the last row are serially readout to the sensing node, a new row of pixels can be loaded on the analog registers. The parallel and serial readout operations repeat until the entire pixel array is completely readout. The major advantage of this architecture is the high fill factor in every pixel (nearly 100%). However, the imaging focal plane has to be protected from incident light during the readout period, which significantly constrains the sensor's operating speed, as exposure and readout operations cannot proceed simultaneously. Therefore, the three-phase CCD device is restricted to low-speed applications, with a typical working speed of under 10 frames/s.

An alternative approach to increase the imaging speed is the inter-line scheme, as shown in Fig. 2.3 (b). Columns of photosensitive pixels and masked storage pixels are alternated over the entire pixel array [29]. As the charge transfer channel is located adjacent to each photosensitive column, the integrated charges can be shifted to the transfer channel immediately. The storage array is then readout by a combination of parallel and serial shift registers, and the imaging array is exposed for a new frame. The inter line transfer architecture allows a very short integration period by electronically controlling the exposure. Although the inter-line transfer device has the advantage of a higher frame rate with better imaging quality, it suffers from reduced resolution and lower sensitivity, as over 50% of the surface is occupied by the storage transfer array.

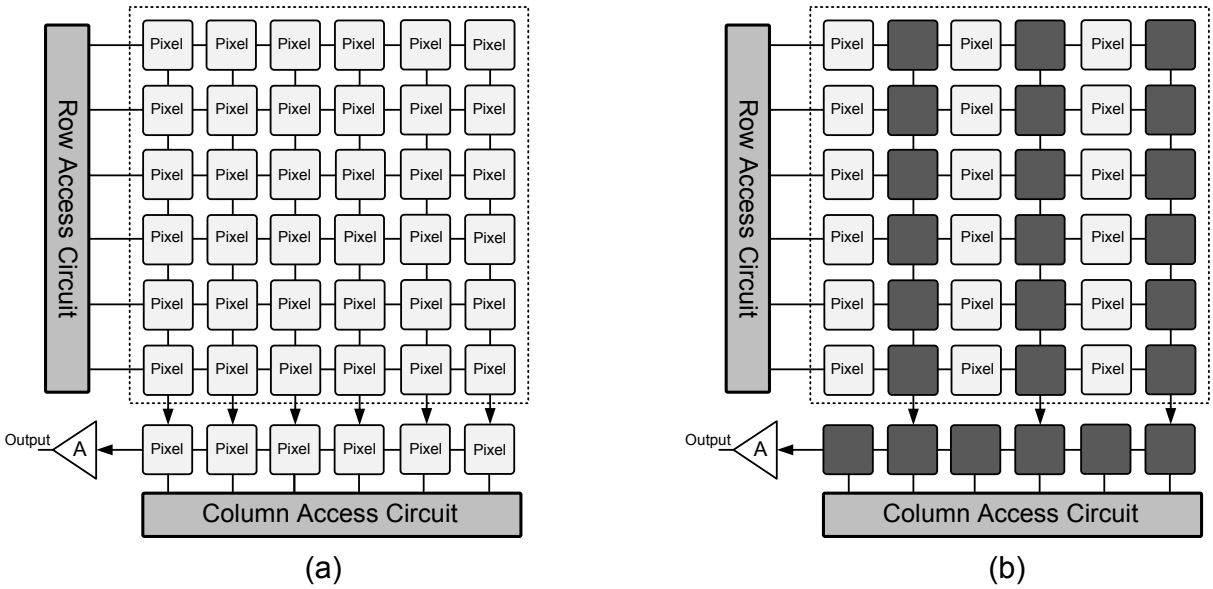


Figure 2.3: (a) Full-frame charge transfer scheme for CCD image sensors. (b) Interline charge transfer scheme for CCD image sensors.

2.2 Fundamentals of CMOS Image Sensors

The CCD image sensor replies on a dedicated and costly process. In contrast, the CMOS image sensor can be fabricated using standard CMOS technology which is compatible to that of common integrated-circuits (ICs). There is a fundamental difference in the readout strategy for CCD and CMOS image sensors. For CCD image sensors, incident photons are converted to electrical charges which are sequentially shifted to the sensing node when readout. The CMOS image sensors can convert photons into charges, voltages or currents based on diverse configurations. Furthermore, each pixel in the entire array can be individually or randomly accessed via row and column shift registers or address decoders. The major contribution of the CMOS image sensor is that image acquisition and processing circuits can be integrated on the same chip by a standard CMOS process.

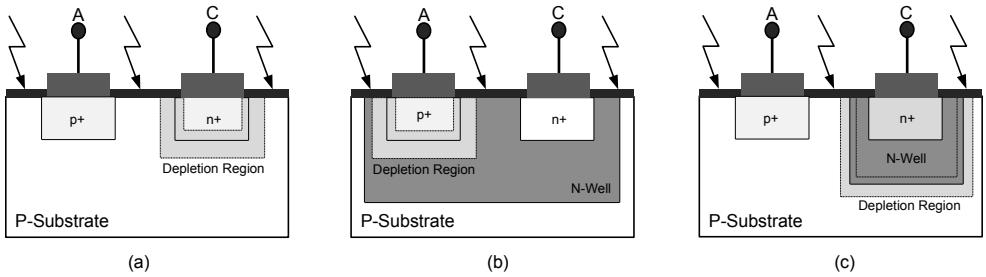


Figure 2.4: Available photodiode structures in CMOS technology: N+/P-substrate, P+/N-well and N-well/P-substrate.

2.2.1 Photodetector

Photodetectors are photosensitive devices that can detect incident light and also convert it to electrical signals. There are limited choices of photodetectors available in the standard CMOS process, which includes PN junction photodiodes, photogates, and pinned photodiodes [30, 31]. Such limitation can be compensated through innovative designs on subsequent readout and processing circuits. This section briefly discusses the operating principles of common photodetectors.

2.2.1.1 Photodiode

In the standard CMOS process, any parasitic PN junction caused by either N-Well or P-Well can be used as a photodetector. As shown in Fig. 2.4, three photodiode structures are commonly used in a standard P-substrate CMOS process: N+/P-substrate, P+/N-well and N-well/P-substrate photodiodes.

Diffusion Substrate Junction: This junction is a widely used photodiode when compared to other junction photodiodes due to its simple layout and susceptibility to lithographic variation. Due to its wider depletion region, the spectral response of this photodiode is much better than that of the P+/N-well photodiode. However, this photodiode structure is vulnerable to the crosstalk noise as the p-substrate is shared by all pixels in the entire array.

Diffusion Well Junction: This photodiode is formed by a P-diffusion on an N-Well. The photon spectral response is lower when compared to the others, as the depletion region for this photodiode is both narrow and shallow. However, the carriers generated outside the N-Well due to the long wavelength light is shielded from the junction caused by the N-Well and P-substrate. The shielding improves the isolation between neighboring photodiodes and thus reduces the crosstalk noise.

Well-Substrate Junction: This photodiode structure is formed by an N-Well and the P-substrate. Since the depletion region is wider and deeper than the other photodiode structures, this photodiode has a better spectral response, and it allows to collect the minority carriers generated in deep substrate. The main disadvantages for this photodiode structure are the substrate noise and the crosstalk from neighboring pixels.

2.2.1.2 Photogate

The structure of a photogate photodetector is analogous to that of the MOS capacitor in CCD technology. Fig. 2.5 (a) illustrates a cross-sectional view of a typical photogate. It consists of a photogate, a floating-diffusion (FD) and a transfer gate to isolate them. When applying a biasing voltage on the gate, a depletion region is produced underneath the gate region. This region works as the accumulation region for the photon-generated carriers. After the integrating period, the accumulated carriers are transferred to the FD region by an appropriate biasing of the transfer transistor and then readout to the external processor. The main advantage of this structure is that integrating and sensing nodes are separated, allowing the correlated-double-sampling (CDS) technology to reduce fixed-pattern-noise (FPN). The photogate structure has the inherent disadvantage of lower sensitivity because of its partially transparent polysilicon gate.

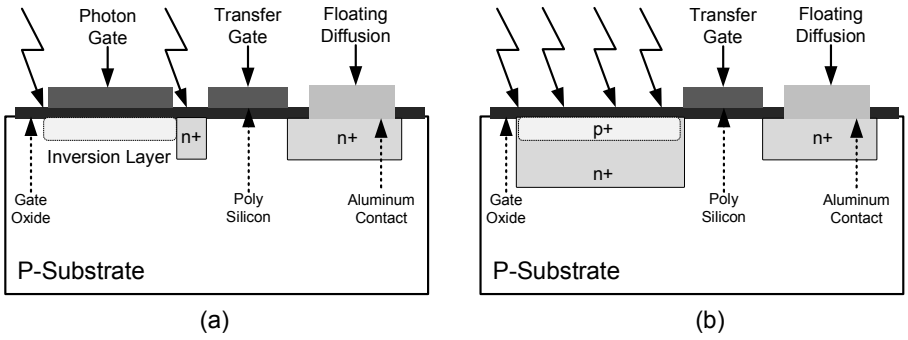


Figure 2.5: Cross-sectional view of the photogate and pinned photodiode.

2.2.1.3 Pinned Photodiode

Photons with lower energy can penetrate deeply into the substrate, while photons with shorter wavelengths are absorbed nearer to the surface. However, because of fabrication defects, the interface between the gate oxide and silicon substrate is known to trap photon-generated electrons, which introduces certain noise to the sensing signal. A pinned photodiode is thus invented to improve the quantum-efficiency (QE) for shorter wavelengths [32]. The topmost surface of the pinned photodiode is a thin P+ diffusion layer, as shown in Fig. 2.5 (b). As a result of such isolation, the accumulation region is completely separated from the surface, rendering the photodiode structure with less noise.

2.2.2 Pixel Structure

All aforementioned photodetectors are used to transform incident light into electrical signals, such as charges, currents, or voltages. However, these electrical signals require additional circuitry for readout, and their combination eventually builds a pixel. The pioneer of the CMOS image sensor originated from a passive-pixel-sensor (PPS) structure. However, the imaging quality of the PPS pixel was so poor that an active-pixel-sensor (APS) was invented to improve it. Unfortunately, APS-based image sen-

sors still suffer from serious fixed-pattern-noise (FPN). An improved APS pixel structure termed as "4T-APS" was then developed by implementing a correlated-double-sampling (CDS) technology. Other pixel structures, such as the digital-pixel-sensors (DPS) and current mode pixels, were also invented in the past two decades. Their advantages and disadvantages are analyzed in this section.

2.2.2.1 Passive-Pixel-Sensor

Historically, the passive-pixel-sensor (PPS) was the first commercially available CMOS image sensor in the market [33]. As shown in Fig. 2.6 (a), a typical PPS pixel simply consists of a photodiode and a switching transistor connected to the column readout bus. The PPS pixel has an extremely high fill factor due to its simple structure. However, because of the large column parasitic capacitance, conventional PPS image sensors suffer serious KTC thermal noise with a low signal-to-noise-ratio (SNR). As shown in Fig. 2.6 (b), a transversal-signal-line (TSL) technology was developed to alleviate the thermal noise limitation [34]. In this design, the large column parasitic capacitance found in the conventional PPS structure is replaced by a small sampling capacitance from the switching transistor. Hence, the thermal noise is reduced to a certain extent at the cost of a small switching noise induced by the additional sampling transistor.

2.2.2.2 3T Active-Pixel-Sensor

Fig. 2.7 (a) and (b) show the circuit diagram of the typical 3-transistors active-pixel-sensor (3T APS) pixel and its cross-sectional view respectively. In comparison to the PPS structure, an APS pixel has an additional reset transistor with a source follower, and it facilitates a better driving capability. The APS-based image sensor is operated with a specific sequence. Firstly, the reset transistor is activated, and the photodiode

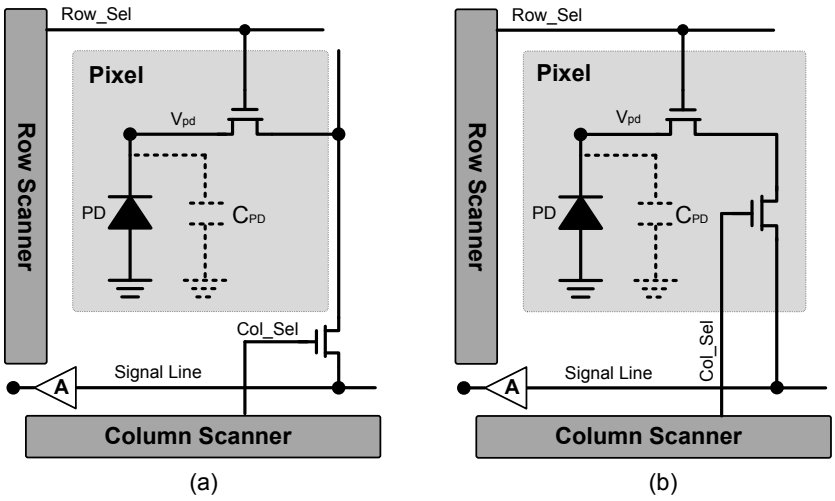


Figure 2.6: (a) System diagram of a conventional passive-pixel-sensor (PPS). (b) System diagram of a transversal-signal-line (TSL) passive-pixel-sensor (PPS).

(PD) is reset into the potential of V_{dd}-V_{th}, where V_{th} is the threshold voltage of the reset transistor. Once the reset transistor is turned off, the photodiode is electrically floated to integrate incident photons. The photon-generated charges change the voltage potential of the photodiode. After the integration phase, the voltage drop on the photodiode is linearly proportional to the incident light intensity. By activating the selection transistor, the accessed pixel exports its photodiode voltage to the column bus. When the readout procedure is completed, the selection transistor is disabled, and the reset transistor is turned on to restart a new integrating operation. Since in-pixel source follower is only activated once during the readout phase, the power consumption in this pixel structure is minimized. However, this APS structure suffers from serious fixed-pattern-noise (FPN) due to the threshold variation and mismatch on the source follower during fabrication.

2.2.2.3 4T Active-Pixel-Sensor

In order to alleviate the FPN limitation in the 3T APS structure, a 4-transistors APS (4T APS) structure is proposed in [35]. As shown in Fig. 2.8, the photon detection

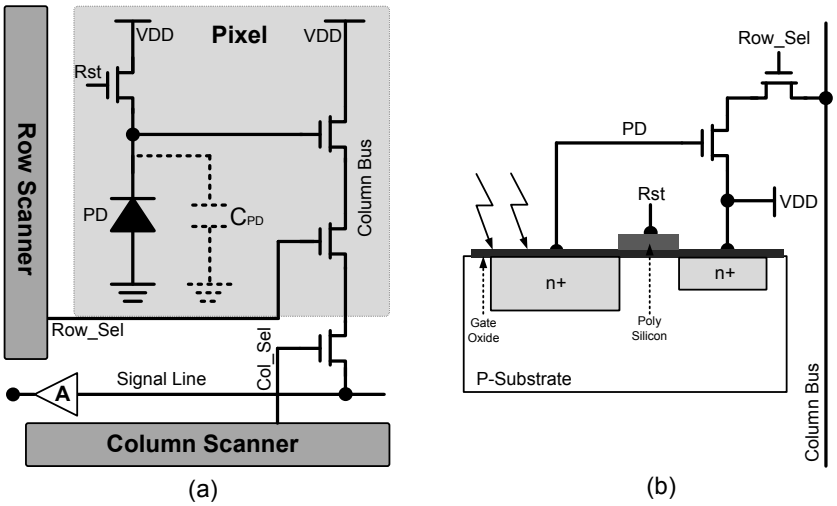


Figure 2.7: System diagram and cross-sectional view of a common 3-transistors active-pixel-sensor (3T APS).

and conversion regions are separated by a transfer transistor. Initially, the photodiode stays in a complete depletion state, and there is no accumulated charge on it. After a certain integration period, the photon-generated charges inside the photodiode are linearly proportional to the incident light intensity. Before transferring the accumulated charges, the floating-diffusion (FD) region is reset by activating the reset transistor. The reset voltage is exported to a correlated-double-sampling (CDS) circuit by the addressing transistor. After the reset voltage is readout, the accumulated charges in the photodiode are transferred to the FD region by switching on the transfer transistor, and the new voltage on the FD region is further exported to the column bus by the source follower. The CDS technology in the 4T APS structure can reduce the fixed-pattern-noise (FPN) caused by the pixel mismatch to a certain extent. Although the 4T APS structure is superior to its 3T APS counterpart in terms of lower noise, the 4T APS has a smaller fill factor due to an additional transfer transistor.

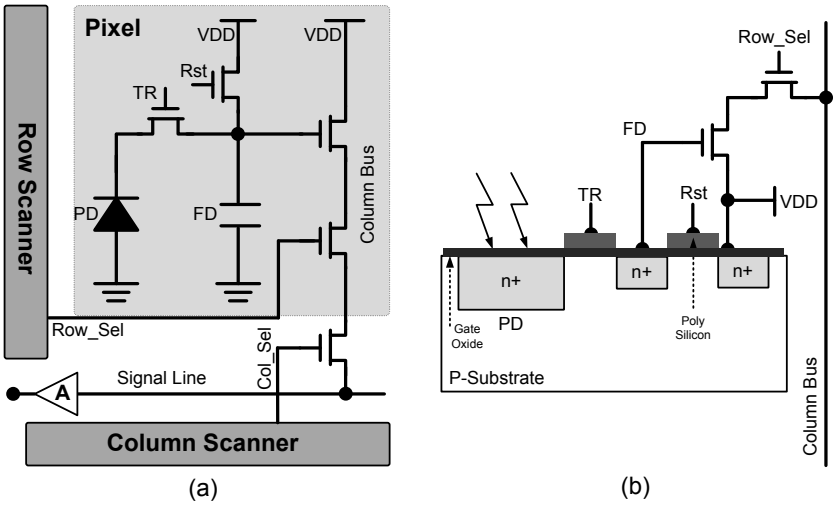


Figure 2.8: System diagram and cross-sectional view of a common 4-transistors active-pixel-sensor (4T-APS).

2.2.2.4 Current Mode Pixel

The outputs of conventional 3T and 4T APS image sensors are analog voltages that indicate the incident light intensity. From the signal processing point of view, current-based operations are more convenient due to the fact that common arithmetic operations, such as summation and multiplication, can be easily implemented by a simple current mirror. Furthermore, the photon current in a photodiode is directly proportional to the incident brightness. As shown in Fig. 2.9 (a), the photon current can be readout to the external by a current mirror [36]. However, under dark conditions, the current caused by the circuit leakage is comparable to that generated from incident photons. The structure in Fig. 2.9 (a) performs poorly in the weak light condition and presents a great fixed-pattern-noise (FPN) due to the mismatch in the current mirror. A feasible solution to the noise problem is shown in Fig. 2.9 (b) [1]. In this structure, after reset, the photodiode voltage is given as $\mathrm{Eq.\ 2.1}$, where L_{sf}, W_{sf} and V_{th} correspond to the length, width and threshold of the source follower respectively. Due to the integrating operation, the photodiode voltage drops to $V_{reset} - \Delta V$. As a consequence, the current

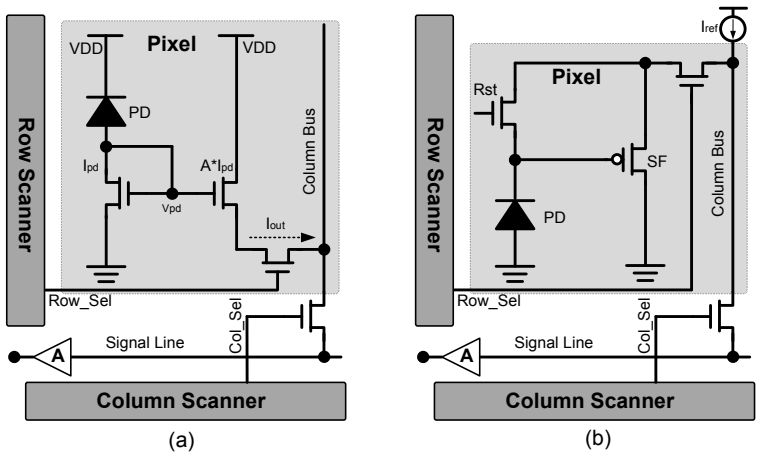


Figure 2.9: (a) Current mode pixel structure. (b) Improved current mode pixel structure with noise reduction [1].

on the source follower path can be expressed as $\mathrm{Eq.\ 2.2}$. Hence, when a given pixel is selected for readout, the eventual current on the column bus is given by $\mathrm{Eq.\ 2.3}$. This current is independent of the threshold V_{th} of the source follower M_{SF}. Hence, the FPN caused by the threshold variation is completely removed.

Besides above structures, there are also a number of digital CMOS image sensors which directly encode light intensity into a digital format inside every pixel. The most common strategies include pulse-width-modulation (PWM) [37, 38] and time-to-first-spike (TFS) [39, 40].

$$V_{reset} = \sqrt{\frac{2L_{sf}}{\mu_n C_{ox} W_{sf}} I_{ref}} + V_{th} \qquad \text{(Eq. 2.1)}$$

$$I_{pix} = \frac{1}{2}\mu_n C_{ox} \frac{W_{sf}}{L_{sf}} (V_{reset} - \Delta V - V_{th})^2 \qquad \text{(Eq. 2.2)}$$

$$I_{out} = \sqrt{2\mu_n C_{ox} \frac{W_{sf}}{L_{sf}}} \Delta V - \frac{1}{2}\mu_n C_{ox} \frac{W_{sf}}{L_{sf}} \Delta V^2 \qquad \text{(Eq. 2.3)}$$

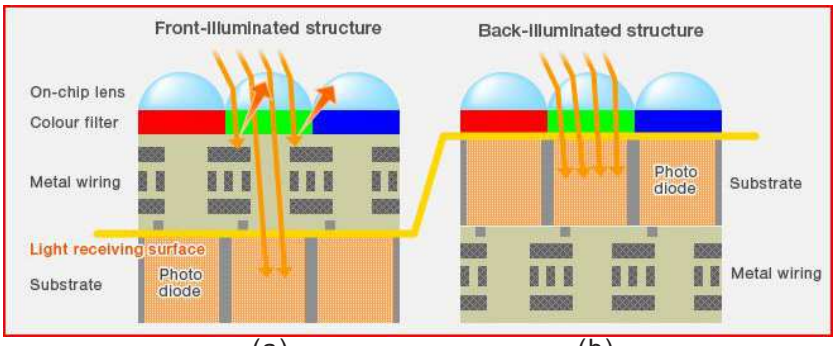


Figure 2.10: Cross-sectional view of front-illuminated and back-illuminated sensors [2].

2.2.3 Back-illuminated Sensor

In conventional image sensors, as shown in Fig. 2.10 (a), photodiodes reside in the bottom silicon substrate, and incident light is propagated to the photon sensing area through top openings. Although part of the incident light is shielded by metal wires, this structure still works effectively for light sensing when pixels are large enough. This scheme is termed as front-illuminated-sensor (FIS). However, the increasing of spatial resolution in image sensors often leads to the shrink of pixel size. More surface area is covered by metal wires, and less incident light can be absorbed by photodiodes. Hence, image sensors suffer a significant drop on light sensitivity. In order to eliminate this problem, designers invented a back-illuminated-sensor (BIS) structure by flipping the silicon wafter during manufacture [41]. As shown in Fig. 2.10 (b), incident light can directly strike photodiodes without passing through metal wires. Hence, light sensitivity and absorption in the image sensor are significantly increased. This technology has been widely used in consumer electronics.

2.2.4 Performance Characteristics

There are a number of important criteria for evaluating the performance of image sensors. These include but not limited to: quantum-efficiency (QE), fill-factor (FF), fixed-pattern-noise (FPN) and dynamic-range (DR). This section briefly describes each criterion.

2.2.4.1 Quantum Efficiency

Quantum-efficiency (QE) quantifies photodetector performance on light detection. It is defined as the fraction of incident photons on the photodetector that effectively creates photo charges. This parameter is mainly determined by the fabrication process used for the doping concentration, and it can be regulated to a minor degree by the geometric arrangement of the photodetector.

2.2.4.2 Fill Factor

Fill-factor (FF) is the ratio of light sensitive area to total pixel size, which is determined by the pixel geometry. A high fill factor is desirable as more pixel area is exploited for light sensing. In CMOS image sensors, the more processing circuit is embedded inside the pixel, the lower fill factor it can achieve.

2.2.4.3 Fixed Pattern Noise

Fixed-pattern-noise (FPN) is the technical term that describes the noise effect in the pixel variation for an image sensor under constant uniform illumination. It is a time invariant noise and mainly caused by device mismatch in the pixel and column circuitry. It can be reduced by various on-chip noise cancelling technologies, such as the correlated-double-sampling (CDS) method [42].

2.2.4.4 Dynamic Range

Dynamic-range (DR) is the ratio of the largest sensed light signal to the smallest detected noise signal. It represents a sensor's capability of detecting the brightest and darkest portions of the intra scene image. Dynamic range can be increased using larger well capacity of the photodetector, and it can also be optimized by utilizing special readout circuits in the pixel design, such as the logarithmic photoreceptor [43].

2.2.5 Comparison between CCD and CMOS

The conventional CCD technology was invented more than four decades ago. Its evolution focuses solely on improving imaging quality through noise reduction. However, CMOS technology also impacts the image sensor design. A variety of CMOS image sensors with diverse structures have been reported in the literature. Furthermore, CMOS image sensors present a major advantage in system-level integration through the implementation of photo detection and signal processing on the same chip. The comparison between CCD and CMOS image sensors and their basic characteristics are summarized in Table. 2.1.

Table 2.1: Comparison of CCD and CMOS image sensors

Parameter	**CCD**	**CMOS**
Image Uniformity	High	Moderate
Power Consumption	High	Low
Complexity	Low	Moderate
Pixel Output	Charges	Voltage/Current
Speed	Low	High
Fill Factor	High	Low

2.3 Challenges of Computer Image Processing

Advancements in the semiconductor industry have reduce the cost of setting up a computer vision system, and even a personal laptop equipped with the webcam is able to handle some basic vision processing tasks. Intensity images from digital cameras are processed by computers in a manner similar to how the human brain manages the visual information derived from eyes. Nowadays, image processing algorithms in computer vision systems have become so complex that it is unfeasible to completely implant them in hardware using the state-of-art technologies. Thus, they are mostly dominated by software implementation. With the increase in spatial and temporal resolutions of image acquisition, the processing speed becomes an bottleneck.

There is a huge amount of raw image data acquired from an image sensor. The analysis on this data flow can easily overwhelm the memory and computation power of the computer. The raw image data is partially redundant and too large to be processed. For instance, in surveillance applications, the consistent static background repetitively reported in the video sequence, which causes redundancy in subsequent operations. In this case, the raw image should be transformed into a set of features which contain relevant information on the original images. Visual tasks, such as pattern recognition and object tracking, can be performed by making use of the extracted features instead of the entire raw image. Image feature extraction is an important procedure on raw intensity images. Common image features include edge, color, texture, and shapes.These features are briefly discussed as the following.

Edge Feature : Edge feature corresponds to a set of points that identify the positions displaying sharp brightness changes in a pixel array. Ideally, by applying the edge detector to the input image, a number of curves indicating the boundaries of objects can be extracted. A variety of edge feature-extraction technology have been proposed

in the literature, including Canny [44], Sobel [45], Laplacian-of-Gaussian (LoG) [46], and Difference-of-Gaussian (DoG) [47]. The choice of edge detector highly depends on its specific application, as every method is advantageous and disadvantageous based on its particular context.

Color Feature : Color is the most direct feature for a polychrome image. A number of color spaces have been developed to model this feature, including red-green-blue (RGB), and hue-saturation-value (HSV). There are a number of color descriptors used to define color features, such as color histogram [48], color moments [49] and color coherence vector [50]. These features describe either the color distribution or color correlation in the selected region of the input image. However, the extraction of color features involves iterative computation on every pixel in the region of interest.

Texture Feature : Texture feature can only be extracted from a group of pixels to define the spatial arrangement of the intensity in the image. Texture feature can be computed in either spatial [51] or spectral domain [52]. The former directly computes pixel statistics to retrieve texture features while the latter transforms the input image into the frequency domain. The most common approach in extracting the texture feature is the Gabor filtering computation, which samples an image in the spectral domain so as to characterize its frequency and orientation as feature vectors.

Shape Feature : Shape feature is an important descriptor defining the geometrical form of the selected region in an image. It comprises a variety of shape parameters including center of gravity, spatial size, circularity ratio, and elliptic variance [53]. The shape feature-extraction algorithm can be classified into: the contour-based method and the region-based approach [54]. In the former, shape features are retrieved by only employing the boundary information of the object. In the latter, all pixels inside the region are taken into account so as to compute the shape representation.

The preceding features do not completely cover all image features used in vision processing. Other features, such as spatial corners and temporal motions, also play

important roles in Typical low level features include edges, corners and motion. High level features, such as color, texture and shapes, require computation on the spatial correlation. Hence, their extractions are much more complex than those for low level features.

The retinas in many vertebrate animals work much smarter than conventional image sensors. They not only detect visual light signals but also process them to extract basic features, such as temporal motions and spatial contours. Since these image features are directly delivered to the center brain via optic nerves, vision processing in biological visual systems is sped up accordingly [55]. Computer vision systems can work in a similar way by implementing feature-extraction functions on the image sensor side. Hence, the image sensor can directly export some specific features to release the computation load in post processors, and this kind of the image sensors is termed as the smart image sensor.

2.4 Innovations of Smart Image Sensors

Instead of transmitting entire images, smart image sensors only deliver valuable features to post processors. It is a significant revolution in image sensor design as it no longer records light intensity but directly encodes specific image features on its focal plane. Therefore, data efficiency is much higher than conventional image sensors by orders of magnitude. In order to be integrated on the focal plane, the selected feature extraction algorithm should be relatively compact in terms of computational efficiency. The aforementioned image features are classified into: low level and high level features. Most of high level features are unsuitable for smart image sensor implementation due to their complexity. Currently, only a few low level feature-extraction functions, such as edge extraction and motion detection, are integrated on smart image sensors.

In biological visual systems, the vertebrate retina acquires and processes the projected image in parallel to perform the lowest level feature extractions, such as the spatial edge extraction. As shown in Fig. 2.11, there are thousands of horizontal cells inside a retina, all of which are responsible for smoothing the acquired image from the photoreceptors through a Gaussian function. The bipolar cells in the retinal nerve compute the differences between the output of both photoreceptors and horizontal cells so as to extract spatial contrast. Smart CMOS image sensors can mimic the smoothing function in retinas on the focal plane by building a resistive network to report both raw and smoothed images simultaneously. Inspired by this biological principle, a variety of artificial retina chips applying with the edge detection function are described in the literature [56–58]. Although the proposed architecture works efficiently in contrast detection, its main drawback is the implementation of the complex resistive network, which requires precise timing control and a huge silicon area.

Spatial contrast only exists in the region where there is an abrupt change in brightness. Therefore, edges can be extracted by computing the difference between a selected pixel with its surrounding neighbors. A number of smart image sensors employing edge detection based on this principle have been proposed in the literature [59–62]. Despite experimental results show satisfactory performance in contrast extraction, the edge extraction image sensors can only report the spatial contrasts without the ability to select foreground objects from the redundant background.

2.5 Motion-Detection Image Sensors

Temporal motion feature is widely used in computer vision fields, as it not only reports the foreground kinetic object but also completely discards the static background, which significantly increases the encoding efficiency. Visual motion analysis also plays an important role in biological visual systems for many insects and animals. The most

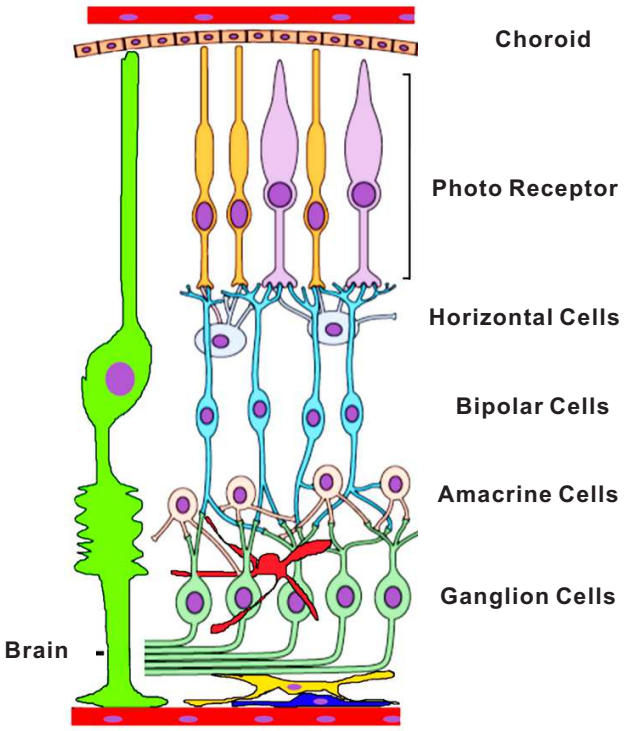


Figure 2.11: Schematic diagram of the biological visual system in vertebrate retinas.

famous example is the frog eye which has the "bug detection" capability to selectively detect small moving objects in its viewing field. Another instance is the housefly which is capable of tracking an active target and extracting the relative motion between the object and background. Visual processing in vertebrates is multifarious. In the visual system of pigeons and rabbits, motion analysis is implemented inside retinas by the ganglion cells as shown in the bottom of Fig. 2.11. Two primitive motion detection and measurement mechanisms were proposed to mimic the biological motion analysis in the literatures [55]. The first scheme is based on direct measurement of the local light intensity changes, termed as the "intensity-based". In the other technique, visual system firstly extracts some basic features, such as edges, corners and blobs, in raw images before measuring visual motions through correlation of these features over time. This type of motion analysis is defined as the "token-based" method. Both

schemes have their own advantages and disadvantages. The following sections will discuss the development of motion-detection image sensors in detail. Comparisons of different technologies utilized in motion-detection image sensors are made at the end of this section.

2.5.1 Token-Based Correlation Mode

In the token-based motion-detection algorithm, image features, such as edges, corners or even blobs, are extracted and then correlated in subsequent frames over time. The displacement of these features indicates visual motions on the focal plane. A variety of motion detection and measurement image sensors have been proposed based on this principle [63–66]. A typical motion sensor is a biologically inspired silicon retina presented in [3]. This image sensor adopts the token-based motion delay and correlation computation to detect and measure visual motions. The conceptual structure of this algorithm is drawn in Fig. 2.12. The cell R mimics the function of photoreceptors and horizontal cells in the retina. The photoreceptor transforms incident light into electrical signals while the horizontal cell smoothes these signals in space. The element E imitates the bipolar cell of the retina by processing signals from photoreceptors and horizontal cells so as to extract spatial edges. The edge detector can extract both "*ON*" and "*OFF*" signals and convert them into digital pulses in parallel. The element M correlates edge pules from the element E and the delayed cell from surrounding pixels via the element D. The corresponding pixel fires a motion pulse at its output only when the travelling time of edge pules from two neighboring pixels matches with the delay temporal coefficient τ. Therefore, the combination of elements M and D can mimic the function of ganglion cells in the retina to sense visual motions. This image sensor is termed as the silicon retina, since it completely mimics the motion sensing function of the biological retina.

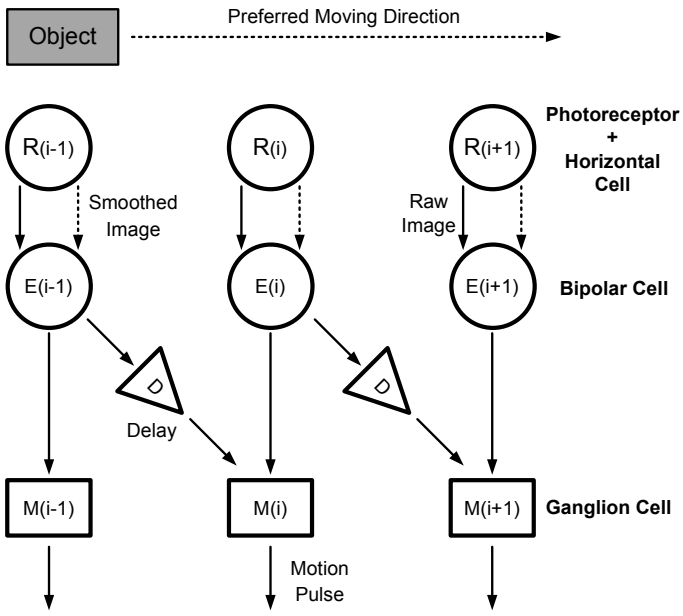


Figure 2.12: Principle of token-based delay and correlation visual motion computation algorithm. [3]

The elements shown in Fig. 2.12 are developed inside every pixel in hardware. The circuit diagram of the edge extractor and delay element are drawn in Fig. 2.13 and Fig. 2.14. Two bipolar-junction-transistor (BJT) based photoreceptors termed as smoothed BJT and isolated BJT are founded inside every pixel. In order to implement the diffusion function of the horizontal cells in retinas, the base region of the smoothed BJT is connected to those in the surrounding neighbors via a NMOS transistor network. For clarity, the smoothing network is not shown here, and a description of the detailed circuitry can be found in the original paper. The isolated BJT directly transforms light intensity into photo current I_{iso}. The difference ΔI between the smoothed image current I_{smt} and the isolated light current I_{iso} detects the temporal zero crossing point of the brightness into the edge signal, which is then further transformed into an edge pulse by a pulse conversion circuitry shown in Fig. 2.14. The edge pulse width is regulated by the biasing voltage V_{pulse} on the discharge path. This extractor can detect both positive and negative light intensity changes to into spatial edges.

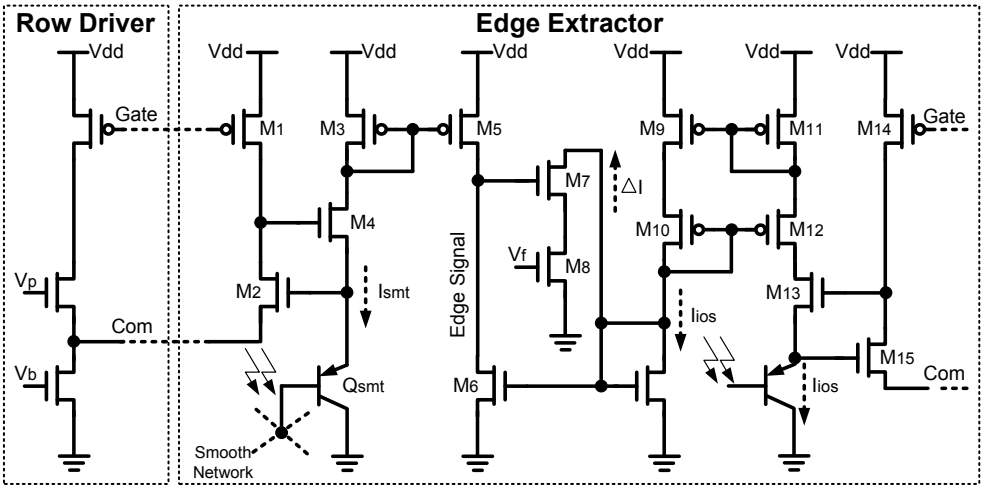


Figure 2.13: Circuit diagram of the edge extractor in a bipolar-junction-transistor (BJT) based silicon retina. [3]

The delay element D generates the delayed edge pulse in both horizontal and vertical directions, as shown in Fig. 2.14. Top two signal paths are dedicated to the narrow delayed edge pulse in the horizontal direction, while the bottom two paths create the edge pulse in the vertical direction. The exact delay time is controlled by regulating the biasing voltages V_{dx} and V_{dy}. In order to reduce the pixel size, all capacitors in the delay element are implemented using the MOS transistor with its source and drain tied together as one plate, and the gate is treated as the other plate. There is a specific circuit in every pixel for correlating the edge pulse from a given pixel with delayed pulses from its four neighbors. The circuit diagram on the correlation block is not presented here due to limited space. Eventually, only the selected motion with the preferred direction and speed can trigger digital pulses from the correlator.

The token-based motion computation image sensor has been implemented with a 32 $\times$ 32 pixel array. Measurement results show satisfactory performance for motion detection. However, due to the complexity of the circuit design, the pixel is extremely bulky with a silicon size of 100 μm $\times$ 100 μm. This limitation restricts the use of token-

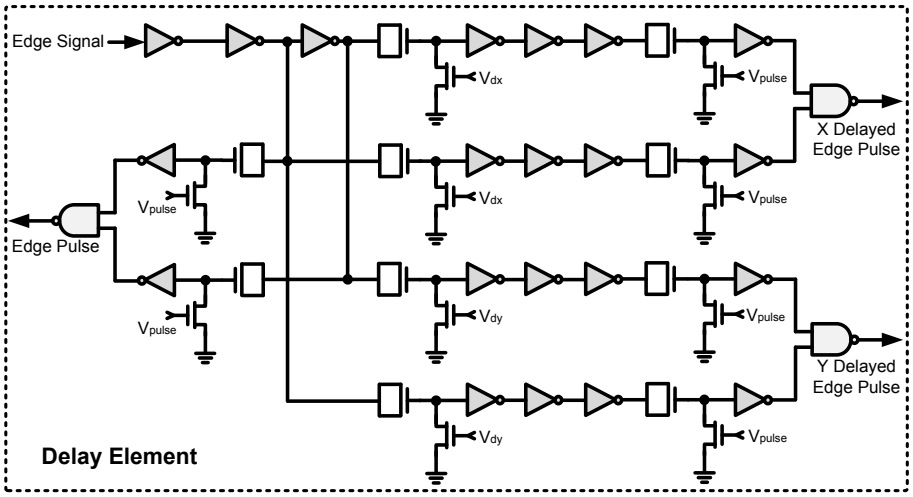


Figure 2.14: Schematic diagram of the delay element in BJT based silicon retina to generate the edge pulses. [3]

based motion-detection technology in high resolution applications. In addition to this, the proposed image sensor can only detect a limited range of the motion speed.

A CMOS motion detector based on the detection and tracking of spot edges for pointing devices is proposed in [4]. Motion in horizontal and vertical directions is estimated by aggregating local information of moving edges on the pixel array. The optical motion-detection system is shown in Fig. 2.15. It consists of a ball with a printed pattern of spots, a LED light source, a focus lens and a CMOS image sensor integrated with photodetectors and processing circuits. A microprocessor works a buffer between the image sensor and a personal computer.

In this system, the LED light source periodically illuminate part of the ball. The displacement of the focused image between two light pulses is estimated by the ratio of edges moving in one direction over the total number of edges in that direction, which is modeled as $\mathrm{Eq.\ 2.4}$ and $\mathrm{Eq.\ 2.5}$.

$$\frac{\Delta x}{P} = \frac{\sum R - \sum L}{\sum E_x} \qquad \text{(Eq. 2.4)}$$

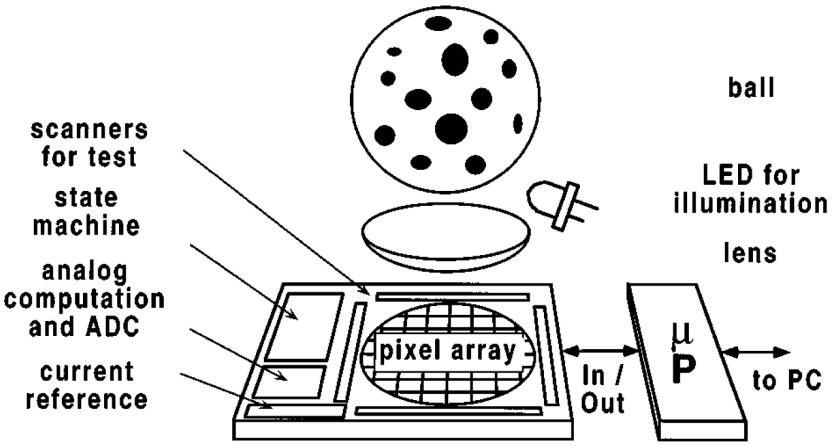


Figure 2.15: Pointing device microsystem proposed in [4].

$$\frac{\Delta y}{P} = \frac{\sum U - \sum D}{\sum E_y} \qquad \text{(Eq. 2.5)}$$

where $\sum R$ and $\sum L$ represent the number of vertical edges moving to right and left, $\sum U$ and $\sum D$ correspond to the number of horizontal edges moving to up and down, and $\sum E_x$ and $\sum E_y$ indicate the total number of vertical and horizontal edges.

The block diagram of the proposed pixel is shown in Fig. 2.16. Every pixel is composed of three modules: an image acquisition circuit, an edge detection block and a logic computation unit. The image acquisition circuit is shown in Fig. 2.17, and it is composed of a photodiode and a current amplifier. Instead of periodically switching transistor T_3 to restore the capacitor C for a new frame integration, transistor T_3 is biased with a constant voltage V_bn, and its saturation current I_sat3 is set around 100 pA. Hence, the exposure time of the pixel array is controlled by the LED light pulse. The output current I_ph from the image acquisition block is modeled as $\mathrm{Eq.\ 2.6}$, and it is mirrored to the edge detection block through transistors T_6 - T_10.

$$I_{ph} = \frac{\beta_5}{2n}\left(V_0 + \frac{1}{C}\left(I_{PD} - I_{sat3}\right)t - V_T\right)^2 - I_b \qquad \text{(Eq. 2.6)}$$

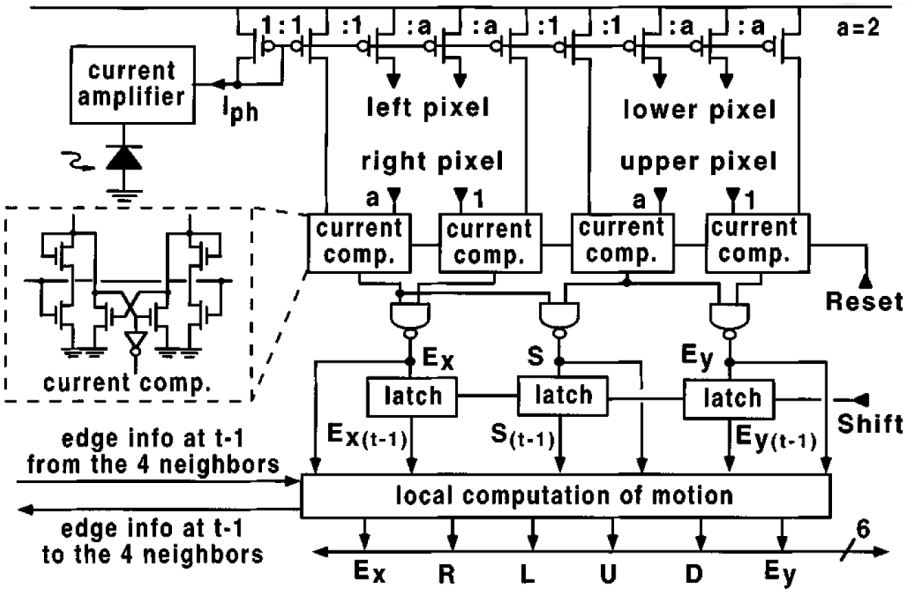


Figure 2.16: Block diagram of the smart pixel proposed in [4].

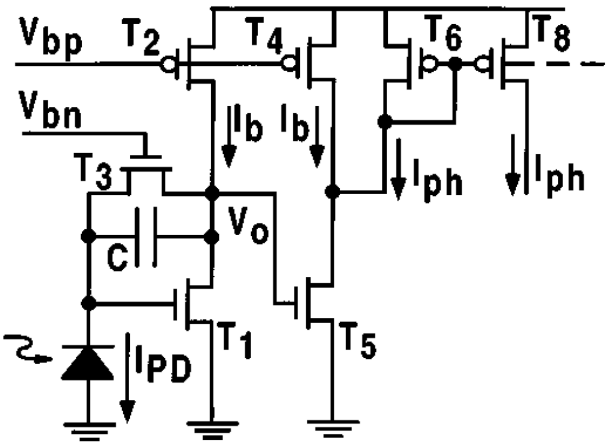


Figure 2.17: Image acquisition circuit of the smart pixel proposed in [4].

where V_T is transistor threshold voltage, V_0 is a constant voltage in the biasing path, t is the exposure period controlled by the LED light pulse, $\beta_5 = \mu \times C_{ox} \times \frac{W_5}{L_5}$, and n is the slope factor of the transistor T_5.

Photon currents I_ph from the a given pixel and from its right and upper neighbors are fed into the latched current comparators controlled by a global signal “Reset”. The

reset signal is precisely synchronized with the LED light pulse. Vertical edges E_x and horizontal edges E_y are detected from the NAND gate output. The gradient of edges S is also determined to differentiate between negative and positive edges at two consecutive light pulses. E_x, E_y and S are stored into respective latches after illumination is turned off.

The edge information (E_x, E_y) and the sign S at time t in a pixel and the information in its four neighbors at time $t-1$ are correlated by the local motion computation block. There are six global analog buses connected to the computation block in every pixel. Each wire carries the sum of edges moving in four directions ($\sum R$, $\sum L$, $\sum U$,$\sum D$) and the total number of edges in vertical and horizontal directions $\sum E_x$ and $\sum E_y$. They are further processed by an off-array analog computation ADC to extract $\frac{\Delta x}{P}$ and $\frac{\Delta y}{P}$ based on $\mathrm{Eq.\ 2.4}$ and $\mathrm{Eq.\ 2.5}$. Experimental results show that the motion detector can extract ball movements in the range of 0 - 11.8 in/s with a resolution over 800 dpi, which is an attractive solution for pointing devices.

2.5.2 Intensity-Based Discrete Mode

In general, an image may contain multiple kinetic objects travelling together. Ambiguity arises in their correlation and correspondence when their features are extracted over frames. Currently, the exact correspondence process in the human vision system is not completely understood. The intensity-based motion-extraction technology provides a reliable solution for sensing visual motion. This method is developed based on the relationship between the intensity gradient and changes at a given pixel in time domain, as modelled in $\mathrm{Eq.\ 2.7}$.

$$\frac{dI(x,y,t)}{dt} = I_x\mu + I_y\nu \qquad \text{(Eq. 2.7)}$$

where $I(x, y, t)$ represents the light intensity in a given pixel at a certain moment; $\frac{dI(x,y,t)}{dt}$ the temporal intensity change on that pixel; I_x and I_y are the image gradients in X and Y directions at the same pixel; μ and ν are the motion velocity in horizontal and vertical directions, and they are termed as optical flow features. It should be noted that visual motion can be detected by monitoring the variation of light intensity with time.

In conventional CCD and CMOS image sensors, light intensity is presented by the gray-scale value inside every frame. Hence, visual motion can be extracted by monitoring the changes in pixel value over frames. This approach, termed as the temporal difference algorithm, computes the distinction between two temporal consecutive frames in order to detect visual motion on the focal plane. Detailed analyses on the temporal difference algorithm is presented in the next section. The first hardware implemented temporal difference algorithm for an image sensor was developed through a PPS structure [5].

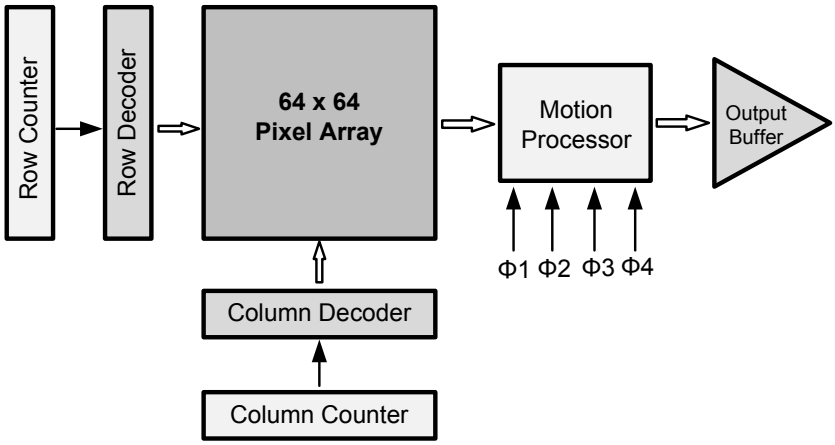


Figure 2.18: System architecture of the motion-detection image sensor based on passive-pixel-sensors (PPS). [5]

The system block diagram of the motion-detection image sensor based on passive-pixel-sensor (PPS) is shown in Fig. 2.18. Main building blocks include a 64 $\times$ 64 pixel

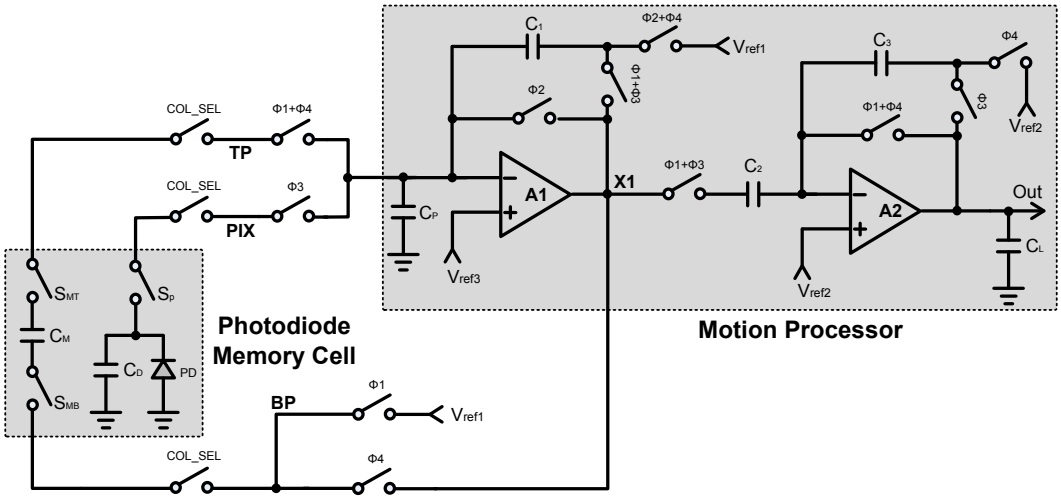


Figure 2.19: Schematic diagram of the data path from a passive pixel to the global motion processor. [5]

array, row and column decoders, a motion processor and an output buffer. Each pixel contains a photodiode formed by an n+ diffusion on a p-substrate and a capacitor (C_M) consisting of a polysilicon layer over an implanted active region. Once a pixel is selected by the row and column decoders, the current frame value from the photodiode (PD) and the previous frame value from the capacitor (C_M) are readout to the global motion processor. Their difference is computed by this processor and then reported to the external via the global buffer.

Data path from the selected pixel to the motion processor is shown in Fig. 2.19. This motion processor comprises of two stages: a charge amplifier A_1 and a subtractor A_2. In this figure, C_D represents the capacitance of the photodiode and C_P is the parasitic capacitor affecting the virtual ground. The motion processor operates in four non-overlapping clock phases Φ1 to Φ4. At the beginning of a scanning cycle, the feedback capacitors, C_1 and C_3 are charged to reset voltages V_{ref3}-V_{ref1} and V_{ref2}-V_{ref2} as a consequence of the operation during the phase Φ4 in the last scanning period. In the phase Φ1, the signal charge of C_M is injected to the charge amplifier

A_1. The output of the amplifier A_1 (X_1) can be expressed as $\mathrm{Eq.}\ 2.8$, where $V_{CD}(i-1)$ represents the previous frame voltage stored on the memory. When the second amplifier is configured as an unity gain buffer, the output of amplifier A_1 is loaded onto capacitor C_2. Hence the voltage on this capacitor ($V_{C2}(i, \Phi_1)$) can be given as $\mathrm{Eq.}\ 2.9$, while C_3 maintains its reset voltage.

$$V_{X1}(i, \Phi_1) = V_{ref1} + \frac{C_D}{C_1}(V_{ref3} - V_{CD}(i-1)) \qquad \text{(Eq. 2.8)}$$

$$V_{C2}(i, \Phi_1) = V_{X1}(i, \Phi_1) - V_{ref2} \qquad \text{(Eq. 2.9)}$$

In the phase Φ2, the charge amplifier A1 and capacitor C_1 are reset to V_{ref3}-V_{ref1} again. During the phase Φ3, the photodiode signal charge is readout to the charge amplifier A1, the output voltage on the operational amplifier A1 ($X_1(i, \Phi_3)$) is shown in $\mathrm{Eq.}\ 2.10$. At the same time, the subtractor computes the difference between the sampled voltages in phases Φ3 and Φ1 so as to generate an output voltage ($V_{Out}(i, \Phi_3)$), as shown in $\mathrm{Eq.}\ 2.11$.

$$V_{X1}(i, \Phi_3) = V_{ref1} + \frac{C_D}{C_1}(V_{ref3} - V_{CD}(i)) \qquad \text{(Eq. 2.10)}$$

$$V_{Out}(i, \Phi_3) = V_{ref2} + \frac{C_D}{C_1}\frac{C_2}{C_3}(V_{CD}(i) - V_{CD}(i-1)) \qquad \text{(Eq. 2.11)}$$

In the phase Φ3, the photodiode is automatically reset to V_{ref3} so as to start a new integration cycle. It can be seen from $\mathrm{Eq.}\ 2.11$ that the subtractor directly exports the difference between two consecutive frames. During the phase Φ4, the signal charge

on C_1 is reloaded into the memory C_M. Meanwhile, both C_1 and C_3 are reset to the initial voltage to restart computation for the next cycle.

The output of this image sensor is the analog voltage indicating the intensity difference between two consecutive frames. Since each pixel only contains a photodiode and a capacitor as an analog memory, the fill factor of the proposed pixel in the image sensor is as high as 50%. Meanwhile, power consumption on both the readout and processing paths is minimized, as electrical charges are directly exported to the global motion processer. However, due to the lack of isolation, the parasitic large capacitance from the column bus significantly degrades photon charges. Furthermore, the leakage current on the memory device also influences the analog computation by reducing the accuracy of the motion detection.

Motion-detection accuracy can be improved by using the active-pixel-sensor (APS) to implement the temporal difference algorithm. A pioneering APS-based motion-detection image sensor is proposed in [6]. By implementing an analog memory inside every pixel, the sensor directly exports the temporal difference between two consecutive image frames.

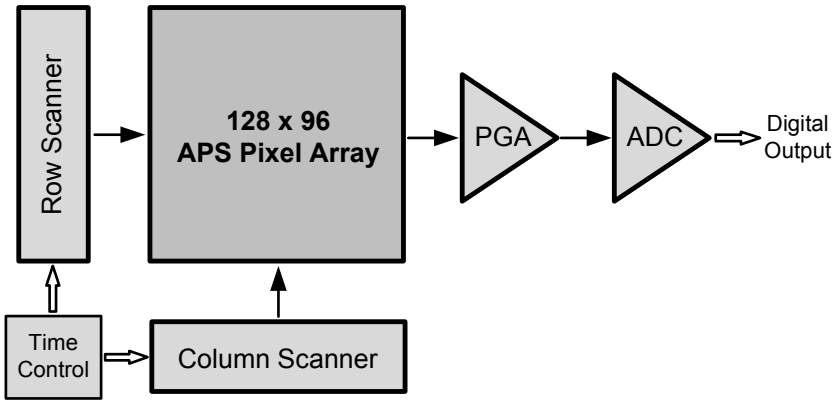


Figure 2.20: System architecture of the motion-detection image sensor based on active-pixel-sensors (APS). [6]

The system architecture of the motion-detection image sensor is shown in Fig .2.20. Main building blocks include a 128 $\times$ 96 pixel array, row and column scanners, a global programmable-gain-amplifier (PGA) and an analog-to-digital-convertor (ADC). The pixel circuit diagram is shown in Fig. 2.21. Two sample and hold blocks, comprising of analog switches (M2 and M3) and memory devices (MOS capacitors of M8 and M9), are presented in every pixel. When a given pixel is addressed by the row scanner, it sequentially exports the current frame intensity voltage from the photodiode and the previous frame voltage from the capacitor to the global amplifier. The difference between these two voltages is computed by the programmable amplifier and then digitalized into the digital format by the on-chip analog-to-digital-converter (ADC).

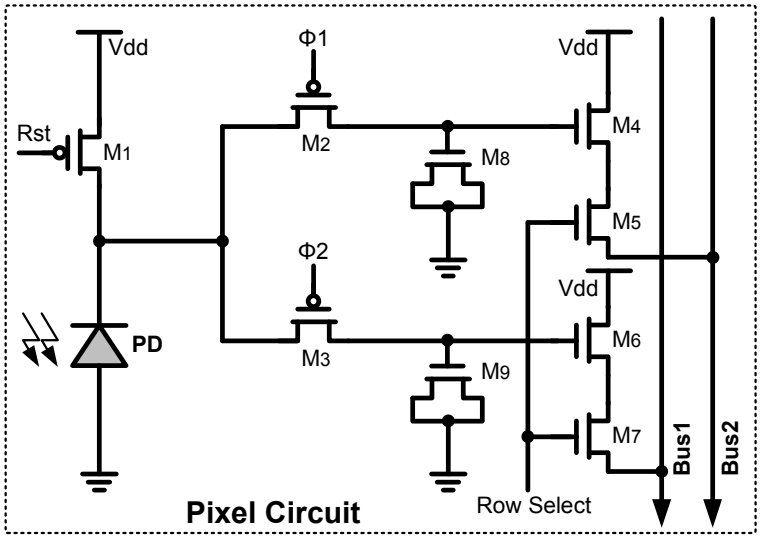


Figure 2.21: Schematic diagram of the smart pixel in the APS-based motion-detection image sensor. [6]

There are two operating modes for this image sensor: intra frame and frame difference modes. In the first mode, only one sample and hold path is activated during the readout in every frame cycle. The other one is only accessed in the reset phase and then disabled during the exposure phase. The integrated voltage is sampled on

one meomory while the other one stores the reset voltage. Their difference generates an analog intensity image with the suppressed fixed-pattern-noise (FPN). In the frame difference mode, two sampling switches turns on alternatively in every frame period. Hence, the integrated intensity voltage in the current frame does not influence the previous one stored on the capacitor. A programmable amplifier computes their difference for implementing the temporal difference algorithm. Fig. 2.22 shows two sample images captured by the proposed motion-detection sensor in the intra frame and frame difference modes respectively. Test results show that this image sensor functions well in extracting visual motions on the focal plane.

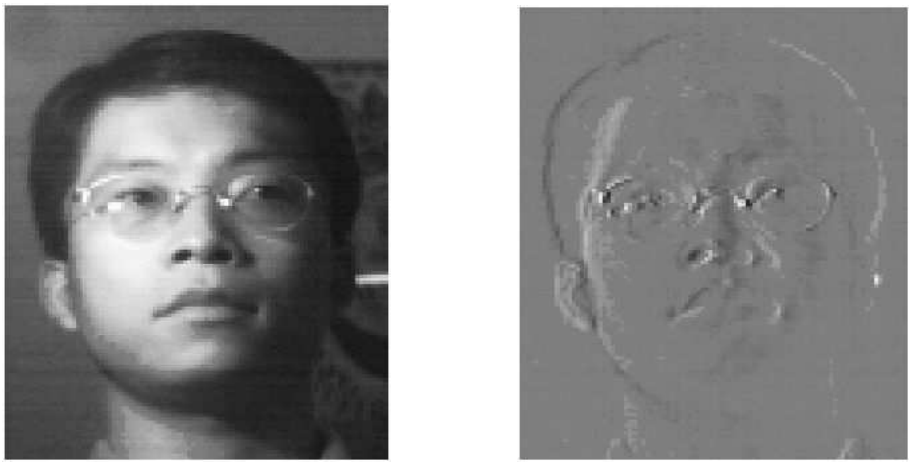


Figure 2.22: (a) Sample image captured in the intra frame mode. (b) Sample image captured in the frame difference mode. [6]

Since each pixel is implemented with an NMOS transistor as the source follower in the readout path, the driving capability to the column bus is significantly improved when compared to the PPS structure. Moreover, the analog memory is implemented by an NMOS transistor, which dramatically reduces the pixel size. When the sampling switch is turned on, photon charges on the photodiode are shared with the analog

memory due to the lack of isolation. The charge sharing limitation reduces the computation accuracy of the temporal difference algorithm. In addition to this, the vertical accessing strategy results in a slight row wise mismatch on the pixel readout. The computation accuracy can be improved using the pipeline readout technology proposed in [67, 68]. This technology minimizes the leakage current on the storage element with a constant latency for every pixel pair.

There is an inherent parasitic capacitor on the PN junction of the photodiode. This capacitor can also be implemented as an analog memory to store the intensity voltage from previous frames. Inspired by this principle, a motion-detection image sensor with a compact pixel structure is proposed in [7]. As shown in Fig. 2.23, each pixel cell comprises of a photodiode and an NMOS transistor which reports the photodiode voltage for charge amplification, when scanned by the vertical shift register.

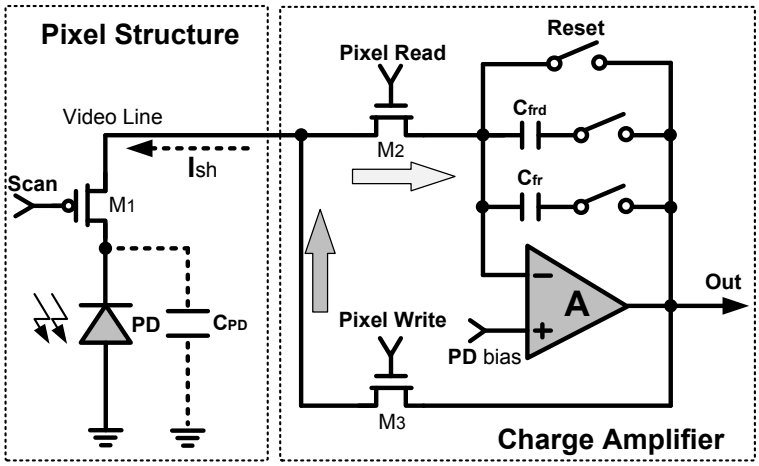


Figure 2.23: Schematic diagram of the motion-detection pixel with a column level charge amplifier. [7]

The implementation of the motion-detection algorithm is explained by the timing diagram shown in Fig. 2.24. After the charge amplifier is initialized by the reset pulse, the video line is balanced to the“pd bias” voltage V_{pdbias}. Once a given pixel is accessed by the vertical scanner, the accumulated photo charge inside the pixel during the

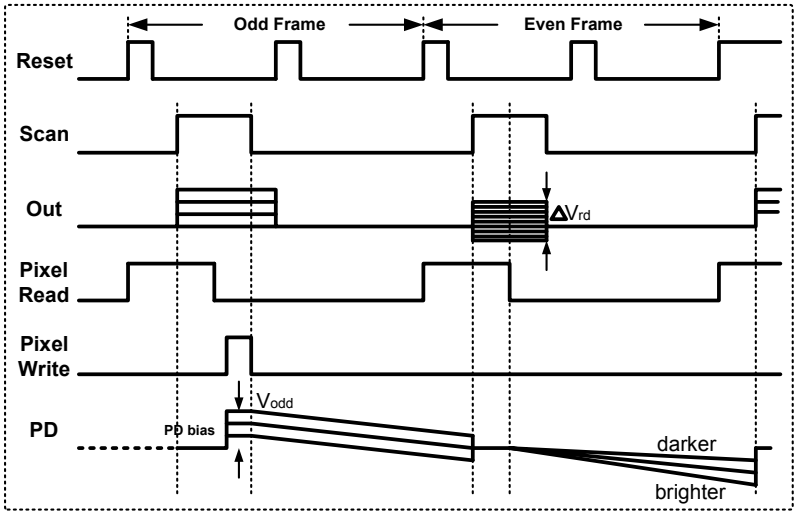


Figure 2.24: Timing diagram of the motion-detection operation. [7]

odd frame is completely transferred to the feedback capacitor C_f and the potential on the photodiode is clamped to the voltage of V_{pdbias}. Hence, the output of the charge amplifier (V_{Out}) can be given as $\mathrm{Eq.\ 2.12}$, where I_{odd} is the photo current in the odd frame and T_{int} is the integration period for the current frame.

$$V_{out} = V_{pdbias} + (I_{odd} \times T_{\mathrm{int}})/C_f \qquad (\mathrm{Eq.\ 2.12})$$

After the pixel readout phase, the "pixel read" switch M_2 is disabled, and the "pixel write" switch M_3 is activated. The photodiode is then reset into V_{out}, allowing the photo charge accumulated in the odd frame to be added to the photodiode. After that, photodiode starts integrating a new frame, leading to a voltage drop which is linearly proportional to the light intensity. This procedure is a subtraction of the odd frame to the even frame in the form of electrical charges. At the end of the integration, the voltage on the photodiode can be expressed as $\mathrm{Eq.\ 2.13}$, where I_{even} is the photo current in the even frame and C_{pd} is the capacitance of the photodiode.

$$V_{pd} = V_{pdbias} + (I_{odd} \times T_{\text{int}})/C_f - (I_{even} \times T_{\text{int}})/C_{pd} \quad \text{(Eq. 2.13)}$$

Since capacitance C_{pd} is designed to be close with that of the feedback capacitor C_f, the voltage variation on the photodiode V_{pd} can be further approximated as $\text{Eq. } 2.14$, which has a linear relationship to the photo current difference between two consecutive frames.

$$V_{pd} = V_{pdbias} + (I_{odd} \times T_{\text{int}} - I_{even} \times T_{\text{int}})/C_f \quad \text{(Eq. 2.14)}$$

After the even frame integration, the charge amplifier is reset again, and the "pixel read" switch is then activated to allow buffering the voltage variation on the photodiode to the feedback capacitor C_{fd}. The eventual output voltage ΔV can be given as $\text{Eq. } 2.15$. Higher computation sensitivity can be achieved by substituting C_f with a smaller capacitor C_fd in the second sampling phase.

$$\Delta V = (I_{odd} - I_{even}) \times T_{\text{int}}/C_f \quad \text{(Eq. 2.15)}$$

The voltage output from the charge amplifier is converted to the digital format by the analog-to-digital-convertor (ADC) array. Fig. 2.25 shows the sample images captured by the proposed image sensor on a scene involving a flying bird. The moving target is visibly extracted from the static background (b). Compared to other motion-detection image sensors, this sensor features the most compact pixel structure with a maximum fill factor of 70%. However, there are many limitations to this vision sensor. For example, in order to achieve the subtraction of two consecutive images, the desired capacitance of the photodiode C_pd should be equal to that of the feedback

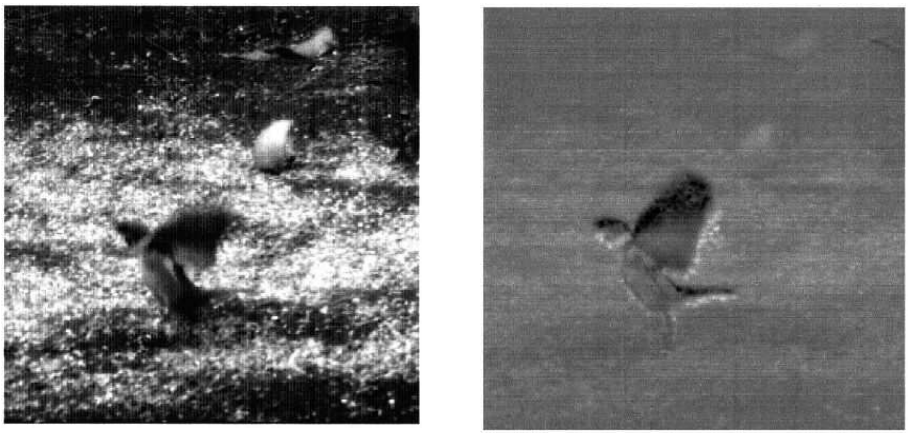


Figure 2.25: (a) Sample intensity image captured at 256 frames/s. (b) Visual motion extracted from the same scenario. [7]

capacitor C_f. Nevertheless, the capacitance of C_pd is highly dependent on its voltage, which varies over the integration period and hence strongly affecting the computation accuracy. Furthermore, the charge processing circuit is sensitive to noise, such as thermal noise and switching noise.

A common limitation of the temporal difference algorithm is the memory consumption, as it is compulsory to store previous frame information. If this memory device is integrated inside each pixel, the fill-factor (FF) will be reduced. In order to achieve a high fill factor, the floating-diffusion (FD) in the 4-transistors active-pixel-sensor (4T APS) is proposed as an analog memory to store previous frame information [8, 69]. By adopting a dynamic resolution strategy, the proposed image sensor can report a high resolution image on the stationery background and a low resolution image on the selected region of motion so that the common motion blurring issue in conventional image sensors is completely alleviated.

Fig. 2.26 shows the block diagram of the proposed multi resolution image sensor. Under normal operations, the sensor provides a 256 $\times$ 256 10-bit intensity image at a speed of 30 frames/s. In the presence of visual motion on the focal plane, an

on-chip motion comparator can generate a 128 $\times$ 128 binary motion image. Based on the motion information, a region-of-interest (ROI) containing the kinetic object is extracted by an off-chip complex-programmable-logic-device (CPLD). The region of interest can be accessed by a column selector and two row address decoders, which can access the object region and the static background separately. In the selected region of motion, 2 $\times$ 2 pixels are spatially merged by the charge accumulation on the floating-diffusion (FD) region. This special readout strategy allows a high signal-to-noise-ratio (SNR) even within a short integration period so as to alleviate the common visual motion blur problem. The proposed image sensor has two output channels from the correlated-double-sampling (CDS) and analog-to-digital-convertor (ADC) blocks. One channel exports the normal intensity image scanned by the shift registers, while the other reports an intensity image on the region of interest accessed by address decoders. A multi-resolution image containing all kinetic objects without any motion blur is then reconstructed through off-chip processing.

The pixel in this smart image sensor is developed based on the common 4-transistor active-pixel-sensor (4T APS). The pixel structure is shown in Fig. 2.27. Four neighboring pixels are integrated together as a super pixel through a common readout sharing strategy [70]. Every pixel is independently accessed by the row and column decoders and registers. For example, the photon charges accumulated on one photodiode can be transferred into the common floating-diffusion (FD) region when both row and column transfer transistors are activated by TX(R) and TX(C) from global decoders. Similarly, the floating diffusion (FD) region is reset into the balance voltage when both RST(R) and RST(C) signals are enabled. An inter pixel switch (IS), controlled by the selection switch (BS), connects the floating diffusion regions in the upper and lower pixels and is only activated in the motion-detection mode.

The system diagram on the motion detector is shown in Fig. 2.28 (a). The motion-detection process starts from the signal readout on the lower pixel when the signal

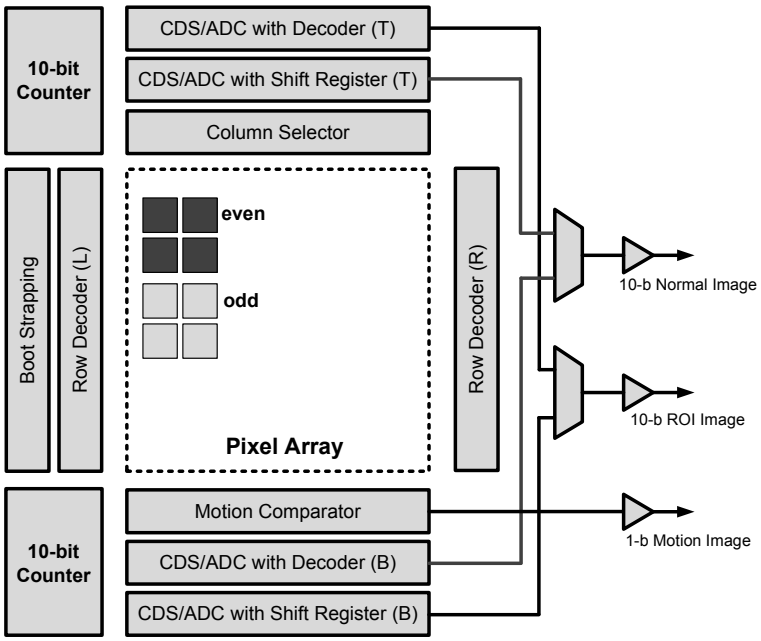


Figure 2.26: System architecture of the multi-resolution motion-detection image sensor. [8]

on the upper pixel has already been exported one row cycle ago. After readout on the lower pixel, half of the photon charge in the floating diffusion (FD) on the lower row is transferred to the upper row when the inter pixel switch (IS) is activated. After half of the integration time $T_{\mathrm{int}}/2$, both the previous frame charges stored in the upper FD and the newly integrated current frame charges in the lower pixel are successively sampled and compared in the motion comparator. As a result, a 1-bit motion signal with a spatial resolution of 128 $\times$ 128 is generated from the motion comparator. After another $T_{\mathrm{int}}/2$, the newly integrated charge is added to the remaining charge in the lower FD and the combined signal is read out to completely build the intensity image. Fig. 2.28 (b) shows a timing diagram on this motion-detection procedure.

Fig. 2.29 shows the sample images acquired by the multi resolution motion-detection image sensor. Sample (a) shows a normal image acquired at a speed of 15 frames/s with a rotating character “M” in the center of the scenario. It shows a serious motion

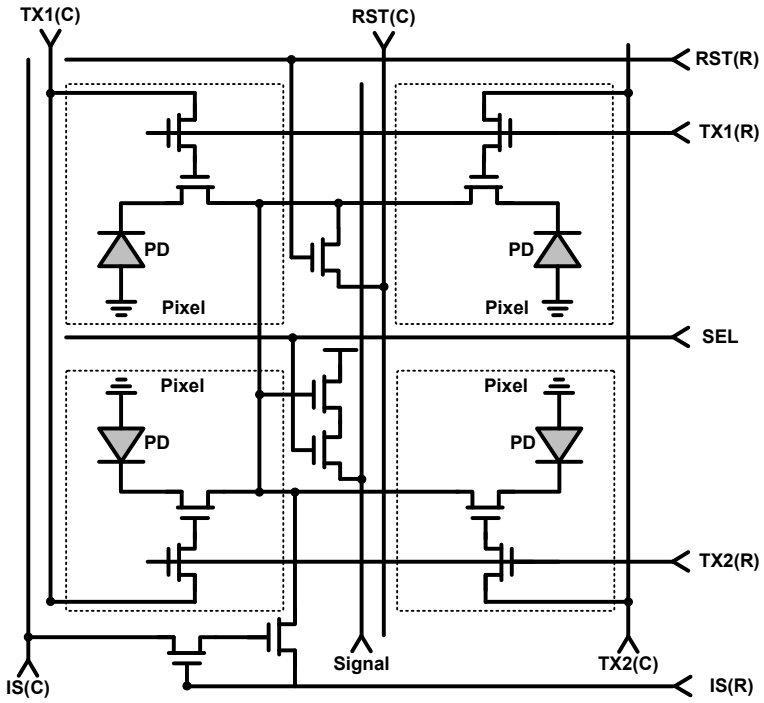


Figure 2.27: Schematic diagram of the multi-resolution super pixel with a combination of 2 $\times$ 2 sub-pixels. [8]

blur effect in the region of motion. In sample (b), the character “M” is acquired in a low resolution with a high frame rate, and the stationery background is captured in high resolution with a low frame rate. Here, the motion blur caused by the rotating object “M” is effectively suppressed. The function is achieved at the cost of a complex readout strategy, despite the compactness of the motion-detection circuit. Furthermore, the off-chip timing control and the post image processing are compulsory in order to operate this image sensor.

The temporal difference algorithm requires a long integration period in order to acquire every image frame. Due to the low temporal resolution, this scheme is unable to detect fast moving objects. The temporal resolution should be designed to be as high as possible so as to detect fast motion, and hence it requires an extreme short exposure time. An adaptive integration image sensor that can detect fast moti-

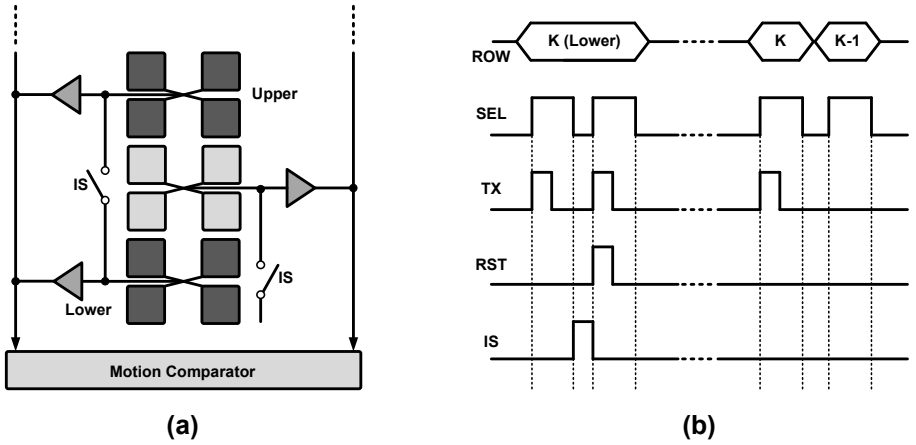


Figure 2.28: (a) System level schematic of the motion-detection circuit. (b) Timing diagram of the motion-detection operation. [8]

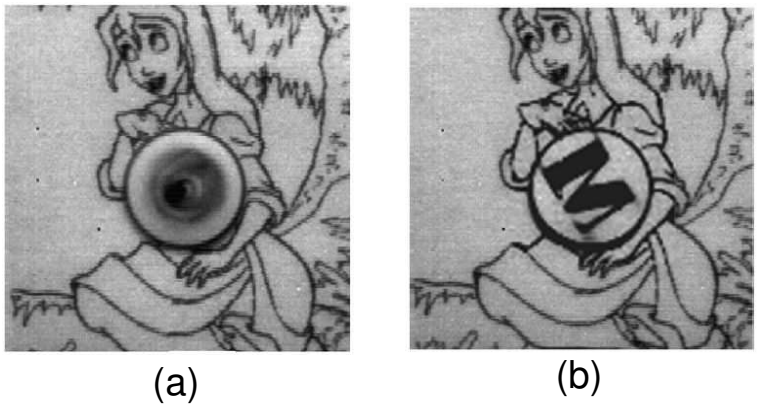


Figure 2.29: (a) Sample intensity image without motion blur suppression. (b) Sample intensity image when motion blur suppression is applied. [8]

ons is proposed in [9]. The sensor only activates motion flag signals for those pixels detecting effective motions and selectively reports their intensity information to the external. Hence, the sensor is quite appealing for image compression and motion blur suppression applications.

The motion-detection scheme in this image sensor is shown in Fig. 2.30. After a minimum integration interval Δt, the voltage difference ΔV between the newly integrated voltage V_{PD} on the photodiode and the prior voltage value V_{CD} on the capacitor

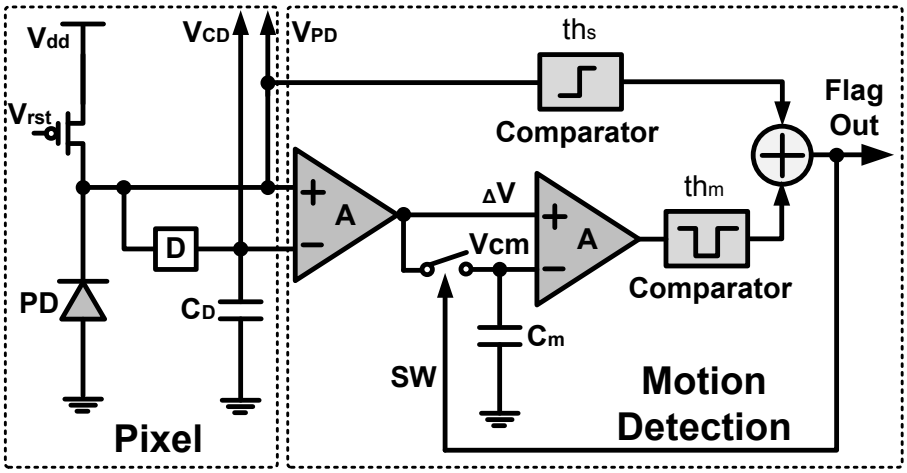


Figure 2.30: Block diagram of the motion and saturation detection circuits in the computational image sensor. [9]

C_D is extracted by the first comparator. C_m is a memory that stores the ΔV in the last Δt integration cycle. If the difference between the current ΔV and previous ΔV_{Cm} exceeds a defined threshold th_m, the relevant pixel is determined as a significant intensity change to trigger a motion. In this case, the sensor activates a motion flag signal for this pixel as well as outputs V_{PD} and V_{CD} to the external processor. Meanwhile, memory V_{Cm} will be replenished by the new ΔV, and the photodiode will be reset to start a new integration. If no motion is detected in this pixel, the photodiode continues its integration operation without resetting. The motion processing block is shared by the pixels in the same column. Since the motion detection is implemented in a row parallel scheme, this vision sensor works much faster than the aforementioned motion-detection sensors. Other intensity-based integration mode motion-detection image sensors have been proposed in the literature as well [71–73]. However, the fundamental principle of the motion-detection algorithm is the same temporal difference computation, and its detailed implementation has been described in the literature.

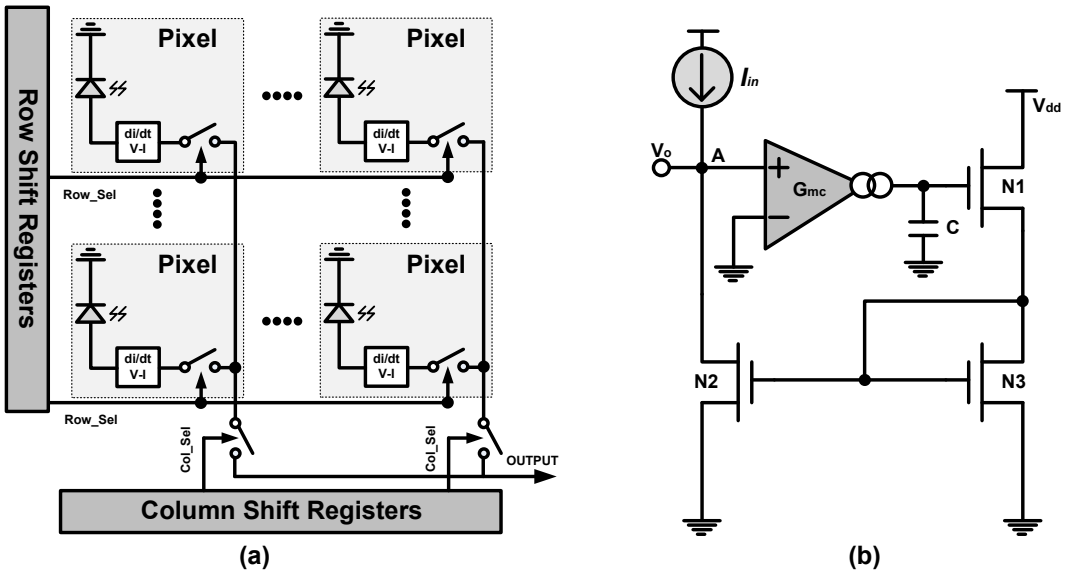


Figure 2.31: (a) System architecture of the current mode motion-detection image sensor. (b) Current mirror differentiator used in the pixel to detect current change. [10]

2.5.3 Intensity-Based Continuous Mode

In conventional frame-based image sensors, light intensity can only be acquired by an integration operation. Photo current can be directly readout by a logarithmical photoreceptor. The light intensity is directly transformed into electrical signals without integration [74]. Visual motion detection on the focal plane can be implemented by continuously monitoring the photo current or voltage changes over time. A motion-detection image sensor, based on this scheme, is proposed in [10]. When a temporally invariant but spatially variant image is projected onto a pixel array, the photo current on every photodiode is constant over time. If there are active motions on the focal plane, the varying light intensity will change the photo currents on relevant photodiodes. A current differentiator is implemented inside every pixel so as to track photo current variations. A given pixel is treated as detecting visual motions only when the change of its photo current exceeds a threshold.

The system architecture of the current mode motion-detection image sensor is

depicted in Fig. 2.31(a). Each pixel comprises of a photodiode, a current differentiator and a *V-I* convertor. The output voltage of the current differentiator is converted into current by a voltage to current convertor, which is then further sampled by the row and column shift registers. In real applications, the sampled current is summed up to implement some logic functions.

$$\frac{V_o}{I_{in}} \approx \frac{sC}{G_{mc}(g_{m1}//g_{m3})} \qquad \text{(Eq. 2.16)}$$

The current differentiator used in every pixel is developed based on a current mirror as shown in Fig. 2.31(b). The differentiator is constructed by adding a transconductance amplifier, driving a capacitor C, to the feedback loop of a current mirror. The input current is injected to node A, and its output voltage V_o is observed at the same node. The output voltage is the first order time derivative of the input current given in $\mathrm{Eq.\ 2.16}$, where V_o and I_{in} correspond to output voltage and input current respectively, G_{mc} is the gain of the transconductance amplifier, and g_m is the transconductance of each individual transistor. Because of the delay in charging capacitor C, there is a large overshoot on the output of the differentiator when a step current input is present. This differentiator can be characterized in terms of its response time T_R and the normalized response voltage O_R, defined in equations $\mathrm{Eq.\ 2.17}$ and $\mathrm{Eq.\ 2.18}$, where V_{omax} is the saturation voltage for this differentiator, t_1 and t_2 correspond to the period when the output voltage is equal to $V_{o\max}/\sqrt{2}$, and v_f is the final output voltage of this differentiator. The measured response period and normalized voltage of the current differentiator depends on the slew rate on the input current. Experimental results indicate that the current mirror differentiator is capable of differentiating input photon current in the subnanoampere range.

$$O_R = \frac{v_{o\max} - v_f}{v_{o\max}} \qquad \text{(Eq. 2.17)}$$

$$T_R = t_2 - t_1 \qquad \text{(Eq. 2.18)}$$

Since the current differentiator is implemented in the analog domain, this sensor occupies a much smaller area than its digital counterpart and does not require any analog to digital conversion. However, since most of the transistors used in this image sensor work in the subthreshold region, mismatch and offset issues are inevitable during its fabrication. Furthermore, due to the sequential scanning readout strategy in this sensor, external processors are not able to respond to fast visual motion despite detection by the pixel array. Hence, the sensor is prevented from high speed applications.

Continuous differentiation of the light intensity can also be realized in the voltage format. The first voltage-based continuous motion-detection sensor was proposed in [11]. The schematic diagram of this motion-detection pixel, embedded with the intensity voltage differentiator, is shown in Fig. 2.32. The processing elements consist of a photoreceptor and a voltage differentiator. A logarithmic photoreceptor is adopted in the pixel to covert photo currents into intensity voltages. In this design, several transistors are serially connected to the photodiode so as to compensate for the voltage drop on the source follower. The temporal differentiation on the intensity voltage is implemented with a feedback differentiator using a simple inverter as the amplifier. The feedback resistor is realized by an operational-transconductance-amplifier (OTA) and its resistance determines the lowest frequency of the differentiation. The relationship between the biasing current (I_{bias}) and the equivalent output resistance (R_{eq}) is given in $\text{Eq. } 2.19$, where $k = 1.38 \times 10^{-23} J/K$ is the Boltzmann constant, and n is the subthreshold ideality factor. Therefore, a small biasing current in the subthreshold region can yield a desirably high resistance in the order of GΩ. The transform function $H(s)$

of this differentiator is expressed as $\mathrm{Eq.\ 2.20}$, where R is the equivalent resistance and C is the feedback capacitance. This equation reveals that the intensity voltage from the photoreceptor is differentiated by the processing circuit in the time domain. By comparison with some thresholds, the output voltage from the differentiator can be digitalized into motion events.

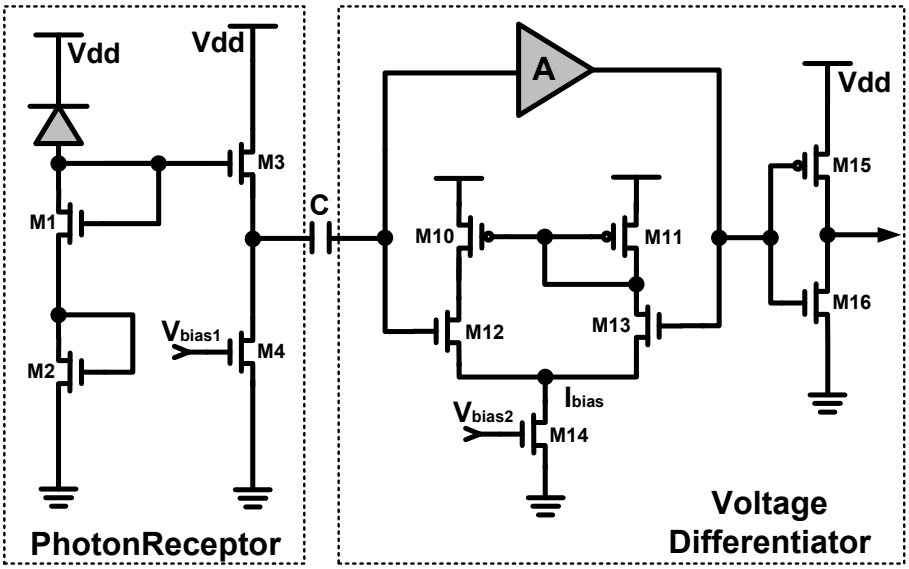


Figure 2.32: Schematic diagram of the motion-detection pixel circuit embedded with the intensity voltage differentiator. [11]

$$I_{bias} = \frac{kT}{nqR_{eq}} \qquad \text{(Eq. 2.19)}$$

$$H(s) = A\frac{A \times RCs}{1 + A + RCs} \cong A \times RCs \qquad \text{(Eq. 2.20)}$$

There are many outstanding advantages of this motion-detection image sensor. The sensor features a high dynamic range for various lighting conditions due to the presence of a logarithmic photodiode. Furthermore, all processing circuits are implemented in the analog domain. Hence, no additional analog to digital conversion is

required. Moreover, visual motion is continuously monitored by the pixel array in an approach that is analogous to the natural behavior of insects, which makes it suitable for artificial vision applications. However, there are also a number of limitations to this sensor. Since the intensity voltage of the logarithmic photoreceptor is directly connected to the differentiator without any amplification, the sensor only has low motion sensitivity. Although pixels in the sensor continuously monitor visual motion in the viewing field, transient motion events cannot be attended to immediately due to the sequential readout scheme.

Although the continuous motion-detection image sensors reported in [10] and [11] can detect fast motion activities, they share a common limitation in their sequential scanning readout strategy. Each pixel is periodically granted with the access to data bus no matter whether there is active motion detected in the pixel or not. Redundant data is produced in this scheme, while active motion events undergo certain delays in response. Fortunately, the asynchronous address-event-representation (AER) readout scheme alleviates such problems by reporting only active pixels with events. This approach significantly reduces data redundancy and preserves precise timing information. A variety of image sensors based on the AER strategy were proposed in [24, 75–77]. The AER communication protocol is also implanted on smart feature-extraction image sensors. The first motion-detection image sensor, based on the asynchronous protocol, is proposed in [12]. This image sensor continuously reports asynchronous address events to record reflection changes on the focal plane, and has been successfully exploited in various applications including neuromorphic engineering [78–80], object tracking [81–83] and pattern recognition [84–86].

Fig. 2.33 shows the block diagram of the proposed temporal contrast vision sensor. The main building blocks include a 128 $\times$ 128 pixel array, row and column address encoders, event handshaking communication logic, and arbiters. Inside every pixel,

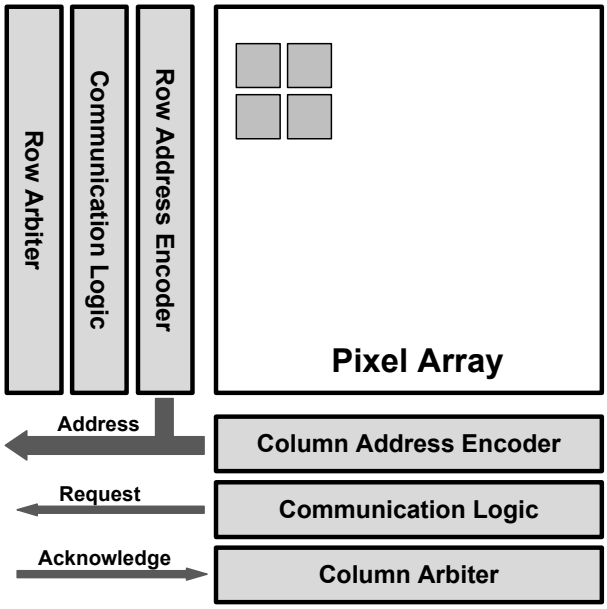


Figure 2.33: Block diagram of the asynchronous temporal contrast address-event-representation (AER) image sensor. [12]

there is a dedicated circuit for detecting relative intensity changes and converting them into binary motion events. Once a given pixel detects active motion as the "*ON*" or "*OFF*" event, it firstly sends a row request signal to the row address encoder and arbiter. Many row requests may be reported to the row arbiter together. However, only one of them can be acknowledged each time. The active pixels in the acknowledged row send their column requests to the column encoder and arbiter at the same time. A pixel is reset to start a new motion-detection cycle only when it receives both row and column acknowledgement signals. Row and column request flags in the event firing pixel are disabled immediately after the reset procedure. As a result, a global acknowledgement signal is fed back to this pixel from the external receiver, which further desserts both row and column acknowledgement signals. At the same time, global request and acknowledge signals are reset into the initial state again, which completes a basic 4-phase asynchronous handshaking communication.

The schematic diagram of the temporal contrast detection pixel is shown in Fig.

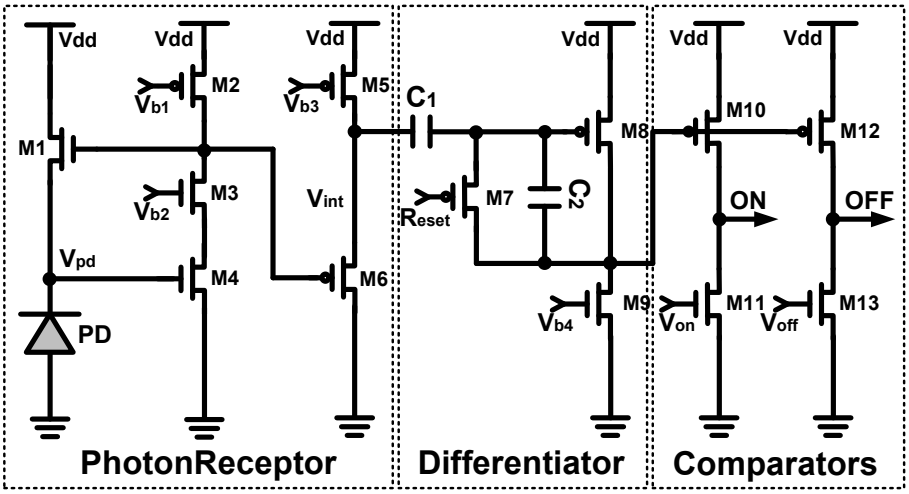


Figure 2.34: Schematic diagram of the continuous motion-detection pixel. [12]

2.35. The proposed pixel comprises of three building blocks including a photoreceptor, a differentiator and two compact comparators. The photoreceptor is implemented with a logarithmic photodiode and a gain control circuit. The logarithmic photoreceptor features an extreme dynamic range over 100dB, and it automatically controls the internal gain to adapt itself for different illumination conditions. However, this photoreceptor suffers a substantial mismatch caused by transistor threshold variations. The differentiator is implemented with a simple common source inverting amplifier and two capacitors C_1 and C_2. The gain of the voltage differentiator is determined by the well matched capacitor with a ratio of C_1/C_2. Hence, incident light intensity is converted to analog voltage $V_{\rm int}$ and its variation $\Delta V_{\rm int}$ is further amplified by the delta modulator. The output of this differentiator is directly connected to two compact comparators. When this output voltage exceeds the "V_{on}" or "V_{off}" threshold, a binary motion event "*ON*" or "*OFF*" is generated correspondingly.

Fig. 2.35 shows the sample images collected from this temporal contrast image sensor. The indoor "face scene" image (a) was acquired at night under illumination from a 15W desk lamp. The outdoor "driving scene" (b) was acquired in daylight with

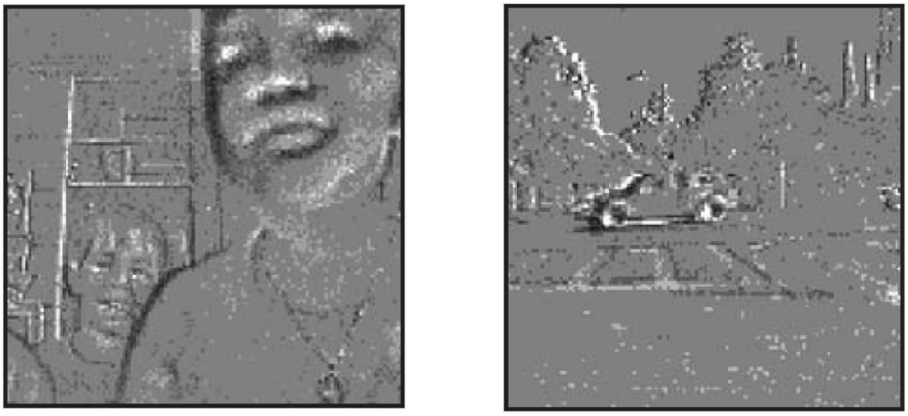


Figure 2.35: (a) Sample image of “faces” with indoor illumination by a desk lamp.(b) Sample image of “street” in outdoors under daylight. [12]

the sensor placed on a car dashboard. It demonstrates that the sensor can report temporal contrast visual motion in complex scenarios with a high dynamic range. Inspired by the continuous voltage differentiating strategy, a number of temporal contrast motion-detection image sensors were reported in the literature [87–90]. They all have the same motion-detection scheme but differs in terms of additional functionalities such as the intensity extraction, noise suppression and so on. Despite the high data encoding efficiency in motion detection, address event based image processing is still a challenge due to its asynchronous nature. If the motion event processing can be implemented on the focal plane, then the entire visual system can be more compact and efficient.

Another intensity-based continuous-mode motion estimation image sensor was proposed by Alan Stocker in [13]. The proposed image sensor extracts motion velocities based on the hardware implementation of optical flow computation. Horn and Schunck defined optical flow in relation to the spatial and temporal changes of image brightness in [91]. It is assumed that the total image intensity $E(x, y, t)$ does not

change with time, which is modelled as below.

$$\frac{d}{dt}E(x,y,t) = 0 \qquad \text{(Eq. 2.21)}$$

Expanding above equation based on Taylor series leads to the following expression.

$$F \equiv \frac{\partial}{\partial x}E(x,y,t)\mu + \frac{\partial}{\partial y}E(x,y,t)\nu + \frac{\partial}{\partial t}E(x,y,t) = 0 \qquad \text{(Eq. 2.22)}$$

Where $\mu = \frac{dx}{dt}$ and $\nu = \frac{dy}{dt}$ correspond to the optical flow vector. The above equation can be simplified as below.

$$F \equiv E_x\mu + E_y\nu + E_t = 0 \qquad \text{(Eq. 2.23)}$$

where $E_x = \frac{\partial}{\partial x}E(x,y,t)$ and $E_y = \frac{\partial}{\partial y}E(x,y,t)$ are spatial derivatives of the image intensity, $E_t = \frac{\partial}{\partial t}E(x,y,t)$ is the temporal derivative of image brightness.

The above equation has two unknowns in every spatial location, and there are infinite number of solutions. Horn and Schunck introduced a global constraint of smoothness to solve this problem. It tries to minimize the distortion in optical flow and choose the optimal solution with the most smoothness. The optical flow is then formulated as the minimization of a global energy function, as expressed below.

$$\int\int F^2 + \lambda\left(\frac{\partial^2\mu}{\partial^2 x^2} + \frac{\partial^2\mu}{\partial^2 y^2} + \frac{\partial^2\nu}{\partial^2 x^2} + \frac{\partial^2\nu}{\partial^2 y^2}\right)dxdy \to \min \qquad \text{(Eq. 2.24)}$$

where λ is the regulation constant. Larger value of λ results in a smoother optical flow. The above function can be minimized by solving the associated Eluer-Lagrange equations, and the Lagrange function is $L = F^2 + \lambda\left(\frac{\partial^2\mu}{\partial^2 x^2} + \frac{\partial^2\mu}{\partial^2 y^2} + \frac{\partial^2\nu}{\partial^2 x^2} + \frac{\partial^2\nu}{\partial^2 y^2}\right)$. Hence, the solution of $\mathrm{Eq.\ 2.24}$ must meet the following expressions.

$$\begin{aligned}\lambda\nabla^2\mu - E_x\left(E_x\mu + E_y\nu + E_t\right) = 0\\ \lambda\nabla^2\nu - E_y\left(E_x\mu + E_y\nu + E_t\right) = 0\end{aligned} \qquad \text{(Eq. 2.25)}$$

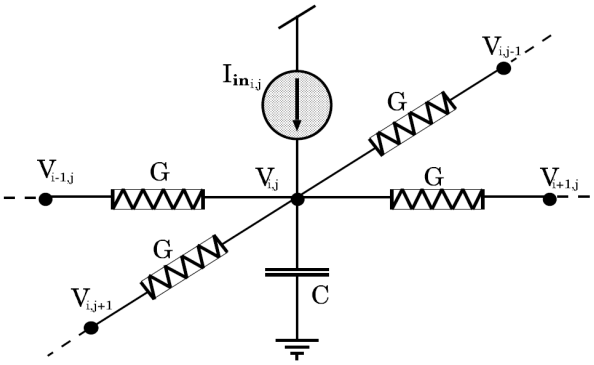


Figure 2.36: Schematic diagram of a single node on the resistive network. [13]

where $\nabla^2 = \frac{\partial^2}{\partial^2 x^2} + \frac{\partial^2}{\partial^2 y^2}$ denotes the Laplacian operator. In practice, the Laplacian operation in a regular grid can be approximated by a discrete five-point approximation. Hence, the above equation can be rewritten as below.

$$\begin{aligned} \lambda\left(\mu_{i+1,j} + \mu_{i-1,j} + \mu_{i,j-1} + \mu_{i,j+1} - 4\mu_{i,j}\right) - E_{x,i,j}F(i,j) = 0 \\ \lambda\left(\nu_{i+1,j} + \nu_{i-1,j} + \nu_{i,j-1} + \nu_{i,j+1} - 4\nu_{i,j}\right) - E_{y,i,j}F(i,j) = 0 \end{aligned} \qquad \text{(Eq. 2.26)}$$

where i and j indicate the pixel position. The above equation can be implemented in hardware by a resistor and capacitor network, as shown in Fig. 2.36.

Based on Kirchhoff's current law, the voltage on the center node can be expressed as followed.

$$C\frac{dV_{i,j}}{dt} = G(V_{i+1,j} + V_{i-1,j} + V_{i,j-1} + V_{i,j+1} - 4V_{i,j}) + I_{in,i,j} \qquad \text{(Eq. 2.27)}$$

where $V_{i,j}$ represents the voltage on the capacitor C, $I_{in,i,j}$ is the input current, and G is the conductance between two neighboring nodes. In the steady state, the above equation becomes as below.

$$G(V_{i+1,j} + V_{i-1,j} + V_{i,j-1} + V_{i,j+1} - 4V_{i,j}) + I_{in,i,j} = 0 \qquad \text{(Eq. 2.28)}$$

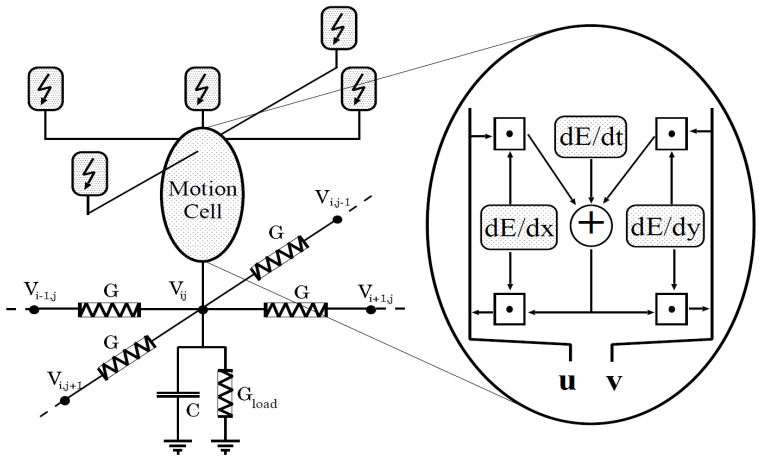


Figure 2.37: Schematic diagram of a single motion cell. [13]

Hence, $\mathrm{Eq.\ 2.26}$ can be implemented in hardware by a resistor network shown in Fig. 2.36. Their correspondences are summarized as below.

$$\begin{aligned} G &\leftrightarrow \lambda \\ I\mu_{in,i,j} &\leftrightarrow -E_{x,i,j}(E_{x,i,j}\mu_{i,j} + E_{y,i,j}\mu_{i,j} + E_{t,i,j}) \\ I\nu_{in,i,j} &\leftrightarrow -E_{y,i,j}(E_{x,i,j}\nu_{i,j} + E_{y,i,j}\nu_{i,j} + E_{t,i,j}) \end{aligned} \qquad (\mathrm{Eq.\ 2.29})$$

A diagrammatic sketch of a single motion cell with the resistor network is shown in Fig. 2.37. In the proposed sensor, there are two parallel resistor networks. The node voltage $U_{i,j}$ and $V_{i,j}$ represent the optical flow vector μ and ν respectively. For the sake of simplicity, only one resistive network layout is shown in Fig. 2.37. The input current $I\mu_{in,i,j}$ and $I\nu_{in,i,j}$ are computed by a negative current feedback loop modulated by the spatial and temporal intensity gradients. By regulating the conductance G on the resistive network, different smoothness constraints λ are realized.

The detailed circuit schematic of a single motion unit is shown in Fig. 2.38. The optical flow vector have to be able to take on positive and negative values with respect to some reference potentials. Hence, the circuit in a motion unit is symmetrically designed, so that positive and negative values are generated on separate lines. The

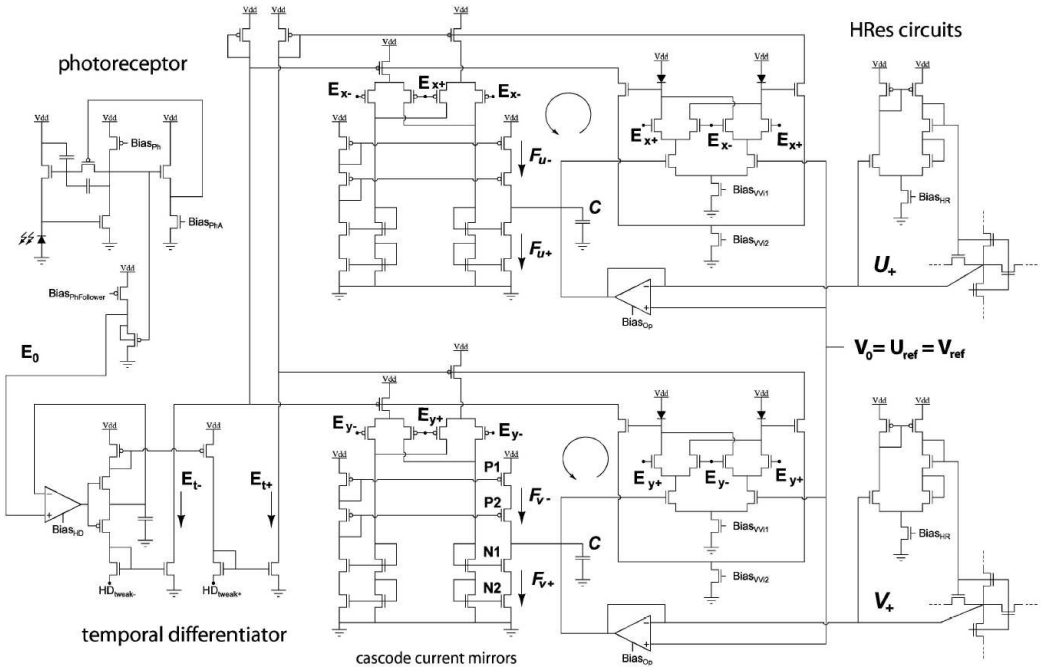


Figure 2.38: Circuit diagram of a single motion cell. [14]

actual value for a motion vector is computed by the difference of two potentials. Experimental results show that the proposed resistor network with the motion cell is able to precisely estimate visual motions by exporting accurate optical flow values. However, the analog implementation of the optical flow computation is achieved with a high hardware cost. Every pixel has more than 200 transistors and consumes a silicon area of 170 $\times$ 170 μm^2 [92].

2.5.4 Summary and Comparison

The original work on motion detection is developed based on the conventional frame-based strategy, which compares differences between two temporal consecutive frame images. Inspired by biology, a number of asynchronous temporal contrast image sensors were developed in recent years. Such image sensor is termed as a “silicon retina”, and its output is compatible with other silicon neuron chips through a com-

mon address-event-representation (AER) communication strategy. The following table summarizes existing motion-detection image sensors and their major characteristics. Temporal contrast image sensors consume much more power than temporal difference ones due to their parallel motion detection strategies.

Table 2.2: Comparison of existing motion-detection image sensors in literature

	Chi [72] '05	Kim [73] '13	Lichtsteiner [12] '08	Bardallo [87] '11
Technology	0.5 μm 2P 3M	0.35 μm 2P 4M	0.35 μm 2P 4M	0.35 μm 2P 4M
Array Size	90 $\times$ 90	128 $\times$ 128	128 $\times$ 128	128 $\times$ 128
Pixel Size	25.2 $\times$ 25.2 μm^2	16 $\times$ 21 μm^2	40 $\times$ 40 μm^2	35 $\times$ 35 μm^2
Readout Scheme	Frame-Based	Frame-Based	AER-Based	AER-Based
Fill Factor	17%	42%	8.1%	8.7%
Supply Voltage	3V	3V	3.3V	3.3 V
Power Consumption	0.8 mW	1.02 mW	24 mW	130 mW

Chapter 3

Temporal Difference Motion-Detection Image Sensor

This chapter describes two frame-based temporal difference image sensors for the motion-detection applications. The first smart image sensor is a compact vision system which integrates motion detection and image processing on the same chip. It not only detects visual motions in the viewing field into binary events but also sequentially processes them so as to localize three kinetic objects in parallel. This image sensor was evaluated by the experiments in the traffic surveillance applications. The second image sensor is developed based on the 3D integrated circuit technology and it is designed to extract either temporal motions or spatial contours from the viewing scene. In addition to the feature extraction capacity, this image sensor has an extreme fill factor because of the tier-stacking technology in the 3D process. This chapter is organized as follows: firstly, the principle and limitations of the temporal difference algorithm is deeply analyzed; This is followed by a detailed description on the motion-detection image sensor with the object localization capability; Next, the 3D integrated feature-extraction image sensor is presented; The last section concludes this chapter with a comparison of other feature-extraction image sensors reported in the literature.

3.1 Temporal Difference Algorithm

There are three common algorithms widely utilized for motion detection in the computer vision field: background subtraction, temporal difference, and optical flow.

- *Background Subtraction:* Visual motion is detected by computing the distinction between a predefined background model and the captured images [93]. However, the eventual result of the motion detection in this approach is highly dependent on the accuracy of the background model. Moreover, the unfixed background information has to be updated dynamically when the content in the viewing scene changes significantly, which leads to complex modeling work in post image processors. This algorithm is limited to motion detection but also can be utilized for object extraction.

- *Temporal Difference:* In this method, the active motion is extracted by analyzing differences between two successive temporal images [94]. The main advantage of this algorithm is its simple implementation, as it only requires one buffer memory to store previous intensity image frames in post image processors. Unfortunately, this technology only detects spatial contours on active objects and fails to extract the entire body on monotonous-textured objects.

- *Optical Flow:* Each incoming image is broken into gridding blocks which are then compared with successive images to calculate motion vectors [95]. Although this approach can provide the best performance among all motion-detection algorithms, the repetitive calculation severely aggregates the computational load on post processors. A hardware implemented optical-flow algorithm onto image sensors has been reported in [13].

3.1.1 Principle Analysis

Among all motion-detection algorithms, the temporal difference method is the least complex and most feasible to be implemented in hardware on the focal plane. This chapter presents two smart image sensors which integrate the temporal difference algorithm on-chip in order to extract motion features. The temporal difference computation in these image sensors can be modeled as $\mathrm{Eq.\ 3.1}$.

$$P_{diff}\left(x,y,N\right) = P\left(x,y,N\right) - P\left(x,y,N-1\right) \qquad \text{(Eq. 3.1)}$$

where N is the frame index in a vide sequence and $P(x,y,N)$ represents the intensity value of the pixel (x,y) in the N_{th} frame image. The temporal difference $P_{diff}(x,y,N)$ can be digitalized to a motion event image by comparing with two user defined thresholds as below.

$$\begin{cases} E(x,y,N) = 1, & P_{diff}(x,y,N) > TH_p \\ E(x,y,N) = 0, & TH_p > P_{diff}(x,y,N) > TH_n \\ E(x,y,N) = -1, & P_{diff}(x,y,N) < TH_n \end{cases} \qquad \text{(Eq. 3.2)}$$

where TH_p and TH_n denote the positive and negative thresholds respectively. When the brightness change on a pixel exceeds the positive threshold, a positive motion event is detected. Similarly, if the intensity change triggers the negative threshold, there is a negative motion event fired accordingly. Fig. 3.1 illustrates the temporal difference algorithm. Subfigures (a) and (b) correspond to two temporal consecutive image frames. Their intensity difference in absolute value is shown in subfigure (c). The motion event image is as shown in subfigure (d), in which the white and black dots represent individual positive and negative events.

3.1.2 Parameter Analysis

The output of the temporal difference algorithm is constant when there are two fixed images as the input. However, the eventual binary motion events are flexible because

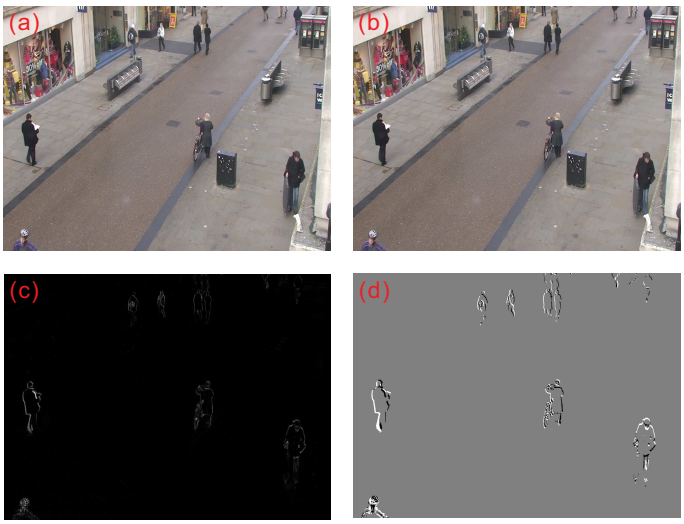


Figure 3.1: Demonstration of the temporal difference algorithm. Subfigures (a) to (d) correspond to two consecutive images, their intensity difference in absolute value and the motion image.

of the tunable thresholds. Fig. 3.2 shows such effect by varying the threshold in the temporal difference algorithm. In this simulation, the thresholds in subfigures (a) to (d) are intentionally chosen as 5, 10, 40 and 80 respectively. As shown in Fig. 3.2, there is a decrease in event activity when the threshold is increased. In order to investigate this effect further, the simulation in Fig. 3.3 sweeps the threshold and records the number of events per frame in the motion image. It shows that there is an inverse relationship between the event activity and the threshold value.

Fig. 3.2 reveals that the thresholding operation can remove the discrete background noise to a certain extent. However, it is not effective for all types of noise. For example, in the simulation of Fig. 3.4, “salt” and “pepper” noise is inserted into the intensity image in subfigure (b) with a spatial density of 0.5%. The binary motion image in subfigure (d) shows that such discrete noise cannot be completely removed by the thresholding operation.

In the conventional active-pixel-sensor (APS) structure, an intensity image is acqui-

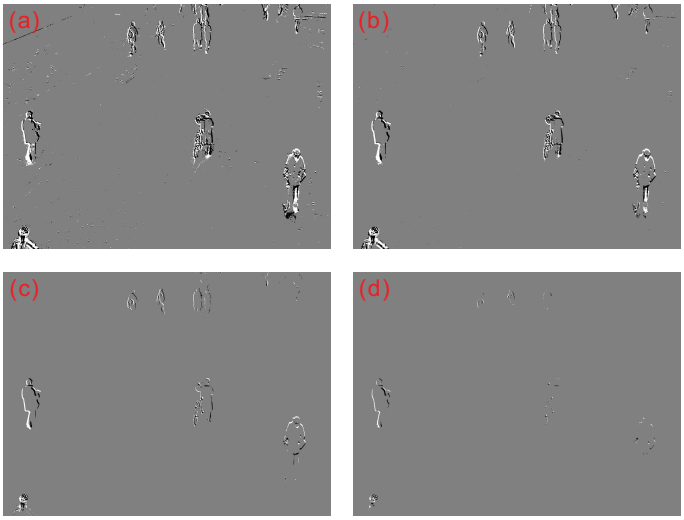


Figure 3.2: Demonstration of the thresholding effect in the temporal difference algorithm.

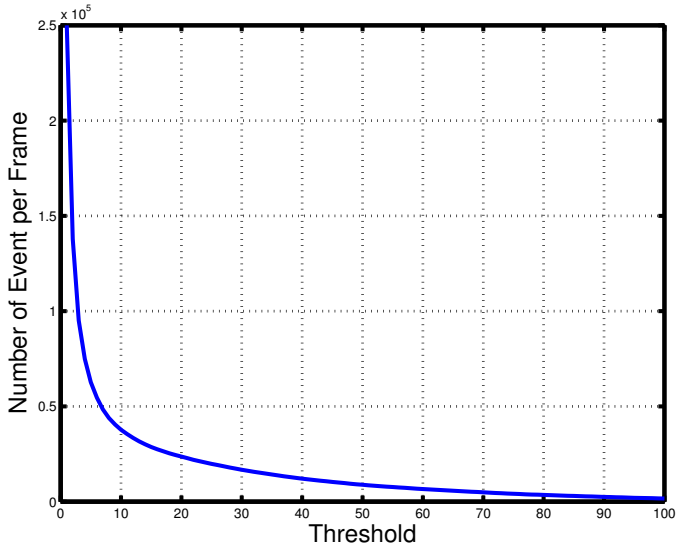


Figure 3.3: Relationship between the threshold and motion event density in the temporal difference algorithm.

red by an integration operation within a certain period, which is modeled as Eq. 3.3, where $P(x, y, N)$ represents the intensity value of the pixel (x, y) in the N_{th} frame image and $I(x, y, t)$ indicates the light intensity along with time. In this equation, the

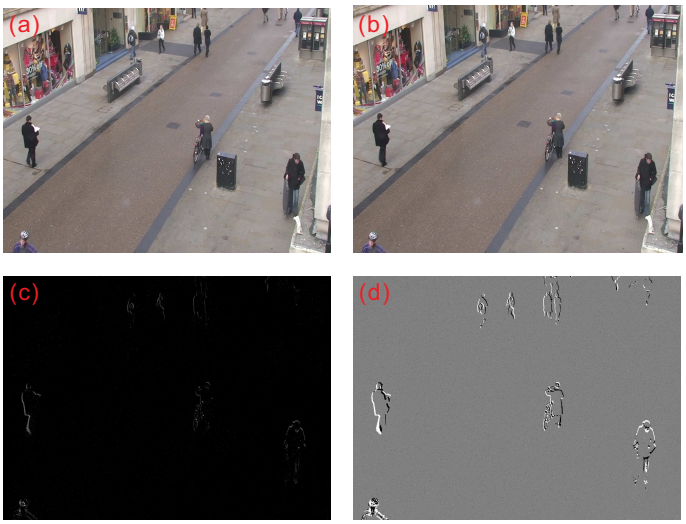


Figure 3.4: "Salt" and "Pepper" noise effect in the temporal difference algorithm. Subfigures (a) to (d) correspond to the test images inserted with noise, their intensity difference and the motion event image.

integration period is fixed as the time interval T. Such integration operation averages the light intensity within the sampling period T, and it smoothens brightness variations. Therefore, the performance of the temporal difference algorithm is highly sensitive to the exposure setup of the image sensor. Fig. 3.5 illustrates this limitation. In terms of simplicity, examples (a) and (b) have the same light intensity change in a periodical waveform. Although there are intensity changes in the viewing scene, in the exposure setup of the example (a), the temporal difference algorithm fails to detect visual motion as consecutive frames (red and blue ones) always have the same intensity value. When the exposure period is delayed by a 1/4 cycle, as shown in the example (b), the temporal difference algorithm can effectively detect visual motion as consecutive frames have different intensity values. Fig. 3.5 reveals that the optimal performance in the temporal difference algorithm is achieved only when the integration setup of the image sensor matches the motion speed, and this is termed as the "aliasing effect".

$$P(x, y, N) = \int_{t-T}^{t} I(x, y, t)dt \qquad \text{(Eq. 3.3)}$$

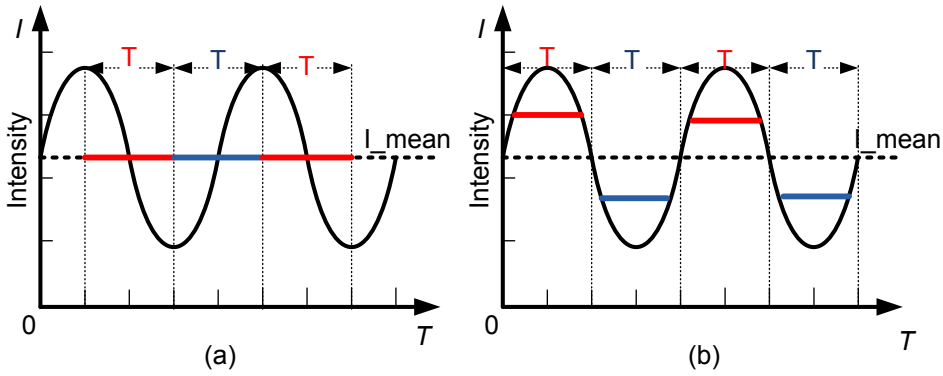


Figure 3.5: Demonstration of the synchronization problem in the temporal difference algorithm.

3.2 Moving Object Localization Image Sensor

This section describes a frame-based motion-detection image sensor with an object localization capacity [96]. This image sensor not only transforms visual motions in the viewing field into binary events, but also processes them on the fly so as to localize and track three kinetic objects in parallel. This image sensor has been evaluated in the traffic surveillance applications with satisfactory results in detecting and tracking kinetic objects. In this section, the system architecture is first illustrated, followed by detailed descriptions of pixel operation and event generation. Experimental results are presented at the end of this section.

3.2.1 System Architecture

Fig. 3.6 shows the system architecture of the temporal difference image sensor with on-chip moving object detection and localization functions. The main building blocks include a 64 $\times$ 64 pixel array, a motion event generator, an object localization unit with additional address controller and decoders. The photodiode inside every pixel is used to capture an analog intensity image, and the capacitor is designed to store

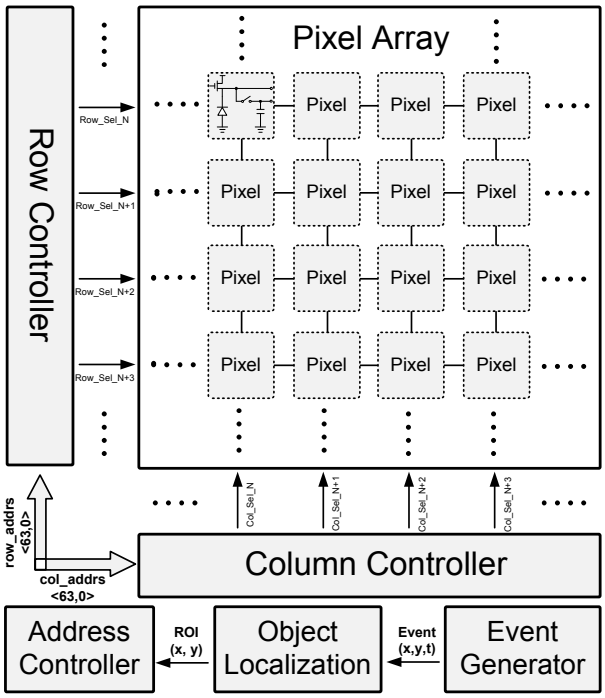


Figure 3.6: System diagram of the motion-detection image sensor with the object localization capability.

previous frame information. When a pixel is accessed by the row and column controllers, it exports two temporal consecutive frame images simultaneously. The global event generator computes and digitalizes their difference into binary motion events by comparing with two predefined thresholds. When the scene illumination and object reflectance are constant, the changes in brightness on the pixel array are induced by motion activities in the viewing field. Eventually, the visual motion on the focal plane is extracted, while the stationery background is completely discarded. The binary motion events are processed by the object localization unit immediately after generation. At the end of each frame, information on the position and size of the largest active object are reported to the address controller at the same time. Equipped with the window address information, the image sensor activates row and column controllers to access the region-of-interest (ROI) for capturing an intensity image on the main moving object.

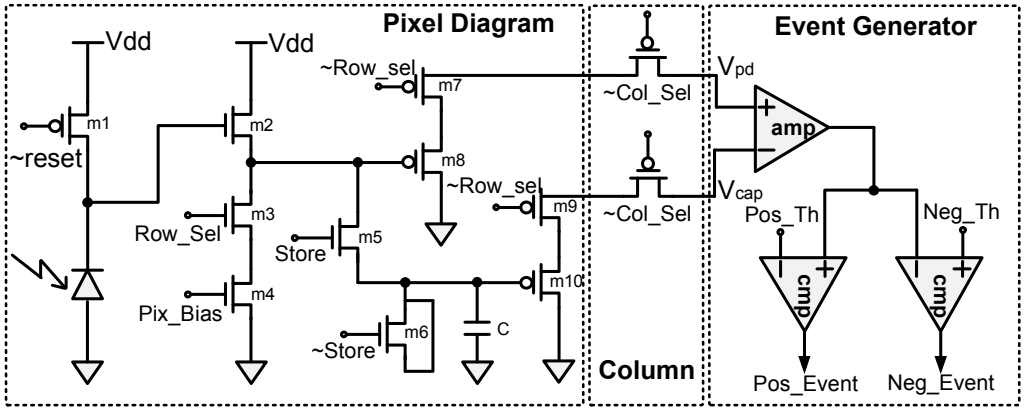


Figure 3.7: Schematic of the signal path from a select pixel to the global event generator.

3.2.2 Pixel Structure

Fig. 3.7 shows the signal path from the pixel to the event generator in the temporal difference image sensor. Each pixel consists of a photosensitive photodiode, a PMOS reset transistor (m1), a unity gain buffer (m2 - m4), a sampling circuit (m5 - m6 and capacitor C) and two sets of source followers (m7 - m10). The photodiode is implemented by a N-Well on the P-substrate. The PMOS reset transistor enables the photodiode to be initialized into the power supply voltage with a wider voltage swing. In order to reduce the voltage level shift in the source follower, a specialized low threshold transistor (m2) is utilized in the circuit, which suppresses the level shift from 500 mV into 300 mV. The voltage shift is further compensated by the readout circuit formed by the PMOS source followers (m7 - m10). In order to reduce the body effect, all PMOS transistors are designed in separate N-Wells with their bulk voltages tied to the source nodes. An additional transistor (m3) works as a power saving device that turns on the source follower path only when the pixel is accessed for readout.

The operation on the pixel follows a rolling sequence of "***reset***", "***integration***", "***readout***" and "***sampling***". At the beginning of the cycle, the photodiode is initialized

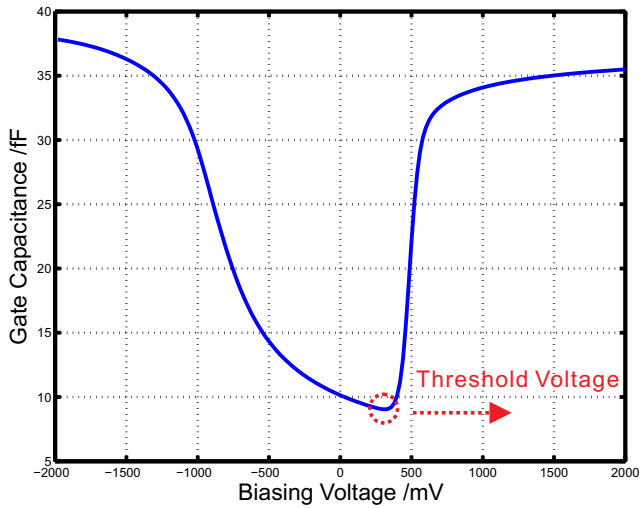


Figure 3.8: Equivalent gate capacitance for an NMOS transistor in the relationship to its gate biasing voltage.

to the power supply voltage (V_{dd}) by activating the reset transistor (m1). The reset transistor (m1) is then switched off to begin the "***integration***" phase. After the exposure period, the pixel is accessed for "***readout***" by the row select signals ("*Row_Sel*" and "$\sim$*Row_Sel*"). The newly integrated voltage on the photodiode (V_{pd}) and the stored voltage on the capacitor (V_{cap}) are then simultaneously readout to column buses. Next, in the "***sampling***" procedure, the enabled transistor (m5) is switched on to update the voltage on the capacitor with the newly integrated intensity voltage. A dummy transistor (m6) works as a charge injection canceling device to reduce the voltage variation during the switching on the store transistor (m5).

The capacitor in Fig. 3.7 is implemented with an NMOS transistor. The MOS transistor-based capacitor features a larger capacitance density when compared to the common metal-isolator-metal (MIM) capacitor structure. Fig. 3.8 shows the relation between the equivalent capacitance and the gate voltage on an NMOS transistor-based capacitor (W: 2 μm and L: 2 μm) in a 0.18 μm CMOS process. This NMOS transistor has the minimum capacitance ($\sim$9.8 fF) when its gate voltage approaches

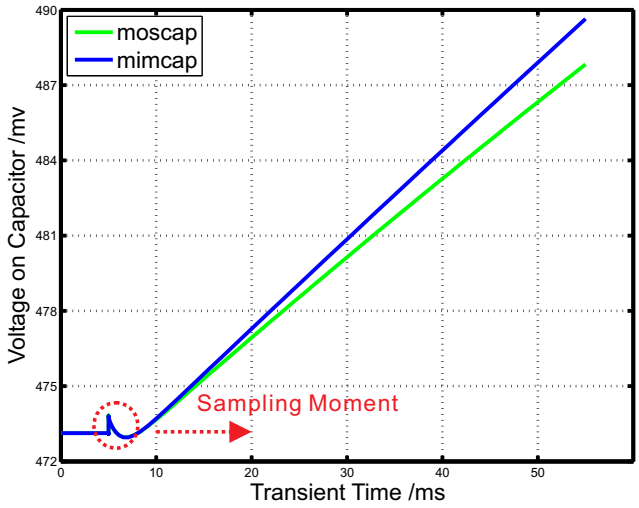


Figure 3.9: Simulation of the voltage variation for capacitors developed on MOS and MIM structures.

the threshold value ($\sim$475 mV).

Due to the current leakage in the pixel circuit, the sampled intensity voltage varies along with time. The minimum frame rate designed for this image sensor is 20 frames/s, which indicates that the sampled voltage on the capacitor has to be maintained for 50 ms. Meanwhile, the global event generator amplifies the intensity voltage difference by a gain factor of 10, and the smallest threshold window is about 350 mV. Therefore, for the longest integration period, the maximum tolerant voltage variation on the capacitor is about 17.5 mV. In order to achieve such a small voltage variation within a period of 50 ms, the minimum required MIM-based capacitor is $\sim$360 fF based on the simulation in the Spectre. This capacitor occupies a silicon area over 22.8 μm $\times$ 22.8 μm. However, a MOS-based capacitor with the same capacitance only occupies a silicon area of 9.3 μm $\times$ 8.7 μm, which saves over 84% space when compared to the MIM-based structure. Fig. 3.9 shows the transient voltage variations on the MIM-based and MOS-based capacitors. In this simulation, the sampling procedure starts at 5 ms, and the initial voltage is chosen as the threshold value of $\sim$473 mV. Hence, the

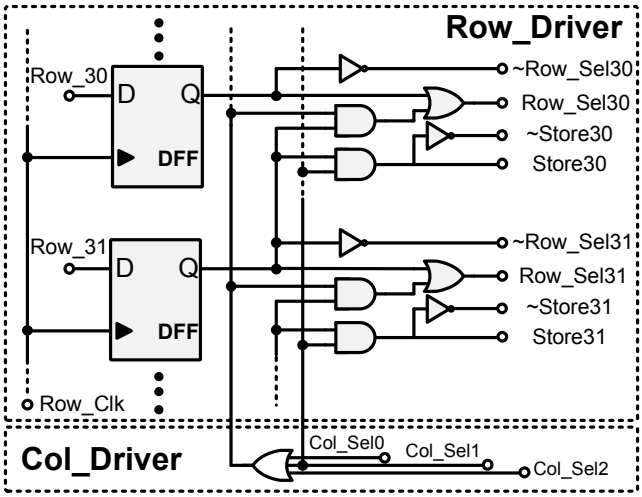


Figure 3.10: Schematic diagram of the row controller in the motion-detection image sensor.

MOS-based capacitor has the minimum capacitance ($\sim$360 fF). Within the integration period $\sim$50 ms, both capacitors are able to maintain the voltage variation less than 15 mV. However, the MOS-based capacitor is selected as the memory device in the pixel, as it is superior to its MIM-based counterpart with less silicon consumption.

3.2.3 Row Controller

The "***reset***", "***integration***", "***readout***" and "***sampling***" operations on the pixel array are performed using the row-based rolling strategy. As shown in Fig. 3.10, the row controller is implemented by a chain of D-type-flip-flops (DFF) with the associated control logic. Row and column addresses are provided by a global address controller. Each node of the row address chain directly generates control signals for the respective row. When a given row is accessed by the row controller, all pixels in the same row simultaneously export their intensity voltages to the column buses. The column readout chain then sequentially scans every column to export the intensity voltages to the global event generator.

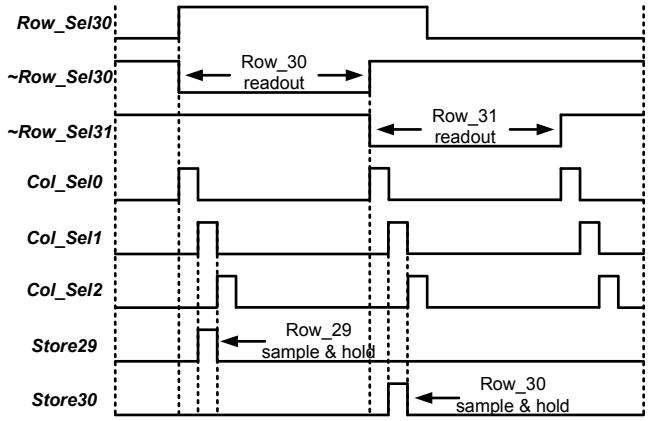


Figure 3.11: Timing diagram of the row controller in the motion-detection image sensor.

When all pixels in the selected row are readout, capacitors in the current row are overwritten with the newly integrated voltage in the "***sampling***" operation. This procedure is executed only when the row controller selects next row for readout. Transistors (m3 and m5) are activated at different moments to avoid the loss of charge on the capacitor (C) due to the malfunction of the source follower (m2 - m4). In the "***sampling***" operation, as shown in Fig. 3.11, the row select signal ("*row_sel30*") activates not only during the complete 64 column cycles when the 30th row is selected for readout but also in the first 3 column cycles when the 31rst row is accessed ("*row_sel31*"). Within this period, the storage transistor (m5) is turned on only when the 2th column is accessed. The "***readout***" operation is performed in a column-by-column manner, while the other operations, such as "***reset***" and "***sampling***", are executed in the row parallel scheme. Therefore, pixels in the same row have different integration times, and it is a column wise mismatch in the pixel array.

3.2.4 Event Generator

In the "***readout***" procedure, two intensity voltages one from the photodiode (V_{pd}) and the other from the capacitor (V_{cap}) are copied onto column buses simultaneously. Their

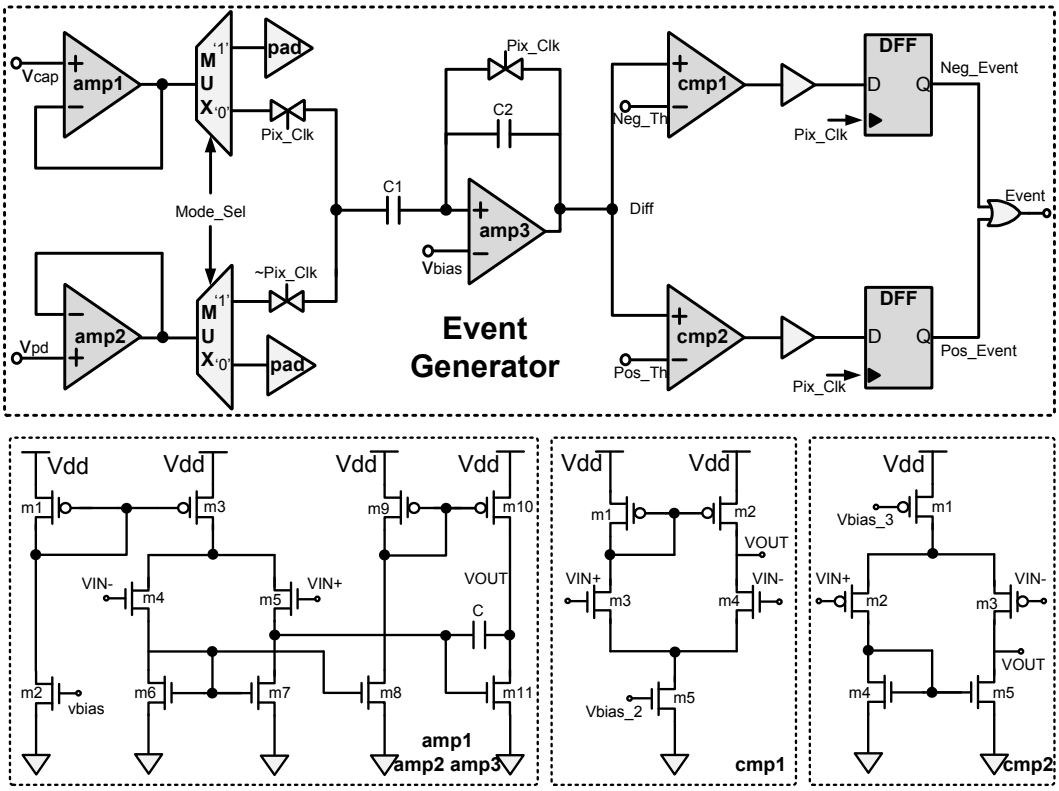


Figure 3.12: Block diagram of the motion event generator with detailed circuits of the amplifier and comparator.

difference is computed by the global event generator, as shown in Fig. 3.12. This event generator comprises of four stages: a unity gain buffer; a switched-capacitor differentiator; two comparators; and two sampling blocks. The image sensor may operate under the intensity mode or the motion mode. Mode switching is controlled by an enabling signal ("*Mode_Sel*") on the multiplexer. Under the intensity mode, the unity gain buffers directly drive the analog pads in order to export the intensity voltages into external analog-to-digital-convertors (ADCs). The maximum frame rate and desired maximum pixel clock frequency of this image sensor is set at 100 frames/s and 400 KHz respectively. In order to stabilize the analog output within the 1/4 pixel clock cycle, the unity-gain-bandwidth (UGB) of the amplifier should be over 1.83 MHz. Amplifiers "*amp1*" to "*amp3*" are designed in the same structure, as shown in Fig. 3.12. Both

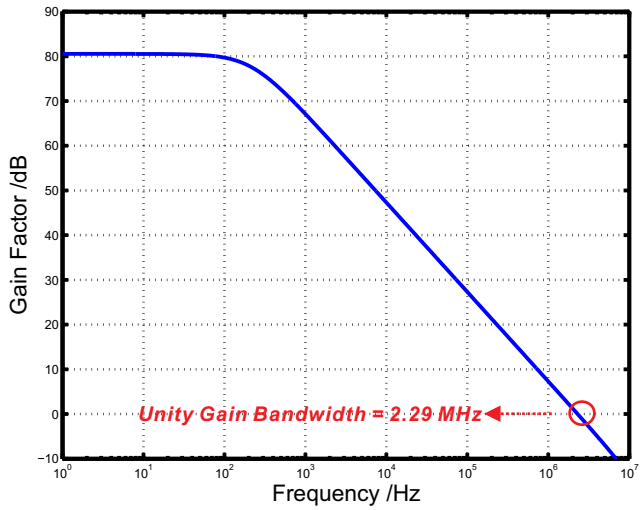


Figure 3.13: Frequency response of the gain factor of the amplifier.

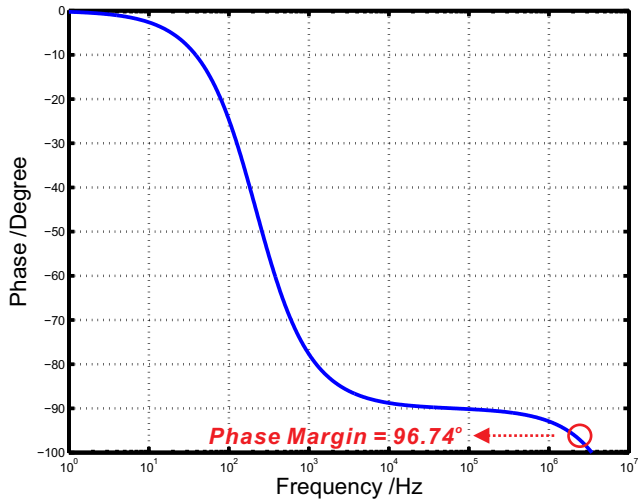


Figure 3.14: Frequency response of the phase shift of the amplifier.

Fig. 3.13 and Fig. 3.14 characterize the amplifier with the simulation results from the Spectre. Under the 25 pF loading condition, the designed amplifier has a gain factor of $\sim$ 80 dB with a unity-gain-bandwidth (UGB) of $>$ 2 MHz and the phase margin of $\sim 96.7^o$. The overall power consumption of this amplifier is about 100 μW under a supply voltage of 1.8 V.

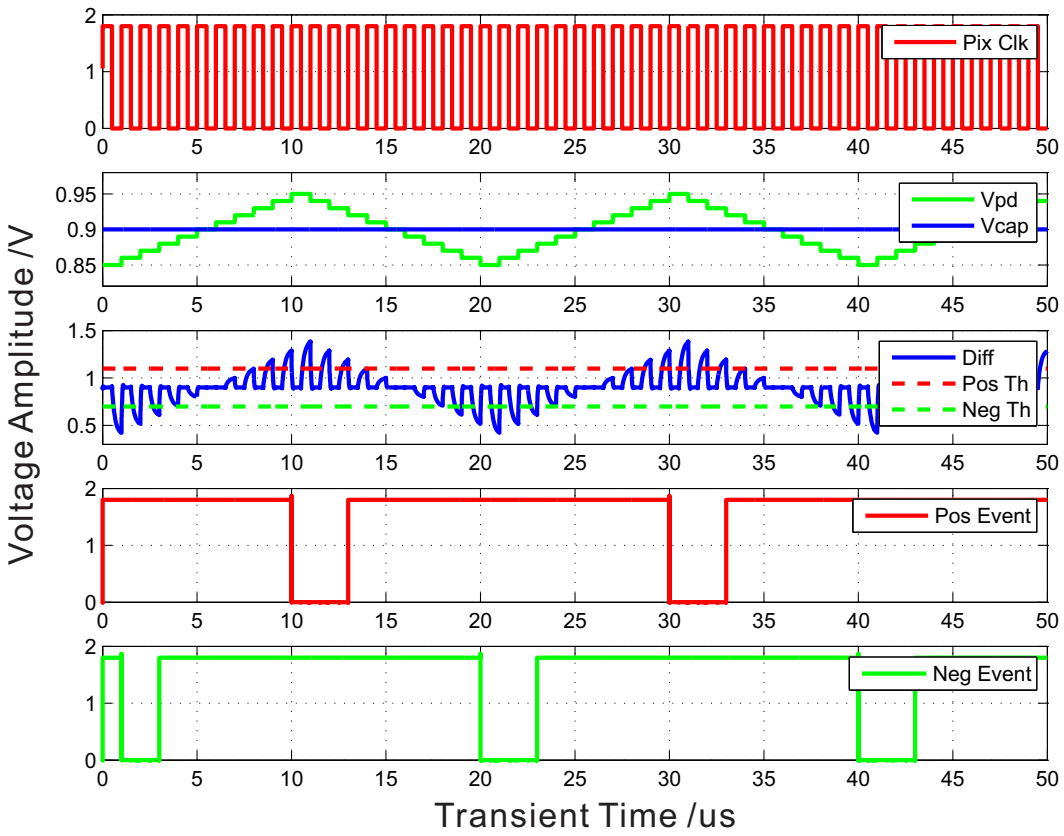


Figure 3.15: Transient simulation of the event generation in the motion-detection image sensor.

$$V_{\text{Diff}} = \frac{C_1}{C_2} \times (V_{cap} - V_{pd}) \qquad \text{(Eq. 3.4)}$$

Under the motion mode, the unity gain buffers are connected to the differentiator. The difference between the voltages from the capacitor (V_{pd}) and photodiode (V_{cap}) is amplified by a factor of C1/C2, as modeled in Eq. 3.4. In this image sensor, the capacitors C1 and C2 are designed with a capacitance of 20 pF and 2 pF respectively. The difference in the intensity voltages is hence amplified 10 times in the delta modulator. The balance voltage in the modulator is centered at 900 mV (V_{dd}/2). The threshold voltages ("*Pos_Th*") and ("*Neg_Th*") are regulated externally. Eventual binary motion

events are sampled by two digital flip-flops. Fig. 3.15 shows the transient simulation of the event generator. For the sake of simplicity, the voltage from the capacitor (V_{cap}) is assumed to be constant, while the voltage on the photodiode (V_{pd}) is modeled as a stair case function precisely synchronized with the pixel clock ("*Pix_Clk*"). The comparator thresholds ("*Pos_Th*" and "*Neg_Th*") are symmetrically set as 700 mV and 1100 mV respectively. The simulation result indicates that the event generator circuit effectively computes and digitalizes the difference between the two intensity voltages into a binary event, once it exceeds the predefined threshold. Finally, binary motion events are simultaneously reported to an external processor and an internal object localization unit.

3.2.5 Object Localizer

In this image sensor, there is an on-chip hardware-implemented event-clustering algorithm that processes motion events so as to localize the largest active object in the viewing scene. In this algorithm, motion events are grouped into three separate clusters based on their spatial distances. Since each cluster represents a potential object, the event-clustering algorithm can detect and localize three active objects in parallel. In the motion mode, the size and position of the largest object is extracted by the event-processing algorithm at the end of each frame. The image sensor switches into the intensity mode to report an analog image of the main object in the region-of-interest (ROI), when necessary. The clustering algorithm was implemented as the object localizer in digital circuits using Verilog hardware-description-language (HDL). Therefore, this image sensor works independently as a compact vision system without the need for external memories and processors. The object clustering algorithm was proposed and implemented by my project partner Dr. Zhao Bo, and it has been described in great details in [96, 97].

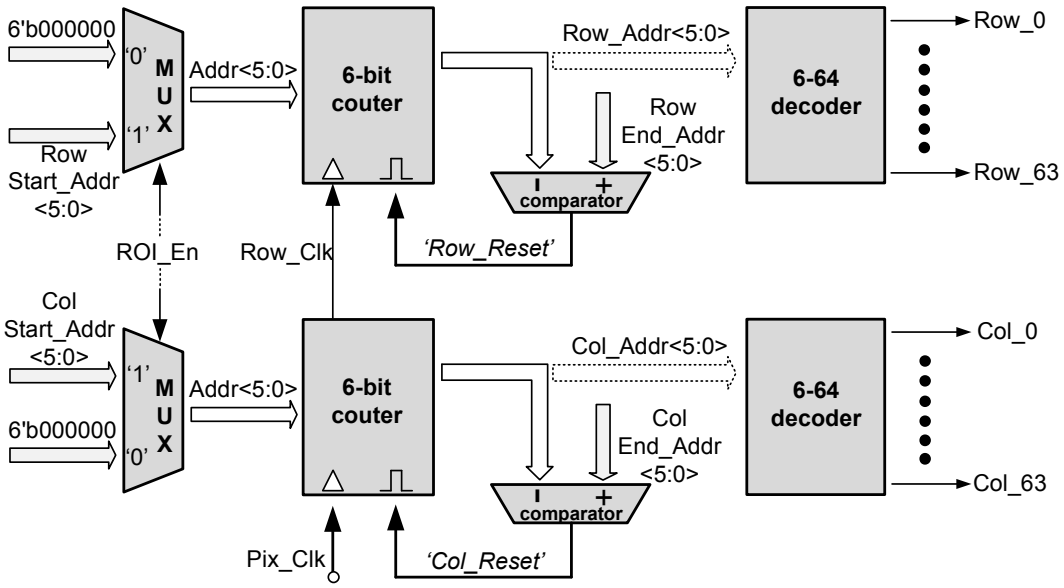


Figure 3.16: Schematic diagram of the global address controller in the motion-detection image sensor.

3.2.6 Address Controller

This image sensor accesses either part or the entire pixel array by regulating the active region window in the address controller, as shown in Fig. 3.16. This function is implemented using two multiplexers controlled by an enable signal ("*ROI_En*"). By default, this image sensor accesses the entire pixel array under the intensity mode in order to report a full resolution image. Under the motion mode, the hardware-implemented event-clustering algorithm sequentially processes the motion events fired from the event generator. At the end of each frame, the position and size of the main object ("Row_Start_Addr", "Row_End_Addr", "Col_Start_Addr" and "Col_End_Addr") are reported to the address controller. When necessary, the external processor switches this image sensor to access the region-of-interest (ROI) so as to report an intensity image on the main active object.

3.2.7 VLSI Implementation

This motion-detection image sensor with object localization capacity was implemented in hardware using UMC 0.18 μm CMOS process. Table. 3.1 summarizes the main characteristics of the image sensor. Under a spatial resolution of 64 $\times$ 64, the maximum speed and power consumption of the image sensor are 100 frames/s and 0.4 mW respectively. The microphotograph in Fig. 3.17 highlights the main building blocks of the sensor. This image sensor has a total die size of 1.5 mm $\times$ 1.5 mm, including I/O pads. Each pixel occupies a silicon area of 14 μm $\times$ 14 μm with a fill factor of $\sim$32%.

Table 3.1: Chip characteristics of the motion-detection image sensor

Process Technology	UMC 0.18 μm 1P6M CMOS
Die Size	1.5 $\times$ 1.5 mm^2
Pixel Array	64 $\times$ 64
Pixel Size	14 $\times$ 14 μm^2
Pixel Complexity	10 transistors
Fill Factor	32%
Readout Strategy	sequential scan
Frame Rate	100 fps
Supply Voltage	1.8 V
Power Consumption	pixels array + motion detection 0.4 mW, object localization 8.63 μW (@100fps)

3.2.8 Experimental Results

In order to evaluate the image sensor, a testing platform was developed based on the Opal Kelly XEM 3010 board, as shown in Fig. 3.18. The board contains a Xilinx Spartan-3 field-programmable-gate-array (FPGA) chip (XC3S1500), a Micron 32 MBytes synchronous-dynamic-random-access-memory (SDRAM) and a universal-serial-bus (USB) 2.0 port on this board. The development board was programmed to operate the image sensor through control signals, including the system clock and the address

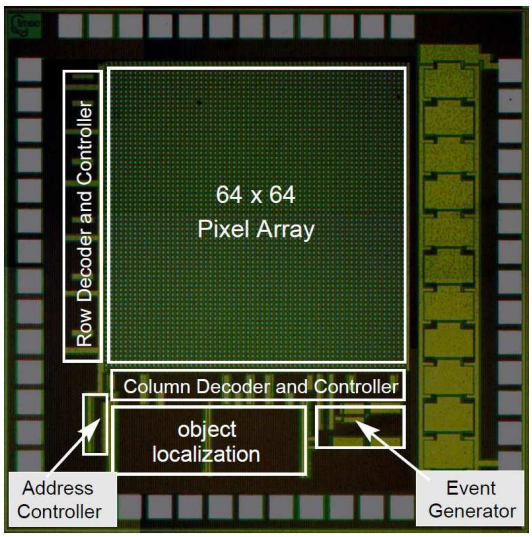


Figure 3.17: Microphotograph of the motion-detection image sensor with main building blocks highlighted.

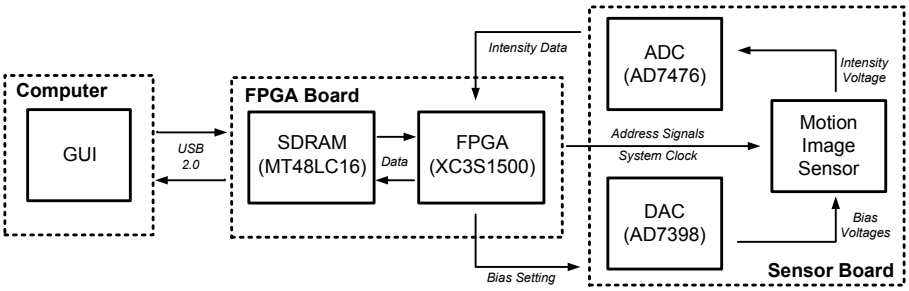


Figure 3.18: Block diagram of the testing platform for the motion-detection image sensor.

window. The image sensor was soldered on a customized printed-circuit-board (PCB), which integrates a 12-bit ADC (AD7476) and a 12-bit DAC (AD7398). All these components are controlled by the central FPGA chip via a serial-peripheral-interface (SPI). Since the data bandwidth for the USB interface is over 40 Mbit/s, ideally, this testing platform can transmit over 800 frames of 64 $\times$ 64 resolution image per second to the host computer. However, in practice, the maximum working speed of this platform is about 100 frames/s, which is limited by the frame rate of the image sensor. On

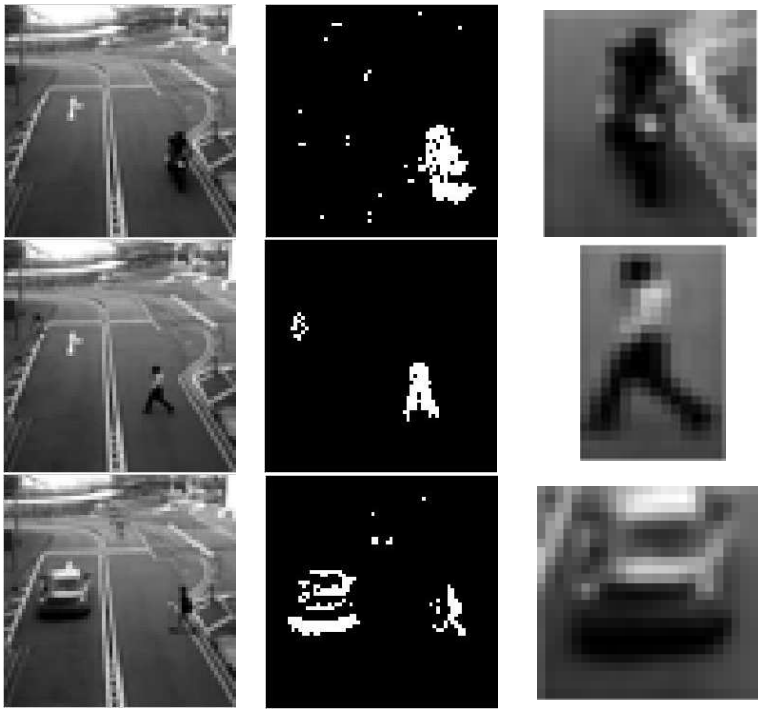


Figure 3.19: Sample images acquired by the motion-detection image sensor in a traffic surveillance application. Column (a) shows the full resolution intensity image. Column (b) shows the binary motion event image. Column (c) shows the intensity images on the regions-of-interest (ROI).

the computer side, a graphic-user-interface (GUI) was developed to display captured images and regulate motion sensitivity as well as other operational parameters for the image sensor. Fig. 3.19 shows the sample images captured by this image sensor in a traffic surveillance application. In this experiment, the largest active object is assumed to be the main target. Column (a) shows the full resolution gray-scale images captured in the intensity mode. Column (b) shows the binary motion images collected in the motion mode. Column (c) shows the intensity images captured on the regions-of-interest (ROI).

3.3 Evolution from 2-Dimension to 3-Dimension

In general, conventional CMOS image sensors are fabricated on a single-layer wafer. The implementation of the feature-extraction function requires complex processing circuits, which leads to a larger pixel size with a lower fill factor in every pixel. Moreover, the post image processing algorithm aggregates this limitation when it is integrated beside the pixel array on the image sensor. Fortunately, an emerging 3D integrated-circuit (3DIC) technology provides an appealing approach to alleviate the conflict between the loss of fill factor and the implementation of feature-extraction functionality. In this technology, multiple silicon-on-insulator (SOI) wafers are vertically stacked and interconnected together using 3D inter-tier vias, and it enables the image sensor to implement light sensing and feature extraction onto sperate tiers. Since the top tier is fully covered by photodiodes, the 3D image sensor is superior in terms of its sensing performance and maintains an extreme fill factor even when it implements the feature-extraction functionalities.

3.4 3D Integrated Feature-Extraction Image Sensor

In this section, a smart feature-extraction image sensor developed using novel 3D integrated-circuit (3DIC) technology is proposed. This smart image sensor has two major advantages when compared to the motion-detection image sensor reported in last section: Firstly, a significant improvement from 32% to 97% on the fill factor is achieved; In addition to the motion-detection function, a contrast-extraction capability was developed in this 3D image sensor, which makes it more appealing for computer vision applications.

3.4.1 Sensor Architecture

Fig. 3.20 shows the block diagram of the 3D feature-extraction image sensor. The main building blocks include a 64 $\times$ 96 pixel array, row and column scanners, an NMOS resistor network and a global analog buffer. Every pixel is equipped with an analog memory to store the intensity voltage from the photodiode. After an exposure period, row and column scanners sequentially access the pixel array and export the intensity voltages on photodiodes and capacitors to the global buffer. There are two operating modes for this image sensor: "***motion mode***" and "***contrast mode***". In the motion-detection mode, the embedded capacitor works as an analog memory to store the previous frame image. This image sensor exports both the current and previous frame images to the post processor simultaneously, which compares their difference so as to detect visual motions in the viewing field. In the contrast-extraction mode, the image sensor firstly exports the intensity image on photodiodes in every pixel and stores them in capacitors on the pixel array. The raw image stored on the capacitors is smoothed by a low pass filtering function through the activation of the NMOS resistor network for a certain period. The image sensor then exports the smoothed image to the external processor. The spatial contour in the viewing scene is extracted by computing the difference between the raw and blurred images.

3.4.2 Pixel Structure

The circuit diagram of the smart pixel and the global readout path is shown in Fig. 3.21. The proposed pixel consists of a photodiode, a memory device (capacitor), an in-pixel voltage buffer, three operational switches (SWA, SWB and SWC), and four PMOS transistors. A series of PMOS reset transistors allow the photodiode to be initialized into the power supply voltage (Vdd), and a wider intensity voltage swing is achieved accordingly. Since there are both row and column accessing transistors in every pixel,

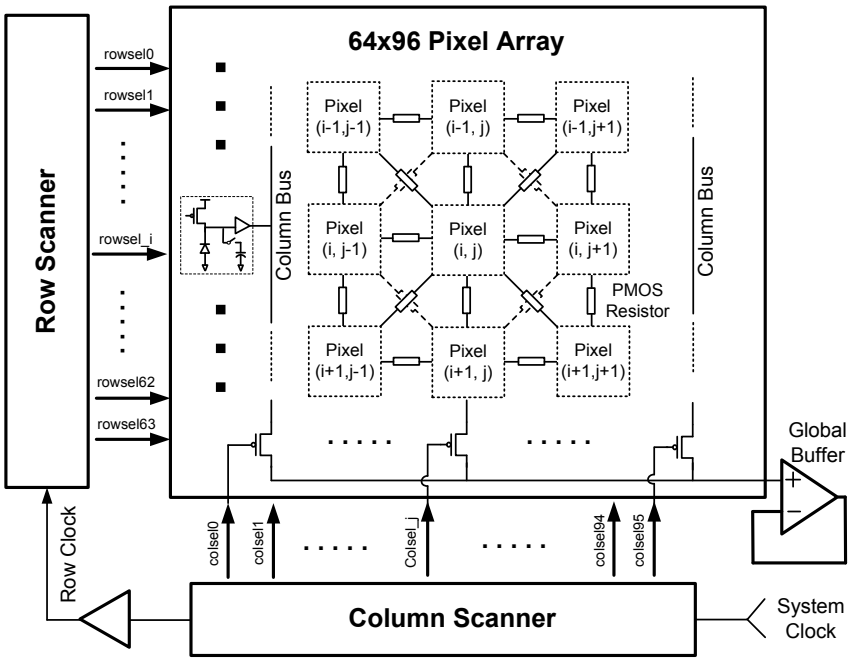


Figure 3.20: System architecture of the 3D integrated feature-extraction image sensor.

all pixels in this image sensor can be addressed individually for reset and readout, instead of a common row wise pixel operation used in conventional APS-based image sensors. The analog memory is implemented by a MOS transistor. The design of the MOS-based capacitor is the same as the one discussed in last section. In this 3D SOI process, the threshold voltage for the NMOS transistor is about 350 mV, and the minimum frame rate of the image sensor is designed at 40 frames/s. The external 8-bit ADC has a resolution of 5.85 mV in the 1500 mV reference. Therefore, the capacitor should be capable of holding the threshold voltage with a variation less than 5.8 mV for a period of 25 mS. Fig. 3.22 shows the transient voltage on the capacitor after it samples the threshold voltage at 10 ms. The voltage variation is only 5 mV within a period of 25 ms. The physical dimension for this MOS-based capacitor is 9.8 μm $\times$ 9.5 μm with an equivalent capacitance of 390 fF. The voltage jump at the sampling moment is caused by the charge injection from the switching transistor.

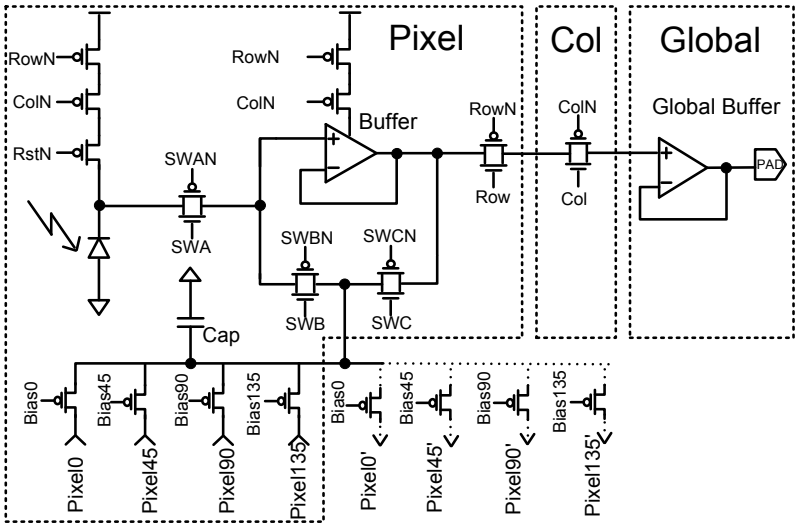


Figure 3.21: Schematic diagram of the signal path from the selected pixel to the global analog buffer in the 3D integrated feature-extraction image sensor.

There is an in-pixel buffer driving the intensity voltage onto the column bus. This buffer is developed based on a common 5-transistor amplifier structure. The parasitic capacitance on the column bus is about 320 fF (64 $\times$ 5 fF). The desired maximum operating speed for this image sensor is $\sim$160 frames/s in a resolution of 64 $\times$ 96, which indicates that the maximum pixel clock frequency is about 980 KHz. In order to stabilize the intensity voltage within the 1/4 pixel clock cycle, the unity-gain-bandwidth (UGB) for this amplifier should be over 5 MHz. Fig. 3.23 and Fig. 3.24 show circuit simulation results on this amplifier in Spectre. It shows that the gain factor for this amplifier is about 50 dB and the unity-gain-bandwidth (UGB) is $\sim$ 5.1 MHz with a phase margin of over 92^o. Moreover, the power consumption of this amplifier is less than 1.8 μW under a power supply voltage of 1.5 V. In order to further reduce power consumption, an enabling path is embedded between the power supply and the amplifier, and it is activated only when the pixel is accessed for readout.

As shown in the system diagram, every pixel connects to its surrounding neighbors through eight PMOS transistors so as to implement the resistor network. Since each

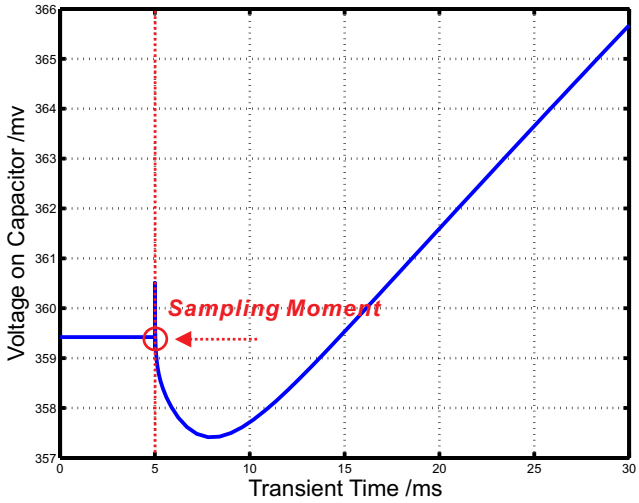


Figure 3.22: Transient simulation on the voltage variation of the MOS-based capacitor along with time.

PMOS transistor is shared by two adjacent pixels, four PMOS transistors are assigned into every pixel. The biasing voltages on the gates of these transistors are individually controlled by the external DACs. Three complementary switches (SWA, SWB and SWC) are designed to configure the readout path inside every pixel. They are activated in specific sequences to implement feature-extraction functions in this image sensor.

3.4.3 Analog Buffer

The global unity gain buffer is implemented by an operational amplifier, as shown in Fig. 3.25. The maximum speed for this image sensor is designed at 160 frames/s with a spatial resolution of 64 $\times$ 96. Hence, the maximum pixel clock speed is about 1 MHz. In order to stabilize the analog output within the 1/4 pixel clock cycle, the unity-gain-bandwidth (UGB) for this operational amplifier should be over 5 MHz. As shown in Fig. 3.26 and Fig. 3.27, this amplifier has a gain factor around 90 dB with a phase margin more than 100^o. The unity-gain-bandwidth (UGB) is about 5.5 MHz in

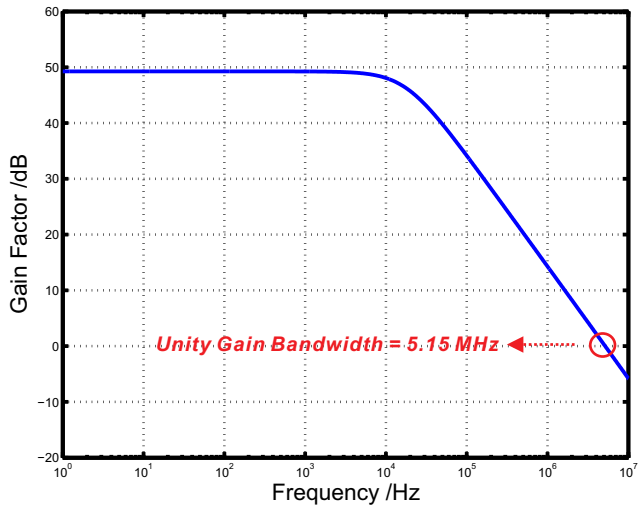


Figure 3.23: Frequency response on the gain factor of the in-pixel amplifier.

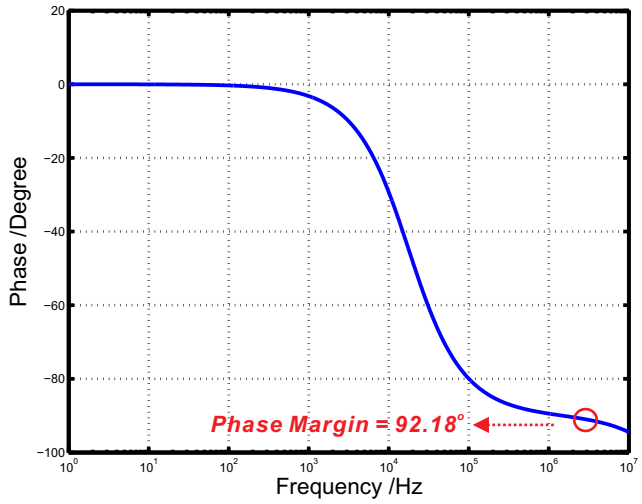


Figure 3.24: Frequency response on the phase shift of the in-pixel amplifier.

a 25 pF loading condition. The total power consumption for this amplifier is about 0.4 mW under a power supply voltage of 1.5 V.

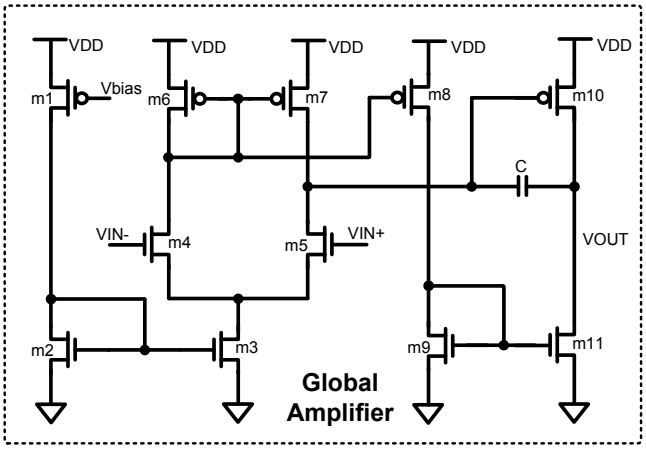


Figure 3.25: Circuit diagram of the operational amplifier in the global analog buffer.

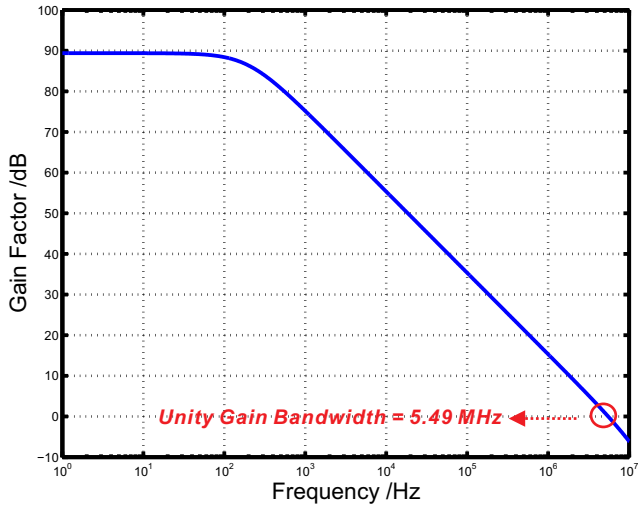


Figure 3.26: Frequency response on the gain factor of the global amplifier.

3.4.4 3D Integration

In order to take full advantages of the 3D integration technology, the proposed smart pixel is separated into three discrete blocks, which are fabricated in three individual SOI tiers as sketched in Fig. 3.28. The top layer is fully covered by photodiodes. To optimize the photon sensitivity and noise performance, the layout of the photodetector is implemented in a specific manner. For better sensitivity, two vertical PN junctions

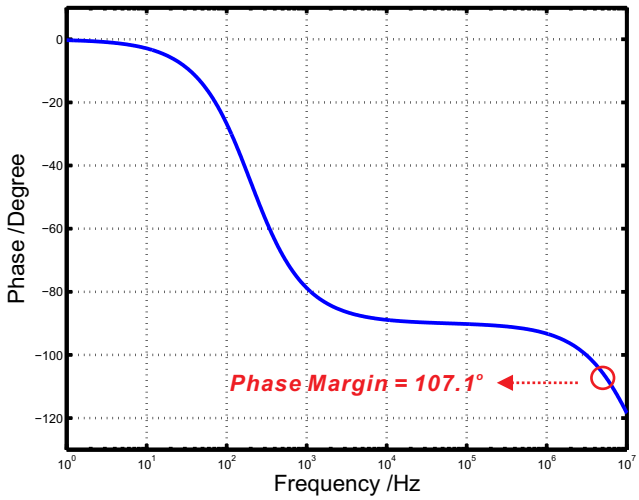


Figure 3.27: Frequency response on the phase shift of the global amplifier.

are combined together to implement a single photodiode, with their N-type regions connected through metal wires. In order to reduce the leakage current between the N-type and P-type regions, the photodiode is designed in a rectangle shape and shielded by a poly silicon layer. This design allows a thin gate oxide to be deposited on the photodiode, which further suppresses the leakage current. The middle tier contains the reset circuit, the in-pixel analog buffer and the operational switches. The resistor network and the analog memory reside in bottom layer. The connection between each individual block is accomplished by the spatial 3D inter-tier vias, shown as the yellow color bar in Fig. 3.28. As shown in Fig. 3.29, the top tier is flipped over during fabrication [15]. Hence, photodiodes in this image sensor are back illuminated. Since no metal wire or contact resides in the photon sensing area, this 3D image sensor features an extreme fill factor of 97%.

3.4.5 Motion Detection

In this image sensor, a temporal difference algorithm is employed to detect visual motions. As shown in Fig. 3.21, each pixel is embedded with a MOS-based capacitor as

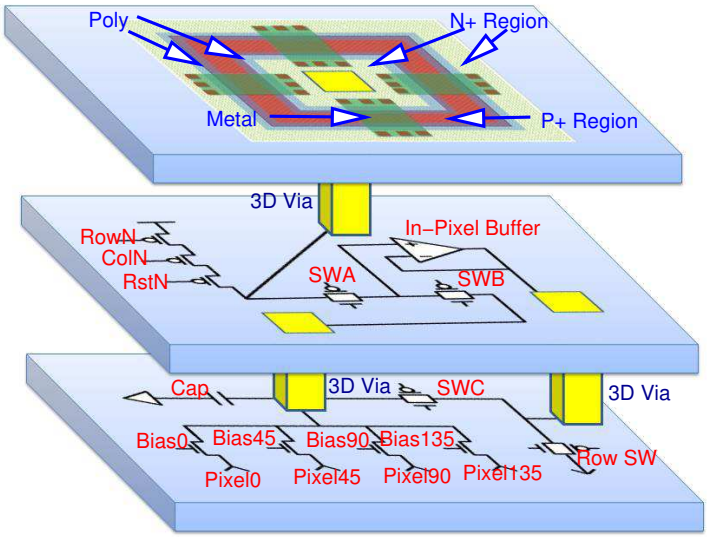


Figure 3.28: 3D integration of the smart pixel in the feature-extraction image sensor.

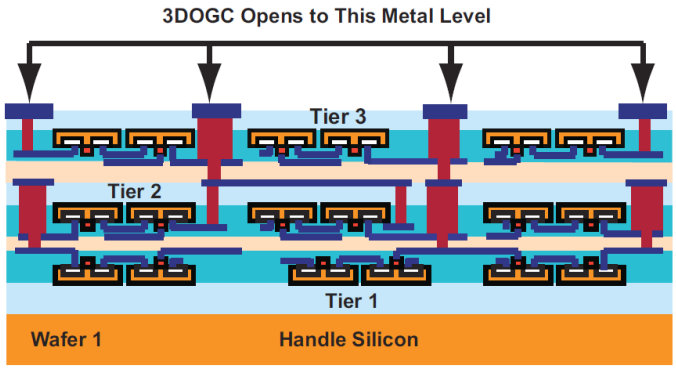


Figure 3.29: Cross-section view of the three-tier assembly in 3D SOI process [15].

an analog memory to store previous frame information. This image sensor works in a rolling shutter scheme, in which every pixel is periodically accessed by row and column scanners. The addressed pixel turns on the switch SWB and disables both switches SWA and SWC so as to report the previous frame intensity voltage from the in-pixel capacitor to the external processor. Switch SWB is then disabled, and switches SWA and SWC are turned on simultaneously. The newly-integrated intensity voltage on the

photodiode for the current frame is exported to the global buffer, and the capacitor is also updated in parallel. Therefore, two analog intensity voltages are presented in sequence to the post processor within a complete pixel readout cycle. Their difference is computed and further converted into a binary event by the thresholding operation. After the readout procedure, the pixel is reset into the power supply voltage, and a new integration phase starts capturing another frame image.

3.4.6 Contour Extraction

Visual contrast resides in regions involving abrupt changes of light intensity. The Laplacian-of-Gaussian (LOG) operation is usually applied on an intensity image in computer vision to extract spatial contour. Gaussian convolution in software is a time-consuming task because of complex computations involving numerous multiplications. Fortunately, the LOG function can be closely approximated by the difference-of-Gaussian (DOG) operation. This image sensor implements the Gaussian filtering function on-chip using a resistor and capacitor network so that it directly exports both the raw intensity image and the Gaussian smoothed one. Hence, the post processor extracts the spatial contours by computing their difference.

As shown in Fig. 3.20, every pixel is cross-connected to its surrounding neighbors through a resistor network. Such network is implemented with PMOS transistors, as shown in Fig. 3.21. Each PMOS transistor connects the capacitors of two neighboring pixels. The voltage gap between the source and drain of the PMOS transistor is so small that the transistor works in the linear region as a MOS-based resistor when its gate end is properly biased. The equivalent resistance of this transistor in relation to the gate voltage can be modeled as $\mathrm{Eq.\ 3.5}$.

$$R_{eq} = \frac{1}{\mu_p C_{ox}(V_{GS} - V_{th})} \left(\frac{L}{W}\right) \qquad \text{(Eq. 3.5)}$$

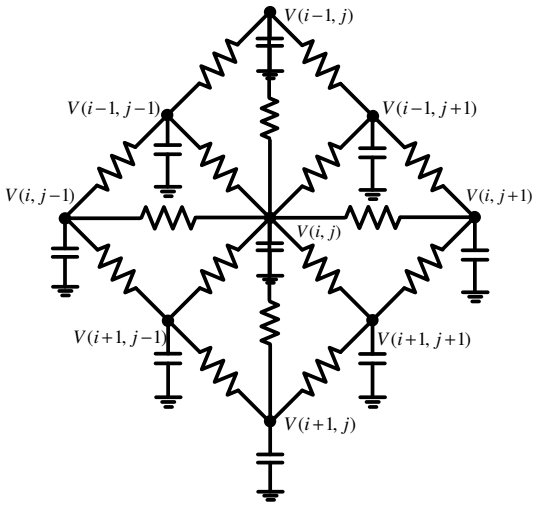


Figure 3.30: Equivalent resistor and capacitor network in the 3D integrated feature-extraction image sensor.

where μ_p and C_{ox} are the process-dependant parameters. The physical size of the transistor (L and W) is fixed after fabrication. Theoretically, a variable resistor is achieved by regulating the gate biasing voltage V_{th}, and the entire pixel array then becomes a programmable resistor capacitor network as shown in Fig. 3.30.

When the resistor network is turned on, the stored intensity charges on the capacitors are redistributed on the pixel array. This resistor and capacitor network can be treated as a resistor-capacitor (RC) transmission line, which implements a low pass filtering operation on the voltages stored on the capacitors. In both horizontal and vertical directions, the resistor-capacitor (RC) network diffuses the image stored on the capacitors with a Gaussian filter, which is modeled as the following equation.

$$V(x, y, t) = V(x, y, 0) \otimes G\left(x, y, \sqrt{\frac{t}{RC}}\right) \qquad \text{(Eq. 3.6)}$$

where

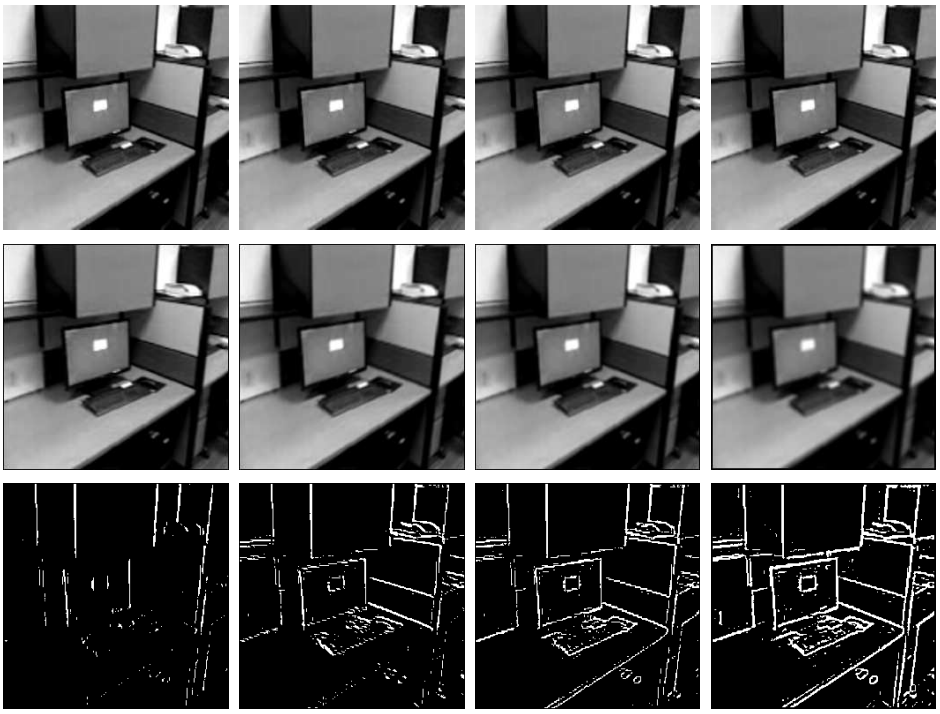
$$G\left(x, y, \sigma\right) = \frac{1}{\sigma} \exp\left(\frac{-(x^2 + y^2)}{4\sigma^2}\right) \qquad \text{(Eq. 3.7)}$$

It is shown that the intensity voltage in a given pixel at a certain moment $V(x, y, t)$ represents the convolution of the initial voltage $V(x, y, 0)$ with a Gaussian filter $G(x, y, \sigma)$ whose width is proportional to the square root of the duration time t. One of the critical parameters in this Gaussian diffusion is the variable factor σ, which is controlled by the resistance R and capacitance C as well as the time duration t. Therefore, with proper settings, the raw intensity image is convoluted with a Gaussian function and diffused to a certain extent as desired. The distinction between the raw image and the smoothed one indicates the spatial contour in the viewing field.

In comparison to the motion-detection mode, the contrast-extraction mode differs in the timing sequence. After the reset and exposure procedures, this image sensor sequentially exports the intensity voltages on the pixel array to the external processor. When a pixel is addressed for readout, it turns on switches SWA, and SWC simultaneously, and thus it exports the intensity voltage and updates the capacitor at the same time. The resistor network is activated for the Gaussian filtering once the entire frame readout is completed. The pixel array is then scanned again to export the diffused intensity voltages on the capacitors by sequentially turning on the switch SWB in every pixel. Since four PMOS resistors are separately controlled by external DACs, this image sensor can execute the Gaussian filtering in four specific orientations to extract spatial contours.

Fig. 3.31 shows the simulation results on the contrast-extraction function in different diffusion settings. The sample images from top to bottom correspond to the original image, the diffused one and the spatial contour after thresholding. In column (a), only the horizontal resistor network is activated, while the other directions are switched off. Therefore, the raw intensity image is horizontally smoothed, and it extracts only vertical contours. Both the horizontal and vertical contrasts are detected through the diagonal diffusion, as shown in column (b). Columns (c) and (d) correspond to the

full orientation configuration when the entire resistor network is activated simultaneously. The extracted edges in column (d) are thicker than those in column (c) because of the longer diffusion period.



(a) (b) (c) (d)

Figure 3.31: Demonstration of spatial contour extraction. Samples from top to bottom correspond to the original intensity images, the diffused ones and the binary spatial contrast.

3.4.7 VLSI Implementation

This 3D feature-extration image sensor was fabricated using MITLL 3D 0.18 μm CMOS FDSOI process. Every wafer tier contains one poly and three metal layers. Fig. 3.32 shows the microphotograph of the top tier with the 64 $\times$ 96 pixel array and the testing structure highlighted. The sensor has a total die size of 1.45 mm $\times$ 1.5 mm including the I/O pads. Fig. 3.33 shows the photon generated current in relation to the illumination light intensity. The horizontal axis represents the light intensity, while the vertical

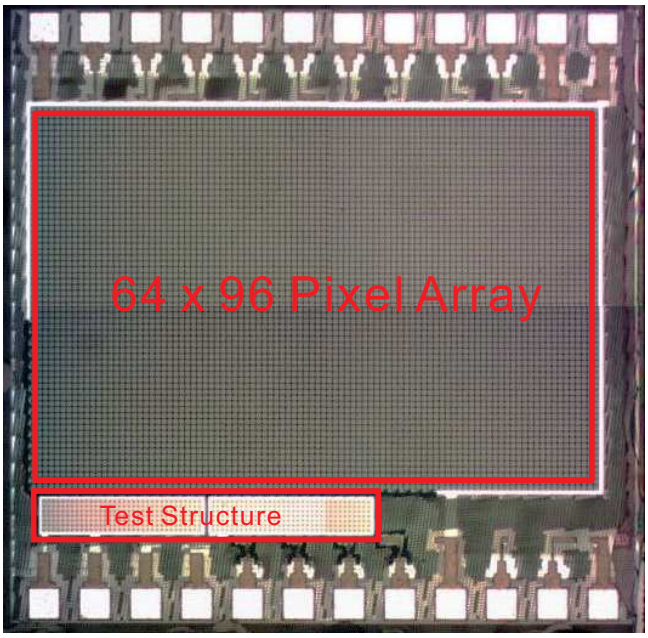


Figure 3.32: Microphotograph of the 3D integrated feature-extraction image sensor with the main building blocks highlighted.

axis corresponds to the current measured from the photodiode. It reveals that the photon current has a linear response to the incident light brightness, which is modeled as below.

$$I_p = 3.5 \times 10^{-14} \times I_{\text{int}} \qquad \text{(Eq. 3.8)}$$

where I_p denotes the photon current in Amperes and I_{int} is the illumination intensity in Lux. The biasing voltage is uncorrelated to the photon sensitivity of the photodiode.

3.4.8 Experimental Results

In order to characterize the image sensor, a testing platform based on the Opal Kelly XEM 3010 board was developed. As shown in Fig. 3.34, there is a Xilinx Spartan-3 FPGA chip (XC3S1500), a Micron 32 MBytes SDRAM and a high speed USB 2.0 interface on this board. The major function of this development board is to operate

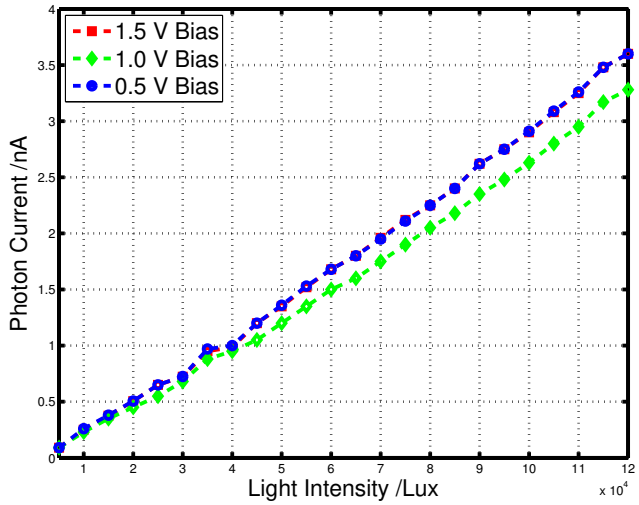


Figure 3.33: Relationship between the measured photo current and the light illumination intensity.

the image sensor via the control signals including system clock, address signals and biasing voltages. The image sensor is soldered on a customized printed-circuit-board (PCB) which implements a 8-bit ADC (AD7277) and four 10-bit DACs (AD7398). Therefore, the analog intensity voltage exported from this image sensor is directly converted into the digital format. Moreover, in the contrast-extraction mode, this image sensor diffuses the stored image with a low pass Gaussian filter in the selected orientation by regulating the biasing voltages on the resistor network via dedicated DACs. The digital image is directly reported to the FPGA chip for motion detection and contrast extraction. An 64×96 resolution intensity image under a 8-bit gray scale only occupies 49.2 Kb storage space. Therefore, the SDRAM chip on the FPGA board can store up to 5208 frames of intensity images. The data transmission bandwidth for the USB interface is over 40 Mbit/s. Hence, the testing platform can ideally transmit over 800 frames of intensity images to the host computer in one second. In practice, the working speed of the testing platform is limited by the frame rate of the feature-extraction image sensor. A graphic-user-interface (GUI) was developed on the computer side so

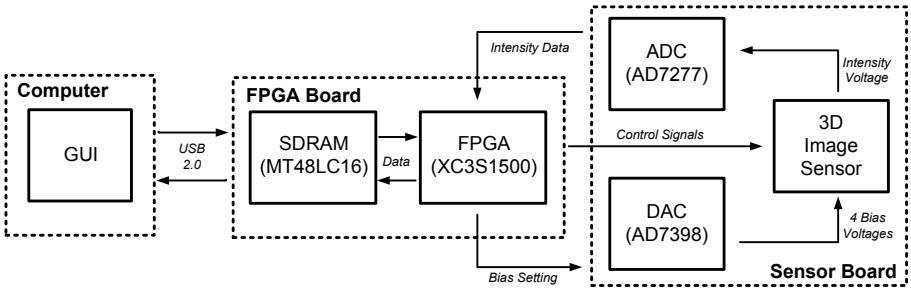


Figure 3.34: Block diagram of the testing platform for the 3D integrated feature detection image sensor.

as to display images sent from the FPGA board and regulate operational parameters for the image sensor whenever necessary.

The main parameters of the image sensor are summarized in Table. 3.2. The pixel of the image sensor occupies a silicon area of 14 $\times$ 14 μm^2 with a fill factor of 97%. Every pixel contains 22 transistors, and the maximum frame rate of the image sensor is 160 frames/s. The total power consumption for this image sensor is about 0.8 mW under a power supply voltage of 1.5 V. The main power consumption resides in the global analog buffer because of the direct driving on the analog pad. The fixed-pattern-noise (FPN) for this image sensor is 3.5% with the dark current of 2.6 fA.

Table 3.2: Chip characteristics of the 3D feature-extraction image sensor

Process Technology	MITLL 0.18 μm 3D FDSOI process
Die Size	1.45 $\times$ 1.5 mm^2
Array Size	64 $\times$ 96
Pixel Size	14 $\times$ 14 μm^2
Pixel Complexity	22 transistors
Fill Factor	97%
Fixed Pattern Noise	3.5%
Dark Current	$\sim$12.6 fA
Frame Rate	160 fps
Supply Voltage	1.5 v
Power Consumption	0.8 mW

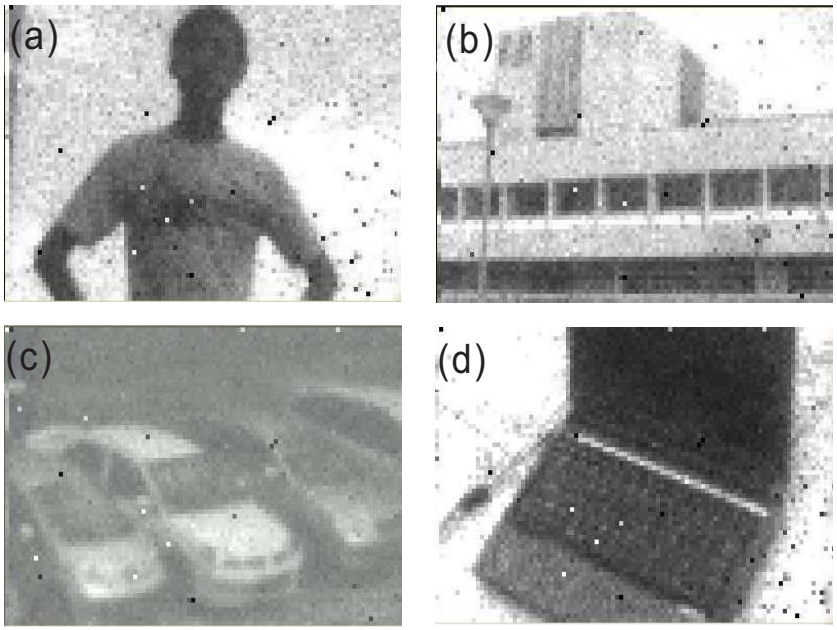


Figure 3.35: Sample images captured from the 3D integrated feature-extraction image sensor. Example photos taken on a human body (a), a building (b), a carpark (c) and a laptop (d).

Fig. 3.35 shows the sample images captured by this 3D feature-extraction image sensor. The pictures were taken on a human body (a), a building (b), a carpark (c) and a laptop (d). Due to the low resolution, the sample images look fuzzy when compared to those captured by the high resolution commercial cameras. There are many discrete and fixed “hot pixels” on the sample images, which are induced by the defects during fabrication and packaging. Additionally, the 3D feature-extraction image sensor also suffers from severe fixed-pattern-noise (FPN). Testing difficulties were encountered on motion detection and contour extraction due to the malfunctions of the in-pixel capacitors. Neither motion images nor contour pictures were available from this image sensor at present.

3.5 Summary and Discussion

As discussed at the beginning of this chapter, the temporal difference algorithm has the least complexity in computation, which makes it feasible for implementation in the pixel array. This chapter reports the development of two smart image sensors based on this algorithm for motion-detection applications. Table. 3.3 summarizes and compares the main characteristics of the motion-detection image sensors reported in this chapter and in the literature.

The image sensors listed in Table. 3.3 have almost similar spatial resolutions. However, image sensors reported in this chapter are more compact in terms of the pixel size with higher fill factors. Although the 3D feature-extraction image sensor suffers from serious fixed-pattern-noise (FPN) due to its process limitation, it has great potential for the implementation of smart image sensors. In neuromorphic engineering, many parallel processing algorithms require complex resistor and capacitor networks with massive connections on the pixel array when they are implemented in analog VLSI circuits. The 3D integration technology is an optimal solution for these applications, as it elegantly solves the conflict between the increase of the circuit complexity and the decrease of the fill factor. Also, the vertical stacking structure in the 3D process enables designers to implement cross-connections between pixels more freely, when compared to the conventional CMOS process. The design of the 3D feature-extraction image sensor is an important attempt to implement this idea.

The major limitation of the smart image sensors reported in this chapter is that they extract only low-level features, including temporal motions and spatial contours. In order to extract high-level features for advanced computer vision tasks, such as pattern recognition and object tracking, post vision processors have to implement complex processing on low-level features, which consumes tremendous system resources. It is

Table 3.3: Comparison of temporal difference motion-detection image sensors

	Kim [98] '09	Kim [73] '13	image senor (1)	image sensor (2)
Technology	0.5 μm 2P 3M	0.35 μm 2P 4M	0.18 μm 1P 6M	0.18 μm 3D 1P 3M
Array Size	64 $\times$ 64	128 $\times$ 128	64 $\times$ 64	64 $\times$ 96
Pixel Size	29 $\times$ 28 μm^2	16 $\times$ 21 μm^2	14 $\times$ 14 μm^2	14 $\times$ 14 μm^2
Fill Factor	23%	42%	32%	97%
FPN	0.4%	0.28%	0.3%	3.5%
Max Frame Rate	30 fps	30 fps	100 fps	160 fps
Supply Voltage	3V	3V	1.8V	1.5 V
Power Consumption	1.02 mW	1.06 mW	0.4 mW	0.8 mW

an elegant solution to integrate motion detection and motion processing on the same chip so that smart image sensors can export some customized futures for specific application so as to simplify and accelerate post processors.

Chapter 4

Event-Clustering Motion-Detection Image Sensor

This chapter describes an event-based motion-detection image sensor for visual object tracking applications. The proposed smart image sensor is a visual system that integrates both motion detection and event processing on the same chip. It not only detects visual motion in the viewing field as binary events, but also processes them in parallel to create specific event-flow features by an event-clustering algorithm. The smart image sensor also adopts a hybrid readout strategy to export motion events so that it is compatible with both frame-based and event-based image processing algorithms. System-level simulations under various scenarios indicate that the proposed image sensor is effective for object tracking applications. This chapter is organized as follows: firstly, the existing frame-based and event-based image acquisition and processing strategies are investigated; the event-clustering algorithm in the proposed smart image sensor is then analyzed in detail, followed by a description on system-level simulations. The conclusion of this chapter compares the proposed smart image sensor with the others reported in the literature.

4.1 Parallel Event Processing

4.1.1 Frame-Based Image Acquisition and Processing

In conventional active-pixel-sensor (APS) based image sensors, an intensity image $I(t)$ can only be acquired by an exposure operation within an integration period T. Hence, the image sequence I_{seq} reported from a vision sensor can be modeled as $\mathrm{Eq.\ 4.1}$, which is precisely synchronized with artificial timing signals. In a post vision system, the video sequence I_{seq} is consecutively processed by its frame order. Such image acquisition and processing methodology is termed as the "frame-based" mechanism.

$$I_{seq} = \{I(t_0),\ I(t_0 + T),\ ...\ I(t_0 + N \times T)\} \qquad (\mathrm{Eq.\ 4.1})$$

This frame-based strategy has been the standard practice since the invention of digital cameras in 1970s. However, it suffers from a variety of fundamental limitations, listed as below.

- ***Image Redundancy*** In practice, there is a huge amount of static background information in the viewing field, which takes up most of the space in the acquired image. The redundant background is repeatedly reported in the video sequence acquired from an image sensor [99]. Furthermore, the data redundancy issue is aggravated with the increase in image acquisition speed. Despite the fact that background information can be completely removed by the frame-based feature-extraction image sensors reported in last chapter, the redundant pixels projected to the background are still periodically accessed by image sensors, and post processing on them leads to massive computation and memory consumption.

- ***Response Latency*** In a frame-based image sensor, every pixel is periodically accessed in a constant sequence to export the entire frame of an intensity image.

When a given pixel detects motion activity, the output access is not immediately granted to this pixel until it is addressed for readout. Under the frame-based mechanism, readout sequence on the pixel array is irrelevant to the imaging content, and the readout channel is mostly occupied by redundant pixels, which causes time delays in sensor's response to active pixels with higher priority [12]. Therefore, the frame-based strategy is unsuitable for high-speed vision applications.

- ***Iteration Complexity*** In frame-based image processing, all pixels in every frame from a video sequence have to be iteratively processed. During this procedure, massive operations including multiplication and convolution are performed on intensity images [100]. This time consuming process severely limits the operating speed of a computer vision system, and the increase of spatial resolution in image sensors significantly aggravates this problem. Hence, repetitive operations in frame-based image processing are inefficient and waste tremendous computational resources.

In comparison with frame-based image sensors, vertebrate eyes work more efficiently by parallel processing sensed images on retinas. In biological vision systems, thousands of retinal neurons work independently in a continuous mode to extract basic image features, such as temporal motions or spatial contrasts. These features are encoded into a series of neural spikes that are instantly relayed to the brain via neural networks. Since fundamental image features are extracted on retinas, post vision processing in the brain is sped up accordingly. Therefore, the parallel processing mechanism in biological visual systems provides an elegant solution to the bottlenecks in conventional frame-based vision systems. However, it is a challenging work to implement full parallel image processing on image sensors due to its complexity.

4.1.2 Event-Based Image Acquisition and Processing

Neuromorphic engineering is an emerging discipline involved in developing very-large-scale-integration (VLSI) circuits to mimic biological nervous systems. Inspired by vertebrate eyes, researchers invented smart silicon retinas by implementing parallel image processing in every pixel to generate a stream of pseudo neuron spikes termed as "events", in order to encode image features extracted from the focal plane. The event sequence E_{seq} from smart silicon retinas is modeled as $\mathrm{Eq.}\ 4.2$. It shows that every event is labelled with its timing and positional information.

$$E_{seq} = \{E(t_0, addr_0),\ E(t_1, addr_1),\ ...\ E(t_n, addr_n)\} \qquad (\mathrm{Eq.}\ 4.2)$$

In general, communication between neuromorphic chips is achieved by a common address-event-representation (AER) protocol. Artificial retinas that export address events are termed as event-based image sensors, which are superior to their frame-based counterparts, and their advantages are listed below.

- ***Low Redundancy*** The AER communication protocol solely exports the active pixels that detect feature events, and discards the redundant pixels without triggering events. This strategy significantly improves readout efficiency, and reduces bandwidth requirement in post vision systems [12].

- ***High Speed*** Unlike conventional frame-based image sensors, no pixel clock is embedded in event-based silicon retinas. The communication rate of address events only depends on the handshaking speed of digital interfaces. Therefore, smart silicon retinas are quite appealing for real-time applications [12].

Despite the advantages outlined above in event-based image sensors, vision processing of address events is still a challenge because of their asynchronous nature.

During the readout procedure, the AER communication module randomly accesses the pixel array, and thus there is no spatial and temporal correlation within address events. In post vision processing, it is common to assign address events based on a constant time slice to build pseudo frames. However, this approach presents the high possibility that those events fired from the same scene at the same moment are distributed into separate frames, and there are inevitable confusions or errors found in subsequent processing stages. The time slice to rebuild pseudo frames must be adapted to the motion activities detected from image sensors [26]. Hence, full event-based vision processing has become an important research topic in recent years.

4.2 Object Tracking in Computer Vision

In computer vision processing, the objective of object tracking is to localize and associate the target of interest in a video sequence over time. It has a variety of applications including human computer interaction, security surveillance, and video compression. Existing object tracking algorithms are dominated by conventional frame-based approaches, which continuously analyze every frame from a video sequence to track objects. Frame-based visual object tracking is time consuming because of massive computations on huge amounts of image data. In recent years, researchers have developed a number of event-based object tracking algorithms by using smart silicon retinas. However, these event-based approaches are not compatible with their frame-based counterparts due to the asynchronous nature of event encoding and communication.

4.2.1 Frame-Based Tracking Algorithms

There are two fundamental procedures involved in the frame-based object tracking algorithm: feature representation and tracking computation. In the first step, an object

of interest is represented by its unique features, so that it can be distinguished in the feature space. There are several common features wildly used in object tracking algorithms, which are listed below.

- ***Feature Points*** An object is simply described by a set of feature points like the centroid and corners [101]. This feature representation is efficient in tracking small objects. However, it fails to track objects in complex environments such as non-rigid objects and collision scenarios.

- ***Color Intensity*** The color histogram for an object is almost consistent along with the frames in a video sequence, and it is insensitive to the size and shape of the object [102]. However, the accuracy of the color intensity model highly depends on illumination variations in the viewing scene.

- ***Spatial Contour*** In general, there are abrupt intensity changes on the boundaries of an object. Spatial contours are easily extracted by specific edge-detection algorithms [103]. Moreover, there is inherent immunity to illumination changes for spatial contour features, and they are widely used in object tracking applications.

The second step involves the association of potential objects with specific features in a video sequence by object tracking algorithms. Several popular frame-based tracking algorithms in computer vision are briefly explained below.

- ***Point Tracking*** Since an object can be represented by a set of feature points, the most direct method to track an object is the association of its feature points in a video sequence. Point-based object tracking algorithms can be divided into two categories: the deterministic approach and the statistic method. The most famous methodology among them is the Kalman filter algorithm [104], which tracks objects by predicting their positions and speeds based on their current

states. Due to its simplicity, the point-based method only tracks small objects travelling under constant speed or acceleration.

- ***Kernel Tracking*** Appearance-based kernel tracking algorithms are widely used because of their superior efficiency and accuracy [105]. There are two common approaches in kernel-based tracking algorithms: template matching and mean shift. Template matching method searches the incoming image exhaustively to find a region that is the most similar to the object template defined in advance. Instead of brute search in the template matching method, mean-shift tracker computes similarities between the target object and potential regions on their intensity histograms [106].

- ***Silhouette Tracking*** Silhouette-based trackers find the object region in every frame by using the shape model on the tracked object [107]. Silhouette trackers can be divided into two categories: shape matching and contour tracking. In the shape matching method, an object shape template is searched within the entire frame to find the region most similar to it. In contrast to the silhouette matching method, contour tracking is performed by evolving the object contour in previous frames to a new position in the current frame. The main advantage of silhouette tracking is its robustness to the shape variation in the tracked object.

In frame-based object tracking algorithms, every pixel in each frame from a video sequence has to be iteratively accessed and processed. Therefore, the increase in the frame rate and spatial resolution of image sensors significantly aggravates the computation load in post vision processors. As discussed in last section, event-based feature-extraction image sensors completely discard redundant pixels, and they only export the active pixels with feature events. Hence, image processing based on address events from event-based image sensors is more efficient. In recent years, a

variety of event-based object tracking algorithms have been proposed, and they show comparable performance to their frame-based counterparts.

4.2.2 Event-Based Tracking Algorithms

Among all feature-extraction smart silicon sensors reported in the literature, a dynamic-vision-sensor (DVS) is the most outstanding one to detect visual motions on its focal plane [12]. In this biomimetic imaging device, relative intensity changes are quantized as a stream of asynchronous address events to identify scene reflectance changes. Since address events directly encode the object movement in the viewing field, the DVS sensor is quite appealing for object tracking applications, and it is adopted as a front-end motion detector in many computer vision applications.

In recent years, based on the DVS sensor, researchers proposed a number of event-based visual object tracking systems. A real-time particle tracking system by using an event-based DVS was proposed in [108]. The tracking algorithm is implemented by grouping motion events based on their spatial and temporal correlations. Another event-based Hough transform algorithm is developed to track micro-spheres in fluid mechanic applications [82]. However, this approach requires a tremendous amount of computation to extract limited features such as lines or circles. An event-driven kernel-based tracking framework is proposed in [81, 83], in which multiple targets in specific shapes can be tracked in parallel. However, this algorithm requires tuning a number of parameters to extract shape features. An event-driven vision system that integrates motion detection, event processing, and pattern recognition is proposed in [27]. This system is capable of recognizing and tracking a rotating dot in a certain size. However, it requires massive hardware components and extreme setup costs, which prevents this system from being used in practical applications.

Raw address events from a DVS sensor are only granted two basic features indicating their firing position and timing information. In the event-based object tracking systems outlined above, it is compulsory to preprocess address events so as to extract specific features such as contours or shapes for subsequential processing procedures. This process involves multiple operations including segmentation, repositioning and convolution, which increases the computational load in post processors and decreases the operating speed of the vision system. In order to simplify post vision processing, it is feasible to integrate some event processing algorithms on the focal plane of silicon retinas so that they can directly export customized features for specific applications, rather than solely reporting raw address events.

4.3 Event-Clustering Image Sensor

In this chapter, a smart feature-extraction image sensor that integrates motion detection and event processing on the same chip is proposed. The image sensor exports address motion events and special clustering features simultaneously. The main contribution of this image sensor is the hardware implementation of an event-clustering algorithm to on-chip process motion events. The event-clustering algorithm is customized for object tracking applications, and its output is termed as an “event-flow” feature. Simulation results show that event-flow features can significantly simplify post image processing in visual object tracking systems.

In this image sensor, motion-detection and event processing are implemented by separate circuits, and they work independently and continuously in parallel. Once motion events are generated by the motion-detection circuit in every pixel, they are immediately processed by the event-clustering algorithm. The transient response from the event-clustering algorithm represents the spatial density and temporal frequency of motion events on the focal plane, and it is defined as the “event-flow” feature, which

is presented as an analog voltage on every pixel. Hence, those inactive objects without motion activities are completely discarded by the proposed image sensor. During the readout procedure, only pixels with sufficient event-flow features can be exported to an external processor via a customized hybrid AER circuit. Since the data interface of the proposed image sensor is developed using a universal AER protocol, existing event-based image processing algorithms can directly adopt this sensor as a motion detector. Furthermore, there is an artificial frame synchronization signal embedded in the output event flow, and the proposed sensor is also compatible with conventional frame-based algorithms.

4.3.1 Clustering Algorithm

The event-clustering algorithm is inspired by the behavior of fish in water. When there is a fish swimming beneath the water surface, ripples are created and propagated from near to far. The sites closer to the fish always display larger waves than those further away. The ripples possess a number of inherent characteristics, such as spatial size and peak position, which are valuable features for object tracking applications. For example, in order to roughly track a swimming fish, observers can simply search and follow the peak point in ripples, as it is extremely close to the swimming fish in space.

Similarly, the event-clustering algorithm in the proposed image sensor is designed to create artificial ripples of event-flow features on the focal plane. The smart pixel array in the image sensor detects motion activities and converts them into binary events. Simultaneously, in the event-clustering algorithm, every event generates a transient spike on its firing position, and every spike covers a certain area in the pixel array. When multiple events fire together in the object region, a number of mountain-shaped responses on the event-flow feature are created accordingly. These mountains correspond to the artificial ripples of event-flow features on the focal plane, and they are important features for object tracking applications.

Recently, in the pattern recognition field, researchers proposed a similar idea to process motion events in [109]. In the proposed paper, a time-surface feature is built by providing a spatiotemporal context around every event. Simulation results based on different databases prove that the time-surface feature is effective for pattern recognition. However, the proposed event-processing algorithm can only be executed in software on the computer side. The event-clustering algorithm proposed in this thesis is dedicated for the on-chip implementation in hardware, so that event-flow features can be created on the focal plane. The event-flow feature may have the potential to be applied in pattern recognition fields, and this needs further investigations.

4.3.2 System Architecture

Fig. 4.1 shows the system architecture of the smart event-clustering image sensor. The main building blocks include a pixel array, a coupling capacitor network, AER readout circuits, and a global analog buffer. Every pixel is cross-connected with its surrounding neighbours via eight coupling capacitors drawn as solid and dashed lines. The event-clustering algorithm is on-chip implemented in hardware by the coupling capacitor network with an identical cluster unit circuit in every pixel, and the output of this algorithm is presented on the pixel array as analog voltages, which are termed as event-flow features. During the readout procedure, row and column AER circuits selectively access the active pixels with their event-flow features exceeding a user-defined threshold. A global unity gain buffer is adopted to export the event-flow feature from the addressed pixel to an external processor.

4.3.3 Pixel Structure

The pixel structure of the smart event-clustering image sensor is shown in Fig. 4.2, and it can be divided into three blocks: a motion detector, an event-clustering circuit and

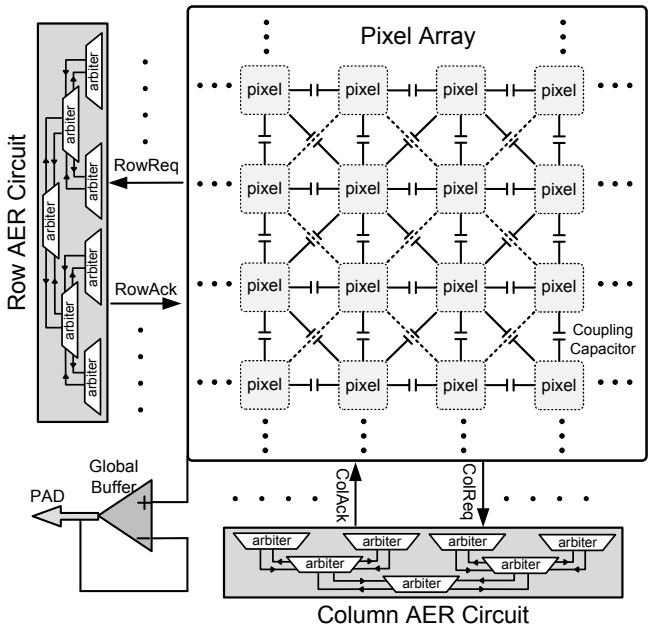


Figure 4.1: System architecture of the event-clustering motion-detection image sensor.

an AER readout module. The motion detector is used to detect and transform visual motions into binary events. There is an identical cluster unit inside every pixel, and the circuit diagram of this unit is drawn in Fig. 4.4. As shown in Fig. 4.2 (c), every cluster unit has a separate cluster point, which is cross-connected with the other eight in the neighbouring pixels through a coupling capacitor network. This configuration is designed to implement the event-clustering algorithm on the pixel array. Motion events fired from the front-end motion detector are immediately processed by the event-clustering circuit in every pixel. The transient response of the event-clustering algorithm is presented as an analog voltage on every cluster point, which is further exported to an external processor via the AER readout module. Since motion detection and event processing are implemented by separate circuits, they work independently in parallel.

There are three reset signals used in the proposed smart pixel: “Global_Reset”, “Motion_Reset” and “Read_Reset”. In the beginning, a “Global_Reset” pulse is applied

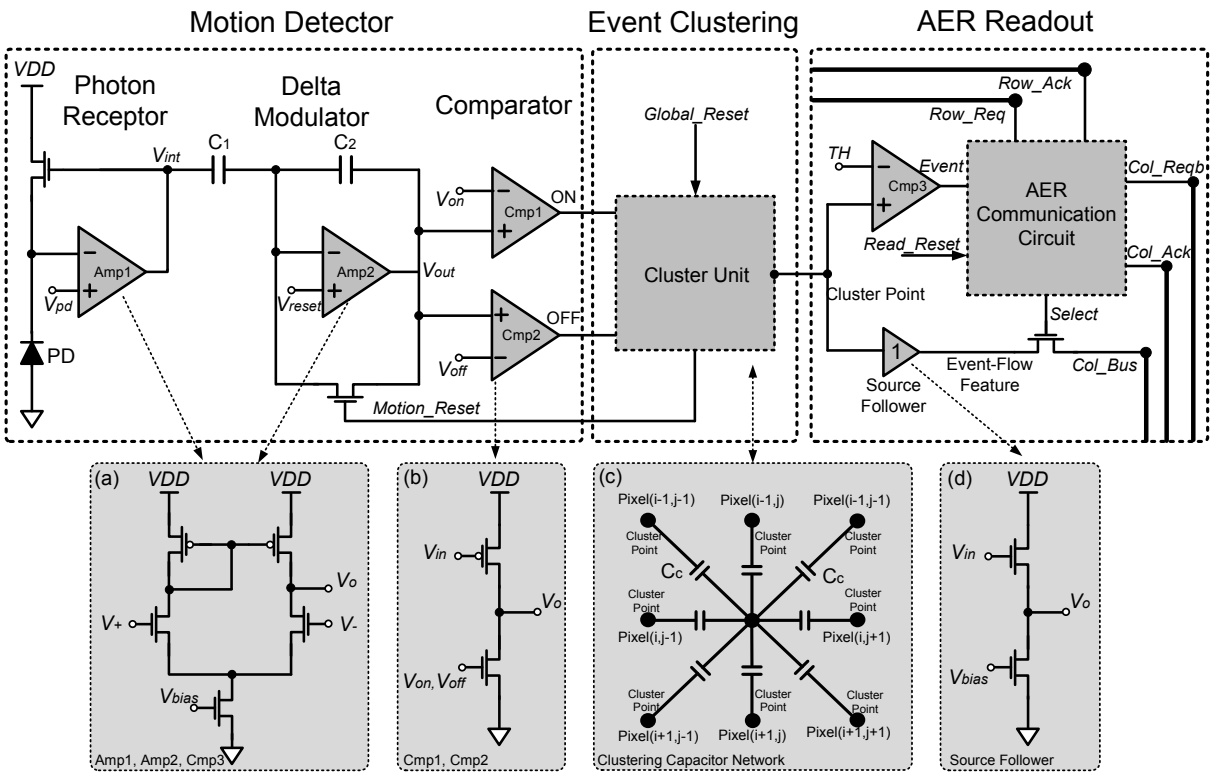


Figure 4.2: Schematic diagram of the smart pixel in the event-clustering image sensor.

to the pixel array. The motion detector and cluster unit in every pixel are initialized for motion detection and event clustering simultaneously. A "Motion_Reset" pulse is activated by the cluster unit only when the pixel detects a motion event, and it solely restores the motion detector to start a new motion-detection operation. An external processor has to activate the "Read_Reset" signal before it starts to readout event-flow features. More detailed descriptions on these reset signals will be presented in the following sections.

4.3.3.1 Motion-Detection Process

In the motion-detection block, there is a photon-reception circuit formed by a logarithmic photodiode with a negative feedback amplifier $Amp1$. The light intensity on the photodiode is directly converted into an analog voltage V_{int}. The main advantage of

this architecture is that biasing voltage V_{pd} on the photodiode remains constant even when there are significant brightness changes on it [110]. Intensity voltage V_{int} is reported to a delta modulator formed by an amplifier $Amp2$ with capacitors C_1 and C_2. This modulator is used to amplify the intensity voltage variation ΔV_{int} from the photon receptor, which is modeled as below:

$$V_{out} = V_{reset} + \frac{C_1}{C_2} \times \Delta V_{int} \qquad \text{(Eq. 4.3)}$$

where V_{out} and V_{reset} correspond to the output and reset voltages respectively. $\mathrm{Eq.\ 4.3}$ reveals that the gain factor of this modulator depends on the capacitor ratio of C_1/C_2. As depicted in Fig. 4.2 (a), the operational amplifiers $Amp1$ and $Amp2$ are designed based on the same architecture.

As shown in Fig. 4.2, there is a reset transistor in the delta modulator, and the switching signal "Motion_Reset" on this transistor is controlled by the cluster unit. In the beginning, when a "Global_Reset" pulse is applied to the image sensor, it activates the "Motion_Reset" signal via the cluster unit in every pixel. Hence, all delta modulators in the pixel array are initialized for motion detection. When a given pixel triggers a motion event, it automatically generates a self-reset pulse "Motion_Reset" from the cluster unit onto the delta modulator which is immediately restored for a new motion-detection operation. Since the "Motion_Reset" signal only restores its own motion detector, every pixel works independently and continuously to detect visual motions in parallel. This design differs from conventional DVS frameworks where pixel reset can only be achieved by external AER signals, which leads to lower accuracy on motion detection, as an event-firing pixel can not be restored immediately until it is accessed by the AER readout module. Two compact comparators $Cmp1$ and $Cmp2$ are designed to digitalize the analog voltage output V_{out} from the delta modulator into binary "ON" and "OFF" events when it exceeds the thresholds defined by the biasing voltages V_{on} and V_{off}.

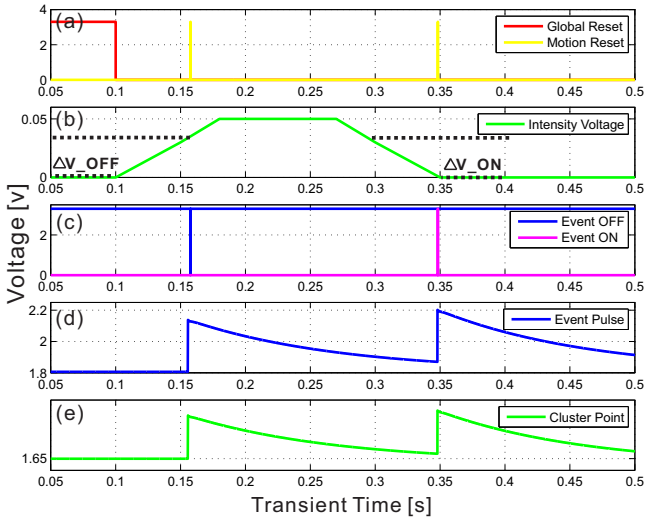


Figure 4.3: Circuit simulation results on motion detection and event clustering. Curves from (a) to (e) correspond to respective signals in different stages.

Fig. 4.3 shows circuit simulation results on motion-detection when a given pixel consecutively fires two motion events. In the beginning, as shown in sub-figure (a), a "Global_Reset" pulse initializes the image sensor for motion detection and event clustering. The intensity voltage variation along with time V_{int} is shown in sub-figure (b). When the relative change of the intensity voltage ΔV_{int} exceeds the predefined threshold "ΔV_OFF", depicted as the interval between two dashed lines, an "OFF" event is triggered accordingly, shown as the first spike in sub-figure (c). Since the proposed image sensor has separate "ΔV_OFF" and "ΔV_ON" thresholds, it can detect both the increase and decrease in light intensity. Hence, as shown in sub-figure (c), there is an "ON" event fired when the intensity voltage variation ΔV_{int} triggers "ΔV_ON" threshold.

4.3.3.2 Event-Clustering Process

The circuit diagram of the cluster unit is shown in Fig. 4.4. This cluster unit has two major functions: Firstly, it generates a "Motion_Reset" signal whenever there is an

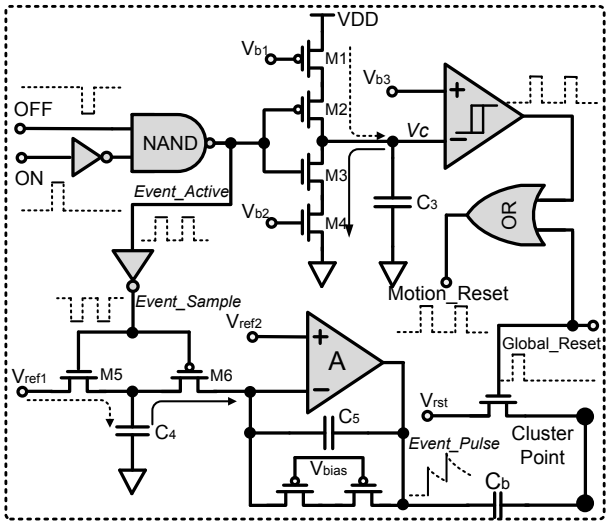


Figure 4.4: Circuit diagram of the cluster unit inside every pixel in the event-clustering image sensor.

event fired in a pixel; secondly, it implements the event-clustering algorithm to transform binary events into transient pulses on the cluster point. In this figure, dashed lines represent idle states, while solid lines correspond to event-firing conditions. Initially, events “ON” and “OFF” are reset into low and high states respectively. Whenever there is an event fired from the front-end motion detection circuit, either event “ON” or event “OFF” toggles its state for activation. Events “ON” and “OFF” are mutually exclusive, and only one of them can be activated at any moment.

When a given pixel detects a motion event, it must be restored immediately by a “Motion_Reset” signal to start a new motion-detection cycle. This self-reset signal is generated by a dynamic logic circuit with a hysteresis comparator in the cluster unit. Initially, “Event_Active” from the NAND gate is reset in the low state. It only turns on the pull up path (M1 - M2), and voltage V_c on capacitor C_3 is charged to the power supply *VDD*, which forces “Motion_Reset” to remain in the low state. When either event “ON” or event “OFF” is activated by the motion detection circuit, “Event_Active” goes high immediately, and it further turns on the pull down path (M3 - M4) to discharge

capacitor C_3. When voltage V_c exceeds the following threshold of the hysteresis comparator, "Motion_Reset" flips its state instantly, and the relative pixel commences the self-reset operation on its delta modulator in the motion-detector block. Then, either event "ON" or event "OFF" is killed accordingly. Hence, "Event_Active" goes low again to charge capacitor C_3. When voltage V_c exceeds the rising threshold of the hysteresis comparator, "Motion_Reset" toggles its state to complete the self-reset operation. Instead of a common single-threshold comparator, a hysteresis comparator is adopted in the cluster unit to ensure that the reset pulse width is wide enough to completely reset the motion-detection circuit. The pulse width of "Motion_Reset" depends on the hysteresis threshold window as well as the charging and discharging currents on the dynamic logic, which are regulated by the biasing voltages V_{b1} and V_{b2} on transistors M1 and M4. Circuit simulation results depicted in the top curve of Fig. 4.3 show that "Motion_Reset" pulses are instantly created after a pixel detects motion events, and the proposed event-clustering circuit can continuously create event pules as long as there are motion events detected in the pixel.

The cluster unit is also designed to transform binary events into transient analog pules. This function is implemented by an integrator circuit formed by an analog amplifier with the switched capacitors C_4 and C_5. When a given pixel detects a motion event, it generates a digital spike on "Event_Sample" due to the self-reset strategy in "Motion_Reset". As shown in Fig. 4.4, this spike works as the sampling clock for a switched-capacitor circuit formed by transistors M5 and M6 with capacitor C_4. Since the initial logic on "Event_Sample" is high, only transistor M5 is turned on, and capacitor C_4 samples the reference voltage V_{ref1}. When "Event_Sample" is switched into the low state by a motion event, transistor M6 is activated so as to transfer the charge from capacitor C_4 into capacitor C_5, which further leads to a voltage jump on the output of the amplifier. Hence, an analog event pulse V_{ep} is created in the cluster unit. The

voltage jump V_m defines the amplitude of the event pulse V_{ep}. It can be modeled as follows:

$$V_m = \frac{C_4}{C_5} \times (V_{ref1} - V_{ref2}) \qquad \text{(Eq. 4.4)}$$

Eq. 4.4 shows that V_m is tunable by varying the capacitor ratio of C_4/C_5 and the voltage gap between V_{ref1} and V_{ref2}.

The influence of a motion event on the focal plane could not remain forever, and it must completely disappear within a certain period. In this image sensor, every analog event pulse V_{ep} is designed with an identical lifetime parameter τ, which is regulated by the leakage path of the analog integrator in the cluster unit. As shown in Fig. 4.4, the leakage path is formed by two PMOS transistors in series, and it is non-linearly regulated by the biasing voltage V_{bias}. In practice, V_{bias} is tuned at an appropriate potential to ensure that the leakage path can be treated as a resistor R. Hence, an event pulse V_{ep} fired at the moment of t_0 can be described as follows:

$$\begin{cases} V_{ep} = V_{dc} & t < t_0 \\ V_{ep} = V_{dc} + V_m \times \exp\left(-\frac{t-t_0}{\tau}\right) & t \geq t_0 \end{cases} \qquad \text{(Eq. 4.5)}$$

where lifetime $\tau = R \times (C_b + C_5)$; V_{dc} represents the balance voltage of the integrator; V_m is the amplitude of the event pulse V_{ep}.

Eq. 4.5 reveals that an event pulse V_{ep} drops exponentially towards the balance voltage V_{dc} after its generation. Hence, as shown in Fig. 4.3, binary motion events "ON" and "OFF" are transformed into transient analog pulses V_{ep} depicted in subfigure (d). In this circuit simulation, the lifetime parameter for an event pulse V_{ep} is designed as $\sim$200 ms. When an "ON" event fires in the pixel, the event pulse generated by an "OFF" event is still partially alive. Hence, both the new and old event pulses in the same pixel are accumulated together, this is termed as the temporal overlapping effect.

As shown in Fig. 4.4, the event pulse V_{ep} fired from the analog integrator is directly reported to capacitor C_b in the cluster unit. This capacitor also connects to the "Cluster_Point", which is highlighted as solid circles in Fig. 4.2. The cluster points on two adjacent pixels are connected via a coupling capacitor C_c, which builds an eight-directional capacitor network on the focal plane, as shown in Fig. 4.1. The event-clustering algorithm in the proposed image sensor is implemented by the capacitor network and the cluster unit in every pixel. The response of the event-clustering algorithm is presented on every cluster point as an analog voltage. Since each coupling capacitor C_c is shared by two adjacent pixels, there are four coupling capacitors C_c embedded in every pixel.

In the beginning, every cluster point in the pixel array is initialized to a biasing voltage V_{rst} by the "Global_Reset" pulse, and it becomes a floating point after the reset path is switched off. Such reset operation on the cluster point is only executed once during the system initialization. Every event pulse V_{ep} disappears by itself in the leakage path of the integrator, and the cluster point strictly follows such event pulse changes due to the coupling effect of C_b. Hence, the cluster unit in every pixel works continuously to process motion events in parallel. When a pixel detects a motion event, the "Motion_Reset" pulse generated from the cluster unit solely resets its motion-detector circuit. Hence, "Motion_Reset" signal is fully isolated from the event-clustering circuit to ensure that pixel's self-reset in motion detection has no influence on event-clustering operation in the pixel array.

As shown in the bottom of Fig. 4.3, when an event pulse V_{ep} is triggered by the analog integrator in the cluster unit, due to the coupling effect on capacitor C_b, the voltage variation of the event pulse V_{ep} is instantly propagated into the cluster point in a certain proportion. Simultaneously, due to the coupling effect on the global capacitor network built by capacitors C_c, the neighbouring cluster points surrounding the event-firing pixel

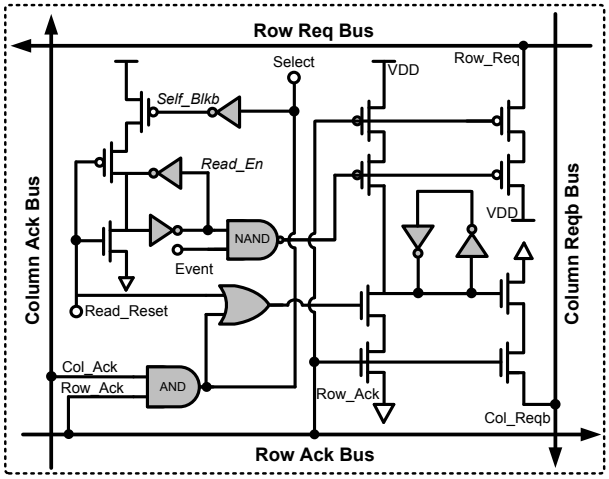


Figure 4.5: Circuit diagram of the AER handshaking logic inside every pixel in the event-clustering image sensor.

also create smaller-amplitude transient spikes. Hence, the event pulse V_{ep} created by a motion event is spread on the pixel array, and the response for a single event in the event-clustering algorithm covers a certain area on the focal plane. When multiple motion events fire closely in space, there will be overlaps on their event-clustering responses, this is termed as the spatial overlapping effect. Therefore, the analog voltage on every cluster point indicates the spatial density and temporal frequency of motion events fired on the focal plane, it is defined as an "event-flow" feature in this thesis.

4.3.3.3 AER-Readout Process

In this image sensor, a frame-driven asynchronous AER readout module is proposed to export only active pixels with sufficient event-flow features to an external processor. Since address events exported from the AER readout module is inherently labelled with a frame start signal, the proposed image sensor is compatible with both frame-based and event-based image processing algorithms. This strategy is designed to reduce the complexity of frame reconstruction for motion events in post processors. As shown in Fig. 4.2, the hybrid AER readout module in every pixel consists of a

comparator, a unity gain buffer and an AER handshaking circuit. The analog buffer and comparator are connected together to the cluster point. In addition, there are four digital buses dedicated for row and column AER communication, and an analog bus "Col_Bus" is used to export the event-flow feature to a global buffer.

Fig. 4.5 shows the schematic diagram of the AER handshaking module, and an example timing diagram of the AER communication is shown in Fig. 4.6. In the AER readout module, there is an enabling circuit on the readout path, and it is controlled by a "Read_En" signal. In order to start event readout, the external processor has to apply a "Read_Reset" pulse onto the enabling circuit in every pixel so that the "Read_En" signal can be initialized into a high state to turn on the readout path. Hence, "Read_Reset" pulses can be treated as pseudo frame-start signals. In this thesis, a readout cycle on the pixel array refers to the time interval between two "Read_Reset" pulses. The enabling circuit in the readout path ensures that pixels are not repeatedly accessed during a readout cycle. After initialization by a "Read_Reset" pulse, the external processor may apply a global threshold onto the comparator in the AER module inside every pixel. Those pixels with their event-flow features exceeding the threshold activate the "Event" signal immediately, and then they send row request signals "Row_Req" to row arbiters. Pixels can send column request signals "Col_Reqb" to column arbiters only when they are granted with row acknowledgement signals "Row_Ack". This strategy is designed to increase the readout speed during the event communication [111, 112]. When a pixel receives both row and column acknowledgement signals, it turns on the unity gain buffer by activating the "Select" signal to copy its event-flow feature onto the analog bus "Col_Bus", which is further exported to the external processor via a global analog buffer. Furthermore, the accessed pixel automatically restores the "Read_En" signal into a low state so as to block such pixel from sending new request until the next readout cycle.

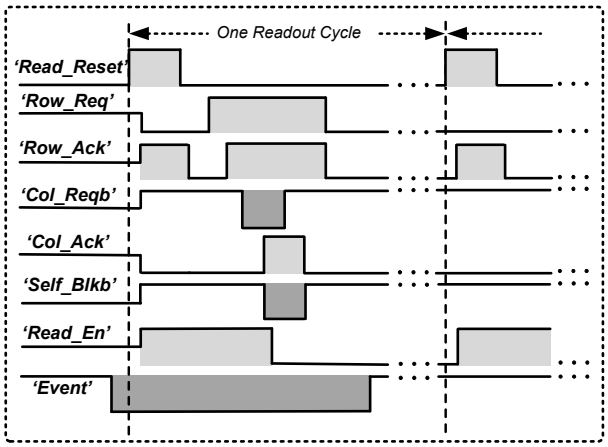


Figure 4.6: Example timing diagram of the AER handshaking communication.

Row and column AER circuits automatically repeat the random accessing procedure until there is no more event request fired on the pixel array, and then the external processor can start a new readout cycle. Address events fired in the time interval between two "Read_Reset" pulses belong to the same frame. By applying a "Read_Reset" pulse, the external processor may arbitrarily abandon the current readout procedure to start a new readout cycle at any moment. The "Read_Reset" pulse is absolutely isolated from "Global_Reset" and "Motion_Reset" signals. Hence, the event-readout procedure has no influence on motion-detection and event-clustering operations. Since address events from the proposed image sensor are inherently labelled with a frame-start signal, the post processor can easily reconstruct a frame of an event-flow feature map by buffering every address event within a complete readout cycle.

4.3.4 Event-Clustering Analysis

In this section, a mathematical analysis was performed on the event-clustering algorithm. Ideally, the global coupling capacitor network is fully symmetrical and infinite in space. Hence, as shown in Fig. 4.7, the equivalent capacitance on every cluster point

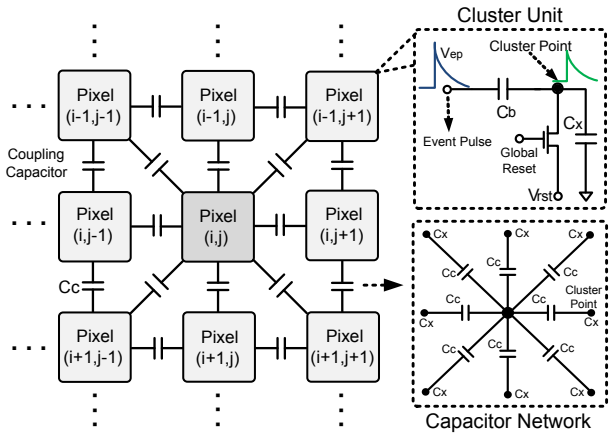


Figure 4.7: Analysis on the event-clustering algorithm with a coupling capacitor network and a cluster unit inside every pixel.

can be treated as the same C_x, which is expressed as the following:

$$C_x = C_b + 8 \times \frac{C_c \times C_x}{C_c + C_x} \qquad \text{(Eq. 4.6)}$$

Since capacitor C_c can be written as λC_b, where λ is the ratio of C_c/C_b, the solution for $\mathrm{Eq.\ 4.6}$ gives an expression for capacitance C_x.

$$C_x = \frac{7\lambda + 1 + \sqrt{49\lambda^2 + 18\lambda + 1}}{2} C_b \qquad \text{(Eq. 4.7)}$$

When a motion event is fired on pixel (i, j) at the moment of t_0, an event pulse V_{ep} is triggered instantly, and it directly drives capacitors C_b and C_x in series. By applying the Kirchhoff current law to the cluster point of pixel (i, j), the following expression can be extracted:

$$C_b \times \frac{d\left(V_{ep} - V\left(i, j, t\right)\right)}{dt} = C_x \times \frac{dV\left(i, j, t\right)}{dt} \qquad \text{(Eq. 4.8)}$$

where $V\left(i, j, t\right)$ represents the transient voltage on the cluster point of pixel (i, j). Since event pulse V_{ep} has been given in $\mathrm{Eq.\ 4.5}$, the solution for $\mathrm{Eq.\ 4.8}$ gains a transient expression on $V\left(i, j, t\right)$ as below:

$$\begin{cases} V\left(i, j, t\right) = V_{rst} & t < t_0 \\ V\left(i, j, t\right) = V_{rst} + \eta \times \exp\left(-\frac{t - t_0}{\tau}\right) & t \geq t_0 \end{cases} \qquad \text{(Eq. 4.9)}$$

where $\eta = \frac{C_b}{C_b+C_x} \times V_m$; τ is the lifetime parameter for an event pulse V_{ep}; and V_rst is the initial voltage on the cluster point after the global reset.

$\mathrm{Eq.\ 4.9}$ shows that the voltage variation of event pulse V_{ep} is coupled into the cluster point in a certain proportion. Hence, the analog voltage on the cluster point exactly follows the same shape of the event pulse V_{ep} with a smaller amplitude. The same analysis can also be applied on the cluster points of the 1st stage neighbours surrounding the event-firing pixel (i, j) within a window of $(i \pm 1, j \pm 1)$. Here, the pixel $(i+1, j)$ is taken as an example to demonstrate the analytical process. Eight neighboring pixels are connected to this pixel via coupling capacitors C_c. However, only three are active followers including pixels $(i+2, j-1)$, $(i+2, j)$, $(i+2, j+1)$ drawn as blue in Fig. 4.8. The other pixels are either located on the same stage drawn as coral, such as pixels $(i, j-1)$, $(i, j+1)$, $(i+1, j-1)$ and $(i+1, j+1)$, or at the front stage drawn as green, such as the event-firing pixel (i, j). Therefore, the transient voltage expression on the cluster point of pixel $(i+1, j)$ can be given as follows:

$$C_c \times \frac{dV\,(i,j,t)}{dt} = C^* \times \frac{dV\,(i+1,j,t)}{dt} \qquad \text{(Eq. 4.10)}$$

where $C^* = C_c + C_b + 3 \times \frac{C_c \times C_x}{C_c+C_x}$. Note that $V\,(i, j, t)$ has been given in $\mathrm{Eq.\ 4.9}$. By solving $\mathrm{Eq.\ 4.10}$, the voltage expression for the cluster point of the 1st stage neighboring pixel $(i+1, j)$ in a single motion event condition is obtained.

$$\begin{cases} V\,(i+1,j,t) = V_{rst} & t < t_0 \\ V\,(i+1,j,t) = V_{rst} + \kappa \times \exp\left(-\frac{t-t_0}{\tau}\right) & t \geq t_0 \end{cases} \qquad \text{(Eq. 4.11)}$$

where $\kappa = \frac{C_c}{C^*} \times \frac{C_b}{C_b+C_x} \times V_m$. Similarly, the transient response on the 2nd stage neighboring pixel can also be derived, which is described as below:

$$\begin{cases} V\,(i+2,j,t) = V_{rst} & t < t_0 \\ V\,(i+2,j,t) = V_{rst} + \gamma \times \exp\left(-\frac{t-t_0}{\tau}\right) & t \geq t_0 \end{cases} \qquad \text{(Eq. 4.12)}$$

where $\gamma = 3 \times \left(\frac{C_c}{C^*}\right)^2 \times \frac{C_b}{C_b+C_x} \times V_m$.

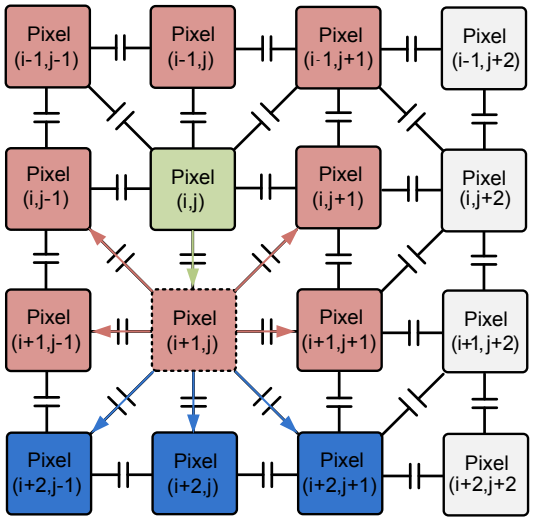


Figure 4.8: Analysis on the coupling capacitor network when the center pixel (i, j) fires a motion event.

Transient responses on the cluster points in the pixels that are further away from the event-firing position can be derived in the same manner. In order to evaluate the modeling above, a 32 $\times$ 32 pixel array embedded with event-clustering circuits is built and simulated in Cadence by using AMS 0.35 μm CMOS technology. In the circuit simulation, as shown in Fig. 4.8, only the center pixel depicted in blue fires a motion event at the moment of 1.0 s, while its neighbouring pixels marked with red and green remain in a standby mode. The initial voltage V_{rst} on every cluster point is reset at 900 mV. A single event creates an event pulse V_{ep} with the amplitude V_m set as 300 mV. In addition, the coupling capacitor C_c has the same capacitance as capacitor C_b in 30 fF. Dashed and solid curves correspond to results from Matlab modeling and Cadence simulation respectively. Fig. 4.9 shows that the mathematical modeling is consistent with the circuit simulation results, and transient responses on the cluster points in the event-firing pixel (center one in blue) and its surroundings (1st stage in red and 2nd stage in green) have the same shape but different amplitudes.

Circuit simulation results in Fig. 4.9 reveal that the event-clustering response for

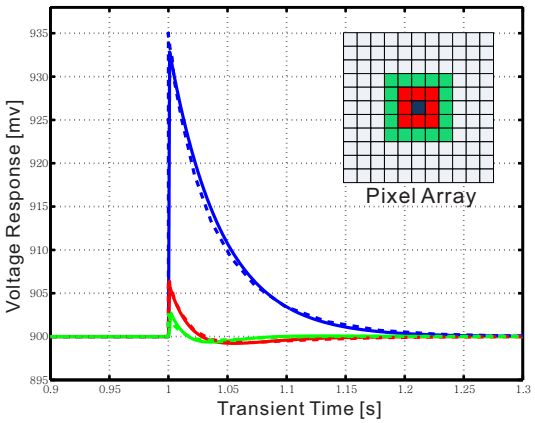


Figure 4.9: Simulation results on the event-clustering algorithm for a single motion event.

a single motion event decreases with the increase in distance from the event-firing position. Therefore, a single motion event creates a mountain-shaped response on the focal plane. By exploiting different capacitor ratios λ, designers can reshape the spatial distribution of the event-clustering response. Fig. 4.10 shows circuit simulation results for single event response under various capacitor ratios λ in Cadence. Analog voltages on every cluster point in the pixel array are sampled at the moment when the center pixel depicted in red triggers a motion event. It shows that a larger λ results in a better coupling effect, which implies that a single motion event creates less response on its firing position but its surrounding pixels receive more response.

Motion events from an active object tend to be spatially concentrated in the regions displaying sharp brightness changes, while random noise events are discrete with lower spatial densities. Fig. 4.11 shows responses on the event-clustering algorithm from multiple events as well as a single event firing together on the focal plane under different capacitor ratios λ. In this circuit simulation, a 3 $\times$ 3 pixel area represents an active region of motion, while an individual pixel indicates discrete noise, and they fire events together at the moment of 0 s. It can be noted that a multiple-event region receives much larger response than the single-event area. Hence, the discrete noise

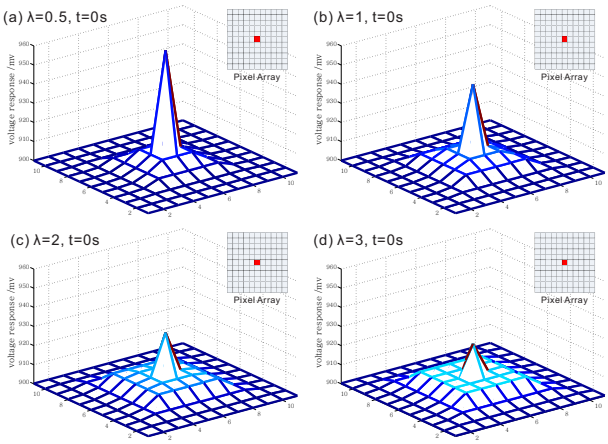


Figure 4.10: Single event response from the event-clustering algorithm under various capacitor ratios λ.

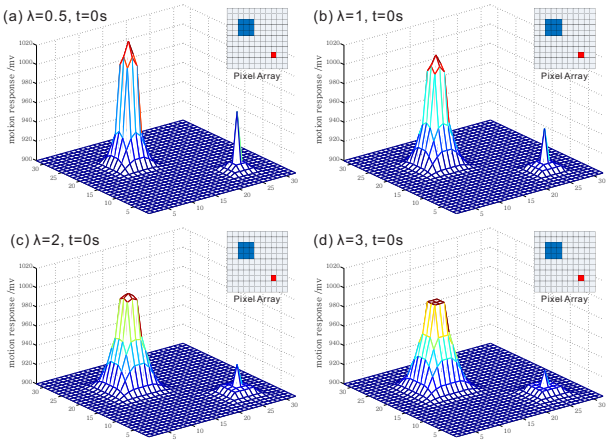


Figure 4.11: Comparison of responses from the event-clustering algorithm under multiple events and a single event.

event effect can be completely removed by a simple thresholding operation. Also, the larger capacitor ratios λ extends the event-clustering response more widely in space.

Fig. 4.12 shows the transient voltage responses of the event-clustering algorithm along with time from the event-firing moment at 0 s. The circuit simulation environment is configured as the same with that in Fig. 4.11. It shows that responses from multiple events and a single event decrease simultaneously, and they almost completely

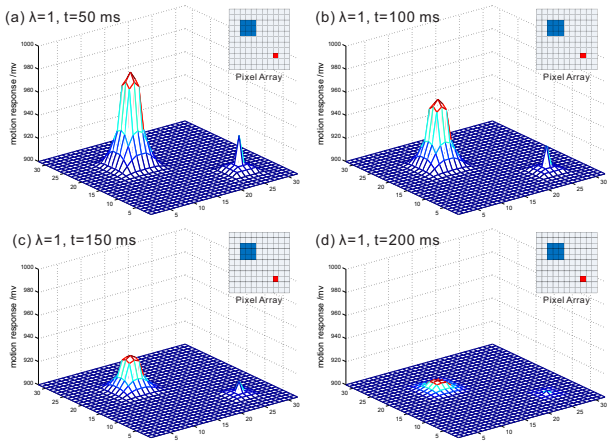


Figure 4.12: Transient responses from the event clustering-algorithm in multiple events and single noise conditions.

disappear in ∼200 ms as long as there is no more event firing within this period in the same region. As described in the previous section, this feature is implemented by the leakage current in the reset path of the analog integrator inside every cluster unit. Lifetime τ for an event pulse V_{ep} is tunable according to practical applications. It is common that a pixel fires a new motion event while the response of the prior event is still partially alive. In this case, the event-clustering response of the new event are accumulated with the prior one. Moreover, due to the fact that the response of a motion event covers a certain area, there will be spatial overlaps when motion events fire closely in space. Therefore, the eventual response of the event-clustering algorithm highly depends on the frequency and density of motion events, which are directly determined by the speed and size of the moving object.

In order to obtain an intuitive relationship between the event-clustering response and the object speed, as shown in Fig. 4.13, it is assumed that a polychrome object is travelling through the viewing field of the image sensor at a constant speed. For the sake of simplicity, this active object is set to cover a 4 $\times$ 4 pixel area when projected to the focal plane. Since the object speed is set at 50 pixels/s, this image

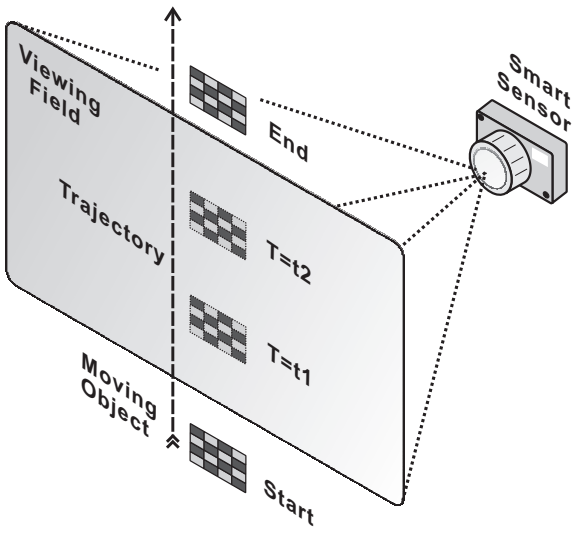


Figure 4.13: Simulation example on a polychrome object passing through the viewing field of the event-clustering image sensor.

sensor periodically generates 4 $\times$ 4 motion events in the object's region with a fixed time interval of 20 ms. At the same time, the lifetime τ for an event pulse V_{ep} in this simulation is configured as $\sim$200 ms. Therefore, there are both temporal and spatial overlapping effects on the event-clustering response along the object travelling trace. Due to the system setup complexity, this simulation is conducted in Matlab by using the mathematic modeling derived above.

Transient voltages on every cluster point in the pixel array at four different moments are drawn in Fig. 4.14. It shows that there is a stable mountain-shaped response on the pixel array with a constant peak voltage of $\sim$1100 mV, when the object travels with a constant speed in the viewing field. In the simulation of Fig. 4.15, the object speed is diversified, and the event-clustering responses at some specific moments are drawn in sub-figures (a) to (d) respectively. One can note that the amplitude of the event-clustering response rises from 1000 mV to 1200 mV with the increase in speed. Moreover, in sub-figure (d), there is a distinct tailing effect along the moving path as the object is travelling so fast that the overlapping effect on the event-clustering response

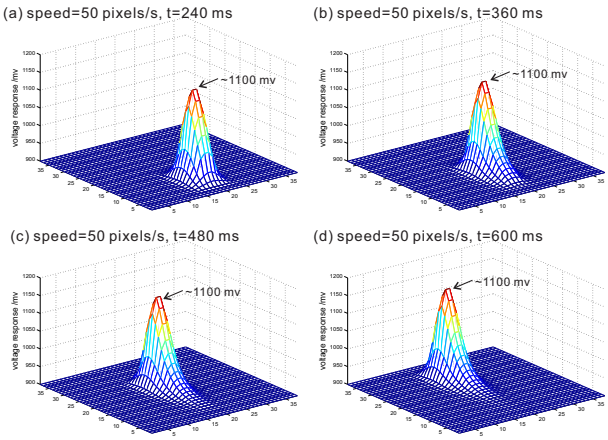


Figure 4.14: Transient response from the event-clustering response in the experiment where the object travelling speed is fixed at 20 pixels/s and lifetime τ of the event pulse V_{ep} is configured as 200 ms.

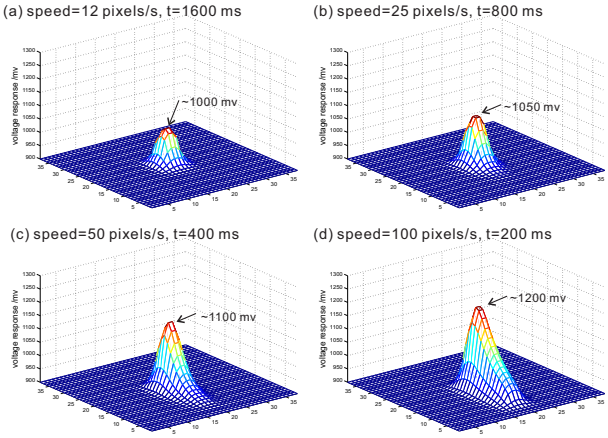


Figure 4.15: Comparison on the responses from the event-clustering algorithm under variant object speeds.

is aggravated.

The amplitude of the event-clustering response is also related to the spatial density of motion events, which is directly determined by the object size. In order to investigate their relationship, the simulation in Fig. 4.13 is repeated on three different-sized objects under various speeds. In Fig. 4.16, the horizontal axis indicates the object speed and the vertical axis represents the peak response voltage from the event-clustering

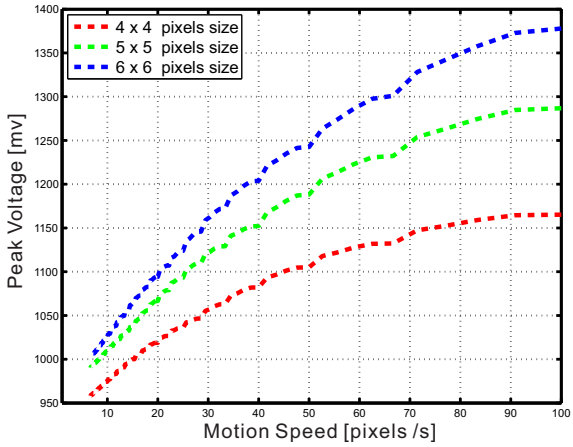


Figure 4.16: Maximum amplitude of the response from the event-clustering algorithm in relation to movement speed and object size.

algorithm. The three curves correspond to objects sized at 4 $\times$ 4, 5 $\times$ 5 and 6 $\times$ 6 pixels respectively.

As shown in Fig. 4.16, the peak amplitude of the event-clustering response is proportional to the travelling speed for the same object. Under the same speed, a larger object creates a bigger response on the event-clustering algorithm. The simulated environment in Fig. 4.13 is idealized as event generation not only relies on the moving speed and physical size of the object but also depends on the object texture and the ambient illumination. In practice, if a kinetic object remains active in the viewing scene, there is always a mountain-shaped event-clustering response surrounding it, which is a customized feature on the focal plane to mimic the natural ripples on the water surface.

4.3.5 Simulation Results

The main contribution of this smart image sensor is the event-clustering algorithm that on-chip processes motion events to extract event-flow features. The proposed event-clustering pixel is designed and simulated in Cadence by using AMS 0.35 μm CMOS

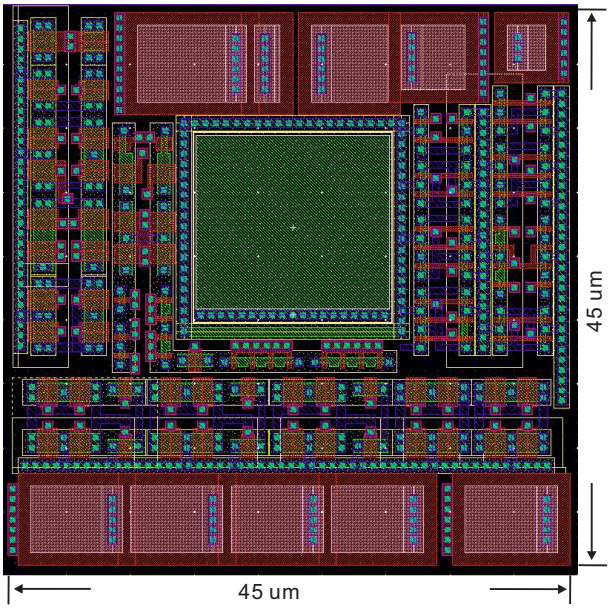


Figure 4.17: Photograph of the smart event-clustering pixel layout.

process. However, the implementation of the event-clustering algorithm on the focal plane is achieved with an extreme hardware cost in every pixel. The photograph of the smart pixel layout is shown in Fig. 4.17. For clarity, it only displays layers below metal-1, and the other top layers are hidden accordingly. Every pixel totally has 108 transistors and 10 capacitors in a silicon area of 45 μm $\times$ 45 μm. The cluster unit and the coupling capacitor network occupy over 40 % of the pixel area, and the fill factor of the smart pixel is $\sim$11.3 %. The main characteristics of the smart pixel are summarized in Table. 4.3. The power consumption per pixel is $\sim$3.07 μW based on simulations, and the total power consumption of the image sensor is estimated over 51 mW under a resolution of 128 $\times$ 128. The proposed event-clustering image sensor is too costly for fabrication, and it has not been implemented in hardware yet. Instead, exhaustive simulation and evaluation have been conducted in this thesis.

The event-flow feature is customized for kinetic object tracking applications. In order to investigate its performance, a number of system-level simulations in different

object tracking algorithms are conducted on a Matlab R2014b platform. Two standard test sequences (crossroad [113] and campus [114]) and one general video sequence (billiard [115]) are adopted in simulations. Information of test sequences and initialization boxes are listed in Table 4.1. The four elements in the initial bounding box are: x, y coordinates of the upper left corner of the target object, as well as its spatial dimensions on length and width. All simulations are conducted on a desktop PC equipped with Intel Core4, 3.0 GHz CPU, 8G RAM, and Microsoft Windows 7 Professional operating system. The tunable parameters in the proposed event-clustering algorithm are fixed in appropriate values as summarized in Table 4.2.

Table 4.1: Information of the test sequences in simulation

Test Sequence	Number of Frames	Resolution	Initial Bounding Box
Billiard	174	1280 $\times$ 720	(568, 125, 39, 39)
Crossroad	141	1280 $\times$ 720	(912, 449, 80, 60)
Campus	196	768 $\times$ 576	(93, 401, 40, 80)

Table 4.2: Parameter setting of the event-clustering algorithm in simulation

Parameters	Values
Coupling Capacitor C_c	50 fF
Buffer Capacitor C_b	50 fF
Capacitor Ratio λ	1
Reset Voltage V_{rst}	50 mV
Event Pulse Lifetime τ	600 ms
Event Pulse Amplitude V_m	500 mV

Simulation in generating event-flow features is demonstrated in Fig. 4.18. The first column corresponds to intensity images from test video sequences. Simulated motion events are generated by applying temporal difference computation on two consecutive frames. Events falling into frame intervals are considered to be fired simultaneously within the second frame period. In general, low-speed frame-based cameras suffer

Table 4.3: Parameters of the event-clustering pixel

Technology	AMS 0.35 μm CMOS
Supply Voltage	3.3 V
Power Consumption per Pixel	3.07 μW
Photodiode Type	N+/P-sub
Pixel Size	45 μm $\times$ 45 μm
Number of Transistor per Pixel	108
Number of Capacitor per Pixel	10
Fill Factor	$\sim$11.3 %

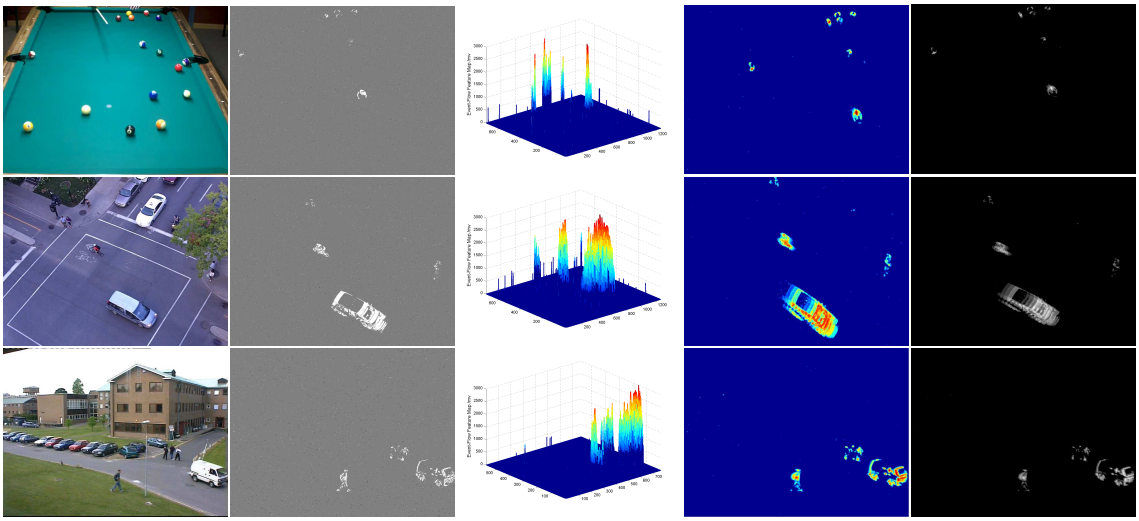


Figure 4.18: Demonstration on event-flow feature generation. Columns from left to right correspond to raw intensity images, motion event images, event-flow feature maps in a 3D view, and top view as well as quantized event-flow feature maps as gray scale images.

from a motion blur effect, and fast motions can not be truly extracted from the video sequence acquired by these sensors [8]. In this simulation, the test video sequences are elaborately planned so that there is only slow visual motion for the object of interest in the viewing scene. This ensures that motion events are precisely located within accurate temporal and spatial resolutions. There is no such constraint in the proposed event-clustering image sensor, as it is able to continuously detect motions with independence of motion speeds.

In order to evaluate the robustness of the event-flow feature for object tracking applications, “salt” and “pepper” noise with a spatial density of 1% is inserted into motion events, as shown in the second column of Fig. 4.18. In the third column, snapshots of the event-flow feature map taken at certain moments are drawn under a 3D view. There are many mountain-shaped responses existing on event-flow feature maps, and these mountain areas correspond to the regions of active objects with motion activities. Hence, event-flow features on the focal plane mimics nature ripples on the water surface. The top view on event-flow feature maps is shown in the fourth column of Fig. 4.18. Brighter colors indicate higher event-flow features, while the blue one represents the static background. Event-flow feature maps are quantized into gray-scale images within 0 - 255 range, as shown in the last column.

There are two object tracking algorithms investigated in this thesis: the first one is a “peak-shift” method, which directly searches the peak location on the event-flow feature map to track a moving object; the other one is developed based on a classical mean-shift framework, which replaces pixel intensity information with event-flow features.

1) Peak-Shift Mode: The amplitude of the event-flow feature is directly correlated to the spatial density and temporal frequency of motion events on the focal plane. In regions containing active objects, there are always distinct event-flow features as long as these objects keep on moving. The most direct way of tracking an object is to follow the peak location of its event-flow features on the focal plane. This approach is termed as the peak-shift algorithm, and it is demonstrated on the billiard video sequence in Fig. 4.19. The peak-shift algorithm is performed with the following steps:

- *Object Initialization:* In the beginning, the spatial dimension (S_x, S_y) and center position (C_x, C_y) of the target object are manually initialized. The peak location

(P_x, P_y) of the event-flow feature in the object region is recorded. There is a relative spatial distance (D_x, D_y) between the peak position (P_x, P_y) and the object center (C_x, C_y), where $(D_x, D_y) = (P_x, P_y) - (C_x, C_y)$. This parameter works as an offset constant to localize the object center when the peak position is detected from the event-flow feature map.

- *Feature Extraction:* After initialization, as shown in subfigure (a), by applying a proper threshold to the event-clustering image sensor, pixels with their event-flow features exceeding the threshold are extracted to an external processor within a complete AER readout cycle, and a new frame of the event-flow feature map is reconstructed on the post processor easily.

- *Region Definition:* Based on the current location of the tracked object, as shown in subfigure (b), a potential region (R_x, R_y, L_x, L_y) drawn with dashed lines is defined to search the peak position of the event-flow features. (R_x, R_y) are the coordinates of the upper left corner of the potential region, and (L_x, L_y) correspond to its spatial dimensions. $(L_x, L_y) = 2(S_x, S_y)$ and $(R_x, R_y) = (C_x, C_y) - (S_x, S_y)$.

- *Object Localization:* The peak position (P_x, P_y) of the event-flow feature in the potential region is extracted, drawn as a red cross in subfigure (c). As shown in subfigure (d), the center location (C_x, C_y) of the object can be updated by simply computing $(C_x, C_y) = (P_x, P_y) + (D_x, D_y)$, where (D_x, D_y) is the offset parameter defined in advance.

The initialization work is executed once at the beginning of the tracking task. The peak-shift tracking algorithm automatically follows the active object by periodically repeating the steps from “feature extraction” to “object localization” in a circular manner.

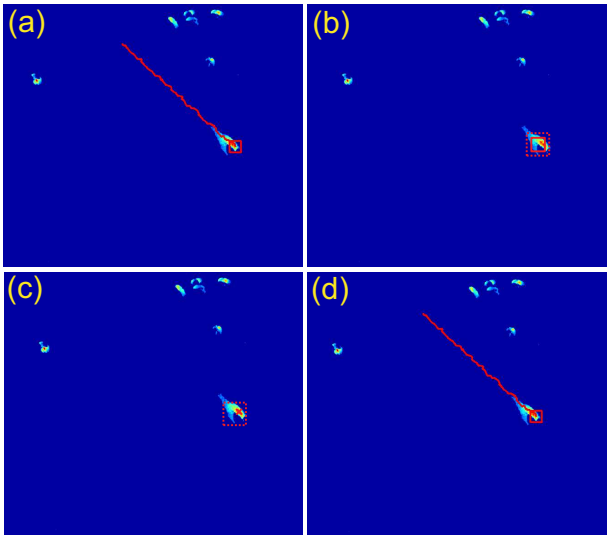


Figure 4.19: Demonstration on the peak-shift tracking algorithm. Images (a) to (d) shows event-flow feature operations in different stages.

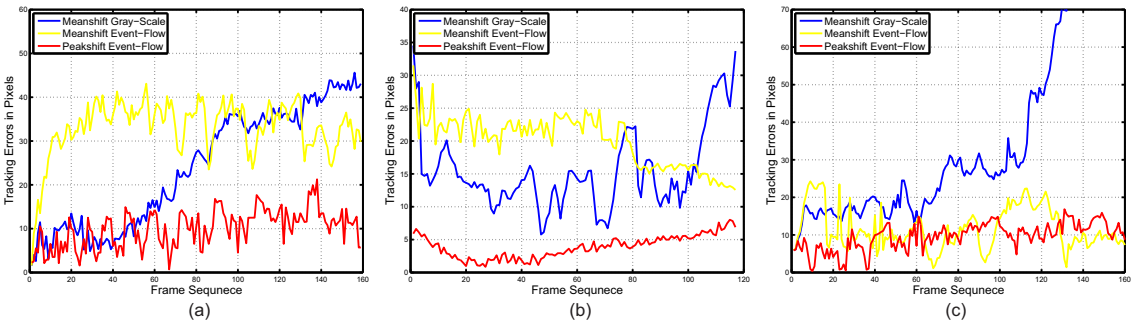


Figure 4.20: Tracking errors versus the frame sequence. Images (a) to (c) correspond to the test sequences of Billiard, Crossroad and Campus respectively.

The main advantage of the peak-shift method is that there is no multiplication or convolution involved during the tracking procedure. Hence, it can be executed with lower computation loads in higher operating speed.

2) Mean-Shift Mode: Asynchronous motion events from conventional dynamic-vision-sensors (DVS) are incompatible with common frame-based object tracking algorithms. However, the proposed event-clustering image sensor is designed with the hybrid AER readout strategy. Hence, post processors can randomly take snapshots

on every cluster point in the pixel array to extract event-flow features, and a frame of the event-flow feature map can then be easily reconstructed by buffering every address event within a complete readout cycle. In addition, the event-flow feature on every pixel has distinct amplitudes. Thus, the event-flow feature map can be quantized into a digital image within a range of 0 - 255, as shown in the final column of Fig. 4.18. Therefore, conventional frame-based object tracking algorithms can be directly executed on digital images obtained from event-flow feature maps.

In computer-based image processing, the mean-shift algorithm is a well-known object tracking method for its superior performance. In general, this frame-based algorithm employs pixel intensity as its input feature and tracks the interested objects by computing their intensity distributions [106, 116]. In this thesis, a specialized mean-shift framework is implemented by replacing pixel intensity values with digitalized event-flow features. Since the hybrid AER readout circuit only exports active pixels with sufficient event-flow features, the static background in the viewing scene is completely discarded. Also, a complete readout cycle to reconstruct a frame of the event-flow map is extremely short. Hence, tracking results in the mean-shift algorithm, based on event-flow features, are reliable and robust.

In this thesis, tracking error is defined as the spatial distance between the center of a tracked object and its actual position. The eventual performance for a tracking algorithm is evaluated based on its tracking error. The ground truths of two standard test sequences are inherently provided from the data source, while the ground truth of the general video sequence is manually labeled. Three parameters are used to quantify the tracking performance: maximum error, minimum error and mean-absolute-error (MAE). MAE can be calculated with the following expression.

$$MAE = \frac{1}{N}\sum_{f=1}^{N}\sqrt{(X_{tr}-X_{gt})^2+(Y_{tr}-Y_{gt})^2} \qquad \text{(Eq. 4.13)}$$

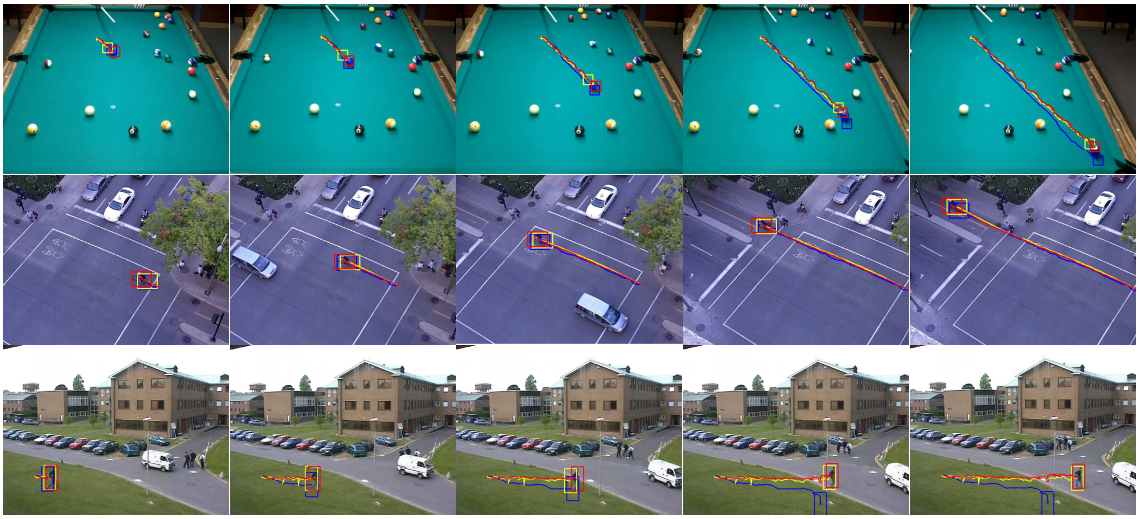


Figure 4.21: Sampled screenshots on tracking results in three test video sequences at different moments. Blue, red and yellow curves correspond to results from different algorithms and features respectively.

where N represents the number of frames in a test sequence, X_{tr} and Y_{tr} denote the center of the tracked object, while X_{gt} and Y_{gt} indicate the ground truth.

Tracking errors along with the frame sequence are shown in Fig. 4.20. In this figure, gray-scale feature-based mean-shift, event-flow feature-based peak-shift and mean-shift frameworks are depicted in blue, red and yellow respectively. The screenshots on tracking results are shown in Fig. 4.21. Images from top to bottom correspond to test sequences on billiard, crossroad and campus at different moments. Table 4.4 summarizes tracking errors in three test sequences under different algorithms and features. Both optimal and sub-optimal solutions are highlighted with bold characters. The peak-shift approach is superior in tracking performance when compared to the other two methods. The MAEs for the peak-shift algorithm in three test sequences are lowest, recorded as 9.98, 8.03 and 9.36 respectively. In the mean-shift framework, the pixel gray-scale and event-flow features have comparable MAEs, however, the former shows greater inconsistency when compared to the latter.

The execution time for every framework to complete the tracking task is recorded in Table. 4.5. It shows that the peak-shift algorithm based on event-flow features is much faster than the other two methods. The total number of multiplication involved during the tracking procedure is recorded in Table. 4.6. It shows that the peak-shift framework based on event-flow features has the least computation load. System-level simulations reveal that event-flow features exported from the proposed event-clustering image sensor significantly accelerate post processing with less computation load in object tracking applications. More demonstration videos on different video sequence for object tracking applications are valid online [117].

Table 4.4: Absolute errors of three object tracking frameworks

Test Sequence	Tracking Error	Gray-Scale Mean-Shift	Event-Flow Peak-Shift	Event-Flow Mean-Shift
Billiard	Max Error	45.6	21.27	43.09
	MAE	**23.97**	**9.98**	33.14
	Min Error	1.58	0.71	1.58
Crossroad	Max Error	34.52	8.03	33.63
	MAE	**15.53**	**4.09**	20.49
	Min Error	5.77	0.85	7.52
Campus	Max Error	147.99	16.84	24.27
	MAE	49.137	**9.36**	**11.47**
	Min Error	5.89	0.37	1.12

Table 4.5: Execution time for three object tracking frameworks

	Mean-shift Gray-scale	Mean-shift Event-flow	Peak-shift Event-flow
Billiard	6.83 s	3.86 s	0.40 s
Crossroad	13.41 s	12.55 s	0.72 s
Campus	8.87 s	10.07 s	0.51 s

Table 4.6: Number of multiplication in three object tracking frameworks

	Mean-shift Gray-scale	Mean-shift Event-flow	Peak-shift Event-flow
Billiard	1979.1K	1058.9K	0
Crossroad	3452.1K	3033.8K	0
Campus	2365.8K	2588.3K	0

4.4 Summary and Comparison

The major contribution of the proposed event-clustering image sensor is the event-clustering algorithm, which is customized to generate specific event-flow features for object tracking applications. In this sensor, motion detection and event processing are implemented by separate circuits. Hence, they work independently and continuously in parallel. A hybrid AER readout module in every pixel only exports active pixels with sufficient event-flow features to an external processor. Since address events reported from this sensor are synchronized with a frame signal, the proposed image sensor is compatible with both frame-based and event-based image processing algorithms. System-level simulations reveal that event-flow features can accelerate post vision processing with much less computation load in object tracking applications.

Table 4.7: Performance summary of the smart image sensors in this report

	Lichtsteiner [12] '08	Kim [98] '09	Bardallo [87] '11	proposed sensor
Technology	0.35 μm 2P 4M	0.5 μm 2P 3M	0.35 μm 2P 4M	0.35 μm 2P 4M
Array Size	128 $\times$ 128	64 $\times$ 64	128 $\times$ 128	128 $\times$ 128
Pixel Size	40 $\times$ 40 μm^2	29 $\times$ 28 μm^2	35 $\times$ 35 μm^2	45 $\times$ 45 μm^2
Scheme	Event	Frame	Event	Event/Frame
Fill Factor	8.1%	23%	8.7%	11.3%
Supply Voltage	3.3V	3V	3.3V	3.3 V
Power Consumption	24 mW	1.06 mW	130 mW	52 mW

The proposed event-clustering image sensor is compared with the other smart image sensors reported in the literature. Table. 4.7 summarizes the main characteristics of these sensors. In the proposed image sensor, the hardware implementation of

the event-clustering algorithm complicates the circuit design with larger pixel size, and the event-clustering circuit in every pixel increases the total power consumption to a certain extent. The other smart image sensors listed in Table. 4.7 only detect visual motions but have no on-chip motion processing capability. The proposed smart image sensor is more appealing as it integrates both motion detection and event processing on the same chip for specific applications.

Chapter 5

Conclusions and Future Work

5.1 Conclusion

As the world is heading towards a smart future, smart image sensors can be applied in many areas of daily life and business, including automated manufacturing, autopilot technology, medical, and etc. One example is a smart image sensor developed by Infineon, which integrates direct measurement of depth and amplitude in every pixel. Hence, it can report both intensity and depth features at the same time, which makes it quite appealing for 3D computer vision applications [118]. Another example is the Exmor IMX230 sensor developed by Sony. The technology has been implemented on digital cameras with the integration of capture and processing on a single CMOS chip. An on-chip phase detection processing function allows the sensor to achieve excellent focus tracking of fast moving objects [119]. The on-chip integration of image processing algorithms allows for more compact imaging module and are beneficial to actualize small, complex and efficient systems.

This thesis proposes a number of smart CMOS image sensors for the implementation of feature-extraction functions on the focal plane. With this implementation, the image feature is directly exported from the image sensor. Hence, post processors become more compact with less computation burden. This thesis also investigates

the potential and feasibility to integrate feature extraction and processing on the same chip, so that a smart image sensor becomes a complete vision system for a specific application.

The first work reported in this thesis is a motion-detection image sensor based on the temporal difference algorithm. This sensor detects visual motions from the focal plane and on-chip processes them to localize the main active object. There are two operating modes for this sensor: intensity mode and motion mode. Under the intensity mode, the image sensor can report intensity images as commercial cameras. Under the motion mode, the sensor reports a binary motion image to encode the visual activities in the viewing field. The sensor can automatically report an intensity image on the region of the main object whenever necessary. Hence, this sensor is compact vision system that integrates moving object detection and localization on the same chip. This image sensor has been published in [96, 120]

The second work presented in this thesis is a feature-extraction image sensor developed using 3D integrated circuit technology with the aim of alleviating the conflict between the drop of fill-factor (FF) and the implementation of feature-extraction functions in the conventional CMOS technology. This sensor has three operating modes: intensity mode, motion mode and contrast mode. Under the intensity mode, the sensor reports an analog intensity image on the viewing scene. In the motion mode, the sensor simultaneously reports two temporal consecutive frame images to detect the motion activities on the focal plane. In the contrast mode, the sensor sequential extracts an intensity image and its smoothed one to extract the spatial contours in the viewing field. Furthermore, the sensor achieves feature-extraction functions while maintaining an extreme fill factor by integrating image acquisition and processing circuits in separate tiers. This image sensor has been published in [121].

An event-clustering motion-detection image sensor that integrates motion detection and event processing on the same chip is proposed. In this sensor, address events

are on-chip processed once they are generated from the motion-detection circuit. The output of this image sensor is a stream of address events associated with analog voltages termed as the "event-flow" features, which represent the density and frequency of motion events on the focal plane. This event-flow feature has been evaluated to be effective in various object tracking algorithms by simulations. This sensor is a significant revolution when compared to the other event-based motion-detection image sensors reported in the literature, as it implements both motion detection and event processing on the same chip. A journal paper reporting this image sensor has been submitted to IEEE Sensors journal [122].

The accomplishments above prove that smart CMOS image sensors significantly relieve the computation burden on post processors for the entire vision system, and it is feasible to integrate the feature extracting and processing algorithms on the same sensor, so as to realize the complete visual system on a single chip.

5.2 Future Work

Numerous challenges still exists in the development of smart CMOS image sensors. The following suggestions may worth investigation in future research.

Compact Pixel: The hardware implementation of the feature-extraction algorithm significantly increases silicon consumption with much more complex pixel circuit. For example, the total number of transistors in the pixel of the proposed event-clustering image sensor is over 100, which makes the pixel and sensor extremely bulky. The pixel design can be compressed to allow smart image sensors to extract valuable features with minimal cost. 3D integration technology is elegant solution by providing higher integration density when compared to conventional CMOS technology.

Autonomous Detection: At present, in many smart image sensors, a number of parameters have to be manually tuned for feature extraction by operators for practical

applications. This configuration is unacceptable for remote sensing applications. One feasible solution is to implement an intelligent self-adopt scheme in the smart image sensor, so that the parameters are automatically updated without human intervention.

Diverse Feature: Currently, smart image sensors can only report limited features, such as contrast, motions or corners. More advanced features should be extracted from the smart image sensor to facilitate post processors. A typical example is the event-clustering image sensor proposed in last chapter, which exports the "event-flow" feature from the general motion events for object tracking applications. Researchers can diversely customize image features into smart image sensors for specific applications.

List of Publications

Journal Papers

(i) **Xiangyu Zhang**, Shizheng Wang, Shoushun Chen and Junsong Yuan, "A Smart Event-Clustering Motion-Detection Image Sensor for Visual Object Tracking Applications," under review in *IEEE Sensors Journal*.

(ii) Bo Zhao, **Xiangyu Zhang**, Shoushun Chen, Kay Soon Low and Hualiang Zhuang, "A 64 $\times$ 64 COMS Image Sensor with On-Chip Moving Objects Detection and Localization," *IEEE Transactions on Circuits and Systems on Video Technology (TCSVT)*, vol. 22, no. 4, pp. 581-588, April 2012.

(iii) Shoushun Chen, Wei Tang, **Xiangyu Zhang**, and Eugenio Culurciello, "A 64 $\times$ 64 Pixels UWB Wireless Temporal-Difference Digital Image Sensor," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 20, no. 12, pp. 2232-2240, April 2012.

Conference Papers

(i) **Xiangyu Zhang** and Shoushun Chen, "A Second-Generation Noise-Immune Motion Detection Image Sensor For Moving Object Tracking Application," *International Symposium on Integrated Circuits (ISIC)*, Singapore, Dec. 2014.

(ii) **Xiangyu Zhang** and Shoushun Chen, "Live Demonstration: A High-Speed-Pass Asynchronous Motion Detection Sensor," *IEEE International Symposium on Circuits and Systems (ISCAS)*, Beijing, China, May 2013.

(iii) **Xiangyu Zhang** and Shoushun Chen, "A Hybrid-Readout and Dynamic-Resolution Motion Detection Image Sensor for Object Tracking," *IEEE International Symposium on Circuits and Systems (ISCAS)*, Seoul, Korea, May 2012.

(iv) **Xiangyu Zhang**, Shoushun Chen and Eugenio Culurciello, "A Second Generation 3D Integrated Feature-Extracting Image Sensor," *IEEE Sensors Conference*, Limerick, Ireland, Oct. 2011.

(v) Bo Zhao, **Xiangyu Zhang** and Shoushun Chen, "A CMOS Image Sensor with on-chip Motion Detection and Object Localization," *IEEE Custom Integrated Circuits Conference (CICC)*, San Jose, USA, Sept. 2011.

(vi) Junwu Zhang, **Xiangyu Zhang**, Zhuangliang Chen, Kye Yak See, Cher Ming Tan and Shoushun Chen, "On-Chip RF Energy Harvesting Circuit for Image Sensor," *International Symposium on Integrated Circuits (ISIC)*, Singapore, Dec. 2011.

Bibliography

[1] L. G. McIlrath, V. S. Clark, P. K. Duane, R. D. McGrath, and W. D. Waskurak, "Design and analysis of a 512× 768 current-mediated active pixel array image sensor," *IEEE Transactions on Electron Devices*, vol. 44, no. 10, pp. 1706–1715, 1997.

[2] SONY, "Sony back-illuminated cmos technology." Available: `https://www.sony.net/SonyInfo/News/Press/200806/08-069E/index.html`.

[3] H.-C. Jiang and C.-Y. Wu, "A 2-d velocity-and direction-selective sensor with bjt-based silicon retina and temporal zero-crossing detector," *IEEE Journal of Solid-State Circuits*, vol. 34, no. 2, pp. 241–247, 1999.

[4] X. Arreguit, F. A. Van Schaik, F. V. Bauduin, M. Bidiville, and E. Raeber, "A cmos motion detector system for pointing devices," *IEEE Journal of solid-state circuits*, vol. 31, no. 12, pp. 1916–1921, 1996.

[5] A. Simoni, G. Torelli, F. Maloberti, A. Sartori, S. E. Plevridis, and A. N. Birbas, "A single-chip optical sensor with analog memory for motion detection," *IEEE Journal of Solid-State Circuits*, vol. 30, no. 7, pp. 800–806, 1995.

[6] S.-Y. Ma and L.-G. Chen, "A single-chip cmos aps camera with direct frame difference output," *IEEE Journal of Solid-State Circuits*, vol. 34, no. 10, pp. 1415–1418, 1999.

[7] S. Mizuno, K. Fujita, H. Yamamoto, N. Mukozaka, and H. Toyoda, "A 256× 256 compact cmos image sensor with on-chip motion detection function," *IEEE Journal of Solid-State Circuits*, vol. 38, no. 6, pp. 1072–1075, 2003.

[8] J. Choi, S.-W. Han, S.-J. Kim, S.-I. Chang, and E. Yoon, "A spatial-temporal multiresolution cmos image sensor with adaptive frame rates for tracking the moving objects in region-of-interest and suppressing motion blur," *IEEE Journal of Solid-State Circuits*, vol. 42, no. 12, pp. 2978–2989, 2007.

[9] T. Hamamoto and K. Aizawa, "A computational image sensor with adaptive pixel-based integration time," *IEEE Journal of Solid-State Circuits*, vol. 36, no. 4, pp. 580–585, 2001.

[10] C. P. Chong, C. Salama, and K. Smith, "Image-motion detection using analog vlsi," *IEEE Journal of Solid-State Circuits*, vol. 27, no. 1, pp. 93–96, 1992.

[11] A. Moini, A. Bouzerdoum, K. Eshraghian, A. Yakovleff, X. T. Nguyen, A. Blanksby, R. Beare, D. Abbott, and R. E. Bogner, "An insect vision-based motion detection chip," *IEEE Journal of Solid-State Circuits*, vol. 32, no. 2, pp. 279–284, 1997.

[12] P. Lichtsteiner, C. Posch, and T. Delbruck, "A 128 $\times$ 128 120 db 15 μs latency asynchronous temporal contrast vision sensor," *IEEE Journal of Solid-State Circuits*, vol. 43, no. 2, pp. 566–576, 2008.

[13] A. A. Stocker and R. J. Douglas, "Computation of smooth optical flow in a feedback connected analog network," in *Advances in Neural Information Processing Systems*, pp. 706–712, 1999.

[14] A. A. Stocker, "Analog integrated 2-d optical flow sensor," *Analog Integrated Circuits and Signal Processing*, vol. 46, no. 2, pp. 121–138, 2006.

[15] MITLL, "Mitll 3d soi process user guide." Available: `http://www.ece.umd.edu/~dilli/research/layout/MITLL_3D_2006/3D_PDK2.3/doc/ApplicationNotes2006-1.pdf`.

[16] J. Ohta, *Smart CMOS image sensors and applications*. CRC Press, 2007.

[17] M. Nixon, *Feature extraction & image processing*. Academic Press, 2008.

[18] J. H. Duncan and T.-C. Chou, "On the detection of motion and the computation of optical flow," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 14, no. 3, pp. 346–352, 1992.

[19] J. P. O'Doherty, P. Dayan, K. Friston, H. Critchley, and R. J. Dolan, "Temporal difference models and reward-related learning in the human brain," *Neuron*, vol. 38, no. 2, pp. 329–337, 2003.

[20] O. Barnich and M. Van Droogenbroeck, "Vibe: A universal background subtraction algorithm for video sequences," *IEEE Transactions on Image Processing*, vol. 20, no. 6, pp. 1709–1724, 2011.

[21] S.-Y. Ma and L.-G. Chen, “A single-chip cmos aps camera with direct frame difference output,” *IEEE Journal of Solid-State Circuits*, vol. 34, no. 10, pp. 1415–1418, 1999.

[22] S. Chen, W. Tang, X. Zhang, and E. Culurciello, “A 64 $\times$ 64 pixels uwb wireless temporal-difference digital image sensor,” *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 20, no. 12, pp. 2232–2240, 2012.

[23] K. A. Boahen, “Point-to-point connectivity between neuromorphic chips using address events,” *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing*, vol. 47, no. 5, pp. 416–434, 2000.

[24] E. Culurciello, R. Etienne-Cummings, and K. A. Boahen, “A biomorphic digital image sensor,” *IEEE Journal of Solid-State Circuits*, vol. 38, no. 2, pp. 281–294, 2003.

[25] G. Johansson, “Visual perception of biological motion and a model for its analysis,” *Perception & Psychophysics*, vol. 14, no. 2, pp. 201–211, 1973.

[26] B. Zhao, R. Ding, S. Chen, B. Linares-Barranco, and H. Tang, “Feedforward categorization on aer motion events using cortex-like features in a spiking neural network,” *IEEE Transactions on Neural Networks and Learning Systems*, vol. 26, no. 9, pp. 1963–1978, 2015.

[27] R. Serrano-Gotarredona, M. Oster, P. Lichtsteiner, A. Linares-Barranco, R. Paz-Vicente, F. Gómez-Rodríguez, L. Camuñas-Mesa, R. Berner, M. Rivas-Pérez, T. Delbrück, *et al.*, “Caviar: A 45k neuron, 5m synapse, 12g connects/s aer hardware sensory–processing–learning–actuating system for high-speed visual object recognition and tracking,” *IEEE Transactions on Neural Networks*, vol. 20, no. 9, pp. 1417–1438, 2009.

[28] W. S. Boyle and G. E. Smith, “Charge coupled semiconductor devices,” *Bell System Technical Journal*, vol. 49, no. 4, pp. 587–593, 1970.

[29] K. Oda, “Interline transfer ccd image sensor with reduced dark current,” Mar. 4 1997. US Patent 5,608,455.

[30] J. S. Lee, R. I. Hornsey, and D. Renshaw, “Analysis of cmos photodiodes. i. quantum efficiency,” *IEEE Transactions on Electron Devices*, vol. 50, no. 5, pp. 1233–1238, 2003.

[31] J. S. Lee, R. I. Hornsey, and D. Renshaw, “Analysis of cmos photodiodes. ii. lateral photoresponse,” *IEEE Transactions on Electron Devices*, vol. 50, no. 5, pp. 1239–1245, 2003.

[32] P. P. Lee, R. M. Guidash, T.-H. Lee, and E. G. Stevens, “Active pixel sensor integrated with a pinned photodiode,” Apr. 29 1997. US Patent 5,625,210.

[33] H. Nabeyama, S. Nagahara, H. Shimizu, M. Noda, and M. Masuda, “All solid state color camera with single-chip mos imager,” *IEEE Transactions on Consumer Electronics*, vol. 1, no. CE-27, pp. 40–46, 1981.

[34] M. Noda, T. Imaide, T. Kinugasa, and R. Nishimura, “A solid state color video camera with a horizontal readout mos imager,” *IEEE Transactions on Consumer Electronics*, vol. 3, no. CE-32, pp. 329–336, 1986.

[35] S. Mendis, S. E. Kemeny, and E. R. Fossum, “Cmos active pixel image sensor,” *IEEE Transactions on Electron Devices*, vol. 41, no. 3, pp. 452–453, 1994.

[36] J. Nakamura, B. Pain, T. Nomoto, T. Nakamura, and E. R. Fossum, “On-focal-plane signal processing for current-mode active pixel sensors,” *IEEE Transactions on Electron Devices*, vol. 44, no. 10, pp. 1747–1758, 1997.

[37] D. G. Chen, D. Matolin, A. Bermak, and C. Posch, “Pulse-modulation imagingreview and performance analysis,” *IEEE Transactions on Biomedical Circuits and Systems*, vol. 5, no. 1, pp. 64–82, 2011.

[38] S. Hanson, Z. Foo, D. Blaauw, and D. Sylvester, “A 0.5 v sub-microwatt cmos image sensor with pulse-width modulation read-out,” *IEEE Journal of Solid-State Circuits*, vol. 45, no. 4, pp. 759–767, 2010.

[39] X. Guo, X. Qi, and J. G. Harris, “A time-to-first-spike cmos image sensor,” *IEEE Sensors Journal*, vol. 7, no. 8, pp. 1165–1175, 2007.

[40] T.-H. Tsai and R. Hornsey, “Analysis of dynamic range, linearity, and noise of a pulse-frequency modulation pixel,” *IEEE Transactions on Electron Devices*, vol. 59, no. 10, pp. 2675–2681, 2012.

[41] S. Iwabuchi, Y. Maruyama, Y. Ohgishi, M. Muramatsu, N. Karasawa, and T. Hirayama, “A back-illuminated high-sensitivity small-pixel color cmos image sensor with flexible layout of metal wiring,” in *Solid-State Circuits Conference, 2006. ISSCC 2006. Digest of Technical Papers. IEEE International*, pp. 1171–1178, IEEE, 2006.

[42] S. K. Mendis, S. E. Kemeny, R. C. Gee, B. Pain, C. O. Staller, Q. Kim, and E. R. Fossum, “Cmos active pixel image sensors for highly integrated imaging systems,” *IEEE Journal of Solid-State Circuits*, vol. 32, no. 2, pp. 187–197, 1997.

[43] M. Loose, K. Meier, and J. Schemmel, “A self-calibrating single-chip cmos camera with logarithmic response,” *IEEE Journal of Solid-State Circuits*, vol. 36, no. 4, pp. 586–596, 2001.

[44] J. Canny, “A computational approach to edge detection,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 8, no. 6, pp. 679–698, 1986.

[45] R. M. Haralick, “Digital step edges from zero crossing of second directional derivatives,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 6, no. 1, pp. 58–68, 1984.

[46] A. Huertas and G. Medioni, “Detection of intensity changes with subpixel accuracy using laplacian-gaussian masks,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 8, no. 5, pp. 651–664, 1986.

[47] P. Perona and J. Malik, “Scale-space and edge detection using anisotropic diffusion,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 12, no. 7, pp. 629–639, 1990.

[48] J. Hafner, H. S. Sawhney, W. Equitz, M. Flickner, and W. Niblack, “Efficient color histogram indexing for quadratic form distance functions,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 17, no. 7, pp. 729–736, 1995.

[49] K. Van De Sande, T. Gevers, and C. Snoek, “Evaluating color descriptors for object and scene recognition,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 9, pp. 1582–1596, 2010.

[50] R. C. Gonzalez, R. E. Woods, *et al.*, “Digital image processing,” 2002.

[51] D.-C. He, L. Wang, and J. Guibert, “Texture feature extraction,” *Pattern Recognition Letters*, vol. 6, no. 4, pp. 269–273, 1987.

[52] B. S. Manjunath and W.-Y. Ma, “Texture features for browsing and retrieval of image data,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 18, no. 8, pp. 837–842, 1996.

[53] M. Yang, K. Kpalma, and J. Ronsin, “A survey of shape feature extraction techniques,” *Pattern Recognition*, vol. 15, no. 7, pp. 43–90, 2008.

[54] D. Zhang and G. Lu, “Review of shape representation and description techniques,” *Pattern Recognition*, vol. 37, no. 1, pp. 1–19, 2004.

[55] S. Ullman, “Analysis of visual motion by biological and computer systems,” *Readings in Computer Vision, chapter Recovering Scene Geometry*, pp. 132–144, 1987.

[56] J. Hutchinson, C. Koch, J. Luo, and C. Mead, “Computing motion using analog and binary resistive networks,” *Computer*, vol. 21, no. 3, pp. 52–63, 1988.

[57] J. G. Harris, C. Koch, E. Staats, and J. Luo, “Analog hardware for detecting discontinuities in early vision,” *International Journal of Computer Vision*, vol. 4, no. 3, pp. 211–223, 1990.

[58] H. Kobayashi, J. L. White, and A. A. Abidi, “An active resistor network for gaussian filtering of images,” *IEEE Journal of Solid-State Circuits*, vol. 26, no. 5, pp. 738–748, 1991.

[59] E. Funatsu, Y. Nitta, Y. Miyake, T. Toyoda, J. Ohta, and K. Kyuma, “An artificial retina chip with current-mode focal plane image processing functions,” *IEEE Transactions on Electron Devices*, vol. 44, no. 10, pp. 1777–1782, 1997.

[60] P.-F. Ruedi, P. Heim, F. Kaess, E. Grenet, F. Heitger, P.-Y. Burgi, S. Gyger, and P. Nussbaum, “A 128 $\times$ 128 pixel 120-db dynamic-range vision-sensor chip for image contrast and orientation extraction,” *IEEE Journal of Solid-State Circuits*, vol. 38, no. 12, pp. 2325–2333, 2003.

[61] N. Massari and M. Gottardi, “A 100 db dynamic-range cmos vision sensor with programmable image processing and global feature extraction,” *IEEE Journal of Solid-State Circuits*, vol. 42, no. 3, pp. 647–657, 2007.

[62] N. Takahashi, K. Fujita, and T. Shibata, “A pixel-parallel self-similitude processing for multiple-resolution edge-filtering analog image sensors,” *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 56, no. 11, pp. 2384–2392, 2009.

[63] T. Delbruck, “Silicon retina with correlation-based, velocity-tuned pixels,” *IEEE Transactions on Neural Networks*, vol. 4, no. 3, pp. 529–541, 1993.

[64] J. Kramer, “Compact integrated motion sensor with three-pixel interaction,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 18, no. 4, pp. 455–460, 1996.

[65] J. Krammer and C. Koch, “Pulse-based analog vlsi velocity sensors,” *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing*, vol. 44, no. 2, pp. 86–101, 1997.

[66] R. Etienne-Cummings, J. Van der Spiegel, and P. Mueller, “A focal plane visual motion measurement sensor,” *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, vol. 44, no. 1, pp. 55–66, 1997.

[67] V. Gruev and R. Etienne-Cummings, “Pipelined temporal difference imager,” *Electronics Letters*, vol. 38, no. 7, pp. 315–317, 2002.

[68] V. Gruev and R. Etienne-Cummings, “A pipelined temporal difference imager,” *IEEE Journal of Solid-State Circuits*, vol. 39, no. 3, pp. 538–543, 2004.

[69] J. Choi and E. Yoon, “Spatial-temporal multi-resolution image sensor with adaptive frame rates for tracking movement in a region of interest,” Jan. 3 2012. US Patent 8,089,522.

[70] M. Mori, M. Katsuno, S. Kasuga, T. Murata, and T. Yamaguchi, “1/4-inch 2-mpixel mos image sensor with 1.75 transistors/pixel,” *IEEE Journal of Solid-State Circuits*, vol. 39, no. 12, pp. 2426–2430, 2004.

[71] Y. Muramatsu, S. Kurosawa, M. Furumiya, H. Ohkubo, and Y. Nakashiba, “A signal-processing cmos image sensor using a simple analog operation,” *IEEE Journal of Solid-State Circuits*, vol. 38, no. 1, pp. 101–106, 2003.

[72] Y. M. Chi, U. Mallik, M. A. Clapp, E. Choi, G. Cauwenberghs, and R. Etienne-Cummings, “Cmos camera with in-pixel temporal change detection and adc,” *IEEE Journal of Solid-State Circuits*, vol. 42, no. 10, pp. 2187–2196, 2007.

[73] D. Kim and E. Culurciello, “Tri-mode smart vision sensor with 11-transistors/pixel for wireless sensor networks,” *IEEE Sensors Journal*, vol. 13, no. 6, pp. 2102–2108, 2013.

[74] S. Kavadias, B. Dierickx, D. Scheffer, A. Alaerts, D. Uwaerts, and J. Bogaerts, “A logarithmic response cmos image sensor with on-chip calibration,” *IEEE Journal of Solid-State Circuits*, vol. 35, no. 8, pp. 1146–1152, 2000.

[75] C. Shoushun and A. Bermak, “Arbitrated time-to-first spike cmos image sensor with on-chip histogram equalization,” *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 15, no. 3, pp. 346–357, 2007.

[76] C. Posch, D. Matolin, and R. Wohlgenannt, “A qvga 143 db dynamic range frame-free pwm image sensor with lossless pixel-level video compression and time-domain cds,” *IEEE Journal of Solid-State Circuits*, vol. 46, no. 1, pp. 259–275, 2011.

[77] C. Shi, M. K. Law, and A. Bermak, “A novel asynchronous pixel for an energy harvesting cmos image sensor,” *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 19, no. 1, pp. 118–129, 2011.

[78] P. OConnor, D. Neil, S.-C. Liu, T. Delbruck, and M. Pfeiffer, “Real-time classification and sensor fusion with a spiking deep belief network,” *Neuromorphic Engineering Systems and Applications*, p. 61, 2015.

[79] O. Bichler, M. Suri, D. Querlioz, D. Vuillaume, B. DeSalvo, and C. Gamrat, “Visual pattern extraction using energy-efficient 2-pcm synapse neuromorphic architecture,” *IEEE Transactions on Electron Devices*, vol. 59, no. 8, pp. 2206–2214, 2012.

[80] M. Suri, D. Querlioz, O. Bichler, G. Palma, E. Vianello, D. Vuillaume, C. Gamrat, and B. DeSalvo, “Bio-inspired stochastic computing using binary cbram synapses,” *IEEE Transactions on Electron Devices*, vol. 60, no. 7, pp. 2402–2409, 2013.

[81] Z. Ni, A. Bolopion, J. Agnus, R. Benosman, and S. Régnier, “Asynchronous event-based visual shape tracking for stable haptic feedback in microrobotics,” *IEEE Transactions on Robotics*, vol. 28, no. 5, pp. 1081–1089, 2012.

[82] Z. Ni, C. Pacoret, R. Benosman, S. Ieng, *et al.*, “Asynchronous event-based high speed vision for microparticle tracking,” *Journal of Microscopy*, vol. 245, no. 3, pp. 236–244, 2012.

[83] X. Lagorce, C. Meyer, S.-H. Ieng, D. Filliat, and R. Benosman, “Asynchronous event-based multikernel algorithm for high-speed visual features tracking,” *IEEE transactions on neural networks and learning systems*, vol. 26, no. 8, pp. 1710–1720, 2015.

[84] G. Orchard, C. Meyer, R. Etienne-Cummings, C. Posch, N. Thakor, and R. Benosman, “Hfirst: a temporal approach to object recognition,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 37, no. 10, pp. 2028–2040, 2015.

[85] S. Chen, P. Akselrod, B. Zhao, J. A. P. Carrasco, B. Linares-Barranco, and E. Culurciello, "Efficient feedforward categorization of objects and human postures with address-event image sensors," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 2, pp. 302–314, 2012.

[86] A. N. Belbachir, M. Hofstatter, M. Litzenberger, and P. Schon, "High-speed embedded-object analysis using a dual-line timed-address-event temporal-contrast vision sensor," *IEEE Transactions on Industrial Electronics*, vol. 58, no. 3, pp. 770–783, 2011.

[87] J. A. Leñero-Bardallo, T. Serrano-Gotarredona, and B. Linares-Barranco, "A 3.6 μs latency asynchronous frame-free event-driven dynamic-vision-sensor," *IEEE Journal of Solid-State Circuits*, vol. 46, no. 6, pp. 1443–1455, 2011.

[88] T. Serrano-Gotarredona and B. Linares-Barranco, "A 128 $\times$ 128 1.5% contrast sensitivity 0.9% fpn 3 μs latency 4 mw asynchronous frame-free dynamic vision sensor using transimpedance preamplifiers," *IEEE Journal of Solid-State Circuits*, vol. 48, no. 3, pp. 827–838, 2013.

[89] C. Brandli, R. Berner, M. Yang, S.-C. Liu, and T. Delbruck, "A 240 $\times$ 180 130 db 3 μs latency global shutter spatiotemporal vision sensor," *IEEE Journal of Solid-State Circuits*, vol. 49, no. 10, pp. 2333–2341, 2014.

[90] M. Yang, S.-C. Liu, and T. Delbruck, "A dynamic vision sensor with 1% temporal contrast sensitivity and in-pixel asynchronous delta modulator for event encoding," *IEEE Journal of Solid-State Circuits*, vol. 50, no. 9, pp. 2149–2160, 2015.

[91] B. K. Horn and B. G. Schunck, "Determining optical flow," *Artificial intelligence*, vol. 17, no. 1-3, pp. 185–203, 1981.

[92] A. A. Stocker, "Analog vlsi focal-plane array with dynamic connections for the estimation of piecewise-smooth optical flow," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 51, no. 5, pp. 963–973, 2004.

[93] A. Elgammal, D. Harwood, and L. Davis, "Non-parametric model for background subtraction," in *European conference on computer vision*, pp. 751–767, Springer, 2000.

[94] R. Polana and R. C. Nelson, "Detection and recognition of periodic, nonrigid motion," *International Journal of Computer Vision*, vol. 23, no. 3, pp. 261–282, 1997.

[95] 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.

[96] B. Zhao, X. Zhang, S. Chen, K.-S. Low, and H. Zhuang, "A 64 $\times$ 64 cmos image sensor with on-chip moving object detection and localization," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 22, no. 4, pp. 581–588, 2012.

[97] Z. Bo, "A biologically inspired human posture recognition system." Available: `http://www.ntu.edu.sg/home/eechenss/Papers/Thesis-2014-A%20biologically%20inspired%20human%20posture%20recognition%20system.pdf`.

[98] D. Kim, Z. Fu, J. H. Park, and E. Culurciello, "A 1-mw cmos temporal-difference aer sensor for wireless sensor networks," *IEEE Transactions on Electron Devices*, vol. 56, no. 11, pp. 2586–2593, 2009.

[99] J. A. Pérez-Carrasco, B. Zhao, C. Serrano, B. Acha, T. Serrano-Gotarredona, S. Chen, and B. Linares-Barranco, "Mapping from frame-driven to frame-free event-driven vision systems by low-rate rate coding and coincidence processing–application to feedforward convnets," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 11, pp. 2706–2719, 2013.

[100] M. P. Ekstrom, *Digital image processing techniques*, vol. 2. Academic Press, 2012.

[101] P. Tissainayagam and D. Suter, "Assessing the performance of corner detectors for point feature tracking applications," *Image and Vision Computing*, vol. 22, no. 8, pp. 663–679, 2004.

[102] J. Ning, L. Zhang, D. Zhang, and C. Wu, "Robust object tracking using joint color-texture histogram," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 23, no. 07, pp. 1245–1263, 2009.

[103] A. Yilmaz, X. Li, and M. Shah, "Contour-based object tracking with occlusion handling in video acquired using mobile cameras," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 26, no. 11, pp. 1531–1536, 2004.

[104] S.-K. Weng, C.-M. Kuo, and S.-K. Tu, "Video object tracking using adaptive kalman filter," *Journal of Visual Communication and Image Representation*, vol. 17, no. 6, pp. 1190–1208, 2006.

[105] D. Comaniciu, V. Ramesh, and P. Meer, "Kernel-based object tracking," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 25, no. 5, pp. 564–577, 2003.

[106] D. Comaniciu and P. Meer, "Mean shift: A robust approach toward feature space analysis," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 5, pp. 603–619, 2002.

[107] C. Stauffer and W. E. L. Grimson, "Learning patterns of activity using real-time tracking," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 8, pp. 747–757, 2000.

[108] D. Drazen, P. Lichtsteiner, P. Häfliger, T. Delbrück, and A. Jensen, "Toward real-time particle tracking using an event-based dynamic vision sensor," *Experiments in Fluids*, vol. 51, no. 5, p. 1465, 2011.

[109] X. Lagorce, G. Orchard, F. Galluppi, B. E. Shi, and R. B. Benosman, "Hots: a hierarchy of event-based time-surfaces for pattern recognition," *IEEE transactions on pattern analysis and machine intelligence*, vol. 39, no. 7, pp. 1346–1359, 2017.

[110] X. Zhang and S. Chen, "A hybrid-readout and dynamic-resolution motion detection image sensor for object tracking," in *Circuits and Systems (ISCAS), 2012 IEEE International Symposium on*, pp. 1628–1631, IEEE, 2012.

[111] C. Shoushun and A. Bermak, "Arbitrated time-to-first spike cmos image sensor with on-chip histogram equalization," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 15, no. 3, pp. 346–357, 2007.

[112] A. M. T. Linn, C. Shoushun, Y. K. Seng, *et al.*, "Adaptive priority toggle asynchronous tree arbiter for aer-based image sensor," in *VLSI and System-on-Chip (VLSI-SoC), 2011 IEEE/IFIP 19th International Conference on*, pp. 66–71, IEEE, 2011.

[113] J.-P. Jodoin, G.-A. Bilodeau, and N. Saunier, "Urban tracker: Multiple object tracking in urban mixed traffic," in *Applications of Computer Vision (WACV), 2014 IEEE Winter Conference on*, pp. 885–892, IEEE, 2014.

[114] D. Makris, "Pets 2001 dataset." Available: `http://www.cvg.reading.ac.uk/slides/pets.html`.

[115] X. Zhang, “Billiard test video sequence.” Available: https://www.youtube.com/watch?v=LvAxpw5rEsU.

[116] D. Comaniciu, V. Ramesh, and P. Meer, “Real-time tracking of non-rigid objects using mean shift,” in *Computer Vision and Pattern Recognition, 2000. Proceedings. IEEE Conference on*, vol. 2, pp. 142–149, IEEE, 2000.

[117] X. Zhang, “A demonstration video of the tracking results for three test video sequences.” Available: https://www.youtube.com/channel/UC_g01tK8PGUZcmjkz8rNAmg.

[118] “Smart image sensor for depth extraction.” Available: https://www.infineon.com/dgdl/Infineon-REAL3+Image+Sensor+Family-PB-v01_00-EN.PDF?fileId=5546d462518ffd850151a0afc2302a58.

[119] “Sony exmor rs imx230.” Available: https://www.sony.net/SonyInfo/News/Press/201411/14-112E/.

[120] B. Zhao, X. Zhang, and S. Chen, “A cmos image sensor with on-chip motion detection and object localization,” in *Custom Integrated Circuits Conference (CICC), 2011 IEEE*, pp. 1–4, IEEE, 2011.

[121] X. Zhang, S. Chen, and E. Culurciello, “A second generation 3d integrated feature-extracting image sensor,” in *Sensors, 2011 IEEE*, pp. 1933–1936, IEEE, 2011.

[122] X. Zhang, S. Wang, S. Chen, and J. Yuan, “A smart event-clustering motion-detection image sensor for visual object tracking applications,” *under review in IEEE Sensors Journal*.