Edge Detection in a Pixel Array Circuit

Jian Zhang

A Thesis

in

The Department

of

Electrical and Computer Engineering

Presented in Partial Fulfilment of the Requirements
for the Degree of Master of Applied Science at
Concordia University
Montreal, Quebec, Canada

September 2004

Your file Votre référence
ISBN: 0-612-94718-1
Our file Notre référence
ISBN: 0-612-94718-1

ABSTRACT

Edge Detection in a Pixel Array Circuit

Jian Zhang

Edge detection is commonly used to extract important features of an image, while providing a compression of image data, and thus it is one of the most important operations in image processing systems. Such a detection is usually implemented in a system of data processor(s) with its input image signals provided by a camera. However, in many applications, in which the constraints of speed, power and size of the hard-ware are critical, it will be favorable to implement the edge detection in a smart sensor, i.e. a pixel array circuit, in which each pixel operates in parallel with other to perform signal acquisition and processing.

In this thesis, the design of a CMOS pixel array for edge detection is presented. The work of this thesis is in the two aspects, detection algorithms and circuit implementations. A new version of the algorithm for edge detection has been proposed, aiming at facilitating its implementation in an integrated circuit with simple inter-pixel connections and processing units in the pixels. It has been demonstrated that the proposed algorithm can result in a quality of the detection as good as the most commonly used detection algorithms. To implement this algorithm, a pixel array circuit has been designed with simple current-mode modules and logic gates in each pixel. The research effect in the circuit aspect is on solving the problems of the transistor mismatch, which makes identically-designed units behave non-uniformly, and charge injection in the current-mode circuits. With special compensation schemes implemented in the pixel circuit, the uniformity of the processing units integrated in the pixel array increased more than ten times, which improves significantly the quality of the operations in the process of the detection in the circuits. The pixel array circuit can be easily implemented with a standard CMOS technology.

Acknowledgements

I would like to express my gratitude to the following people, who have helped me during this research.

I wish to express my sincere appreciation and gratitude to my Supervisor, Dr. Chunyan Wang, for her guidance, support and encouragement throughout my graduate study. Dr. Wang has motivated me so much with her dedication to this work.

The discussions with my colleagues, Bing Qiu, David Claveau, Yuling Yang and Yuelin Cui, have contributed substantially to this research. I would like to thank them all for their assistance and support along my study.

Finally, I owe special gratitude to my wife, Yuwen Zhao, and my parents during the past years. Without their enormous and unconditional love, understanding and support, I would have never made it so far. I am forever grateful to them.

Table of Contents

List of Figures

List of Tables

1

Introduction

1.1 Image Pre-processing and Smart Sensors

Image processing is widely used in many signal detection systems, such as motion detection and face recognition. An image normally includes a large volume of data while the processing for detection usually deals with only those related to the features of the image. Thus, the image pre-processing, such as edge detection, extracting those features and compressing the data, is usually the first step of image processing after the acquisition. Compared with the original image, the amount of data in the edge map is significantly reduced by extracting the information related to the important spatial structural properties, which facilitates the following processes and their implementation.

A traditional vision system, consisting of a camera and general-purpose or digital signal processor, can perform edge detection. However, such a system requires a complex hardware structure, and a large amount of data needs to be transferred between the camera and processor. Thus, they may not be suitable for those applications that require a small volume, low power dissipation, and/or high detecting speed.

An alternative to a camera-processor system for image acquisition and processing, is a smart sensor scheme. A smart sensor is a circuit designed for the application targeted for, in which photoreceptors and processing circuits are integrated in a pixel array. Lyon optical mouse [1] is one of early smart sensor examples. The structure of a smart sensor is shown in Figure 1.1, consisting of an array of pixels. The light signal is converted to a voltage or current signal by the photoreceptor, and then processed by the circuits in each pixel. Thus, all the pixels in the array perform the operations at the same time, which makes the overall signal processing time short. If the circuits in each pixel is simple enough to occupy a very small space, a high density array can be integrated in a single chip for a high resolution image acquisition and pre-processing. This approach can help to achieve a low-power and high-speed image processing in a small size system.

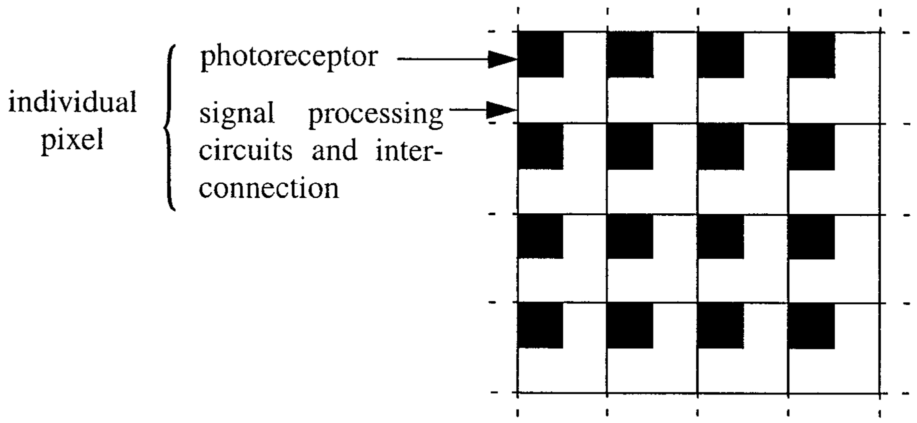


Figure 1.1 Smart sensor array. Signal sensing and processing are performed by the photoreceptor and circuits in each pixel.

1.2 Objectives of the Thesis

Edge detection, compressing the image data and producing the edge map, is one of important stages in the image pre-processing, and widely used in many fields, such as the machine vision and biomedical application [2] [3] [4] [5]. It is a relatively simple processing and suitable to be implemented in a pixel array. Thus, in this thesis, using a smart sensor structure shown in Figure 1.1, a CMOS pixel array for edge detection is going to be studied.

In the design of a pixel array for edge detection, there are some problems to be addressed. They are in two aspects, detection algorithms and hardware implementations.

Edge detection is implemented by the processing circuits and interconnections in a pixel array. As mentioned previously, the processing circuits and interconnections have to be simple for a high pixel density in a chip. This requires an algorithm suitable for such an implementation. Most currently used edge detection algorithms are not suitable for an easy implementation in a smart sensor circuit. Therefore, a study of detection algorithms needs to be conducted in order to identify an algorithm to be both effective and implementable with the VLSI approach.

In the hardware implementations, the requirement of the reasonable resolution limits the silicon area per pixel. The analog processing is often used in the pixel circuit design to minimize the silicon area per pixel. As analog computations are usually based on the device characteristics, which are affected by many elements, such as the process variation and temperature, the characteristics of the processing circuits in a pixel array can have significant spatial variation. In addition, integrated photoreceptors have to be very small due to the resolution requirement. They can thus produce very weak currents, in the range of nano-Ampere or below under a normal light intensity. The transistors in the processing circuits can be driven in the weak inversion mode, which makes the variation of the circuit characteristics serious in the pixel array. Therefore, in the pixel array design, it is indispensable to analyze and solve problems affecting the processing accuracy of the pixel array circuit, in particular those related to the non-uniformity of the device characteristics in the weak current situation.

Addressing the problems presented above, the work of this thesis is first to study the detection algorithms so as to have one that adapts to the implementations in the pixel array, and then to propose a circuit structure for the implementation. Current mode modules will be used in the circuit design and the characteristics of these modules will be studied.

1.3 Organization of the Thesis

This thesis presents the design of a CMOS pixel array for edge detection. The thesis is organized as follows.

Chapter 2 comprises two parts. It first describes algorithms for edge detection, and then presents the work done about pixel circuit designs focusing on two aspects, mismatch/compensation and current-mode processing. Chapter 3 presents an edge detection algorithm adapted to our design and implementation in a pixel array circuit. Chapter 4 is dedicated to the design of a pixel array to implement the algorithm proposed in Chapter 3. The problems of weak current operations and transistor mismatch are addressed and solutions to the problems are presented in this chapter. Chapter 5 concludes the thesis by highlighting the contributions in this thesis and possible future work in the subject area of the thesis is suggested.

2

Background and Related Work

2.1 Introduction

Edge detection of an image is usually performed by calculating the signals of the neighbors around pixels. The procedure of calculation is applied to all the pixels. Thus, it is suitable to implement such a procedure in each pixel so that detection can be effective in a parallel manner. This approach of implementing an edge detection can provide many advantages, in terms of the speed, power dissipation and efficiency of the computation for specific applications, over a camera-processor system.

In a pixel array, the processing circuits and interconnections perform the edge detection operations, therefore, they have to be simple for a high pixel density in a chip. The algorithm to be implemented in the pixel array must meet such requirements in addition to its effectiveness for edge detection. Thus, a study of edge detection algorithms needs to be conducted.

It is very important to have the uniformed characteristics of the processing circuits for a high quality of the signal processing using a pixel array. The device mismatch in CMOS technology hinders the accuracy of the processing circuits, especially in the case of weak photocurrents. Thus, compensation schemes are needed for the pixel circuit design to have a uniformity of the characteristics. Besides, the limited silicon area per pixel requires new methods of circuit implementations. The current-mode processing may satisfy such a requirement with the photocurrent inputs in a pixel array.

In the next sub-chapter, edge detection algorithms will be first discussed. Previous work concerning two circuit design issues in the pixel array, the mismatch/compensation and current-mode processing, are also presented.

2.2 Edge Detection Algorithms

An edge in an image is defined as the boundary between two regions with relatively distinct levels of the light intensity. Based on the gradient of the light intensity between two regions, edges in an image can be classified as two patterns, sharp and ramped edges. Figure 2.1 shows examples of such edges. In Figure 2.1(a), the levels of the light intensity change significantly between the two regions, while the levels of the light intensity change gradually in Figure 2.1(b). The differences of the light intensity among the pixels are computed by a spatial convolution with pre-defined masks.

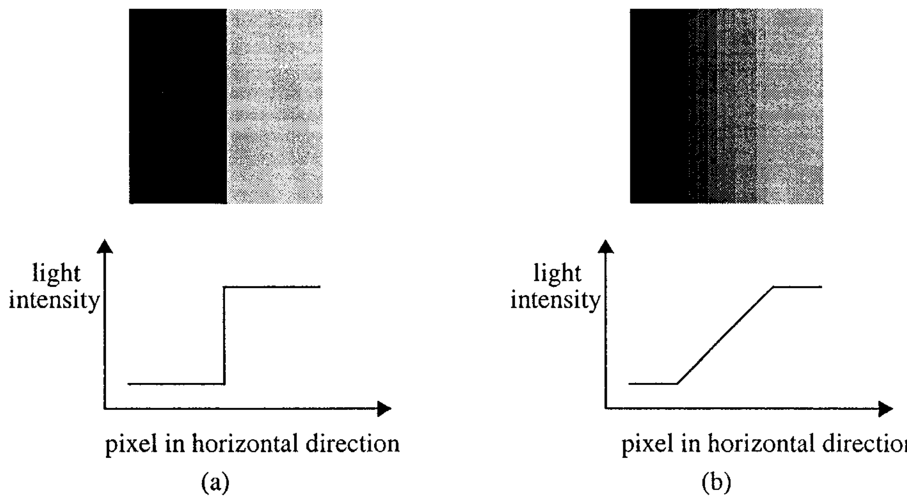


Figure 2.1 Two types of edges in the horizontal direction. (a) sharp edge (b) ramped edge. The black and white color represent the low and upper bounds of the light intensity.

Considering the ramped edges in an image, most of the algorithms for edge detection use masks with a 3x3 size to compute the differences of the light intensity among neighboring pixels. Some examples are the Prewitt [6] and Sobel[7] algorithms of which the masks are shown in Figures 2.2 and 2.3. Evidently, these 3x3-masks require for each pixel to have an access to eight neighboring pixels. Such a requirement may not be difficult in the software programming but leads to dense metal interconnections if the algorithm is implemented in a pixel array [8][9].

NW	N	NE
W	C	E
SW	S	SE

1	0	-1
1	0	-1
1	0	-1

M_a

1	1	1
0	0	0
-1	-1	-1

M_b

Figure 2.2 Masks for the Prewitt algorithm

NW	N	NE
W	C	E
SW	S	SE

1	0	-1
2	0	-2
1	0	-1

M_a

1	2	1
0	0	0
-1	-2	-1

M_b

Figure 2.3 Masks for the Sobel algorithm

The Roberts algorithm [10], with the masks shown in Figure 3.4, can reduce the complexity of the circuit since its masks are of 2x2 format. However, the Robert algorithm may not be efficient for detecting the ramped edges because of its two-pixel width masks.

NW	NE
SW	SE

1	0
0	-1

M_a

0	-1
1	0

M_b

Figure 2.4 Masks for the Roberts algorithm

One of special masks [11] shown in Figure 2.5 is three-pixel wide but reduces number of the signals accessed by the central pixel. Using this mask, four interconnections per pixel are needed in a pixel array, which can simplify the implementations to a certain extent.

NW	N	NE
W	C	E
SW	S	SE

-1	0	1
0	0	0
1	0	-1

Figure 2.5 One of the masks used in the work[11]

Besides the interconnection issue related to the masks used, the operation procedure of an edge detection algorithms is relevant to the complexity of the pixel circuit in the implementation. In the operation procedure of the commonly used algorithms, the differences of the light intensity among neighboring pixels are computed. Then, the differences are summed up to decide edge positions since the summation includes more information in different directions and averages the differences for the purpose of a spatial low-pass filtering. In such a procedure, several different types of the operators, such as subtraction, addition and the absolute value operation, need to be performed, which leads to a complicated hardware structure in the pixel array implementation. Thus, to make an algorithm implementable with the circuit approach, the procedure of the calculations must be redesigned.

2.3 Pixel Circuit Design

Analog circuits are usually used to perform the computations of the algorithm implemented in a pixel array in a space-efficient manner. However, the uniformity of these circuits' characteristics is a critical issue for a high quality of signal processing using a pixel array. It is challenged by the transistor mismatch due to the imperfection of the fabrication process, particularly for the circuits in the weak current operations. Another constraint in the pixel circuit design is the limitation of the area per pixel. The photoreceptor designed in such a limited space is small, which produces a weak current. To process such a weak current with a minimal number of transistors per pixel, new methods of analog techniques are needed. Particularly, the current-mode processing can be suitable for the pixel circuit with photocurrent inputs. Thus, in this section, two issues, the mismatch/compensation and current-mode processing in the circuit design and existing work will be discussed.

2.3.1 Mismatch and Compensation

In the pixel array design, the light signal is sensed by a photoreceptor. In a standard CMOS technology, the p-n junction diode can be used as a photoreceptor. Due to the silicon area limitation per pixel, this photodiode is small and can only produce a weak current under the normal light intensity, in a range of nano-Ampere or below. Thus, the transistor in such a weak current operation is driven in the weak inversion mode (also called as the subthreshold region). In the weak inversion mode, the drain current I_D of a transistor versus its voltages is approximately given by the exponential relationship [12],

$$I_D = 2n\frac{W}{L}\mu_n C_{ox}\Phi_T^2 e^{\left(\frac{V_G - V_{T0}}{n\Phi_T}\right)}\left(e^{\frac{-V_S}{\Phi_T}} - e^{\frac{-V_D}{\Phi_T}}\right) \tag{2.1}$$

where, n is a slope factor between 1 and 2, W and L are the channel width and length of the transistor, m_n is the mobility of charge carriers, C_{ox} is the gate oxide capacitance per unit area, Φ_T is the thermal voltage, V_{T0} is the gate threshold voltage with zero source-to-substrate voltage. The transistor mismatch is an inherent problem in CMOS technologies. The process variation changes the values of W, L and C_{ox}. From Equation (2.1), it can be seen that the variations of these values will make the transistor current vary from its designed value. In the pixel array, the effect of the transistor mismatch are in two aspects, intra-pixel and inter-pixel. Inside a pixel, for example, an output current of a current mirror will differ from its input current because the two MOSFETS can not be made identical. Among pixels, the transconductances of transistors in different pixels will vary from one and another, which produce different currents when a same voltage applied to their gate terminals. The mismatch problem becomes serious when the transistors work in the weak inversion mode, and the current disparity among a pair of transistors in a current mirror structure can reach 50% [13]. Thus, the characteristics of the processing circuits in the pixel array can be seriously affected by the mismatch effect.

The transistor mismatch can be reduced by the special layout technique, for example, using multi-finger transistors and symmetry [14]. But all these technique need a large silicon area, which contradicts the silicon area minimization requirement in the pixel circuit design. Such techniques can hardly be used in the pixel array.

The non-uniformity of the circuit characteristics resulted from the mismatch in a pixel array can also be corrected by a global compensation strategy: applying same input signals to all the pixel circuits and ensuring that all pixel outputs are same. Thus, the mismatch effect on the circuit can be suppressed at the signal level concerned.

There are several circuit designs in which this technique to reduce the mismatch effect was applied in different ways. In the Mead's adaptive retina [15], a feedback loop has been constructed to compensate for the mismatch effect by changing the threshold voltage of the transistor, as shown in Figure 2.6. This has been realized by the ultraviolet activated coupler. Such floating-gate transistors are used in some other image senor design [16][17][18]. However a double poly technology is needed for the circuit fabrication.

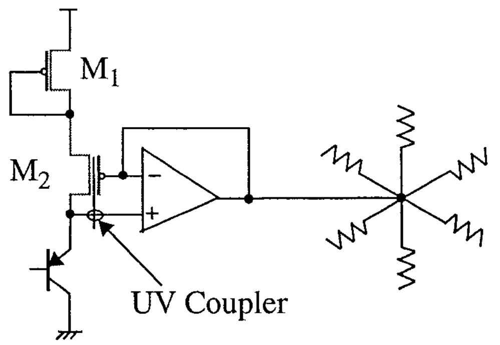


Figure 2.6 Pixel circuit in the Mead's adaptive retina [15]. The output variation among pixels is corrected by adjusting the transistor threshold voltage with the ultraviolet light.

In the work [19], a feedback loop is used to keep the outputs of all the pixels identical when the photodiodes receive the same light signals, as shown in Figure 2.7. The comparator threshold value is memorized in a capacitor. Then, when the photodiode receives a light signal, the output of the pixel will go to the highest or lowest voltage level, depending to a open loop comparison result between the pre-sampled and currently received signals.

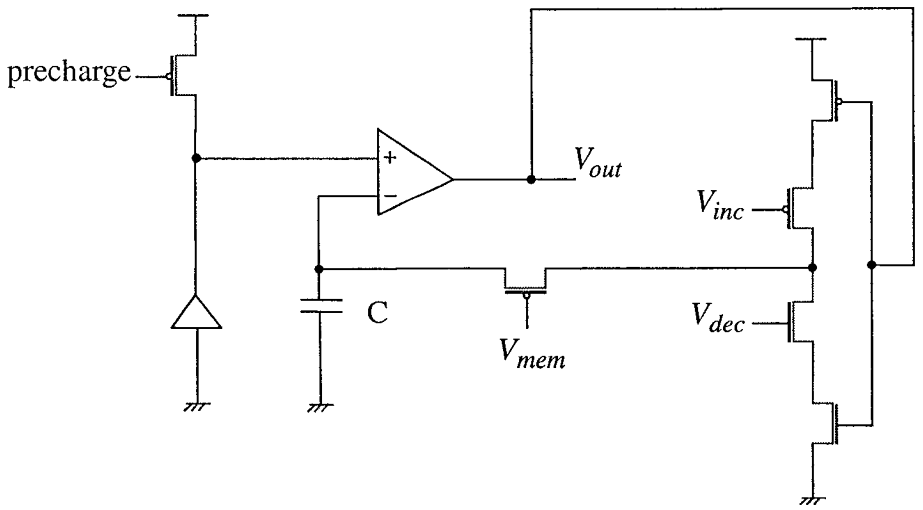


Figure 2.7 Comparator with the mismatch suppressed in the detection sensor [19].

Using the same global compensation strategy mentioned above, a current comparator circuit shown in Figure 2.8 is proposed to make the comparator threshold insensitive to the transistor mismatch [20]. The comparator can set up its own steady state adapting to its circuit parameters. By such a compensation scheme, the non-uniformity of the threshold value in an array of such comparators can be greatly reduced.

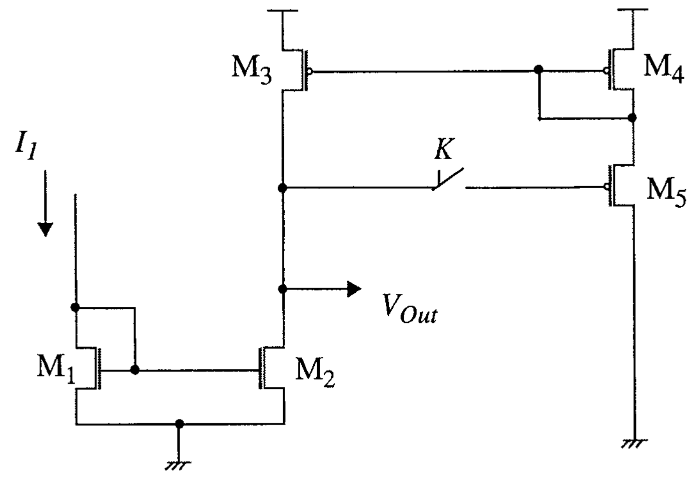


Figure 2.8 Current comparator circuit with its threshold value insensitive to the transistor mismatch [20].

2.3.2 Current-mode Processing

The current-mode processing has been widely used in many applications, such as A/D and D/A converters and analog filters. Besides its wide bandwidth feature, the current-mode approach may result in a simple circuit structure in case that the processed input signal is current. In a pixel array, photodiodes convert the light signals to currents. Thus, using the current-mode modules for the processing circuits in the pixel array, the design can be simplified and a better performance can be expected.

The translinear circuit principle introduced by Gilbert [21] is one of techniques for the current-mode processing, and has been used to design circuits to perform the required operations in the image sensors [16][17]. However, in this case, a current-to-voltage conversion is needed and the circuit operation is related with the transistor transconductance. Thus, the pixel circuit may not be compact and suffer from the mismatch problem.

Another approach to process the photocurrents in a pixel array is the switched-current technique that often involves current mirrors or current memories. Figure 2.9 shows an example of using the current memory and MOS switches to design a multi-purpose sensor [22]. A variety of functions can be performed by combinations of the control signals $S_5S_4S_3S_2S_1S_1$. Using such a technique, the photocurrents can be directly processed in a simple circuit structure and the mismatch problem can be also avoided by using current memories.

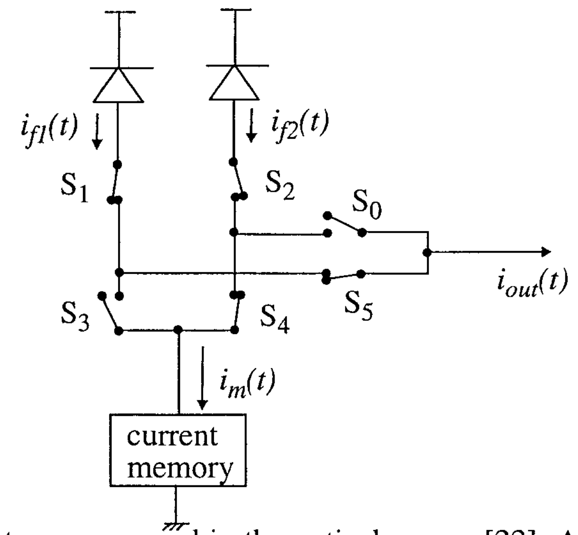


Figure 2.9 Current memory used in the optical sensor [22]. A variety of functions can be performed. For example, if we apply, to the sensor, first a combination of control signals $S_5S_4S_3S_2S_1S_0$ = 110110, followed by the combination $S_5S_4S_3S_2S_1S_0$ = 101010, at the output, we have $i_{out} = i_{f1} - i_{f2}$.

Current memories, combined with MOS switches, are capable of performing many kinds of calculations, such as addition, subtraction, multiplication, division and integration [23]. However, the accuracy of a current memory mainly depends on the ability to protect the voltage in the critical gate terminal, once the voltage is established, from the various noise sources. As shown in Equation (2.1), in the weak inversion mode, the transistor current is exponential to the gate voltage. A variation of several millivolts at the gate voltage will produce a big current change. Among the noise sources, the charge injection from the switch transistor contributes a big part of such a voltage variation. Thus, if current memories can be used in the pixel circuit, the charge injection problem must be solved.

One of the potential solutions to the charge injection problem is the matching scheme to cancel the charge injection effect. This cancellation can be done by introducing the extra charge or current. A dummy switch technique [24], as shown in Figure 2.10, is based on the charge cancellation. The dummy transistor M_2 is chosen to be half size of the switch transistor M_1 to absorb the charge released from the channel of transistor M_1, assuming that the charge in the channel is injected equally to its drain and source sides when the transistor M_1 is turned off. The class-AB two step sampling technique (S^2I) [25][26][27] uses the error current cancellation scheme to improve the current memory accuracy. Its operating principle can be shown in Figure 2.11. The process of memorizing the input current I_{in} is made in two steps. During the first step the input current I_{in} is sampled and stored in the NMOS memory with an error. This is followed by a second step during which the error in the first step is derived from the input current and the current of the NMOS memory, and memorized in the PMOS memory. In the output phase, both the PMOS and NMOS memories deliver their memorized currents in a way that cancels the errors in the two memories. Thus an accurate copy I_o of the input current is produced.

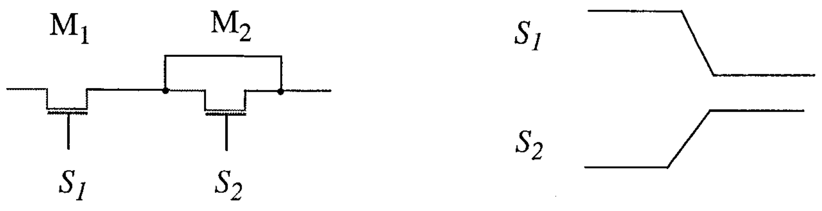


Figure 2.10 Dummy switch technique to reduce charge injection [24]. The charge released from the channel of the transistor M_1 is absorbed by the dummy transistor M_2.

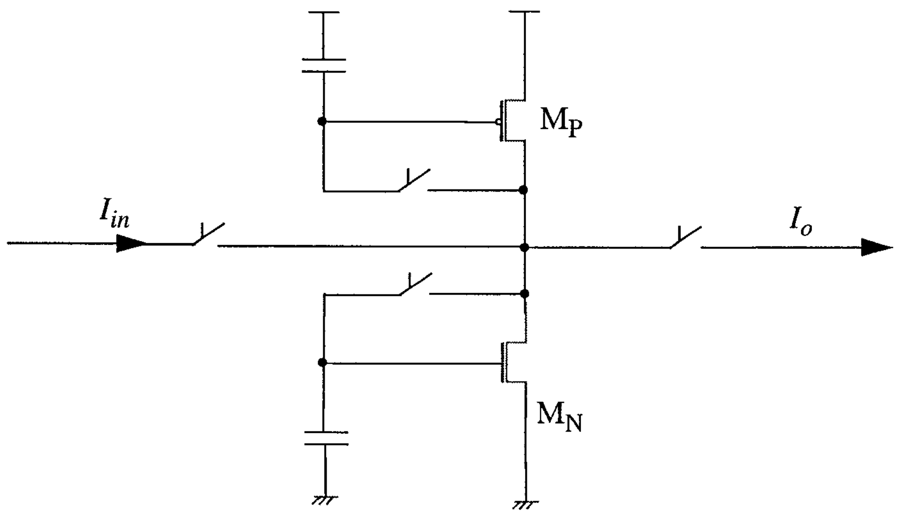


Figure 2.11 Class-AB two-step sampling (S^2I) [26]. The errors in the two memory are cancelled each other to make the output current I_o equal to the input current I_{in}.

Another approach to improve the current memory accuracy against the charge injection problem is diverting the charge released from the switch transistor away from the critical gate terminal. Thus the charge injected to this node will be minimized and the current memory accuracy is improved with less voltage variation at this node. According to such an idea, a current memory circuit with the charge injection reduction [28] was proposed as shown in Figure 2.12. The current memory accuracy is improved by using three capacitors and two steps of current sampling. The charge released from the switch S_2 during the first step sampling is eliminated by the loop through C_2. When the switch S_3 is turned off during the second step sampling, the resulting voltage variation at the critical gate terminal of the transistor M_1 is attenuated by the capacitive divider.

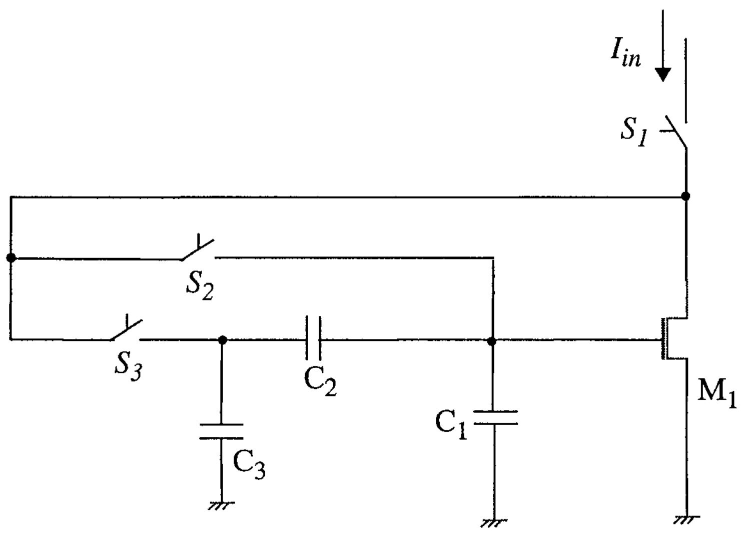


Figure 2.12 Current memory with the charge injection reduced [28]. Three capacitors C_1, C_2 and C_3 attenuate the gate voltage variation of the transistor M_1.

2.4 Summary

In this chapter, the background and related work about the pixel array for edge detection have been studied. This study includes two aspects, edge detection algorithms and the circuit design issues.

The commonly used edge detection algorithms have been discussed, focused on the convolution masks that are related to the metal interconnections in the circuit implementation. A special 3x3-mask can be adopted to the design of the circuit while the computation procedure involving this mask needs to be improved to facilitate a simple hardware implementation in a limited pixel area. Thus, a new version of detection algorithm should be proposed for the work in this thesis.

In the circuit design, the issues of the mismatch effect on the characteristics of pixel circuits and current-mode circuits have been discussed. Some existing work related to these topics are presented. Although some of the techniques provided some effectiveness in solving the problems addressed, they are yet to be improved in the circuit design of this project.

In the next chapter, the algorithm to be implemented in a pixel array are going to be described.

3

Proposed Algorithm for Edge Detection

3.1 Introduction

Edge detection is one of important operations in image processing. Since it extracts the important features of an image and the analysis based on edge detection is insensitive to the change of the overall illumination level, many high-level image processing are based on edge maps of original images. Therefore, it is very important to develop efficient methods for edge detection.

As discussed in Chapter 2, the algorithms with three pixel wide masks, such as Prewitt and Sobel, can efficiently detect ramped edges but will introduce complex interconnections during the hardware implementation. In order to reduce the interconnection complexity and preserve the efficiency to detect ramped edges, a special three-pixel wide mask [11] is the starting point of our study of an effective edge detection algorithm to be implemented in a pixel array.

3.2 Proposed Algorithm

In this section, we present a new version of an algorithm for edge detection aiming at implementation in the pixel array using a CMOS technology. The proposed algorithm uses masks shown in Figure 3.1, which are derived from one of masks in the work [11]. It will be possible, by using these masks, to make the algorithm efficient in detection of ramped edges, and to implement the algorithm in a relatively simple array circuit with four interconnections per pixel. In addition to these masks used, a special operation procedure is needed to explore the information of the accessible pixels and to facilitate the circuit implementation.

NW		NE
	C	
SW		SE

1	0	1
0	0	0
-1	0	-1

M_a

1	0	-1
0	0	0
1	0	-1

M_b

Figure 3.1 Masks for the proposed algorithm. The difference between the signals of pixels in the horizontal, vertical or diagonal direction can be calculated.

3.2.1 Operation Procedure

In most of the algorithms for edge detection, spatial convolutions with well-defined masks are used as shown in Figure 3.2. To execute such a procedure, in each pixel, the signals of the neighboring pixels are used, according to the masks, for calculating the signal differences. The signal differences are summed up. The absolute value of the summation result is then compared with a predefined threshold to determine the binary output signal representing edge information.

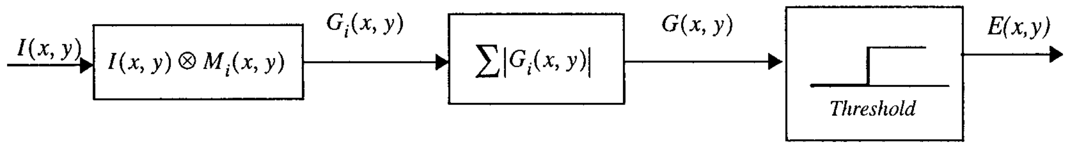


Figure 3.2 General procedure for the classical algorithms. *I(x,y)* is the signal vector in a pixel array *(x,y)*. *$G_i(x,y)$* is the result of the spatial convolution. *G(x,y)* is the absolute value of the sum of the spatial convolution. *E(x,y)* is the edge map.

The objective of the study of the algorithm is to make it implementable with a certain simplicity in a pixel array in which all the pixels are made to operate in the same manner. In order to minimize the number of circuit modules and that of the inter-pixel connections, the procedure of computation in each pixel needs to be designed to have the operations performed in series.

To implement the general procedure described above in a specific way for a simplicity of the circuit implementation and the quality of the operation, the following two issues should be addressed.

1. In most cases, each pixel needs to receive signals from eight neighbors, which requires dense interconnections if the procedure is implemented in a pixel array. However, using the masks shown in Figure 3.1 will alleviate the problem of dense interconnections under certain conditions.

2. The operations included in this procedure are all analog ones that are sensitive to spatial and temporal noises. The noise compensation schemes may be implemented but they need additional circuit modules, which results in lowering pixel resolution. Thus, it is desirable that the number of analog operators is minimized.

Taking into consideration of the two issues, a procedure for edge detection with repetitive operations is presented in Figure 3.3. In this procedure, the result of each subtraction is compared with a threshold, which generates a binary signal, so that the following processing can be done in a digital manner. Thus, simple operators can be used in the procedure, which can simplify the structure of the circuit implementation. The detail of the procedure is described below.

In the procedure shown in Figure 3.3, as the information from four neighboring pixels is needed to determine if the pixel at the center position C is part of an edge. The two signals I_{i1} and I_{i2} are selected among the signals from these pixels. The selection is processed successively according to a sequence of using twelve masks, M_1, M_2...and M_{12}, shown in Figure 3.4. Applying each of the masks, the difference of these two signals, ΔI_i, is calculated, and then is compared with a predefined threshold I_{th}. If $\Delta I_i \geq I_{th}$, then binary signals E_i is equal to “logic-1”, otherwise E_i is equal to “logic-0”. This operation can be expressed by,

$$E_i = Threshold(\Delta I_i - I_{th})$$
$$= \begin{cases} 1, if \Delta I_i \geq I_{th} \\ 0, if \quad \Delta I_i < I_{th} \end{cases} \tag{3.1}$$

Then the same operation will be repeated up to twelve times with the signals selected according to different masks, M_1, M_2...and M_{12}, which produces a sequence of up to 12 binary signals E_1, E_2...and E_{12}. In the next stage, all these binary signals, are summed up by an accumulation operator in the voting block. The accumulation result $\sum_{i=1}^{12} E_i$ is compared with a threshold N_{th} to produce the signal E representing the edge information.

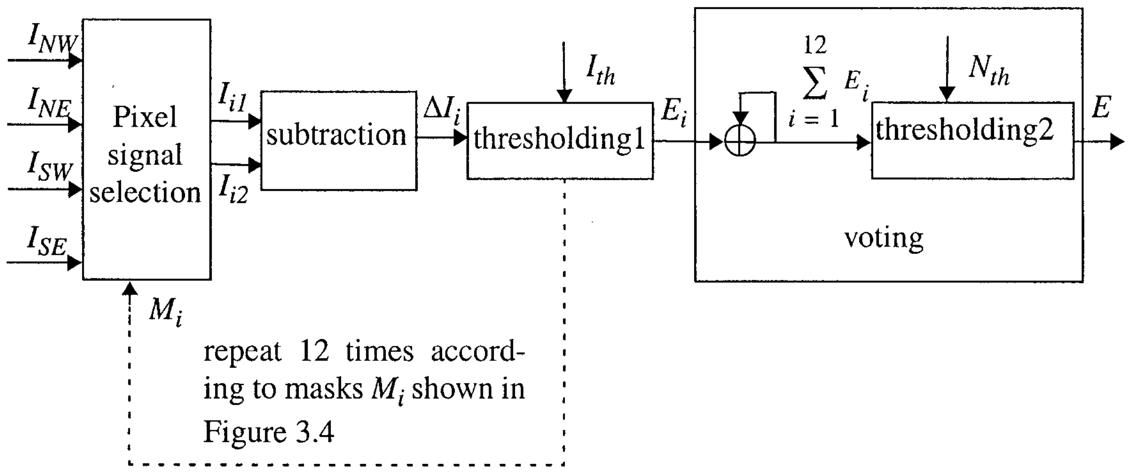
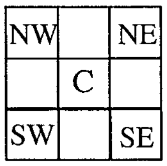


Figure 3.3 Proposed procedure of the algorithm. I_{NW}, I_{NE}, I_{SW} and I_{SE} are the signals from four neighboring pixels. The two signals, I_{i1} and I_{i2} are selected according to the sequence of the masks shown in Figure 3.4.



$$\underset{M_1}{\begin{bmatrix}1&0&0\\0&0&0\\-1&0&0\end{bmatrix}}\quad \underset{M_2}{\begin{bmatrix}-1&0&0\\0&0&0\\1&0&0\end{bmatrix}}\quad \underset{M_3}{\begin{bmatrix}0&0&1\\0&0&0\\0&0&-1\end{bmatrix}}\quad \underset{M_4}{\begin{bmatrix}0&0&-1\\0&0&0\\0&0&1\end{bmatrix}}$$

$$\underset{M_5}{\begin{bmatrix}1&0&-1\\0&0&0\\0&0&0\end{bmatrix}}\quad \underset{M_6}{\begin{bmatrix}-1&0&1\\0&0&0\\0&0&0\end{bmatrix}}\quad \underset{M_7}{\begin{bmatrix}0&0&0\\0&0&0\\1&0&-1\end{bmatrix}}\quad \underset{M_8}{\begin{bmatrix}0&0&0\\0&0&0\\-1&0&1\end{bmatrix}}$$

$$\underset{M_9}{\begin{bmatrix}1&0&0\\0&0&0\\0&0&-1\end{bmatrix}}\quad \underset{M_{10}}{\begin{bmatrix}-1&0&0\\0&0&0\\0&0&1\end{bmatrix}}\quad \underset{M_{11}}{\begin{bmatrix}0&0&-1\\0&0&0\\1&0&0\end{bmatrix}}\quad \underset{M_{12}}{\begin{bmatrix}0&0&1\\0&0&0\\-1&0&0\end{bmatrix}}$$

Figure 3.4 Masks obtained by decomposing those in Figure 3.1. They are used for calculating the signal differences between two neighboring pixels around the center position *C*. The absolute values of the signal differences, in six directions of *NW-SW, NE-SE, NW-NE, SW-SE, NW-SE* and *NE-SW* can be detected using these masks.

It should be noted that using the masks shown in Figure 3.4, each pair of signals are used twice for calculating the differences [29]. For example, M_1 and M_2 are used for calculating $(I_{NW} - I_{SW})$ and $(I_{SW} - I_{NW})$, of which the positive one must be equal to $|I_{NW} - I_{SW}|$. Thus, the absolute value of each of the signal differences in six directions of *NW-SW, NE-SE, NW-NE, SW-SE, NW-SE* and *NE-SW,* as shown in Figure 3.4, is obtained successively. In the proposed procedure, each signal difference ΔI_i, $i = 1, 2$...and 12, is compared with a threshold I_{th}, and twelve binary signals E_1, E_2...and E_{12} are generated. As the negative value of ΔI_i, $i = 1, 2$...and 12, produces the comparison result of "logic-0", among these twelve comparison results, only six, can possibly be "logic-1", which implies that the absolute values of the signal differences in six directions of *NW-SW, NE-SE, NW-NE, SW-SE, NW-SE* and *NE-SW,* are used for edge detection.

The masks in Figure 3.4 show that only the pixel signals at the corner positions in a 3x3 window are used for computing the edge information at the center pixel. Thus, when the algorithm is implemented in a pixel array, each pixel has only four connections to its neighbors, which reduces inter-pixel connections of the circuit. In the operation procedure shown in Figure 3.3, only two operations, the subtraction and thresholding, instead of four operations, the subtraction, summation, absolute value and thresholding for the procedure shown in Figure 3.2, are performed, in a repetitive manner, twelve times, in each of which one of the twelve masks shown in Figure 3.4, is used. These two operations can be implemented in two simple analog blocks, and the rest of the processing can be performed with binary signals. Therefore, when the algorithm is implemented in a pixel array, the circuit can be simplified. The detailed description of the binary processing stage is presented in the following sub-section.

3.2.2 Voting Operation

In the procedure of a classical algorithm for edge detection, the signal differences obtained using the chosen masks are summed up, then the result of summation is compared with a threshold. These operations provide a low-pass filtering feature for reducing the spatial noise, but require analog circuit modules in the implementation, which may cause critical problems, in terms of silicon area per pixel, and increase fixed pattern noise (FPN). A simplified detection method has been reported previously [29], in which an edge decision is made according to the maximal value of the signal differences in all the directions. Although this method can be easily implemented in a pixel array, it does not provide an efficient filtering feature. The development of the proposed algorithm aims at implementing edge detection in a simple circuit structure with the function filtering the spatial noise.

As shown in Figure 3.3, twelve binary signals E_i, $i = 1, 2...$and 12, are generated and they shall be processed to determine the edge signal E. As mentioned previously, half of the twelve logic values are "logic-0" resulting from negative signal differences. Of the other half, some of which can be "logic-1", carry the information of the signal differences in the defined directions. Therefore, it is very important to fully use these signals for determining the edge signal and to explore the possibilities of the threshold operation with N_{th} in the procedure as shown in Figure 3.3.

In the voting block in Figure 3.3, twelve binary signals E_i, $i = 1, 2...$and 12, are summed up. The maximal value of the summation result, $\sum_{i=1}^{12} E_i$, is six. Then, the edge signal E is determined by the comparison of the summation result with a threshold value of N_{th}, $1 \le N_{th} \le 6$, and it can be expressed as,

$$E = \begin{cases} 1, \text{ if } \sum_{i=1}^{12} E_i \ge N_{th} \\ 0, \text{ if } \sum_{i=1}^{12} E_i < N_{th} \end{cases} \tag{3.2}$$

The threshold N_{th} is determined according to the image patterns to be detected. It should be high enough for the procedure to filter out the spatial noise, such as single spots in an image, but not too high to miss edge points. For an image with the normal contrast and illumination, the cases of $N_{th} = 3$ and $N_{th} = 4$ should be discussed.

In Figure 3.5, 3x3 windows are presented to show the cases of $\sum_{i=1}^{12} E_i \ge 3$. In each of these windows, double-arrowed lines represent $|\Delta I_i| > I_{th}$, resulting in $E_i = 1$ in the specified direction. Image patterns shown in Figure 3.6 are some examples of patterns. For a clarity of demonstration, the gray level in these patterns is quantized into seven levels, and the threshold I_{th} is assumed to be one unit of the quantization level. The examples are carefully chosen to show cases in which $\sum_{i=1}^{12} E_i \geq 3$ are barely satisfied. They are critical cases and should be taken into considerations in determining the threshold N_{th}.

If the threshold N_{th} is equal to three, the center pixel in each of the patterns in Figure 3.6 will be considered as an edge point with E equal to 1, as it satisfies $\sum_{i=1}^{12} E_i \geq 3$. However, examining the cases presented in the first three rows, these patterns are more likely to be generated by noise, such as the signal fluctuation in a region or glitches of the light intensity, instead of edges in an image. Thresholding with N_{th} equal to three would include many of such cases, which could produce a considerable number of "fault" edge points. By raising the threshold N_{th} to four, these twelve cases will be filtered out and all the other patterns in Figure 3.6 will be identified as patterns with the edge point at the center pixel.

The choice of the threshold N_{th} should also follow so-called "a majority rule". The signal E_i equal to "logic-1" indicates that the gradient of the signal in one direction among those shown in Figure 3.3, is significant. If the signal gradient in most of the directions across the center pixel is significant, it is likely that the center pixel is an edge point. According to the majority rule, if at least four out of twelve signals E_i, half of which are zeros, are equal to "logic-1", the edge signal E will be set "logic-1", otherwise E will be set "logic-0". Thus, the threshold N_{th} equal to four is suitable for most of the cases. If the input images have special intensity change patterns, the threshold N_{th} may need to be adjusted to a higher or lower level.

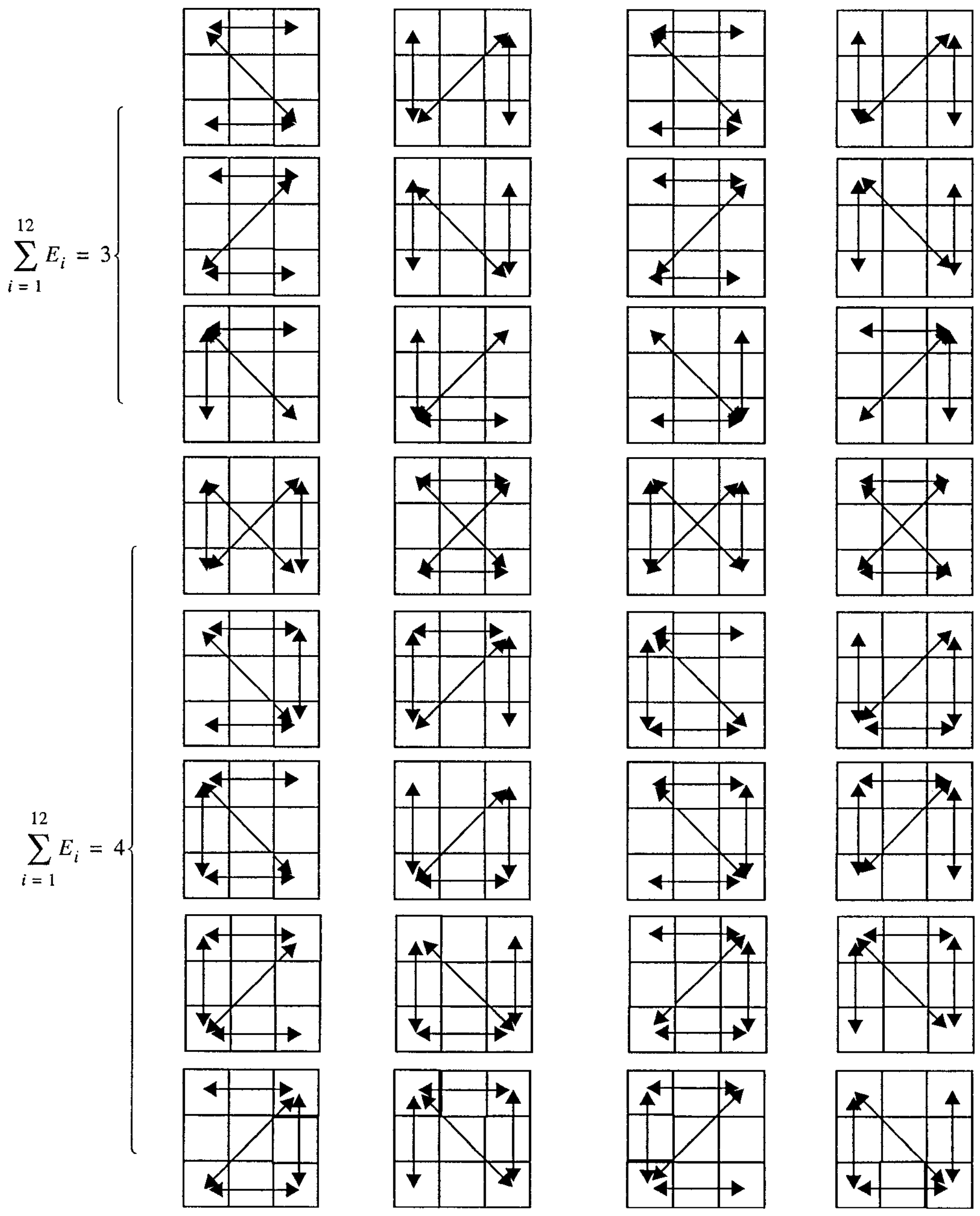


Figure 3.5 All the cases of $\sum_{i=1}^{12} E_i = 3$, 4, 5 and 6. Each of the double-arrowed lines represents $|\Delta I_i| > I_{th}$, resulting in $E_i = 1$ in the specified direction.

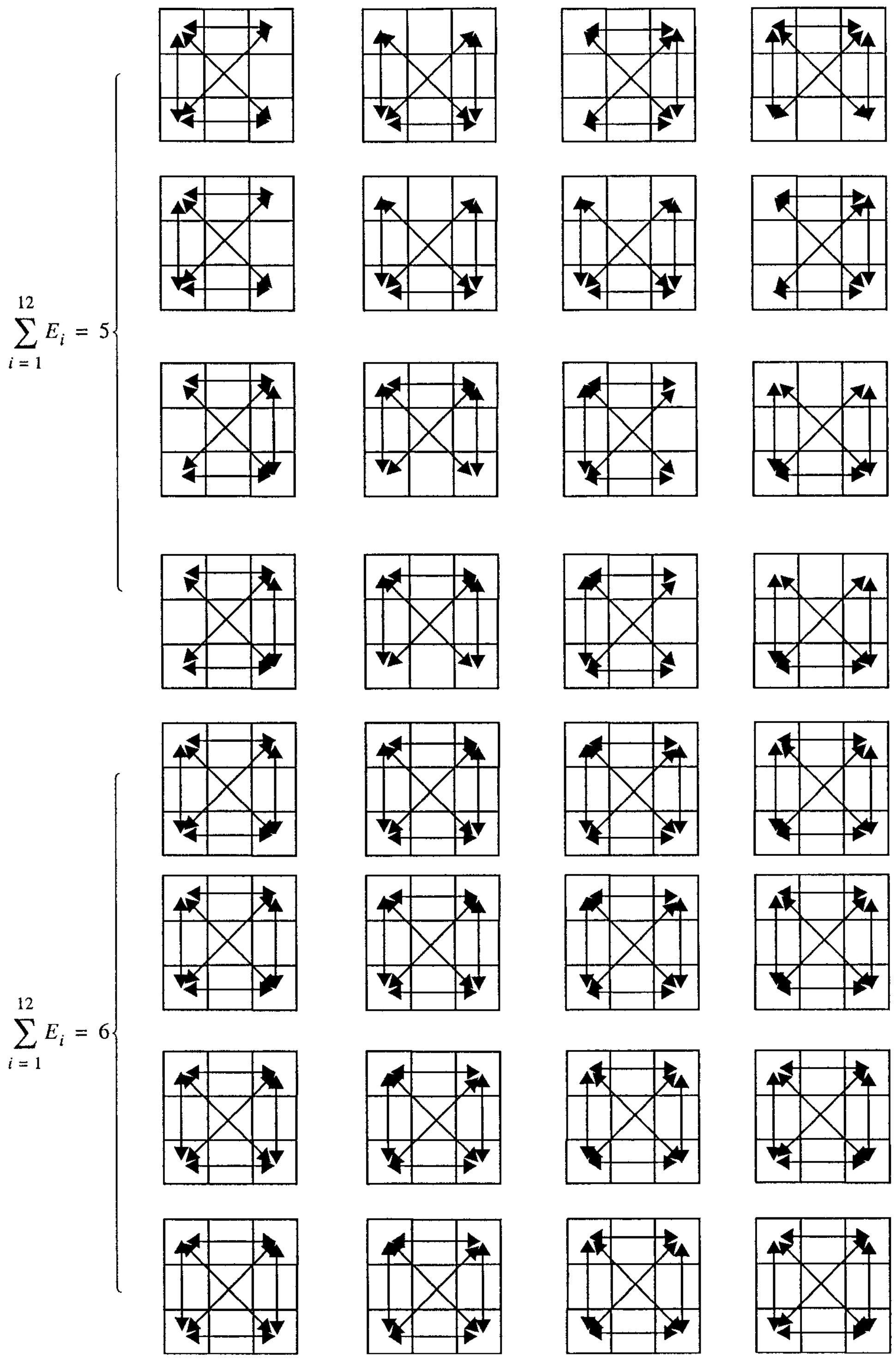


Figure 3.5 *(continued)*

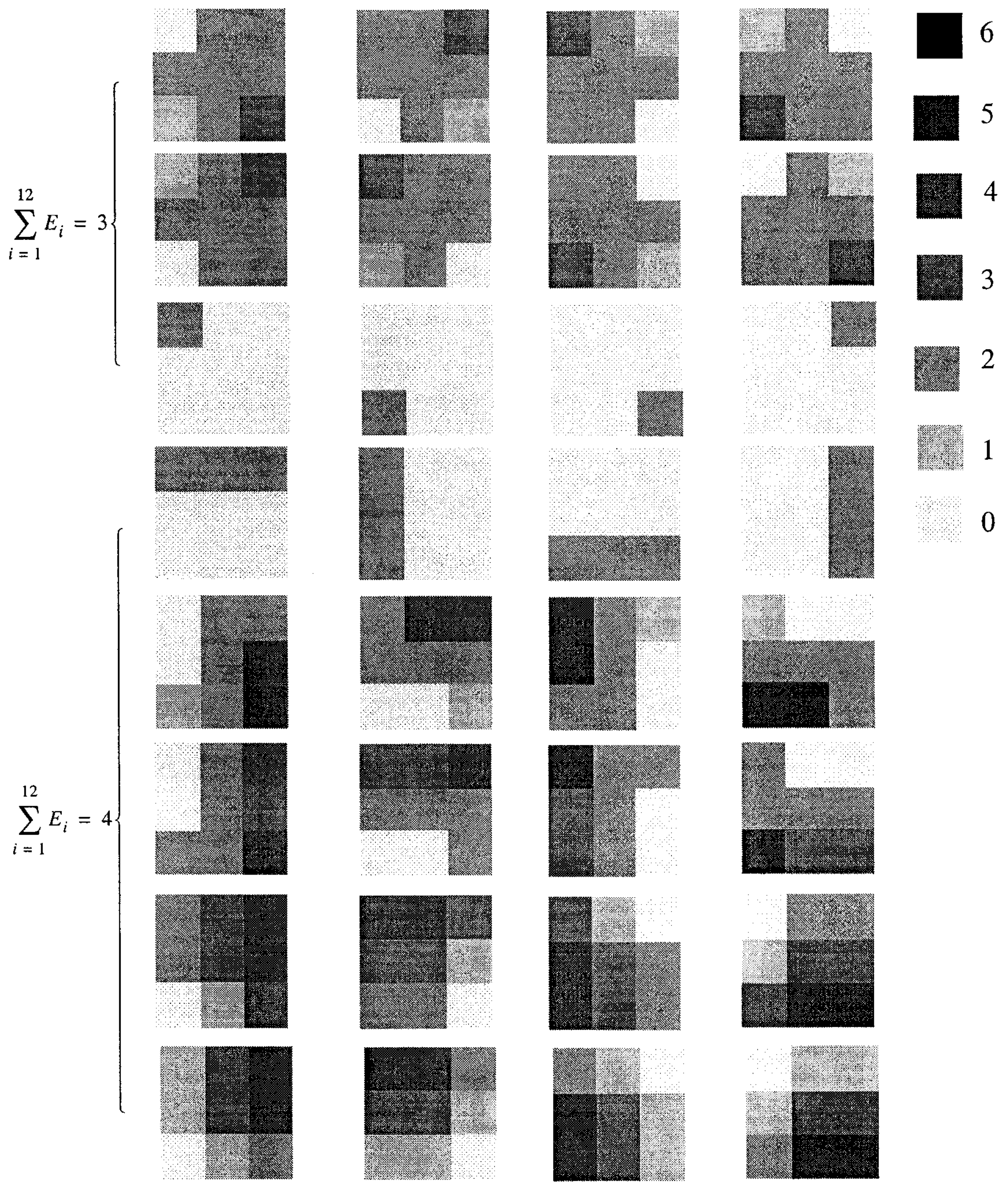


Figure 3.6 Image patterns in 3x3 windows. To express the cases clearly, the gray level of the pixels is quantized into seven levels shown in the right side. If I_{th} is one unit of the quantization level and $N_{th} = 3$, the edge signal E of the center pixel in each pattern will be "logic-1". If $N_{th} = 4$, the edge signal E in the patterns of the first three rows will be "logic-0", and that of the other rows will be "logic-1".

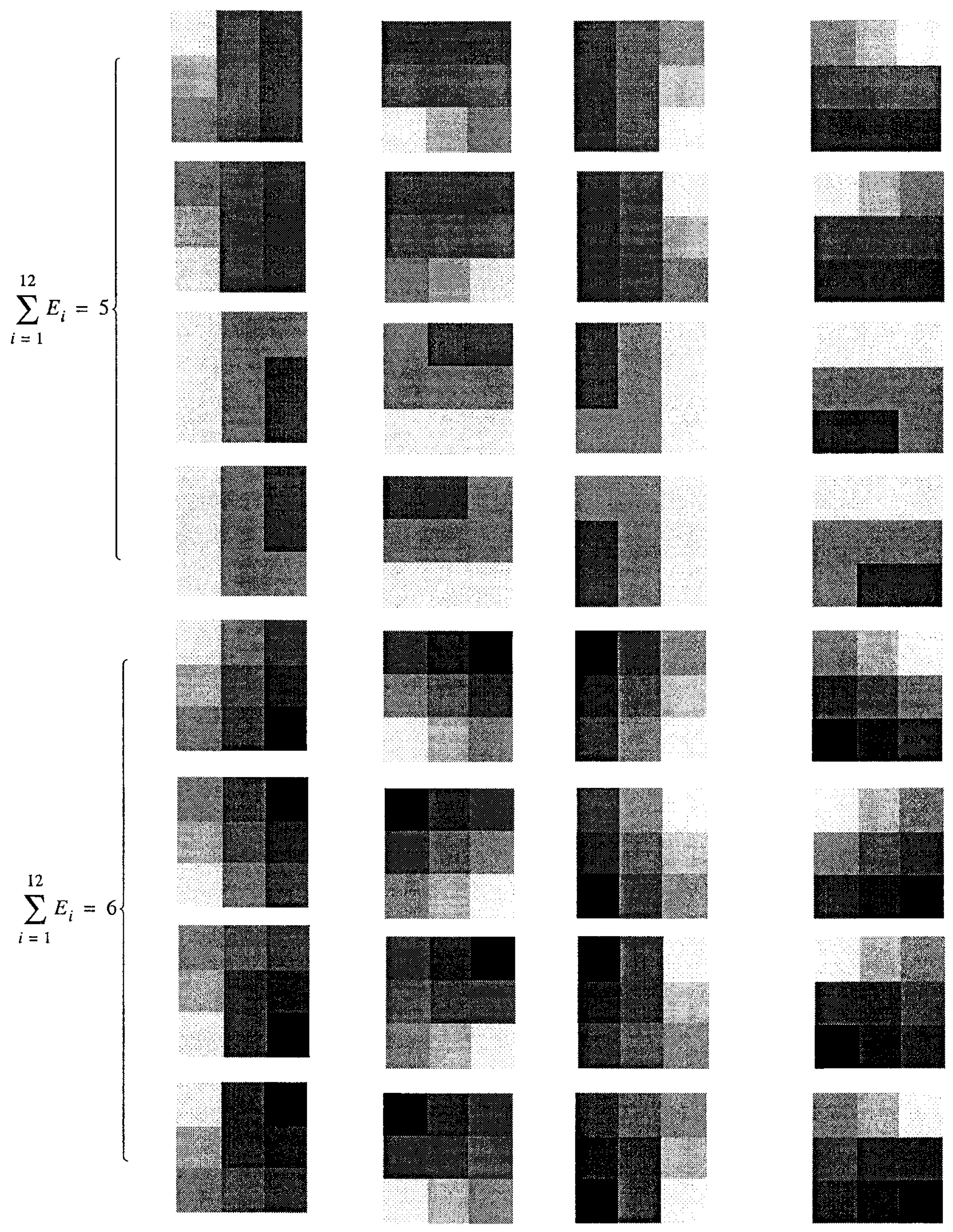


Figure 3.6 *(continued)*

The voting operation explores the signal information of all the pixels in the masks shown in Figure 3.1 to find the edge information. It has a low-pass filtering feature and the spatial noise can be filtered out by carefully choosing the threshold N_{th}. Thus, though the proposed algorithm can lead to a simplified structure for implementation in a pixel array, the performance of edge detection is expected to be comparable with that of the classical algorithms, which can be demonstrated by the simulation results in the next sub-chapter.

3.3 Simulation results

The proposed algorithm has been simulated for an evaluation of its effectiveness. As the Sobel and Prewitt algorithms are commonly used for edge detection, they are chosen as the references for this evaluation. The results are shown in Figure 3.7. It can be seen that the edge map obtained using the proposed algorithm has almost the same quality as those obtained using the other two, although the proposed algorithm requires less computation and data connection.

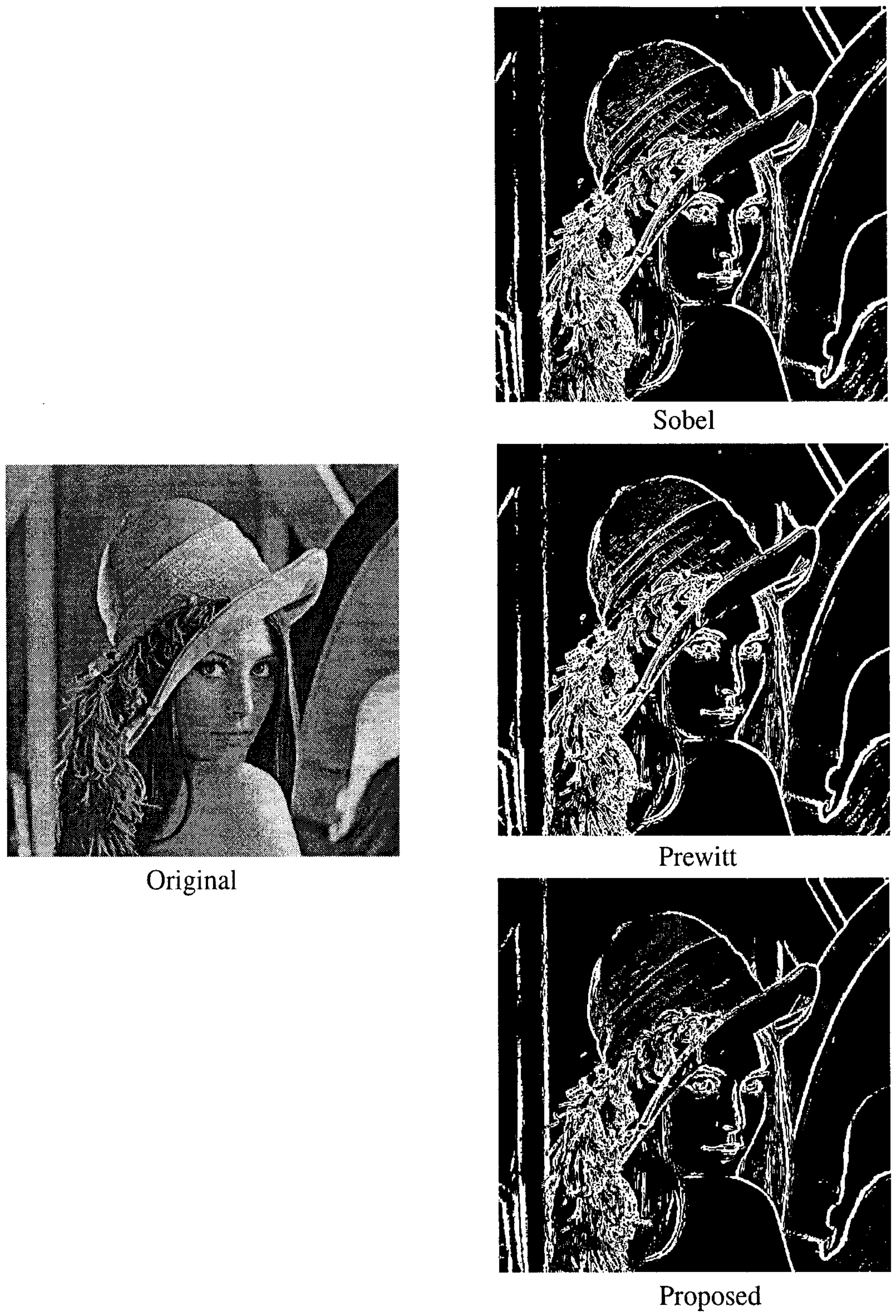


Figure 3.7 Lena original image and simulation results obtained using the Sobel, Prewitt, and proposed algorithms.

3.3 Summary

In this chapter, algorithms for image edge detection are studied, particularly in the view of implementation in a pixel array. As most of the currently used algorithms, the three-pixel wide masks are the basis of the proposed algorithm. Moreover, special masks of three pixel width are chosen, aiming at simplifying interconnections among pixels in the pixel array implementation.

To minimize the area required for every pixel in the implementation of the algorithm, an operation procedure of the detection has been proposed. This procedure involves only simple and repetitive operations. The analog part is minimized to avoid the accuracy problem in the implementation while the digital part is kept simple. Compared with the procedure of the classical algorithms, this procedure requires only two simple analog operators and one simple binary voting block. Besides, a voting block is proposed to replace analog summation and comparison in most of the algorithms, and to produce the edge signal with a comparable quality as that of the classical algorithms.

The efficiency of the proposed algorithm has been verified by the simulation. The comparison of the simulation results with those obtained using two classical algorithm, Sobel and Prewitt, shows that the proposed algorithm can lead to an effective and efficient edge detection as that of the other two.

In the following chapter, an implementation of the algorithm in a pixel array is presented.

4

Pixel Circuit Design

4.1 Introduction

In Chapter 3, an algorithm for edge detection has been proposed to facilitate its implementations in a pixel array. In this chapter, the design of a pixel circuit implementing the proposed algorithm is presented.

In the proposed algorithm, as mentioned in the previous chapter, only the signals in the corner positions of 3x3 windows are involved in the procedure of computation in the center pixel. Therefore, the structure of the pixel array is designed as shown in Figure 4.1. In this figure, each square represents a pixel, which has an access to its four diagonal neighbors. In each pixel, there are a photodiode converting a light signal into a photocurrent, and processing circuits performing the operations of the algorithm.

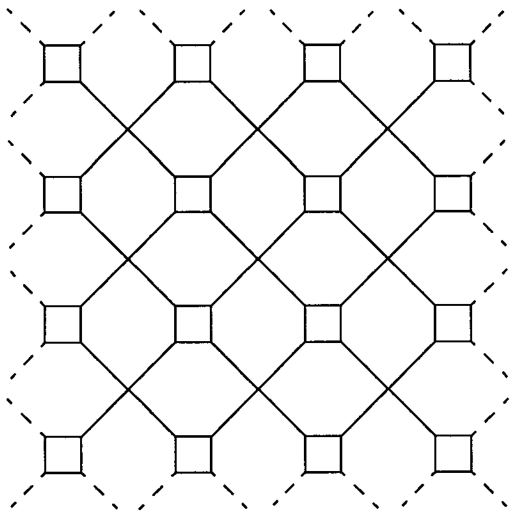


Figure 4.1 Simplified diagram of the pixel array for implementing the proposed algorithm. Each pixel, presented by a square, contains a photodiode for light sensing, and processing circuits for implementing the proposed algorithm in Chapter 3. The photocurrent in each pixel is steered by the control signals into its adjacent pixel through the connections between two adjacent pixels.

To make the pixel array shown in Figure 4.1 to perform the operations defined in the proposed algorithm, the following two considerations need to be taken.

(1) Signal flow of the photocurrents in the pixel array. As a photodiode is used for sensing the light signal in each pixel, the level of a photocurrent from the photodiode can be in a nano-Ampere range or below. It is not easy to copy or amplify such a weak current. To avoid the current's copying, the pixel circuit should be designed to have its operation in the first stage, i.e. current subtraction, performed in such a way that two input photocurrents are applied successively, instead of simultaneously. In this way, at each time every photocurrent needs to flow to only one pixel and no cur-

rent-copying is necessary. Moreover, such an arrangement can facilitate the photocurrent switching.

(2) Mismatch of MOS devices. The mismatch would lead to a non-uniformity of the characteristics of pixels and affect seriously the precision of the operations. This negative effect gets even worse with weak operating currents. Hence, the compensation is needed to obtain an acceptable quality of the operations of the pixel circuit.

In this chapter, the design of the circuit in each pixel is presented. The structure of the pixel array and the procedure of the circuit operations for implementing the proposed algorithm in Chapter 3 are described. The effort of the design is laid on solving the problems related to the weak current operation and the mismatch in the circuit units. The simulation results are also presented to show the significance of the techniques used in the circuit design.

4.2 Circuit in a Pixel

In this section, the pixel circuit to perform the operations in the procedure of the proposed algorithm in Chapter 3, is described. The description of the circuit consists of the following parts. The circuit block diagram shows the structure of each pixel. The current subtractor with a procedure of reducing the charge injection is described. The current comparator is elaborated considering the transistor mismatch. The digital part of the pixel circuit is presented as well.

4.2.1 Block Diagram

According to the procedure of the proposed algorithm presented in Chapter 3, the basic block structure of each pixel is designed as shown in Figure 4.2. In this structure, a photodiode converts a light signal into a photocurrent, and a switch unit controls the flow of photocurrents in the pixels and directs the photocurrent of the pixel to one of its neighbors. The three blocks, current subtractor, current comparator and accumulator, perform the three operations in the procedure of the proposed algorithm, subtraction, thresholding and voting.

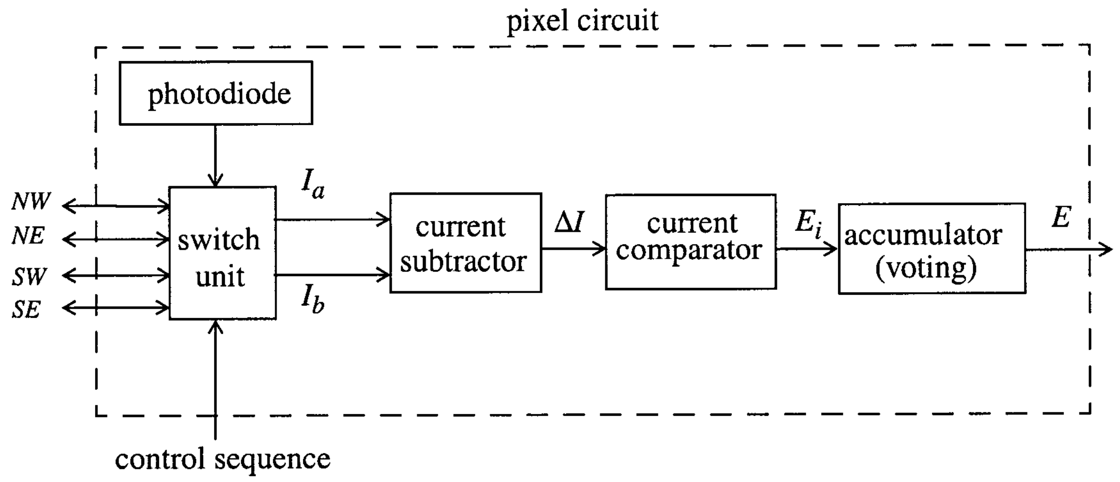


Figure 4.2 Block diagram of the pixel circuit.

In Figure 4.2, the photocurrent inputs of each pixel come from its nearest neighbors in *NE*, *SE*, *NW* and *SW* directions through connections in the pixel array shown in Figure 4.1. The photocurrent of this pixel is also needed to be sent to its neighbors for processing through the same connections in Figure 4.1. Thus, the switch unit in each pixel controls these input and output photocurrents of the pixel, selecting one photocurrent out of the four from the neighbors for the computation in this pixel and sending the photocurrent of this pixel to one of its four neighbors for the calculation in that pixel at the same time.

As discussed previously, to avoid having transistor pairs for current copying, in Figure 4.2 the switch unit enables the successive input of the two operands, currents I_a and I_b, to the current subtractor to produce the current difference ΔI. This current difference is then thresholded in the current comparator, which should be made to be able to discard a negative difference ΔI as required by the proposed algorithm. The last block, i.e. the accumulator, counts the signals E_i in each operation cycle. By applying a control sequence to the pixel circuit, the above operations are repeated to implement the proposed algorithm for producing an edge signal E.

In the following sub-section, the circuits in each block in Figure 4.2 are described in details.

4.2.2 Current Subtractor Implemented by a Current Memory

As mentioned in the sub-chapter 4.2.1, the two currents I_a and I_b in Figure 4.2 for the subtraction should be imported successively to avoid current duplication. The circuit shown in Figure 4.3 can perform a current subtraction in this manner. In this circuit, a basic current memory [28] performs the current subtraction by sampling one of the two input currents, then produces the difference between the sampled current and input current. The transistor mismatch problem is avoided in this circuit since the subtraction is not implemented by transistor pairs.

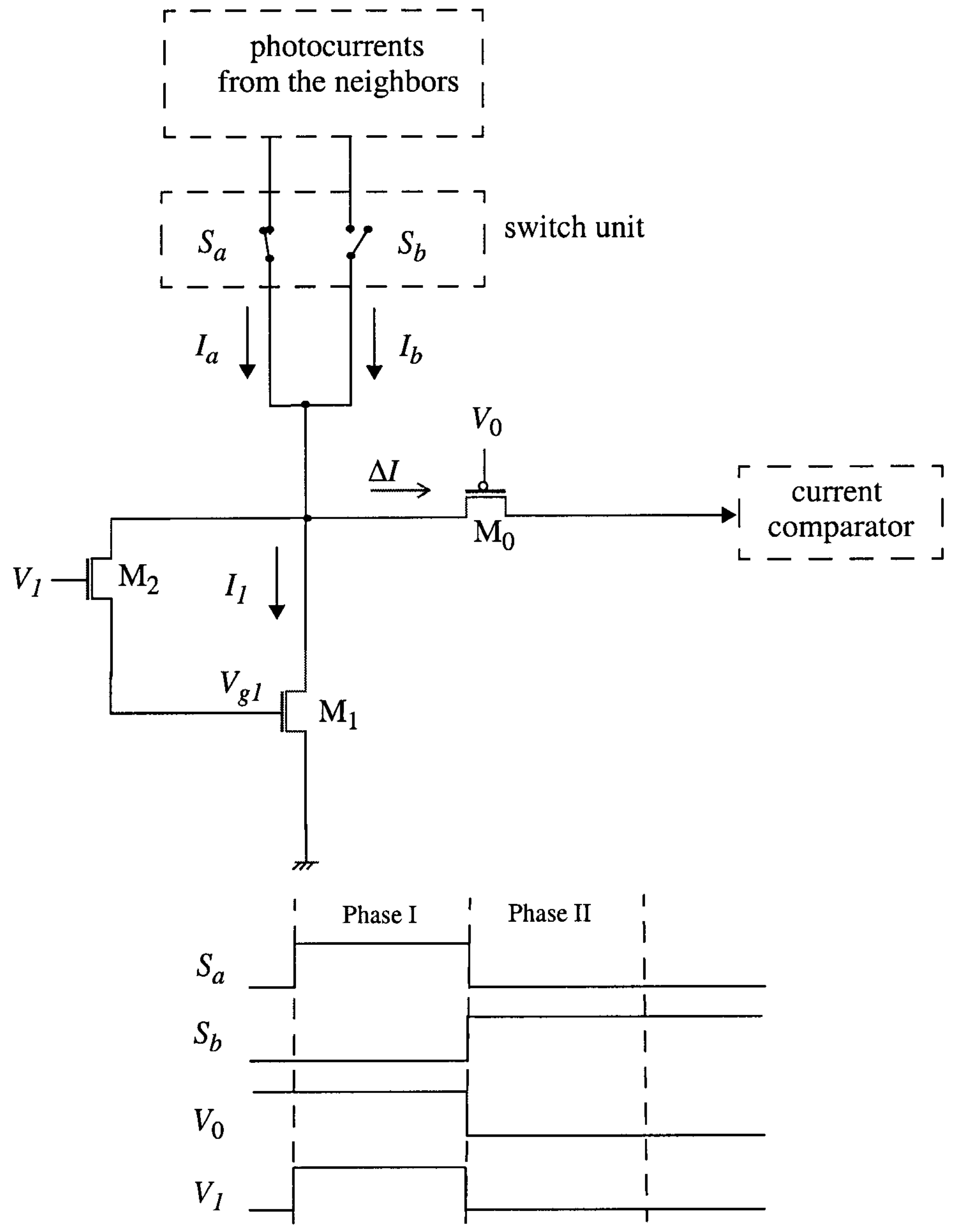


Figure 4.3 Current subtraction scheme to design the current subtractor in Figure 4.2. Two currents from two of the nearest neighboring pixels are applied to the circuit one after another, respectively during the two phases. The current difference ΔI equal to $(I_b - I_a)$ is obtained during Phase II if there is no switch noise.

Though the current memory shown in Figure 4.3 can perform the current subtraction instead of using the transistor pair, the charge injection from the transistor M_2 will change the gate voltage V_{g1} established during the sampling phase, which makes the current I_1 different from the sampled current I_a and thus creates an error. In the case of the weak photocurrent input, the transistor M_1 works in the subthreshold region, whose drain current is exponentially proportional to its gate voltage V_{g1}. Even 1mV variation of the gate voltage V_{g1} can lead to an approximate 5% error between the sampled current I_a and the reproduced current I_1. This error makes ΔI different from $(I_b - I_a)$. Thus, when the current memory is used in a pixel circuit, it is very important to reduce such a variation at the critical gate voltage. Besides, since the input is a weak photocurrent, the current memory should have an acceptable sampling time with the technique applied to reduce the effect of charge injection.

To reduce the effect of charge injection, the technique of charge division [30] is applied to the circuit structure shown in Figure 4.4. In this structure, M_1 is the storage transistor, M_2 and M_5 are used as switches, and M_3 and M_4 serve as capacitors. The gate voltage V_{g1} of M_1 is protected from the voltage variation at node V_{out} by applying a fixed voltage V_b to the gate of M_6, and the drain current variation of M_1 resulted from the channel-length-modulation is also reduced by cascading M_6 with M_1. By applying the control signals V_1 and V_2 to the switches M_2 and M_5, the quantity of charge injected into the node V_{g1} is only a fraction of the charge released when the switch M_2 is turned off. Thus, less voltage variation at the node V_{g1} can be produced. In this circuit, ΔQ_{13}, the charge injected into the node V_{g1}, is approximately given by,

$$\Delta Q_{13} = \frac{1}{1 + \frac{C_4}{C_3} + \frac{C_4}{C_1}} \Delta Q \tag{4.1}$$

where ΔQ is the charge injected into the node V_{g3} from the switch M_2, C_1 is the gate capacitor of M_1, C_3 and C_4 are the gate-to-diffusion capacitors of the dummy MOS transistors M_3 and M_4, respectively. As $\frac{1}{1 + \frac{C_4}{C_3} + \frac{C_4}{C_1}} < 1$, thus ΔQ_{13} is smaller than ΔQ.

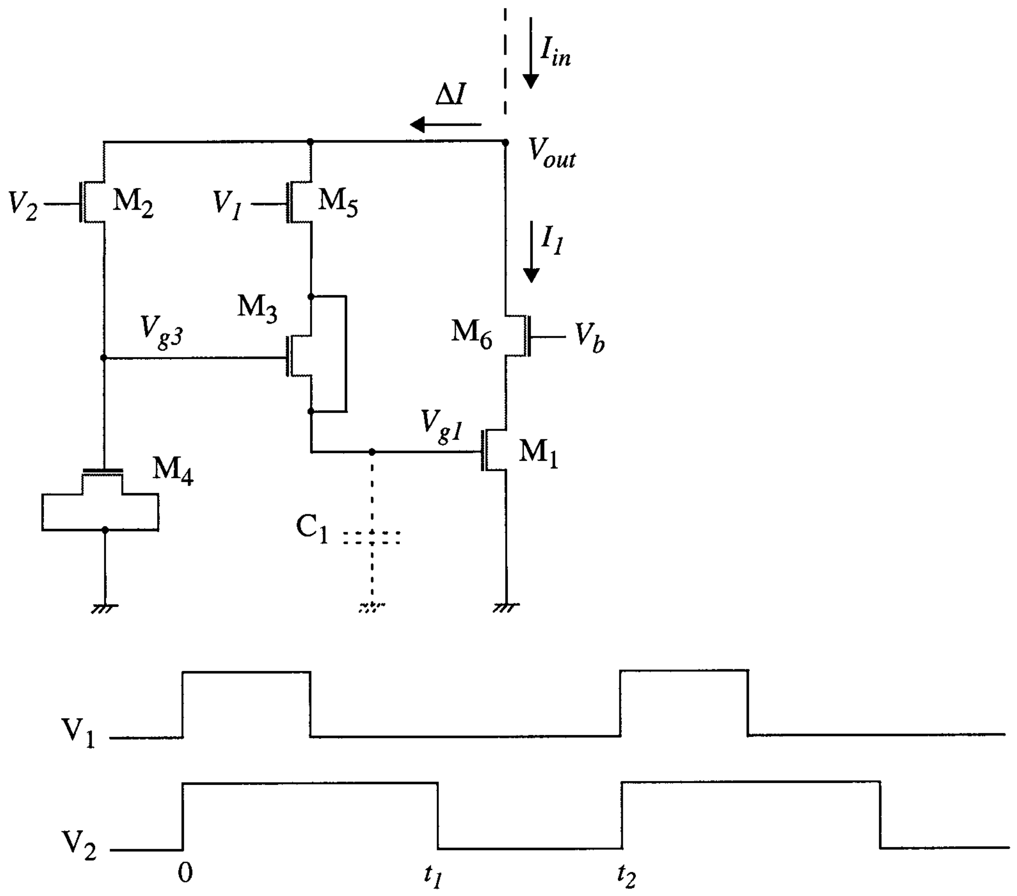


Figure 4.4 Six transistor switched-current (SI) current memory with a procedure implemented to reduce the charge injection effect [30]. The input current is sampled during $(0, t_1)$. The transistor M_1 reproduces the sampled current during (t_1, t_2).

Equation (4.1) shows that maximizing C_4 can improve the efficiency of the reduction of the charge injection. However, a large value of the capacitor C_4 will slow the current sampling. When M_5 is turned off, the voltage V_{g3} is increased gradually to a stable level by charging the capacitor C_4 with a small difference current ΔI between the input current I_{in} and the sampled current I_1. In case of a weak operating current, e.g in a sub-nano Ampere range, a long period of time to stabilize V_{g3}, such as several millisecond, can be anticipated. As the current sampling-reproduction is repeated used in the operation procedure of the pixel circuit, it is necessary to shorten the sampling time.

In Figure 4.4, an increase of the voltage V_{g3} results from an injection of a certain amount of charge to the capacitor at the node of V_{g3}. If this amount of charge is stored in advance and applied to the capacitor when needed, the time to stabilize V_{g3} will be shorter than that of weak current charging situation. Thus, to solve the sampling speed problem of the circuit shown in Figure 4.4, a boosting technique with charge sharing is proposed as follows.

On the basis of the circuit shown in Figure 4.4, two transistors M_7 and M_8 are added for boosting and another transistor M_9 is added to reset the critical gate voltage, as shown in Figure 4.5. The transistor M_8 is a switch enabling a precharge at the node V_{g4}, and M_7 activates a charge sharing between the nodes V_{g4} and V_{g3} during the second part of the sampling phase. When V_1 is equal to V_{DD}, a certain amount of charge is stored at the gate node of M_4 by pre-charging the capacitor at the node V_{g4} and turning off the transistor M_7.

Just after M_3 is turned off, M_7 is turned on by a control signal V_3 and this amount of charge helps to raise V_{g3} at that moment. Since the capacitance of C_4 is designed to be much bigger than that of the branch of C_3 in series with C_1 in order to reduce the effect of charge injection, after charge sharing between the nodes V_{g3} and V_{g4}, the level V_{g3} will be boosted to a value which can be designed to be approximately equal to the stable value of V_{g3}. With the charge from the node V_{g4}, the voltage change rate of V_{g3} depends little on the current difference ΔI (I_{in} - I_1) which can be very weak. Thus, the speed of the current memory is improved very significantly by such a boosting technique. After boosting, the circuit shown in Figure 4.5 works in the same way as that one in Figure 4.4 to reduce the effect of charge injection.

The improvement of the sampling speed does not result in any negative effect in the power dissipation, as there is little difference between the circuits in Figures 4.4 and 4.5 in terms of DC current path. The charge injected into the two circuits to establish the voltage V_{g1} is almost the same. Thus the circuit with the voltage boost technique has nearly the same power dissipation as its counterpart.

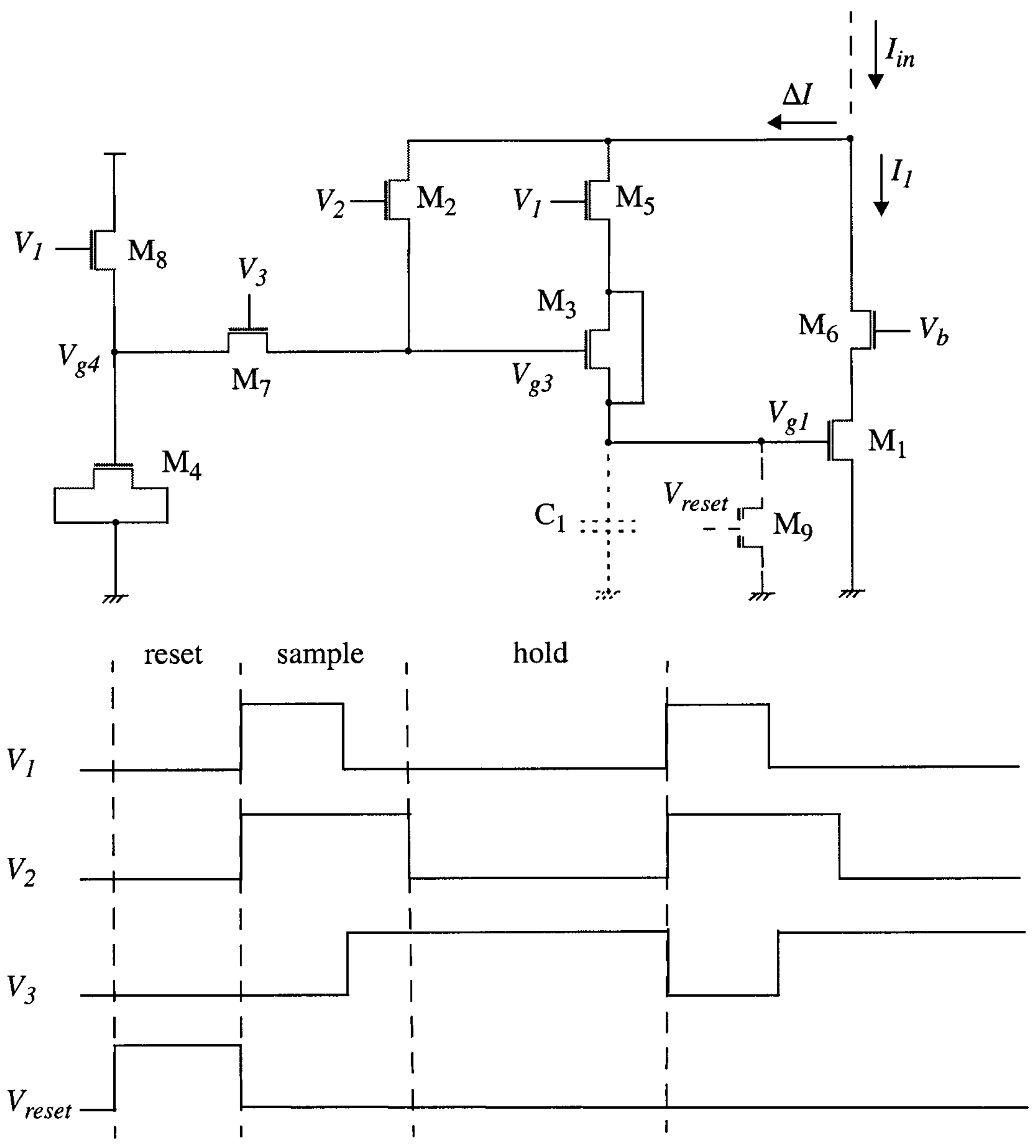


Figure 4.5 Current memory used in the pixel circuit with the same procedure of the charge injection suppression implemented in the circuit shown in Figure 4.4, and a voltage boost technique applied to speed up the current sampling. A high voltage V_{DD} of V_{reset} resets the current memory before current sampling. The current I_{in} is sampled when V_2 is at the high level and afterwards the transistor M_1 reproduces the sampled current. The circuit is sped up when V_1 changes from 0v to V_{DD}.

In the design of the current memory used in the pixel circuit, the charge division technique is applied to reduce the effect of charge injection. The circuit speed is improved by a voltage boost technique. Thus, the current memory unit can have both an accuracy and a sampling time acceptable used in the pixel circuit with the power dissipation in a rang of nano-watts.

4.2.3. Current Comparator with the Compensation of the Transistor Mismatch

To perform the procedure of the proposed algorithm in Chapter 3, a current memory described in the previous part, is used to produce a current difference ΔI that can be positive or negative. This signal ΔI is then compared to a threshold I_{th} in a comparator. If ΔI is positive and greater than I_{th}, the output of the comparator should be high, equal to "logic-1". If ΔI is smaller than I_{th}, including the case of the negative ΔI, then the output should be low, equal to "logic-0".

The current comparator proposed by Freitas and Current [31] shown in Figure 4.6 can perform the above operation. The threshold of the comparator I_{th} is set by adjusting the voltage V_{Thr} at the gate of M_3 to control its current i_{M3}. The input current ΔI is reflected by the current i_{M2} through the transistor pair (M_1, M_2). If $\Delta I < I_{th}$, the voltage V_{Out} will rise to the high level; if $\Delta I > I_{th}$, the voltage V_{Out} will fall to the low level. The voltage V_{Out} is then buffered to produce a binary signal E_i. The operation of the this comparator can be expressed by the following equation,

$$E_i = \begin{cases} 1, & \text{if } \Delta I > I_{th} \\ 0, & \text{if } \Delta I < I_{th}, \text{including } \Delta I < 0 \end{cases} \tag{4.2}$$

which is the required function in the comparator design.

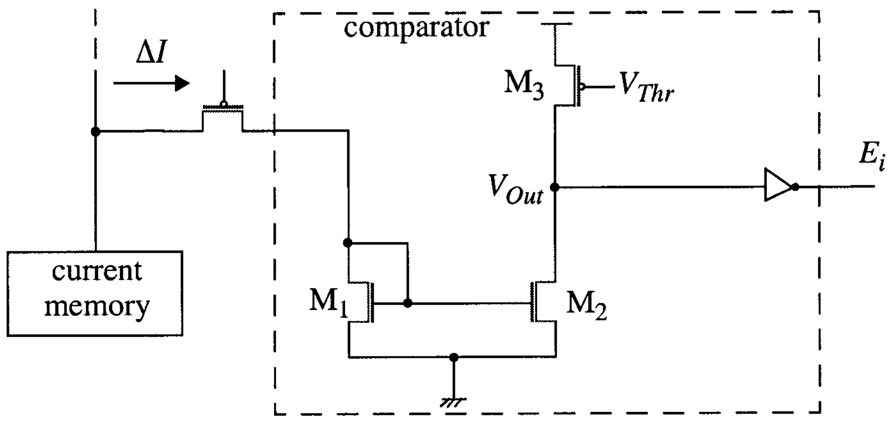


Figure 4.6 Basic scheme of the current comparator. This comparator based on the basic one [31] can operate normally when $\Delta I > 0$. If $\Delta I < 0$, the output voltage V_{Out} remains a high level, regardless the magnitude of the current.

Though the current comparator shown in Figure 4.6 can perform the required function, its threshold suffers from the device mismatch since the threshold relates to the transconductance g_m of M_3 and the current mirroring between M_1 am M_2. It should be noted that, in a pixel array, to achieve a certain uniformity of the comparison threshold, a process compensating for device mismatch is needed, particularly in the case of weak current operation.

In the circuit shown in Figure 4.6, when the input current ΔI is equal to I_{th}, the output voltage V_{Out} should be at a level to make both M_2 and M_3 operate in the saturation region with a high drain-to-source resistance, which is considered as an equilibrium state. In the case of ΔI not being equal to I_{th}, a small derivation of ΔI from I_{th} will result in a significant voltage change of V_{Out}, and the voltage V_{Out} will approach to V_{DD} or V_{SS}. Thus, if the circuit is set to memorize such a current I_{th}, with the mismatch of the devices, the threshold of the comparison will be insensitive to the mismatch [20].

Based on the idea described in the above paragraph, the circuit shown in Figure 4.7 is used for the compensation of the mismatch. In this circuit, the threshold of the comparator is set up by injecting I_{th} into M_1 and meanwhile connecting the node V_{Thr} and V_{Out} by the switch S_1. This connection establishes a V_{Thr}, and makes M_2 and M_3 automatically operate in the saturation region to establish the equilibrium state. If all the pixels in an array do so, the equilibrium state is obtained at each of them, i.e. the threshold I_{th} is memorized in each of the pixels with the device parameters. Assuming that the voltage V_{Thr} is preserved after the switch S_1 is turned off, if the input current ΔI is different from I_{th}, the voltage V_{Out} will be set either at a high level or a low level. In the pixels with the same circuit structure, each pixel can respond in the same manner to the same input current, and produce a uniformed comparison result. Thus, the device mismatch is compensated and the comparators in all the pixels can have the same threshold value.

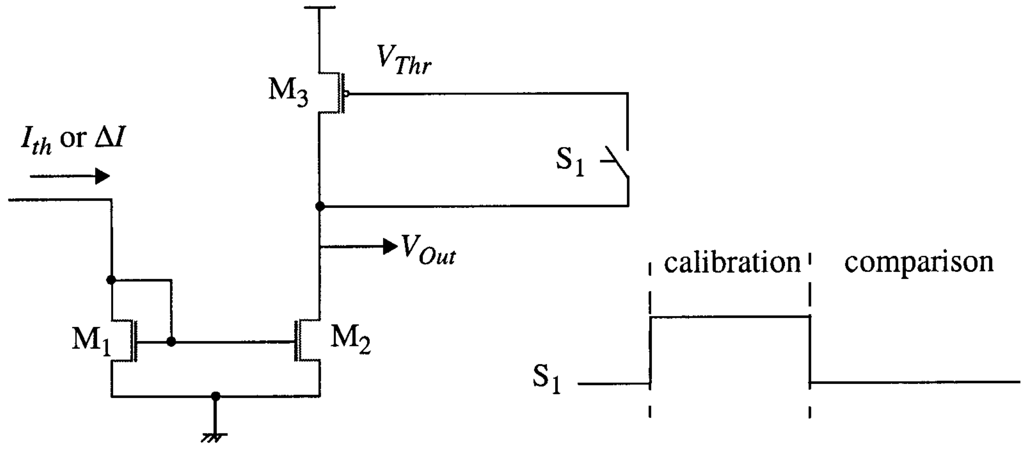


Figure 4.7 Compensation scheme [20] to make the threshold of the comparator insensitive to the transistor mismatch. The comparator is calibrated with the input current ΔI equal to I_{th} when the switch S_1 is on to establish V_{Thr}. During such a calibration phase, the output of the comparator, V_{Out}, is set to make M_2 and M_3 operate in the saturation region. Then, the switch is turned off with the threshold I_{th} memorized as the voltage V_{Thr}, together with the transistor mismatch. After that, during the comparison phase, a signal current ΔI is injected and the output V_{Out} goes to the high or low levels according to the comparison result of ΔI and I_{th}.

It should be noted that the voltage V_{Thr} can be modified during S_1 switching, resulting in a critical change of the current in M_3. The prime cause of the modification of V_{Thr} is the effect of the charge injection. A compensation for charge injection is needed for the circuit to function as described in the preceding paragraph. In the current memory designed previously in the sub-chapter 4.2.2, two phases of the adjustment are used with two switches turned off successively. The charge injection is suppressed by means of a charge division between a dummy transistor and a parasitic capacitor, which is carried out in these two phases. To apply this two-phase charge compensation technique in the comparator design, the following points need to be considered.

(1) Current i_{M2} should mirror the threshold I_{th} correctly. During the compensation process, a current difference Δi between i_{M2} and i_{M3} is used to produce ΔV_{Out}, and through a negative feedback to adjust V_{Thr} back to the level set before the charge injected. Only when M_2 stays in the saturation region during the charge compensation phase, can the reference i_{M2} remain stable and be approximately equal to I_{th}, so that the voltage V_{Thr} can be correctly adjusted to memorize I_{th} at the end of the feedback process.

(2) Based on the first point, V_{Out} should not be allowed to drop too low so that M_2 would not be driven into the triode region, insuring i_{M2} to be able to mirror I_{th} correctly. Thus, at this moment, the charge injected by the switch should increase i_{M3} to make V_{Out} go up, and to prevent M_2 to be driven into the triode region.

(3) Charge produced by the dummy transistor during the second phase, should compensate the charge released from the first switch.

Based on the above points, the comparator is designed as shown in Figure 4.8 with two clock signals V_1 and V_2. The transistor M_4 functions as the switch S_1 in Figure 4.7 and it is chosen to be a NMOS transistor. Turning off M_4 produces the negative charge to the node V_{Thr}, lowering V_{Thr}. Then the current i_{M3} will increase, raising V_{Out} to a higher level, which avoids M_2 to be driven to the triode region. Thus, the current i_{M2} can still mirror the input current I_{th}. During the second phase ($V_2 = V_{DD}$), the voltage V_{Thr} will be brought back through a negative feedback with a constant reference current i_{M2}. On the contrary, if M_4 is a PMOS transistor, when it is turned off, V_{Thr} will increase by the positive charge injected, resulting in a decrease of V_{Out}, which may bring M_2 out of the saturation region. The drain current of M_2 will not reflect I_{th} properly and the voltage V_{Thr} can not be brought back in the second phase.

In the circuit shown in Figure 4.8, a PMOS dummy transistor M_5 is placed to produce the positive charge in order to compensate for that from M_4 at the moment when M_4 is turned off. When the switch M_4 is on, the transistor M_5 works in the on state to ensure a channel created in it by setting V_{g7} as a lower voltage V_{pre}, and then after M_4 is turned off, M_6 is turned on to create a charge distribution for bringing V_{Thr} back. After that when M_6 is turned off, according to the charge division, a large quantity of charge from M_6 will be injected to the transistor M_7. The voltage variation at the node V_{Thr} will be minimized. In this comparator circuit, two fixed voltage V_{b1} and V_{b2} are applied to the gates of M_9 and M_{10} to make the comparator more sensitive to the input current around the threshold value.

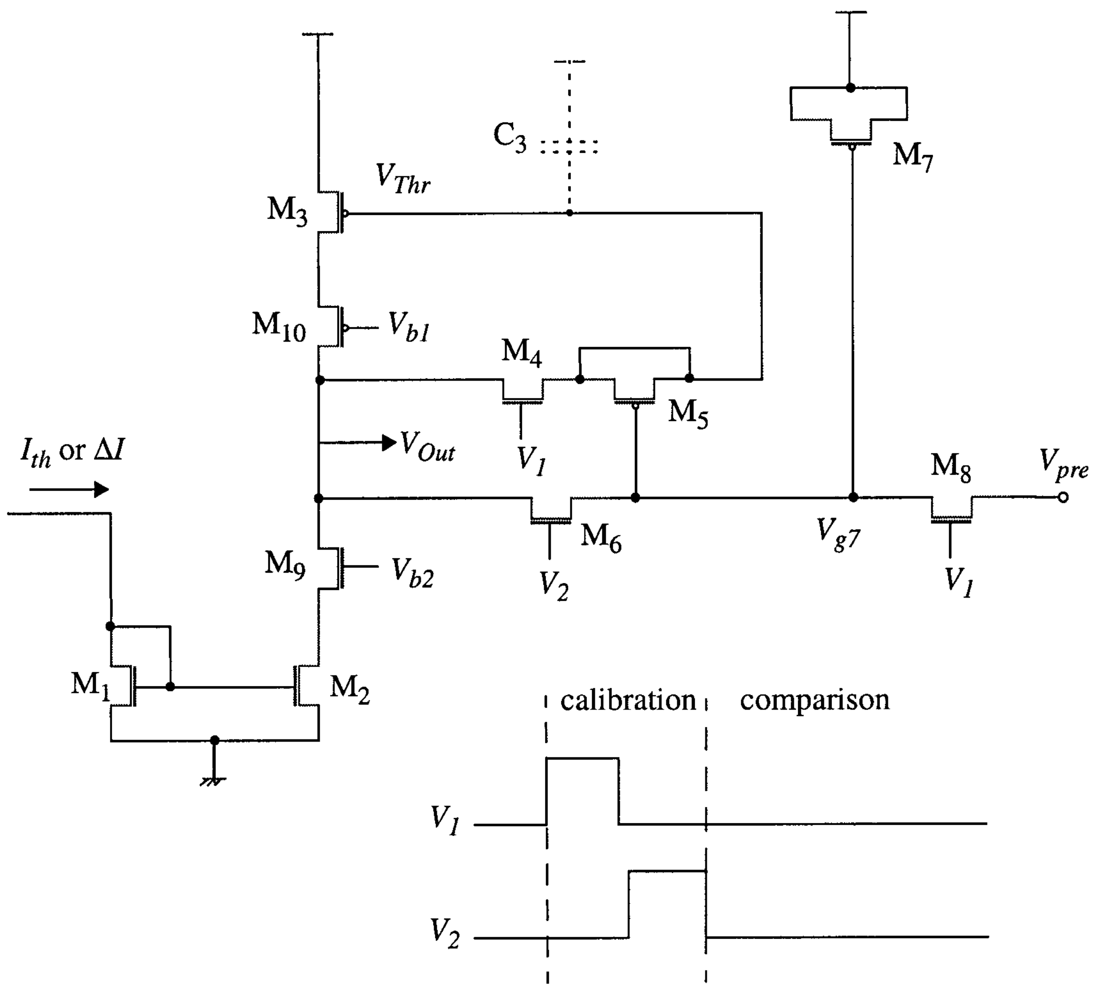


Figure 4.8 Detailed circuit of the current comparator. It is calibrated with the input current I_{in} equal to I_{th} during two phases when V_1 and V_2 are high as shown in the figure. Then, a signal current is injected for comparison with I_{th}. A NMOS switch transistor M_4 is used to ensure I_{th} mirrored correctly by (M_1, M_2) during the two phases of the calibration. The charge injection is reduced by the charge division technique as used in the current memory.

In the design of the comparator, a basic structure is used to perform the required operation. Compensation for the device mismatch and charge injection have been implemented in the circuit of the comparator. Therefore, the comparators in all the pixel will have a uniformed characteristics in terms of the threshold value.

4.2.4. Accumulator Implemented by a Serial Two-bit Half-adder

The voting operation in the pixel calculates how many "logic-1" signals generated from the twelve current comparisons, and then produces the edge signal E. This operation is performed by the accumulator shown in Figure 4.2, which counts the signal E_i and calculates the signal E as follows.

$$E = \begin{cases} 1, \text{ if } \sum_{i=1}^{12} E_i \geq 4 \\ 0, \text{ if } \sum_{i=1}^{12} E_i < 4 \end{cases} \tag{4.2}$$

As mentioned in Chapter 3, only six of twelve binary signals E_i, $i = 1, 2...$and 12, can possibly be "logic-1". The maximal value of the summation result, $\sum_{i=1}^{12} E_i$, is six. Thus, the accumulator can be implemented by a serial two-bit half-adder shown in Figure 4.9.

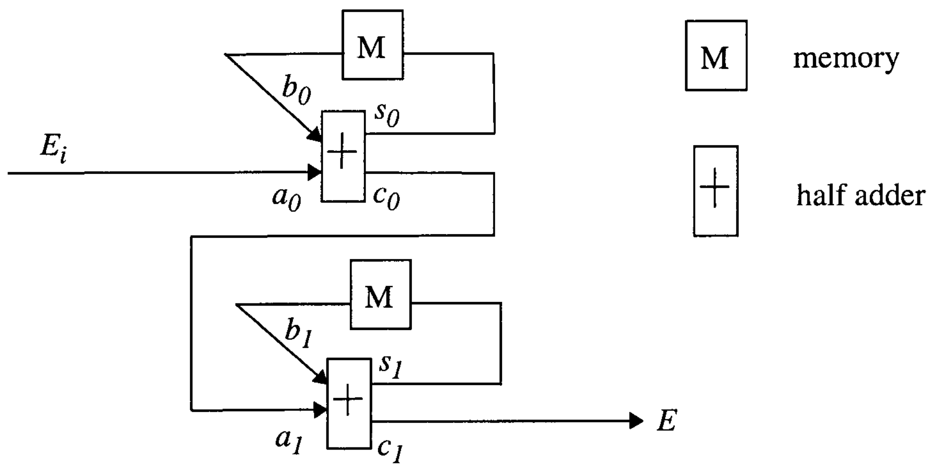


Figure 4.9 Serial two-bit half-adder to perform the operation of the accumulator in Figure 4.2. The input E_i is one of the operands of the first half-adder. The sum of each of two half-adders is memorized, and then fed back as the other operand in the next cycle. The carry c_1 of the second half-adder is the output edge signal E.

In Figure 4.9, two memories store the sums of two adders, which is the accumulation result of E_i until the last computation cycle. Then in the next cycle, another input E_i is added in the first half-adder and the memories are refreshed. The three bits of the adders, $c_1 s_1 s_0$, show the accumulation value of E_i. If at least four out of the signals E_i, $i = 1, 2$...and 12, are equal to "logic-1", the signal c_1 must be "logic-1", which meets the requirement of the operation of the accumulator.

In the two-bit serial adder discussed above, only one half-adder, instead of a full-adder, is needed in each stage, as there are only two inputs in each of the two stage. Since the threshold of the voting is four, i.e. 100 in binary code, the generation of the carry out of the second stage, c_1, indicates a result of E equal to 1. As there are twelve serial inputs in this adders, once c_1 equal to 1 is generated, this "logic-1" signal should be kept till the end of the operation cycle.

Based on the above analysis, the logic functions of the serial adder can be simplified as,

$$s_0 = E_i \oplus b_0 \tag{4.3}$$

$$c_0 = E_i \bullet b_0 \tag{4.4}$$

$$c_1 = E_i \bullet b_0 \bullet c_{m0} \tag{4.5}$$

where b_0 and c_{m0} are the memorized values of the sum s_0 and c_0 in the last computation cycle, and c_{m0} and c_1 stays at the value of "logic-1" if it ever becomes "logic-1".

For the circuits to perform the functions above, the pass logic is used to design the gates and the dynamic memory technique is used to store the required signals. The circuit of the serial adder is shown in Figure 4.10 and is described in detail as follows.

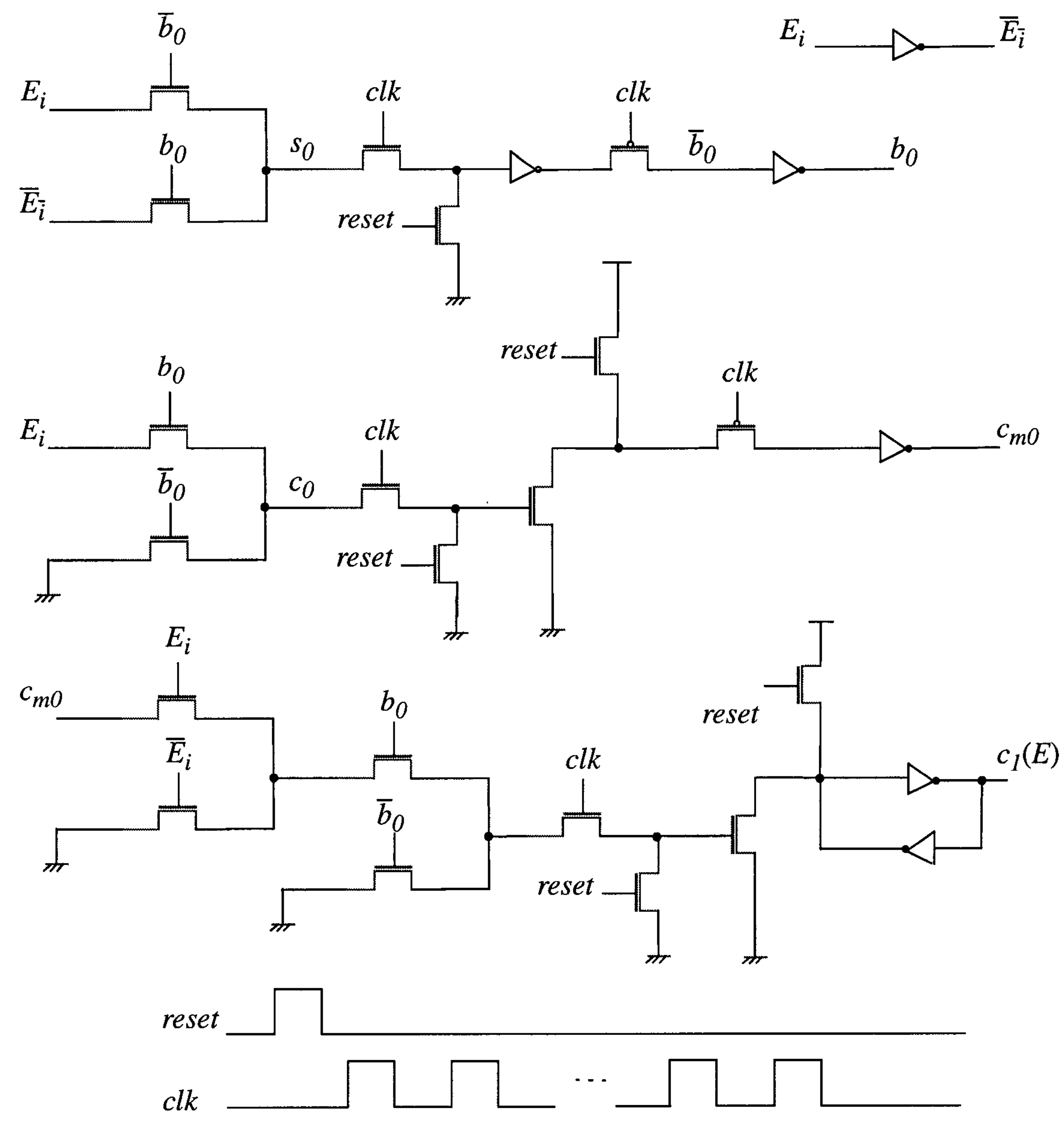


Figure 4.10 Circuit of the serial two-bit half-adder. The signals b_0, c_{m0} and c_1 are reset to "logic-0" with the signal *reset* equal to "logic-1". After that, when *clk* is equal to "logic-1", the logic functions are computed by the gates of pass logic. When *clk* is equal to "logic-0", the memories are refreshed. This operation is repeated for twelve times to produce the output edge signal c_1, i.e. *E*.

The serial adder is first reset with its stored signal, b_0, c_{m0} and c_1, equal to "logic-0". Then, when *clk* is in the high level, the pass logic gates compute Equation (4.3), (4.4) and (4.5) with the input signal E_i, and the stored signals b_0 and c_{m0}. When *clk* is in the low level, three memories are updated, in which the last two memories flip only in one direction, i.e. from "logic-0" to "logic-1". After twelve cycle, the signal c_1 will represent the result of the edge signal *E*.

The accumulator circuit in Figure 4.2 is designed by a structure of a serial two-bit half-adder. By simplifying the operation needed and using pass logic gates and dynamic memory cells, the serial adder is implemented with the minimal number of transistors for saving the silicon area in each pixel.

4.2.5 Switch Unit Controlling the Flow of Photocurrents

As mentioned in the section of the introduction of this chapter, each pixel has four connections to its nearest neighbors in *NE*, *SE*, *NW* and *SW* directions. At each time, every pixel outputs its own photocurrent to one of the four neighbors and at the same time receives a photocurrent from another of these neighbors. Figure 4.11 shows the photocurrent flow in the pixel array. During each clock cycle, every pixel outputs its own photocurrent to one of the four neighbor (e.g. that in the SW) and, at the same time, receives the photocurrent from the neighbor in the opposite direction (e.g. that in the NE). Such input and output are controlled by means of a switch unit included in every pixel. Thus, the photocurrent switching in all the pixels in the array is same and independent from the pixel position, and can be easily arranged. Table 4.1 summarizes the relationship of the flowing direction between the incoming and outgoing photocurrents in each pixel.

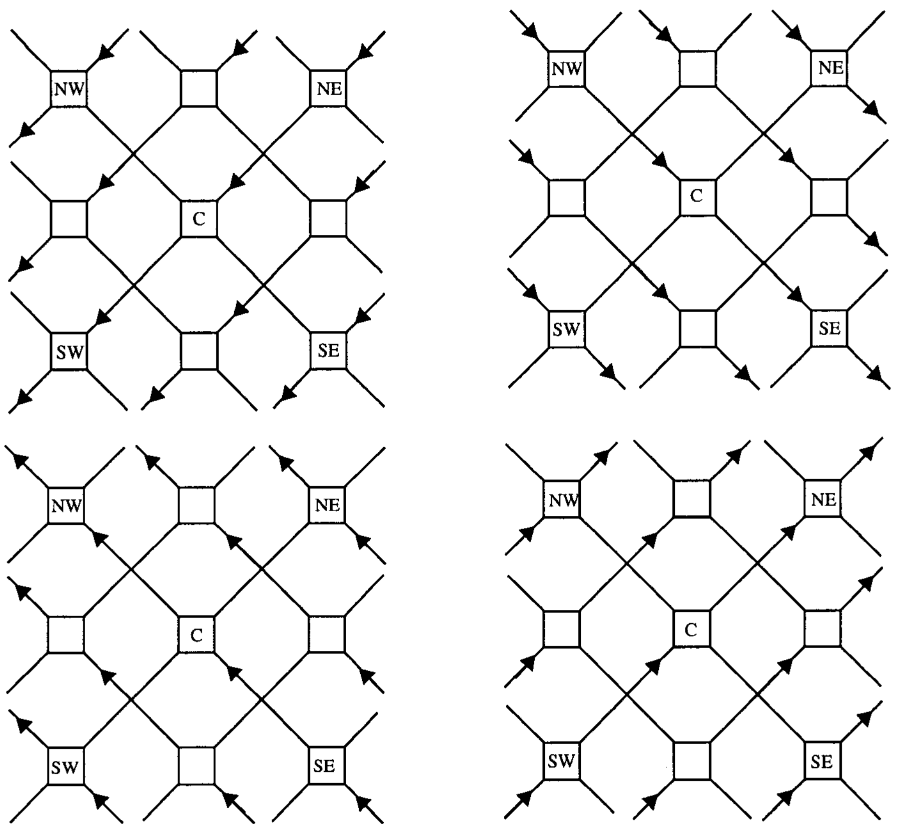


Figure 4.11 Photocurrent flow in the pixel array. In each case, as shown by the arrows, the pixel C receives the photocurrent from one of the neighbors in the *NW*, *NE*, *SW* and *SE* direction, and sends its own photocurrent to one of those neighbors. This current switching is the same for all the pixel, and controlled by the switch unit in each pixel.

	Direction in which the neighbors are located			
incoming photocurrent	*NE*	*NW*	*SE*	*SW*
outgoing photocurrent	*SW*	*SE*	*NW*	*NE*

Table 4.1 Photocurrent flow in each pixel.

According to Table 4.1, the switch unit is designed as shown in Figure 4.12. In this figure, two binary signals b_0 and b_1 are applied to the gates of the transistors to switch photocurrents. At each time, one of the paths connected to the photodiode in this pixel is on and the others are off so that its photocurrent can be sent only to one of its neighbors.

At the same time, the photocurrent of one of the neighbors in the opposite direction flows into this pixel through one of the paths connected to the computation circuit enabled by the same value of the signals b_0 and b_1. Table 4.2 shows the relationship between the control signals b_0 and b_1 and the photocurrent flow.

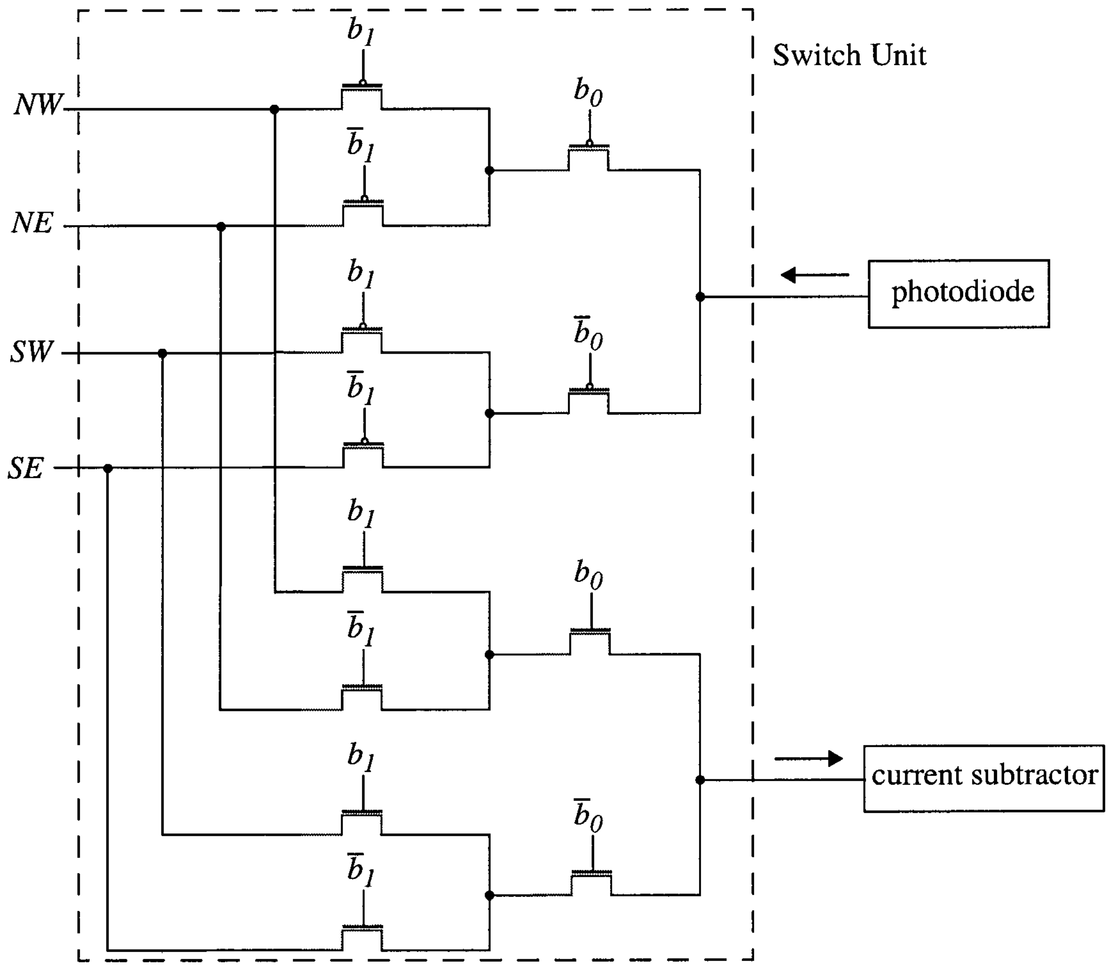


Figure 4.12 Switch unit controlling the flow of photocurrents with two binary signals b_0 and b_1.

b_1b_0	direction of the outgoing photocurrent	direction of the incoming photocurrent
00	*NW*	*SE*
10	*NE*	*SW*
01	*SW*	*NE*
11	*SE*	*NW*

Table 4.2 Control of the photocurrent flow in each pixel.

The switch unit has been designed to control the input and output photocurrents in each pixel. The photocurrent switching in this unit is arranged so that the photocurrent of each pixel flows in the same global direction in the pixel array. The switch units of all the pixels share only two binary control signals. Thus, the number of the control signals for the pixel array is reduced and less silicon area is spared on these control signals.

4.3 Simulations

In the preceded sub-chapter, the design of the pixel circuit was described. Much effort has been made to reduce the effect of transistor mismatch and that of the charge injection in the current-mode circuit, to improve the quality of the operations with the weak photocurrents. In order to demonstrate the effectiveness of the techniques used in the circuit, simulations have been done using HSPICE with the transistor models of a 0.18μm CMOS technology. They are conducted in three steps, with a current memory cell, a single current comparator cell, and with a set of comparator cells.

4.3.1 Current Memory

In the current memory shown in Figure 4.13, two techniques, the charge division and voltage boost, are used to improve the its accuracy and speed, which can be shown in following simulation results respectively.

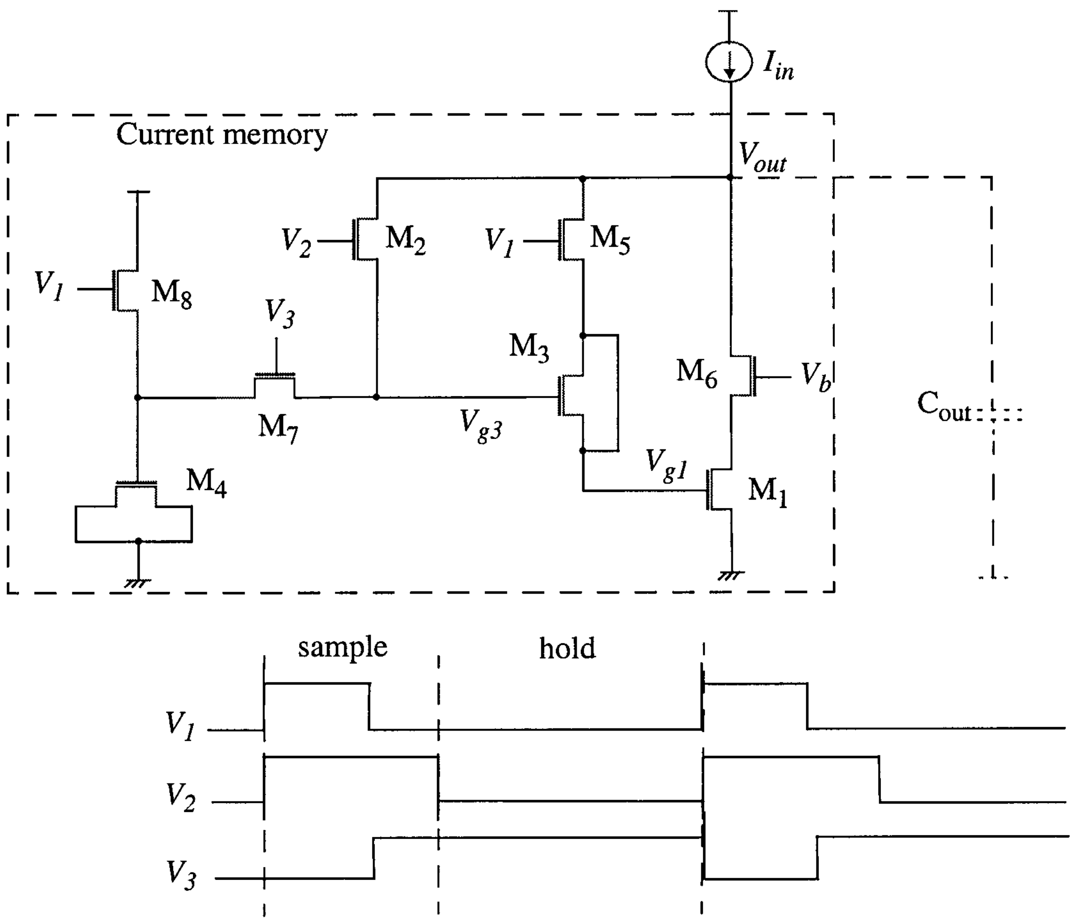


Figure 4.13 Current memory with the charge division and voltage boost techniques implemented. A capacitor C$_{out}$ is added in the simulations to observe the voltage change at V_{out} for the calculation of the current sampling error.

In the circuit shown in Figure 4.13, the transistors M$_1$ and M$_4$ have aspect ratio of 0.5μm/1μm and 2.5μm/2.5μm, respectively. The other transistors are minimum-sized. The input signal current I_{in} is in a range of 5nA to 40nA.

In the simulation, after current sampling, the difference current between the input current I_{in} and the current i_{M1} reproduced by the memory, will charge or discharge the output capacitor C$_{Out}$ at the node V_{out}. By observing the changing rate $\Delta V_{Out}/\Delta t$ from the waveform of the voltage V_{out} shown in Figure 4.14, the error current ΔI, I_{in} - i_{M1}, can be calculated by the following equation.

$$C\frac{\Delta V_{out}}{\Delta t} = \Delta I = I_{in} - i_{M1} \qquad (4.6)$$

The error rate of the current memory observed by the simulation result, $\Delta I/I_{in}$, is shown in Figure 4.15. It can be seen from this figure that the error rate for the input current in the nano-Ampere range, is below 3% as the voltage variation of V_{g1} is less than 1mV. Without this compensation, the voltage variation would be as high as 30mV.

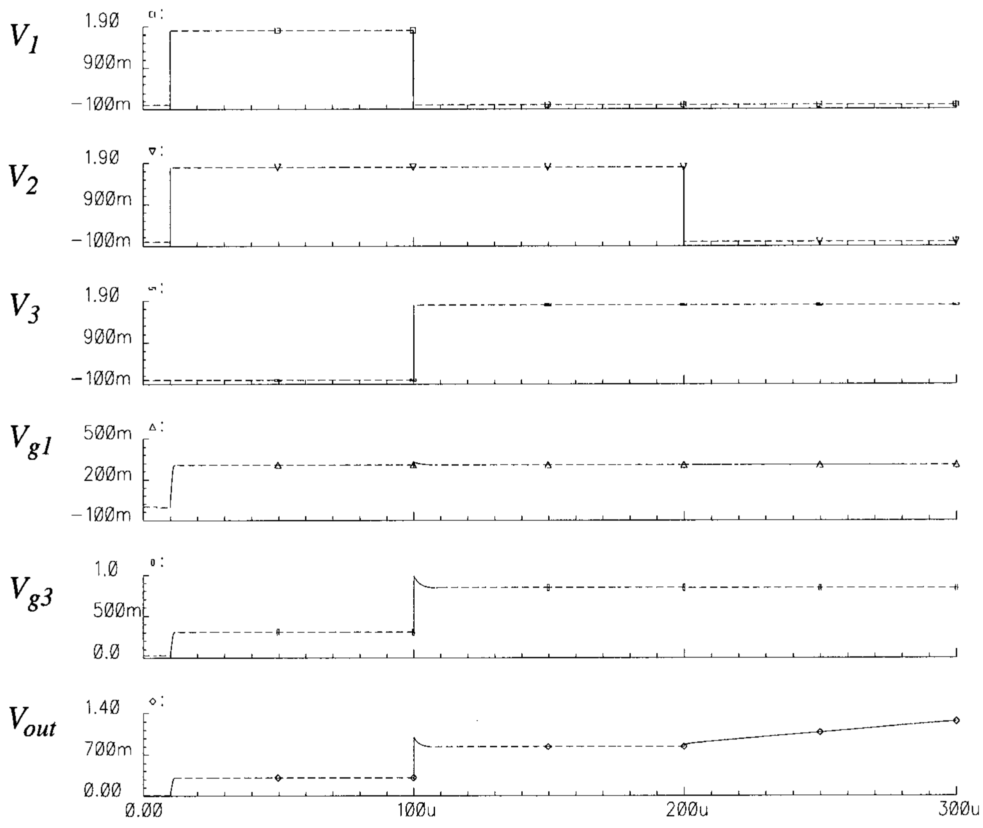


Figure 4.14 Simulated waveforms of the circuit shown in Figure 4.13 when the input I_{in} is equal to 10nA. The memory samples the input current when the control signal V_2 is high and reproduces the current when the signal V_2 is low. After current sampling, the error current between the input and reproduced currents charges the output capacitor C_{Out} to make the output voltage V_2 increase with the time. It is observed that the error current ΔI is 0.12nA.

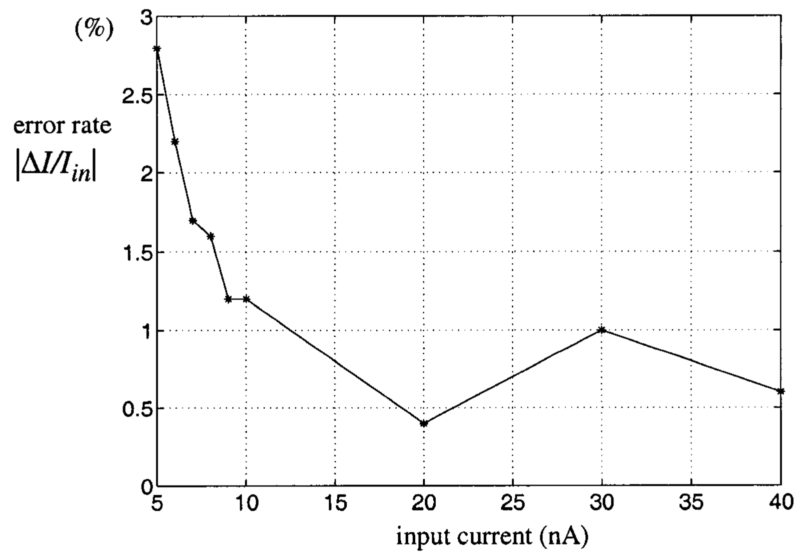


Figure 4.15 Current error rate of the current memory designed by a 0.18μm CMOS technology.

Besides the charge division technique, the effectiveness of the voltage boost technique can also be observed from the waveform of the voltage V_{g3} shown in Figure 4.14. Just after the control signal V_1 goes down, the charge sharing, enabled by the control voltage V_3, boosts the voltage V_{g3} to speed up its variation.

To demonstrate the speed improvement by the voltage boost technique, the circuit without this technique, i.e., without transistors M_7 and M_8 in Figure 4.13, is also simulated for speed comparison. Figure 4.16 shows a clip of the waveforms of the voltage V_{g3} in two circuits when the input current is 5nA. It can be seen that the voltage V_{g3} in the memory with the voltage boost changes faster than that without this technique.

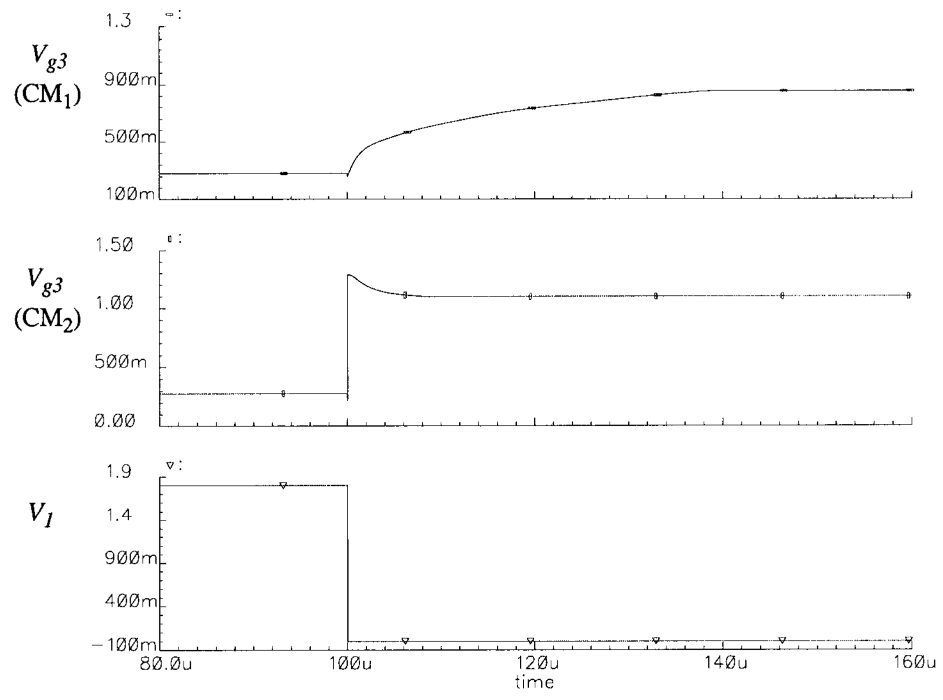


Figure 4.16 Simulation results showing the change of the voltage V_{g3} after the control signal V_1 goes down, in two cells CM$_1$ and CM$_2$, without and with the voltage boost, respectively. The voltage V_{g3} in the memory CM$_2$ changes faster than that in the mem-

By observing the current i_{M1} in the simulations, the comparison of the speed for the two memories to be stabilized, is made as shown in Figure 4.17. After t_1 and t_2, respectively in the two memories, when the control signal V_1 goes down, the difference between the current i_{M1} and the input current I_{in} is less than 10% of the input current I_{in}. The time values are given in Table 4.3. It can be seen that the circuit with the voltage boost technique can be stabilized more quickly than the circuit without this technique.

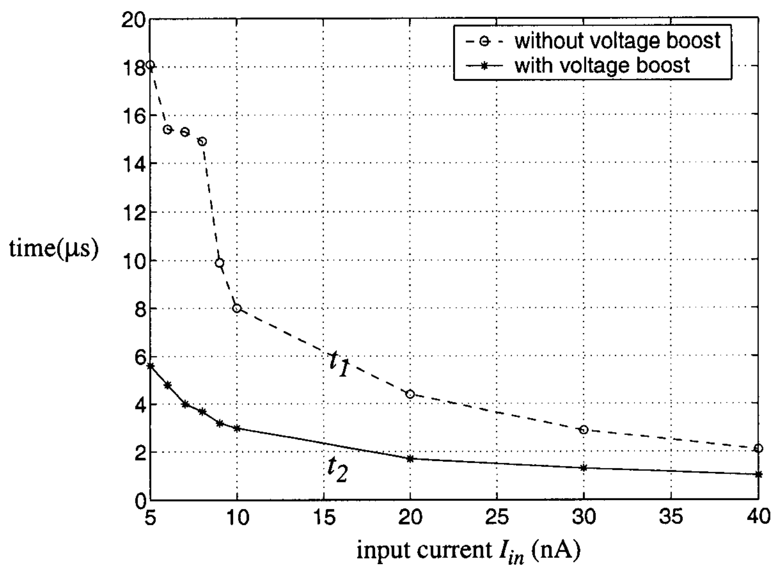


Figure 4.17 Speed comparison of the current memories without and with the voltage boost technique. The duration t_1 is the time needed, without the voltage boost, for V_{g3} to reach the level that supports the current i_{M1} with an error rate less than 10%, and t_2 is that with the boost.

input current (nA)	5	6	7	8	9	10	20	30	40
t_1 (μs)	*18.1*	*15.4*	*15.3*	*14.9*	*9.9*	*8.0*	*4.4*	*2.9*	*2.1*
t_2 (μs)	*5.6*	*4.8*	*4*	*3.7*	*3.2*	*3.0*	*1.7*	*1.3*	*1*
t_1 -t_2 (μs)	*12.5*	*10.6*	*11.3*	*11.1*	*6.7*	*5.0*	*2.7*	*1.6*	*1.1*

Table 4.3 Time needed for the current memories without and with the voltage boost technique to be stabilized in the 10% I_{in} error range after the control signal V_1 goes down.

As discussed in the sub-chapter 4.2.1, the time needed to stabilize the current memory when the control signal V_1 goes down, is the bottleneck of the whole sampling time. Thus, the current memory with the voltage boost technique has a better speed performance since this part of the sampling time is reduced.

In summary, with the charge division and voltage boost techniques implemented, the current memory designed in the pixel circuit can have the acceptable accuracy and sampling speed in the application of the pixel sensor circuit with weak photocurrent situation.

4.3.2 Single Current Comparator

The compensation scheme for the transistor mismatch has been used in the current comparator design. In the following simulations, the operation of the current comparator is shown to demonstrate how the compensation scheme is implemented.

For the circuit shown in Figure 4.18, the transistors M_1, M_2, M_3 and M_7 have aspect ratio of 0.22μm/2μm, 0.22μm/2μm, 0.22μm/3μm and 1μm/1μm. The other transistors are minimum-sized. The voltages V_{b1}, V_{b2} and V_{pre} are equal to 1V, 0.7V and 0.4V, respectively.

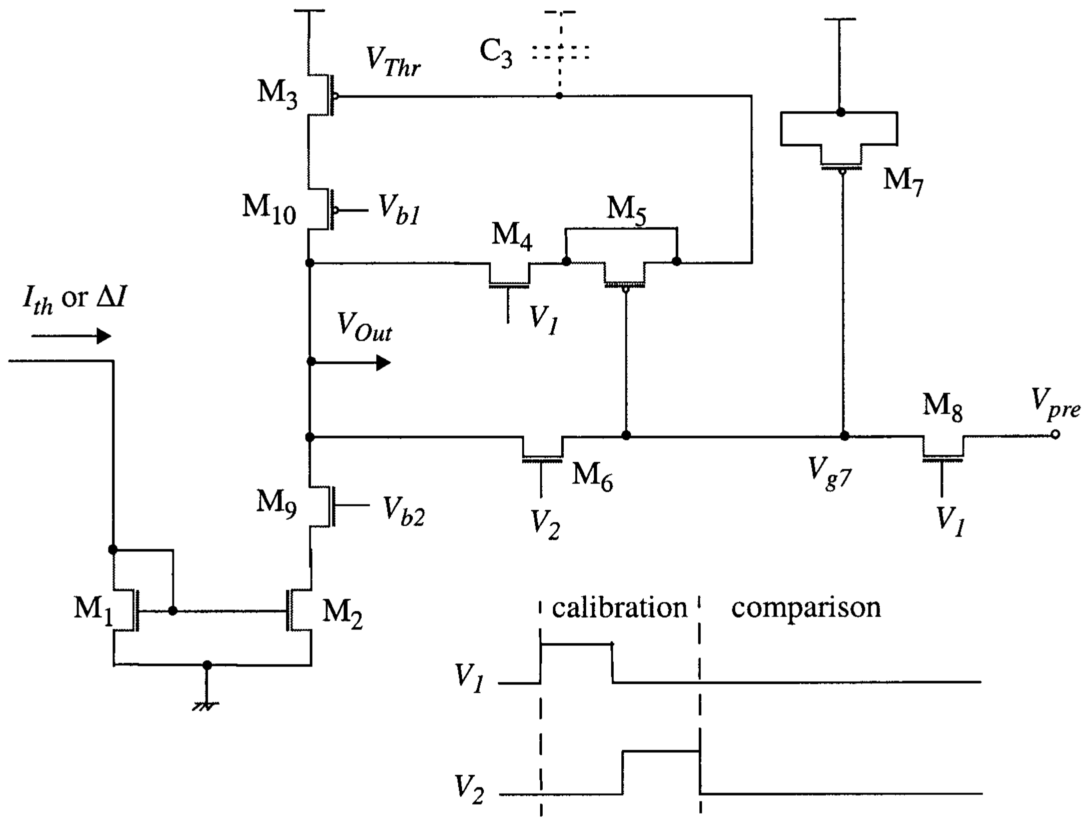


Figure 4.18 Current comparator with the mismatch compensation. During the calibration period, the input current is equal to the threshold value. During the comparison period, the current subtraction result ΔI from the current memory is injected for comparison with I_{th}.

In the simulation shown in Figure 4.19, when the control signal V_2 decreases to 0v, the voltage V_{Thr} is slightly modified, and the voltage V_{Out} changes to a higher level in the case of ($I_{in} = I_{th}$). But, because of the charge division technique implemented by the transistors M_5, M_6, M_7 and M_8 in the circuit, the variation of the voltage V_{Thr} is made to be less than 1mV. Thus, the threshold current I_{th} is memorized at the node V_{Thr} with a very small error ΔI_{th} during the calibration period. It can be observed from Figure 4.19 that such a threshold error ΔI_{th} in the comparator is less than 3%.

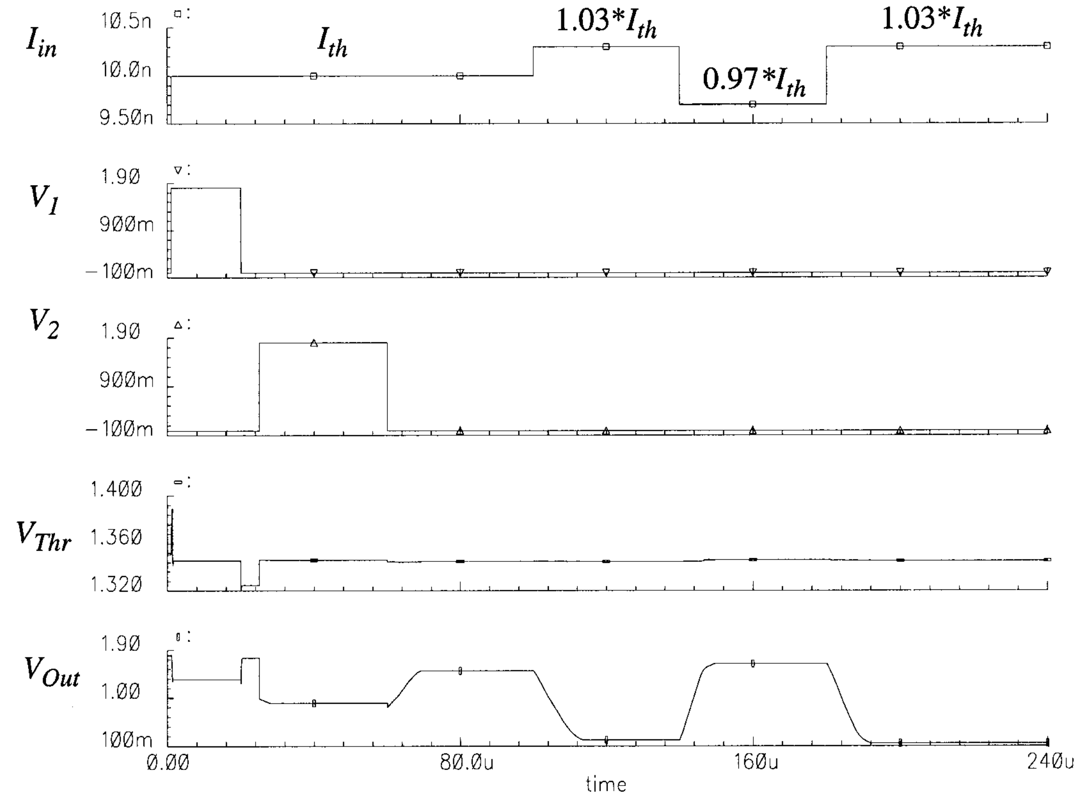


Figure 4.19 Simulated waveforms of the current comparator shown in Figure 4.18. The comparator is calibrated with I_{th}=10nA when the control signals V_1 and V_2 are at the high level successively. After that, with the input current changing from $0.97I_{th}$ to $1.03I_{th}$, the output voltage V_{Out} can change from the level near to V_{DD} to the level near to V_{SS}. The error ΔI_{th} of the comparator threshold is less than 3%.

4.3.2 Evaluation of the Effect of the Mismatch in the Current Comparator Cells and the Efficiency of the Compensation

In order to evaluate the mismatch effect in the current comparator cells and the significance of the compensation scheme used in the circuit, Monte Carlo analysis of two groups of comparators is conducted.

The first group of comparator cells has the basic current comparator structure shown in Figure 4.20. The second group has the same circuit structure shown in Figure 4.19. The non-uniformity of the transistor parameters, is set the same in the two group of comparator cells for the simulations with a current threshold set to be 10nA.

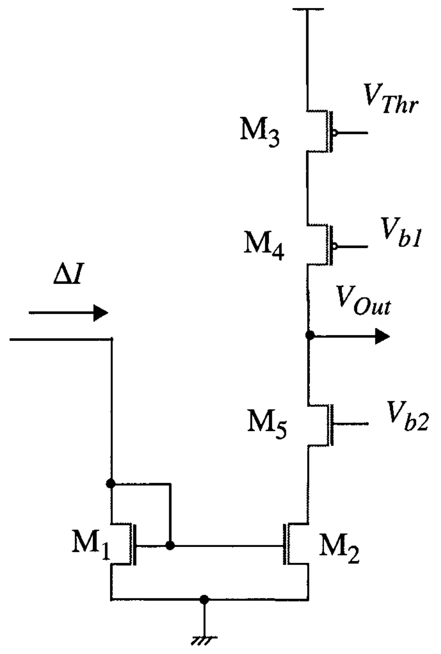


Figure 4.20 Basic current comparator to be simulated in the Monte Carlo analysis. The input current ΔI comes from the subtraction result of the current memory.

Based on the process information given by a 0.18μm CMOS technology, the spatial variation of the transistor parameters shown in Table 4.4 is used to model the mismatch information for Monte Carlo simulations. In such a case of the parameter variation, the maximum current variation in a transistor operating in the subthreshold region can be estimated, approximately equal to 30 percent of the designed value, by using the following relationships,

$$I_D \propto \frac{W}{L}\mu_n C_{ox} \Phi_T^2 e^{\left(\frac{V_G}{n\Phi_T}\right)} \tag{4.7}$$

$$C_{ox} \propto \left(\frac{1}{T_{ox}}\right) \tag{4.8}$$

where, W and L are the channel width and length of the transistor, m_n is the mobility of charge carriers, C_{ox} is the gate oxide capacitance per unit area, n is a slope factor between 1 and 2, Φ_T is the thermal voltage, V_{GS} is the gate-to-source voltage, and T_{ox} is the transistor gate thickness. Using Equation (4.7) and (4.8), the maximum current variation of a pair of transistors in a current mirror, can also be estimated, approximately equal to 70 percent of the designed value.

	Parameter Variation		
	T_{OX} (%)	*W* (%)	*L* (%)
NMOS	*5%*	*10%*	*10%*
PMOS	*5%*	*10%*	*10%*

Table 4.4 Transistor parameter variation for Monte Carlo simulations. T_{OX}: transistor gate thickness; W and L: the channel width and length of the transistor.

For each of the first group of comparator cells, i.e. those having basic current structure shown in Figure 4.20, the current threshold is not equal to any of the other cells in the group, due to the transistor parameter variation. The comparison thresholds of the cells of the second group, i.e. cells with the compensation, are much more uniformed than those in the first group. The Monte Carlo simulation results are shown in Figure 4.21. Under the conditions of the set parameter variations (Table 4.4) and the set current threshold of 10nA, the non-uniformity rate of the comparison threshold, $|(I_{th} - I_{th(nominal)})/I_{th(nominal)}|$, in the cells of the first group can be as high as 163%. Under the same condition, this rate in the cells with the compensation is below 10%. It is shown that the current comparison operation in the cells of the second group is much less sensitive to the spatial parameter variations than the first one. Thus, the efficiency of the compensation scheme is proved.

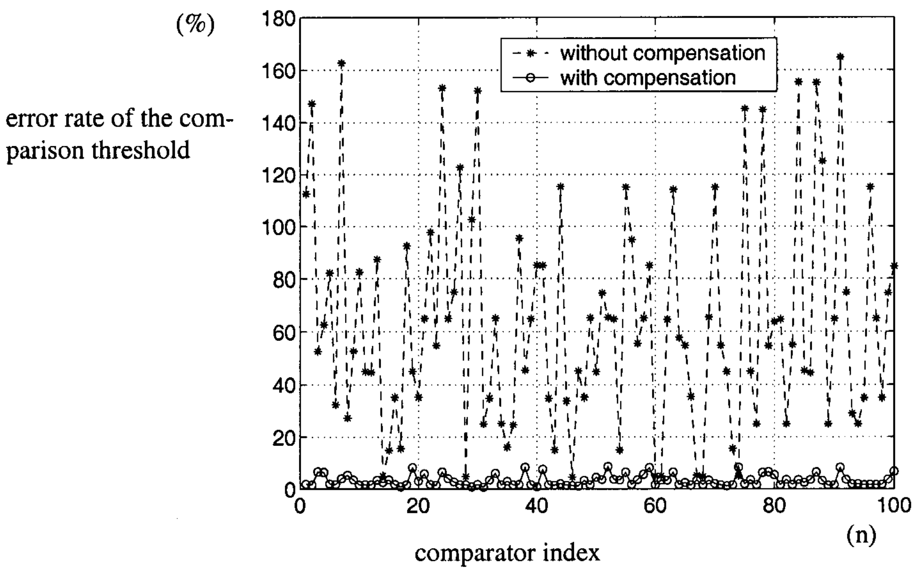


Figure 4.21 Comparison of the uniformity of the threshold of comparator cells with and without the compensation scheme. With the same parameter variation, the errors of two circuits are below 10% and over 100%, respectively.

4.4 Summary

In this chapter, the design of the pixel array circuit implementing the proposed algorithm for edge detection in Chapter 3 has been presented. The structure of the pixel circuit is carefully chosen to be suitable for the weak photocurrent situation. In each pixel, with the photocurrents controlled by the switch unit, three operations, subtraction, comparison and accumulation, are repeatedly performed by a current memory, current comparator and two-bit adder, respectively.

In the current memory circuit, a charge division technique is applied to suppress the effect of charge injection during the transistor switching. The reduction rate depends on the ratio of two parasitic capacitors, without the need of the transistor match. In order to improve the current sampling speed, which is the bottleneck of the detection speed of the pixel circuit, a voltage boost technique is used in this current memory design. In the design of the comparator, a compensation scheme is used to make the threshold of the comparator insensitive to the transistor mismatch. Thus, all the pixel can have the uniform characteristics, which improves the performance of the pixel array for edge detection. As the pixel circuit is designed to receive the photocurrents one after another, the signal flow in the pixel array is easily arranged by the switch unit in each pixel. By simplifying the logic function in the two-bit adder design, the number of transistors in the pixel circuit is minimized.

The effectiveness of the techniques used in the pixel circuit design has been verified by the simulations. The simulation results show that performance of the current memory and comparator is improved by those techniques in the weak current operation. Thus, the pixel array circuit can perform the operations for edge detection.

There are in total sixty-two transistors and one photodiode in each pixel circuit. Using a 0.18μm CMOS technology, it is estimated that the size of each pixel is about 50x50μm^2. No passive element, such as capacitors, is needed in this design. Thus, the pixel array circuit can be fabricated using a CMOS digital technology and the pixel array has a reasonable resolution for edge detection.

In the following chapter, the conclusion of this thesis will be presented and the future work will be also discussed.

5

Conclusions

5.1 Summary of the Work

In this thesis, the design of a CMOS pixel array circuit for edge detection has been presented. The image acquisition and pre-processing are integrated in a pixel array, which can help reduce the power dissipation and increase the image processing speed in a compact system. The work of the thesis has been conducted in two aspects, the algorithm of edge detection and its implementation in a pixel array circuit.

A new version of the algorithm for edge detection has been proposed aiming at facilitating its implementation in a pixel array. Adapting a special 3x3 mask to its spatial convolution, the algorithm can be implemented in a pixel array with four interconnections per pixel, instead of eight in most of the currently used algorithms. Also, to simplify the structure of the circuit in each pixel in the implementation, a new procedure of computation is proposed for the algorithm. In this procedure, simple Boolean functions are designed to replace part of analog processing to minimize inaccuracy in analog implementation. With very simple operations of subtraction, comparison and binary addition in its procedure of computation, the proposed algorithm is like other edge detection algorithm, involving the processing steps such as spatial convolution, absolute value, thresholding, and low-pass spatial filtering. The simulation results have shown that the proposed algorithm yields the processing quality as good as those of the most commonly used ones.

With the proposed algorithm of edge detection, the circuit of the pixel array is designed to be a structure network with pixels of simple processing units and each pixel having four connections to its neighbors. In each pixel, there are a photodiode, a current memory, a current comparator, a two-bit serial adder and MOS switches. The optical signal of the image is acquired while the operation procedure of the detection algorithm is being performed. The main research efforts have been put on the reduction of some negative effects in analog blocks, in particular, the transistor mismatch and charge injection. The transistor mismatch makes the identically designed devices behave non-identically, and thus the characteristics change from pixel to pixel. Current differences in each pixel are calculated in a current memory, instead of current mirror to avoid the mismatch of the transistor pairs. The most critical problem in the current memory, the effect of charge injection is reduced by means of a technique of charge adjustment. Also, a voltage boost procedure is implemented so that the cycle of the operation in the current memory is improved three times. The transistor mismatch could also seriously affect the uniformity of the current comparison in pixels. To this problem, a compensation scheme has been adopted and the uniformity of the current comparison threshold in the pixels is improved more than ten times compared with those without compensation.

In the pixel array circuit, each pixel has sixty-two transistors to perform the operations for the edge detection and compensation procedures to insure the quality of the detection. The circuit can be fabricated using a CMOS digital technology without any extra processing. The silicon area per pixel is about $50x50\mu m^2$ if a $0.18\mu m$ CMOS technology is used for the fabrication. The circuit can be applied in image processing systems for communications and for automatic controls.

5.2 Suggestions for Future Investigations

The coefficients of the signal differences in the proposed algorithm are same, equal to one. However, some edge detection algorithms use different coefficients, such as Sobel using one and two. Thus, such a coefficient extension will make an algorithm more flexible to detect different edge patterns and deserve to be studied.

The capacitor coupling from the control signal at the MOS switch, i.e clock feedthrough in the current memory, can change the voltage at the critical node. To further improve the current memory accuracy, the study of how to reduce this coupling effect should be conducted.

In the pixel array designed in this thesis, the results of edge detection are stored in each pixel. These edge signals can be read out using normal row-by-row or column-by-column techniques. But in a detection system, the edge points are mostly used for further processing, for example, feature extraction. By reading out only those signals in the interested pixel positions, the amount of the data to be further processed can be reduced and the detection speed can be improved. Hence, a study of the read-out circuit in the pixel array needs to be conducted.

References

[1] R.F. Lyon, “The optical mouse, and an architectural methodology for smart digital sensors,” Carnegie-Mellon University’s Conference on VSLI Systems and Computations, October, 1981, Computer Science Press.

[2] R.R. Avent, C.T. Ng, and J.A. Neal, ”Machine vision recognition of facial affect using backpropagation neural networks,” Proceedings of the 16th Annual International Conference of the IEEE Engineering in Medicine and Biology Society, Engineering Advances: New Opportunities for Biomedical Engineers, November, 1994, Pages:1364 - 1365.

[3] J. Derganc and F. Pernus, ”A machine vision system for inspecting bearings,” Proceedings of IEEE 15th International Conference on Pattern Recognition, Volume: 4, September, 2000, Pages: 752 - 755.

[4] M. Gudmundsson, E.A. El-Kwae, and M.R. Kabuka, "Edge detection in medical images using a genetic algorithm," IEEE Transactions on Medical Imaging, Volume: 17, Issue: 3, June, 1998, Pages: 469 - 474.

[5] A. Khashman, "Automatic edge detection of DNA bands in autoradiograph images," Proceedings of the IEEE International Symposium on Industrial Electronics, ISIE '99, Volume: 3, July, 1999, Pages: 1184 - 1188.

[6] J. M.S. Prewitt, "Object enhancement and extraction," In B. S. Lipkin and A. Resonated (eds.), "Picture processing and psychopictorics," Academic Press, New York, 1970, Pages: 75 - 149.

[7] W. K. Pratt, "Digital image processing," Reading, Wiley Interscience, New York, 1991.

[8] N. Kanopoulos, N. Vasanthavada, and R.L. Baker, "Design of an image edge detection filter using the Sobel operator," IEEE Journal of Solid-State Circuits, Volume: 23, Issue: 2, April, 1988, Pages:358 - 367.

[9] S. Hagopian and G. Erten, "Analog Cellular Image Sensor Processor(CISP)," Proceedings of the 43rd IEEE Midwest Symposium on Circuits and Systems, Volume: 2 , August, 2000, Pages: 620 - 623.

[10] L. G. Roberts, "Machine perception of three-dimensional solids," in J.T. Tippet (eds.), "Optical and electro-optical information processing," MIT Press, Cambridge, Massachusetts, 1965, Pages: 159 - 197.

[11] W. Frei and C. Chen, “Fast boundary detection: a generalization and a new algorithm,” IEEE Transaction on Computer, Volume: c-26, No. 10, October, 1977, Pages: 988 - 998.

[12] E. A. Vittoz, "Analog VLSI Signal Processing: Why, Where and How?" Journal of VLSI Signal Processing, Kluwer Academic Publisher, Volume:8, July, 1994, Pages: 27 - 44.

[13] F. Forti and M.E. Wright, “Measurement of MOS current mismatch in the weak inversion region,” IEEE Journal of Solid-State Circuits, Volume: 29, Issue: 2, February, 1994, Pages: 138 - 142.

[14] B. Razavi, “Design of analog CMOS integrated circuits ,” Reading, McGraw-Hill, Boston, 2001

[15] C. Mead, “Analog VLSI and Neural Systems,” Reading, Massachusetts, Addison-Wesley, 1989.

[16] R.A. Deutschmann and C. Koch, “An analog VLSI velocity sensor using the gradient method,” Proceedings of the 1998 IEEE International Symposium on Circuits and Systems, ISCAS’98, Volume: 6, May - June, 1998, Pages: 649 - 652.

[17] A. Pesavento and C. Koch, “A CMOS imager with focal-plane computation for feature detection,” Proceedings of the 2001 IEEE International Symposium on Circuits and Systems, ISCAS 2001, Volume: 3, May, 2001, Pages: 624 - 627.

[18] M. Cohen and G. Cauwenberghs, "Floating-gate adaptation for focal-plane online nonuniformity correction," IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing, Volume: 48, Issue: 1 , January, 2001, Pages: 83 - 89.

[19] F. Lavainne, "Amélioration des qualités optométriques d'une rétine photosensible monolithique à processeurs intégrés," Ph. D thesis, Université de Paris-Sud U.F. R. Scientifique D'Orsay, 1995.

[20] C. Wang, M.O. Ahmad, and M.N.S. Swamy, "Low power current comparator cell for weak current operations," Proceedings of the 2001 IEEE International Symposium onCircuits and Systems, ISCAS 2001, Volume: 1, May, 2001, Pages: 544 - 547.

[21] B. Gilbert, "Translinear circuits: a proposed classification," Electronics Letters, Volume 1, 1975, Pages: 14 - 16.

[22] C. Wang, A.M. Omair, and M.N.S. Swamy, "Design of analog memory based optical sensors," Proceedings of IEEE Region 10 International Conference on Electrical and Electronic Technology, Volume: 2, August, 2001, Pages: 856 - 859.

[23] C. Toumazou, F.J.Lidgey and D.G. Haigh, "Analogue IC design: the current-mode appoach," Reading, Peregrinus on behalf of the Institution of Electrical Engineers, London, 1990.

[24] R. E. Suarez, P.R. Gray, and D.A. Hodges, "All-MOS charge-redistribution analog-to-digital conversion techniques. II," IEEE Journal of Solid-State Circuits, Volume: 10, Issue: 6, December, 1975, Pages: 379 - 385.

[25] N.C. Battersby and C. Toumazou, “Class AB switched-current memory for analogue sampled-data systems,” Electronics Letters, Volume: 27, Issue: 10, May 1991, Pages: 873 - 875.

[26] J.B. Hughes and K.W. Moulding, “S^2I: a switched-current technique for high performance,” Electronics Letters , Volume: 29 , Issue: 16 , August, 1993, Pages: 1400 - 1401.

[27] A. Worapishet, J.B. Hughes, and C. Toumazou, “Low-power high-frequency class-AB two-step sampling switched-current techniques,” IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing, Volume: 50, Issue: 9, September, 2003, Pages: 649 - 653.

[28] S.J. Daubert, D. Vallancourt, and Y.P. Tsividis, “Current copier cells,” Electronics Letters, Volume: 24, Issue: 25, December, 1988, Pages: 1560 - 1562.

[29] C. Wang and F. Devos, “An analog retina for edge detection,” Proceedings of the International Conference on Optical Computing, August, 1994, IOP Publishing Ltd. , Pages: 313 - 316.

[30] C. Wang, M.O. Ahmad, and M.N.S. Swamy, “Design and implementation of a switched-current memory cell for low-power and weak-current operations,” IEEE Journal of Solid-State Circuits, Volume: 36, Issue: 2, February, 2001, Pages: 304 - 307.

[31] D.A. Freitas and K.W. Current, “CMOS current comparator circuit,” Electronics Letters, Volume:19, No. 7, August, 1983, Pages:. 694 - 697.