

# A CMOS Image Sensor with on-chip Motion Detection and Object Localization

Bo Zhao, Xiangyu Zhang and Shoushun Chen  
 VIRTUS IC Design Center of Excellence, School of EEE,  
 Nanyang Technological University, Singapore

**Abstract**—This paper presents a  $64 \times 64$  CMOS image sensor with on-chip moving objects detection and localization capability. The sensor is able to operate at triple modes: temporal difference mode, normal full frame intensity mode and region of interest (ROI) intensity mode. Each pixel is equipped with an analog memory (capacitor) and the whole array is hence capable of storing the previous frame. The rows are sequentially selected for readout and frame difference is digitalized into binary events using a global differentiator. An on-chip hardware-implemented clustering-based algorithm processes the events on the fly and localizes moving objects in the scene. The sensor can switch to ROI mode and shoot a zoomed picture of the object. The proposed image sensor has been implemented using UMC 0.18  $\mu\text{m}$  CMOS technology with a die area of  $1.5 \times 1.5 \text{ mm}^2$ . The power consumption was measured to only 0.4 mW at 100 FPS.

## I. INTRODUCTION

Fast-motion objects detection and localization is important to a number of applications including surveillance, security monitoring and road traffic enforcement [1]. These applications require continuous image acquisition and processing in real time. With the increasing demand on image quality as well as video speed, conventional cameras with nothing or little computing capabilities, fall short of meeting the requirements. Massive quantities of primitive, unimportant image data have to be transmitted and processed before the features of interest are obtained. To make things worse, some systems require multiple sensors to cover a large surveillance area and different view angles, together with the need of higher speed video ( $>100$  frames per second).

These considerations call for smart image sensors that can perform certain image processing on chip and thus reduce the output redundancy. As a matter of fact, a number of image sensors with motion detection capability have emerged during the last decade [2], [3], [4]. These sensors utilize focal-plane frame difference or asynchronous temporal contrast to automatically remove the static background and only output the motion events caused by the moving objects. By using this kind of motion detection image sensor, objects tracking related tasks have been investigated in [5], [6], [7]. However, these systems all require external DSP or FPGA to realize the complex tracking algorithms, which increases the system cost and total power consumption.

In this paper we propose an image sensor with on-chip moving objects detection and localization. The algorithm was implemented on chip without the need of any external computation and storage. The image sensor performs frame

differencing [4] to generate binary motion event, which is processed on the fly to build clusters based on a distance criterion. After the position and size of the active object is obtained, the sensor switches to ROI intensity mode and reports a zoomed picture of the object. By setting programmable threshold of the cluster size, noise objects can be filtered. The aim of this work is to develop a realtime tracking system for low power applications. Our major contribution resides in 1) an efficient clustering-based object localization algorithm, 2) single chip fusion of motion detection and object localization. The rest of paper is organized as follows: Section II introduces the system overview. Section III describes the moving object localization algorithm. The VLSI implementation of the chip is illustrated in Section IV. Section V reports the experimental results and Section VI concludes this paper.

## II. IMAGER ARCHITECTURE

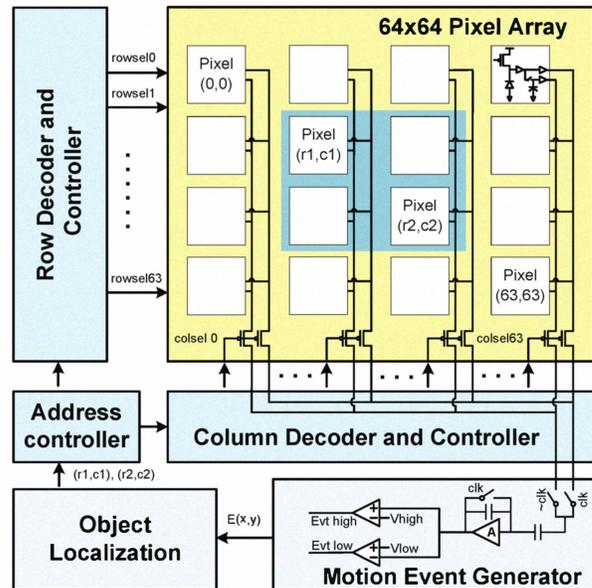


Fig. 1. Block diagram of the image sensor with moving objects detection and localization functionality.

Fig. 1 shows the system architecture of the proposed image sensor. Main building blocks include  $64 \times 64$  pixel array, global event generator, objects localization unit, address controller, row and column decoders. With pixel memory implementation, the image sensor can simultaneously output two consecutive

frames, whose temporal difference are amplified and digitalized to binary motion events by the global event generator [4]. The motion events are processed on the fly to build clusters based on a distance criterion. At the end of the frame, the position and size of the active object is obtained (as the shadowed box in Fig.1). The sensor can be externally triggered to ROI intensity mode and reports a zoomed picture of the object.

### III. MOVING OBJECT LOCALIZATION ALGORITHM

We propose an efficient object localization algorithm and has integrated it seamlessly into the above-mentioned motion detection sensor. The algorithm is implemented in an on-the-fly fashion. It continuously monitors the incoming pixel data (motion event), in which ‘1’ stands for a pixel on a motion object (effective motion event) and ‘0’ for a still background pixel (ineffective motion event). For the sake of clarity, the “event” mentioned in the following part all refer to effective motion event. The key processing element is so-called “cluster”, which is a block of pixels belongs to the same object. As illustrated in Fig. 1, a cluster is visually represented as a rectangle shape and is uniquely defined by its boundary coordinates  $(r1, c1), (r2, c2)$ . By trading off between the immunity against number of noise objects and the implementation complexity, we employs 3 clusters and therefore the chip is able to track 3 objects at the same time.

The details of the clustering-based object localization algorithm is illustrated in Algorithm 1.

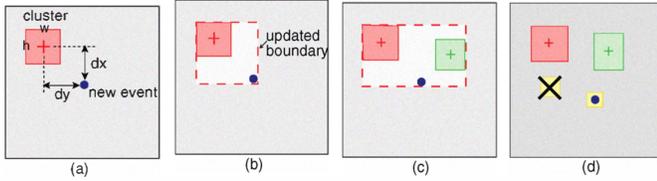


Fig. 2. Clustering of the incoming event. (a) shows the row distance “dx” and column distance “dy” from the current event (blue point) to the center of an existing cluster. (b) illustrates the update process of a cluster, the boundaries are renewed to dash lines by the current event. (c) presents the instant merging operation: the current event belongs to both two clusters at the same time and they are merged together to form a new one (dashed). (d) shows the cluster discarding process, the current event (blue point) doesn’t belong to any of the 3 existing clusters, then the smallest cluster is discarded and a new one is created using the address information of current event.

Fig. 3 shows the intermediate clusters during the sequential scanning of a binary image, which models the manner of output data stream from the image sensor.

### IV. VLSI IMPLEMENTATION

In the proposed algorithm, clusters are updated on the fly with the arrival of each motion event and no external memory is needed. No multiplication or division is involved, enabling hardware friendly on-chip implementation. In this section, we will introduce the VLSI design of the vision chip .

#### A. Motion Detection

The image sensor consists of a  $64 \times 64$  pixel array, each pixel is equipped with an analog memory (capacitor), the whole

#### Algorithm 1 Procedure for event clustering

- 1) Check the number of existing clusters in this frame. If no cluster exists, create a new one defined as “cluster 1” for convenience. A cluster is represented as a rectangle shape with parameters including its boundary coordinates (top left corner:  $(r1, c1)$ , right bottom corner:  $(r2, c2)$ ) and number of events ( $NOE$ ). The parameters of cluster 1 are set up according to the first event.
- 2) If there exists more than one clusters, examine whether the current event belongs to each one of them, based on the criteria:

$$d_x < Th_x \text{ and } d_y < Th_y$$

where  $d_x$  and  $d_y$  are the row distance and column distance between the event and the center of a cluster, respectively (see Fig. 2(a)). Note that  $Th_x$  and  $Th_y$  are not constants, they adaptively increase with the growing of the cluster. Let  $w$  and  $h$  be the width and height of the cluster, then  $Th_x$  and  $Th_y$  can be described as

$$\begin{aligned} Th_x &= Th0 + h/2 \\ Th_y &= Th0 + w/2 \end{aligned}$$

where  $Th0$  is a user-defined threshold.

- 3) If the new event belongs to a single cluster, then corresponding cluster’s features are updated, which means that its center address and boundaries are renewed according to the event (Fig. 2 (b)), and number of events is increased by 1.
- 4) If this event belongs to more than one cluster at the same time, then instance merging is performed. The corresponding clusters and the current event are merged together, forming a new cluster (Fig. 2 (c)). The new cluster’s features are updated accordingly.
- 5) If this event belongs to none of the existing clusters. Meanwhile, cluster resources are still available (e.g. assume we only have three cluster resources, and at present the number of existing clusters is less than three), then a new cluster is created for this event.
- 6) If this event belongs to none of the existing clusters and all available cluster resources are occupied, then the existing cluster with least number of events is discarded and a new one is created based on current event’s information (see Fig. 2 (d)).

array is hence capable of storing the previous frame as a reference image. The rows are first sequentially selected for reset. Later at another round of scanning, the rows of pixels are selected for readout sequentially. Each pixel will output both the new integration voltage on its photodiode and the previous voltage stored on its capacitor. The two voltages are fed into an global event generator circuit which is composed of a global amplifier with a temporal-difference computation circuit based on dual comparison [4]. The event generator computes the difference between the two voltages, and compares it to

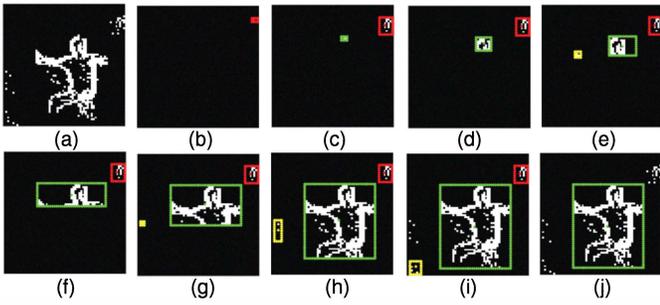


Fig. 3. Evolution of the intermediate clusters during the sequential scanning of a temporal difference image. (a) the test image which models the output data stream from the sensor. (b) The first event of this frame creates cluster 1 (red rectangle). (c) As the events are received and processed sequentially, cluster 1 enlarges and a new cluster (cluster 2, green rectangle) is created since the distance is beyond the threshold. (d) Cluster 2 enlarges, a new cluster (cluster 3, yellow rectangle) is created. (e) shows the instant merging: the new event belongs to both cluster 2 and 3, they are merged together to form to a new cluster 2. (f) presents the status at about half of the frame, a new cluster 3 is created again. (g) shows enlarged cluster 2 and 3 at about 5/6 of the frame. (h) demonstrates the cluster discarding process. The smallest cluster (the yellow one in this case) is discarded and a new one is created. (j) At the end of the frame, based on preliminary knowledge (e.g. the largest object in the scene) or external selection, one cluster is picked out as the aiming object, and its size and position is easily obtained from the description of this cluster.

a positive and negative threshold. A digital motion event is generated if this difference exceeds the thresholds.

Fig. 4 shows the schematic of the pixel. Building blocks include a photo sensing element (photodiode), reset transistor (m1), an internal unity gain buffer made of a NMOS source follower (m2-m4), a sample-and-hold circuit (m5, m6, c) for storing the previous photo sensing voltage and two sets of PMOS source follower readout paths (m7-m8 and m9-m10). In order to address the body effect and improve the linearity, both the NMOS and PMOS source follower transistors are designed in dedicated P Wells and N wells, respectively. This comes at the cost of a minor increased silicon area.

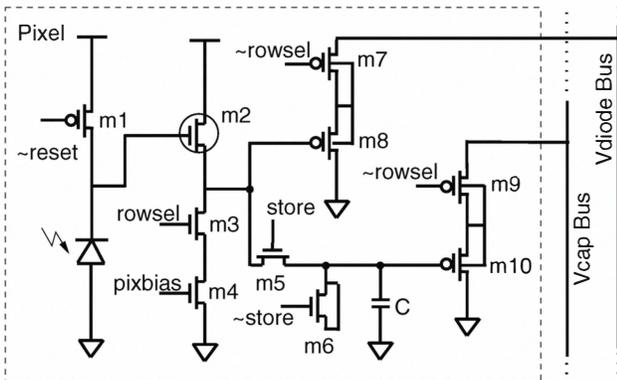


Fig. 4. Schematic of the pixel. Low threshold NMOS follower m2 is implemented and highlighted with circle. Capacitor C is the memory device that stores the previous-frame pixel value.

## B. Object Localization

Followed by motion detection, each active event is sent in parallel to all the clusters in order to determine whether

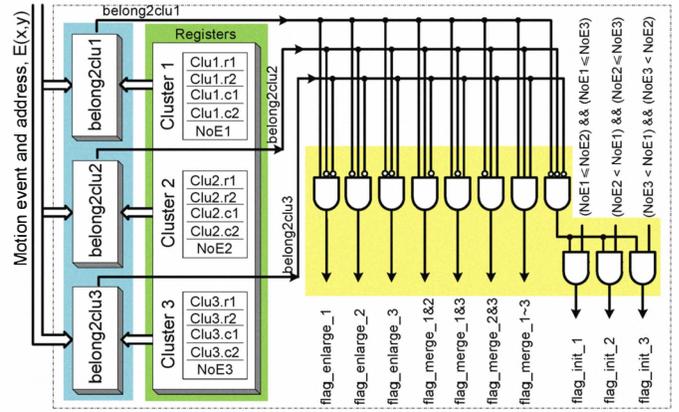


Fig. 5. Generation of the clustering flags. Based on three 1-bit signals (“belong2clu1”, “belong2clu2” and “belong2clu3”), ten clustering flags are generated for further selection of boundary update results.

this pixel belongs to either of them based on the distance criteria as explained in section III. As shown in Fig. 5, the 3 bits of comparison results combined with the information of number of events (NoE) in each cluster, are used to decode a variety of operational flags. For instance, the combination of ‘b100 represents the fact that the motion event only belongs to cluster-1 and therefore cluster-1 should be enlarged. Another case of ‘b111 implies a merge operation of the three clusters. If the event is found belonging to none of the clusters, the cluster are further compared on the basis of their number of events (NoE). The one holding the least number of events is considered as a noise object and will be initialized.

The aforementioned flags then control a series of tri-state buffers to update the cluster registers. In fact, each flag corresponds to a dedicated arithmetic building block which computes possible future boundaries and number of events (NoE). For clarity, here we only show the cluster-1 registers (see Fig. 6). For instance, when “flag\_init\_1” is asserted, cluster-1’s (r1,c1), (r2,c2) are both initiated at the address of the incoming event (x,y) and (NoE) is reset to ‘1’. When “flag\_merge\_1&2” is true, the bounding box which comprises cluster-1, cluster-2 and the event is chosen as the next value of cluster-1.

## V. EXPERIMENTAL RESULTS

The proposed single-chip smart sensor was fabricated using UMC 0.18  $\mu\text{m}$  CMOS process. The chip has a die size of 1.5mm  $\times$  1.5mm (including pads). Fig. 7 shows the chip microphotograph with main building blocks highlighted. Each pixel features an area of 14  $\times$  14  $\mu\text{m}^2$  with a fill-factor of 32%. The object localization unit was implemented with standard cells and occupies a relatively small footprint of 600  $\times$  220  $\mu\text{m}^2$ . Guard rings were extensively used to limit substrate coupling and shield the pixels from the outer-array digital circuitry. Power and ground buses were routed using top layer metal. The chip characteristics are also summarized in Table I.

Fig. 8 reports a few sample images of the image sensor. A demo video has also been made and can be accessed from our

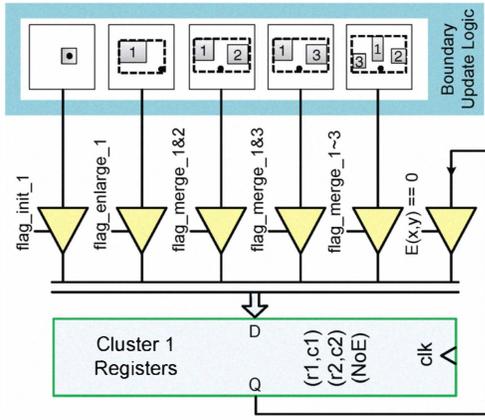


Fig. 6. Update of cluster-1 registers. Cluster-1 related flags control a series of tri-state buffers. Corresponding boundary update results are sent to the bus and further latched to the registers.

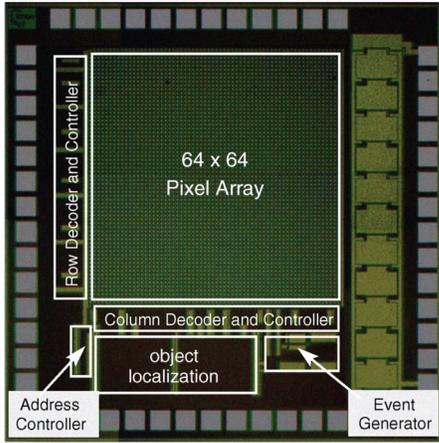


Fig. 7. Chip microphotograph with main building blocks highlighted.

lab website [8]. In this example, we assume the largest object in the scene is the target of interest and one can note that the sensor can locate and zoom to the running person or vehicle quite well. Due to the limited resolution of  $64 \times 64$  pixels, the image quality of the ROI is not very satisfying when the object goes farther. With the successful proof of this concept sensor, we plan to adopt multi-resolution strategy in the future. The proposed localization algorithm works at low resolution and high frame rate. Once the object of interest is determined, the sensor switches to higher resolution and only reports analog image of the ROI.

TABLE I  
CHIP CHARACTERISTICS

Process Technology	UMC 0.18 $\mu\text{m}$ 1P6M CMOS
Die Size	1.5 x 1.5 $\text{mm}^2$
Pixel Array	64 x 64
Pixel Size	14 x 14 $\mu\text{m}^2$
Number of trans/pixel	10
Fill Factor	32%
Readout Strategy	sequential scan
Frame Rate	100fps
Supply Voltage	1.8 v
Power Consumption	pixels array + motion detection 0.4 mW, object localization 8.63 $\mu\text{W}$ (@100fps)

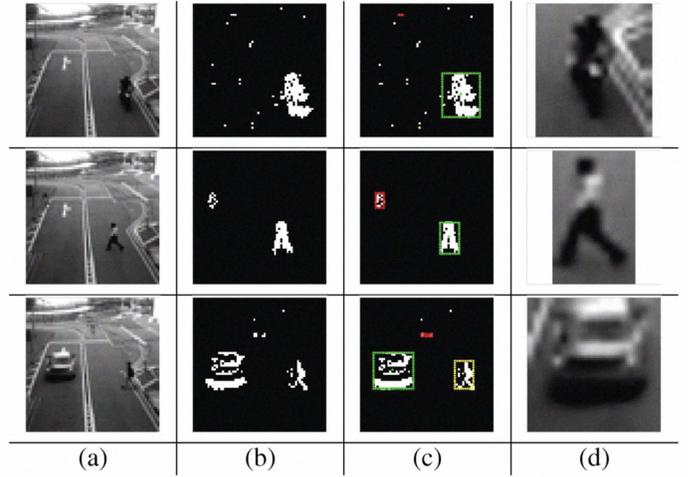


Fig. 8. Sample images of the image sensor. (a) gray level images captured in full frame intensity mode. (b) corresponding images captured in temporal difference mode. (c) the localization results on top of the temporal difference images. (d) the snapshot of the ROI (images are resized in software for better visualization).

## VI. CONCLUSION

A single-chip CMOS image sensor with moving objects detection and localization capability is reported in this paper. Temporal differences are generated and digitalized at focal-plane. With the hardware-implemented clustering-based object localization algorithm, the image sensor can localize moving objects in the scene and take a snapshot of the ROI. The overall system does not require any external computation and storage. It was shown that the processor itself is very compact and consumes little power. The proposed design is an ideal candidate of wireless sensor network node.

## ACKNOWLEDGMENT

This work was supported by Nanyang Assistant Professorship (M58040012) and ACRF Project (M52040132).

## REFERENCES

- [1] Y.-J. Wu, F.-L. Lian, and T.-H. Chang, "Traffic monitoring and vehicle tracking using roadside cameras," in *IEEE Int. Conf. Systems, Man and Cybernetics, SMC '06.*, vol. 6, 2006, pp. 4631–4636.
- [2] S. Mizuno, K. Fujita, H. Yamamoto, N. Mukozaka, and H. Toyoda, "A  $256 \times 256$  compact CMOS image sensor with on-chip motion detection function," *IEEE J. Solid-State Circuits*, vol. 38, no. 6, pp. 1072–1075, 2003.
- [3] P. Lichtsteiner, C. Posch, and T. Delbruck, "A  $128 \times 128$  120 dB 15  $\mu\text{s}$  latency asynchronous temporal contrast vision sensor," *IEEE J. Solid-State Circuits*, vol. 43, no. 2, pp. 566–576, 2008.
- [4] S. Chen, W. Tang, and E. Culurciello, "A  $64 \times 64$  pixels UWB wireless temporal-difference digital image sensor," in *IEEE ISCAS 2010*, 2010, pp. 1404–1407.
- [5] M. Litzberger, C. Posch, D. Bauer, A. Belbachir, P. Schon, B. Kohn, and H. Garn, "Embedded vision system for real-time object tracking using an asynchronous transient vision sensor," in *Digital Signal Processing Workshop*, 2006, pp. 173–178.
- [6] T. Delbruck and P. Lichtsteiner, "Fast sensory motor control based on event-based hybrid neuromorphic-procedural system," in *IEEE ISCAS 2007*, May 2007, pp. 845–848.
- [7] T. Hamamoto, S. Nagao, and K. Aizawa, "Real-time objects tracking by using smart image sensor and FPGA," in *Proc. 2002 Int. Conf. Image Processing*, 2002.
- [8] <http://www3.ntu.edu.sg/home2009/zhao0130/autozoom/demo.htm>.