Suppressing Rolling-Shutter Distortion of CMOS Image Sensors by Motion Vector Detection

Article *in* IEEE Transactions on Consumer Electronics · December 2008

DOI: 10.1109/TCE.2008.4711190 · Source: IEEE Xplore

3 authors, including:

Jung-Bum Chun
Korea Advanced Institute of Science and Technology
4 PUBLICATIONS **41** CITATIONS

SEE PROFILE

Chong-Min Kyung
Korea Advanced Institute of Science and Technology
334 PUBLICATIONS **1,966** CITATIONS

SEE PROFILE

Suppressing Rolling-Shutter Distortion of CMOS Image Sensors by Motion Vector Detection

Jung-Bum Chun, Hunjoon Jung, and Chong-Min Kyung, *Senior Member*, IEEE

Abstract - *This paper focuses on the rolling shutter distortion of CMOS image sensor coming from its unique readout mechanism as the main cause for image degradation when there are fast-moving objects. This paper proposes a post image processing scheme based on motion vector detection to suppress the rolling shutter distortion. Motion vector detection is performed based on an optical flow method at a reasonable computational complexity. A practical implementation scheme is also described.*

Index Terms - CMOS image sensor, rolling-shutter distortion, post-processing technique

I. INTRODUCTION

Since solid-state image sensors replaced films in the consumer electronics market, more and more digital gadgets such as cellular phones and PDA's are increasingly equipped with digital camera function. Image sensor, which converts photons into electrons via photoelectric effect, consists of photodiodes or phototransistors as light-sensing area and peripheral circuitry to control the signal read-out. Image sensors are classified into CCD image sensor and CMOS image sensor (CIS). While CCD can store charges like a memory and can transfer them by means of controlling gate voltages [1], MOSFET's in CIS can only transfer charges from photodiodes to readout circuitry without storing them.

CCD is fabricated through its dedicated fabrication process and is generally known for better image quality than CIS. But a major drawback of CCD is the process incompatibility with CMOS, which makes it difficult to implement peripherals such as timing generator and analog-to-digital converter on the same die with the image sensor. On the other hand, CIS can be built on the same chip as their peripherals due to the process compatibility, which gives CIS an economical advantage over CCD.

Another important difference between CMOS and CCD lies in the signal readout mechanism. While all photodiodes of CCD are exposed to a scene simultaneously to obtain signals corresponding to an image frame, each CIS row, being sequentially accessed, is given a different exposure time window as shown in Fig. 1 (a). We call the readout mechanism of CIS *rolling shutter (RS) mechanism* and that of CCD *synchronous shutter (SS) mechanism*.

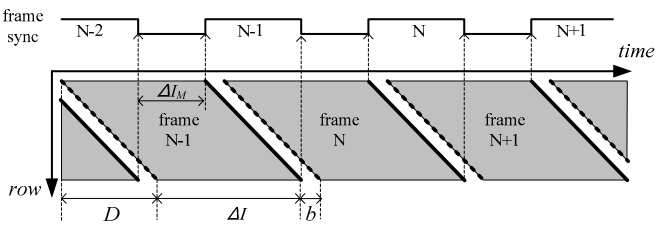
Jung-Bum Chun is with the Electrical Engineering Department, Korea Advanced Institute of Science and Technology, Daejeon, 305-701, Korea (e-mail: jbchun@vslab.kaist.ac.kr).

Hunjoon Jung is with Clairpixel Co., Ltd., Seoul, 153-803, Korea (e-mail: henry@clairpixel.com).

Chong-Min Kyung is with the Electrical Engineering Department, Korea Advanced Institute of Science and Technology, Daejeon, 305-701, Korea (e-mail: kyung@ee.kaist.ac.kr).

Contributed Paper
Manuscript received August 22, 2008

The RS mechanism does not incur any problem as long as the object and the camera are stationary with each other. If either one is moving with respect to the other, then the RS mechanism will produce distorted images as shown in Fig. 1 (b). Three images in the left were taken when the panel is stationary whereas the images in the right were taken when the panel is rotating clockwise. It is shown that distortion patterns are different according to the direction of motion of objects.



(a)

(b)

Fig.1 Rolling shutter mechanism. (a) Integration time (ΔI) of each row, shown as shaded region, is sliding downwards as each row is sequentially accessed to form an image. D and b denote the timing delay between the first and last rows and the blank time between two consecutive frames, respectively. ΔI_M denotes the integration time when a mechanical shutter is employed. (b) Three images in the left are taken when the panel is stationary while the others, showing different distortion patterns according to the direction of motion, are taken when the panel rotates.

This paper proposes a post-processing scheme to reduce the image distortion caused by the RS mechanism. Previous works on the RS mechanism and known alternatives are described in section II. A mathematical analysis of the RS

mechanism is given in section III. In section IV, the implementation scheme for the proposed algorithm is described. Experimental results are given in section V.

II. Previous Works and Other Alternatives

The RS distortion can be completely removed by using a mechanical shutter. By adopting the mechanical shutter, all the photodiodes of CIS are exposed to light during the same time interval, denoted as ΔI_M in Fig. 1 (a), regardless of readout mechanism. Downside of using the mechanical shutter is reduced integration time ($\Delta I \rightarrow \Delta I_M$), and the additional size and cost caused by additional mechanical devices. Since primary applications of CIS are portable devices such as cellulars and PDA's to which imaging is a subordinate function, such overheads due to mechanical shutters is not always justified.

On the other hand, the RS distortion can be reduced by raising the readout speed. In Fig. 1 (a), ΔI denotes the exposure time and D denotes the maximum access time difference between rows. D can be reduced by speeding up the readout while ΔI remains unchanged. However, raising the readout frequency becomes more difficult and requires more power consumption as the number of pixels increases.

El Gamal et al. [2] described a novel CIS architecture where each pixel integrates an analog-to-digital converter to digitize the signal and a latch to store the digitized signal. This architecture enables CIS to operate in the same way as CCD does. However, this architecture is economically impractical due to poor fill factor and poor sensitivity.

Geyer et al. [3] proposed a new camera projection model for camera with the RS mechanism to mitigate the reduction in accuracy and described a framework for analyzing structure-from-motion problems in RS cameras. By parameterizing the velocity of the camera coordinate, the RS effect is applied to the traditional pinhole camera model [4] to derive the projection matrix.

Ait-Aider et al. [5] proposed a technique for pose recovery and 3-D velocity computation by taking the RS effect into account. They took advantage of image deformation induced by the RS mechanism and computed 3-D poses and velocity based on rigid sets of 3-D points.

Liang et al. [6] made the first attempt to correct the RS distortion based on motion vector detection. They found the global motion vector by the block matching and voting method [7]. The block matching technique is similar to that of MPEG 4 where we cannot but sacrifice the accuracy of motion vector to decrease the computation time. Excessive computational load due to the smoothing operation to compensate for inaccuracy of motion vector is not acceptable in mobile devices, major applications of CIS.

In this paper, we utilized an optical flow method which usually produces more accurate motion vector than block matching. To reduce computation, we set center-oriented subwindows, applied the optical flow algorithm and produced one motion vector from those outputs. For low-power consideration, the motion vector detection is performed only when an image capture takes place.

III. Rolling Shutter Analysis

An image sensor array with the RS mechanism is defined in Fig. 2. The sensor has a W x N pixel array and can generate up to F_M frames per second. We define Cartesian coordinates where the origin is located at the top-left corner of the sensor array. In addition, we refer to a new M x N array (in the unit of pixel) located at the center as effective area whose top-left corner is located at (x_1, y_1). The effective area corresponds to the actual image output of the sensor. Generally, the effective area is used for an image output while the rest, called margin area, is utilized by other auxiliary routines like black-level compensation, color interpolation and so on.

Now consider a situation where an image is taken by the image sensor when there is a rectilinearly moving object with velocity v. We assume that integration time ΔI is so small that we can ignore any motion blur in the image and the motion occurs globally throughout the sensing area. We assume that the camera is fixed and the scene moves with a relative velocity. It is possible to find distortion patterns of the effective area with respect to individual components of velocity vector v.

A. Horizontal (x-axis) Motion

CIS is controlled row by row and all the pixels belonging to a row have the same exposure timing.

Consider only a horizontal motion with motion vector $(v_x,0)$ where the moving object is the effective area itself. If the sensor starts reading out the pixel array from the first row after an integration time ΔI, then it takes $H\tau$ to read out the whole array, where H is the number of rows in the array and τ is the time spent to read out a single row.

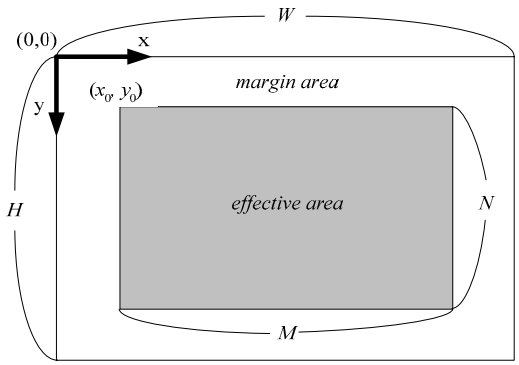


Fig.2 Definitions of related parameters and axes; an image sensor array consists of effective area for actual image output and margin area for other image processing purposes

τ is either a known parameter or can be approximated from other given parameters. When clock frequency f is given, τ is given by W/f since reading a row requires W clocks. If frame rate F_M is given instead of f, a period for a single image frame can be given by $1/F_M - b$ where b is blank time

between image frames as shown in Fig. 1 (a). By equating $H\tau$ with $1/F_M$ - b, τ can be obtained as

$$\tau = \frac{W}{f} = \frac{1 - bF_M}{HF_M} \tag{1}$$

In Fig. 3, the y-coordinate of the k-th row of the effective area is $y_1 + k - 1$ and the time spent by the rolling shutter until it reaches the row is $(y_1 + k - 1)\tau$. Since the row also moves with velocity v_x, multiplying $(y_0 + k - 1)\tau$ by v_x yields $d_{x,k}$, the displacement of the k-th row in the x direction as

$$d_{x,k} = v_x \tau (y_1 + k - 1) \tag{2}$$

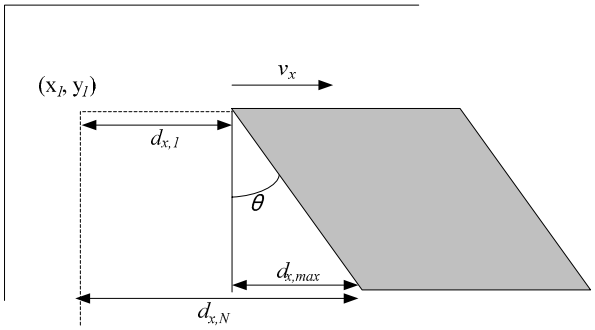


Fig.3 Distortion in a horizontal motion; a rectangular object is distorted into a parallelogram due to the RS distortion

Since $d_{x,k}$ is represented as the sum of a constant plus a component proportional to k, the rectangular area is distorted into a parallelogram, where the so-called *skew angle* θ formed by y-axis and a side of the parallelogram is a good measurement to show the degree of distortion. *Maximum horizontal skew,* $d_{x,\max}$, is defined as the difference between $d_{x,k}$ of the first (k = 1) and the last (k = N) row;

$$\begin{aligned} d_{x,\max} &= d_{x,N} - d_{x,1} \\ &= v_x \tau (y_1 + N - 1) - v_x \tau y_1 \\ &= v_x \tau (N - 1) \end{aligned} \tag{3}$$

θ is thus given by

$$\theta = \tan^{-1} \frac{d_{x,\max}}{N - 1} = \tan^{-1} v_x \tau \tag{4}$$

B. Vertical (y-axis) Motion

Let us consider a vertical motion of the effective area with velocity (0, v_y). To understand the vertical distortion, we define *scan velocity* v_{scan}, denoting the number of rows read out per second, as given by the inversion of τ;

$$v_{scan} = \frac{1}{\tau} \tag{5}$$

The displacement of the k-th row of the effective area in the vertical direction is denoted by $d_{y,k}$. When a capture starts, the rolling shutter starts its readout at velocity v_{scan} and the k-th row of the object starts its downward motion at velocity v_y denoting the number of rows traversed by the moving object in a second. Since the time elapsed until the rolling shutter meets the k-th row of the effective area can be written as $(d_{y,k} + y_1 + k - 1)/v_{scan}$ or $d_{y,k}/v_y$.

By equating the two expressions, $d_{y,k}$ can be obtained as follows.

$$\begin{aligned} d_{y,k} &= \frac{v_y (y_1 + k - 1)}{v_{scan} - v_y} \\ &= \frac{v_y \tau (y_1 + k - 1)}{1 - v_y \tau} \end{aligned} \tag{6}$$

Maximum vertical stretch denoted by $d_{y,\max}$ is defined as the difference between $d_{y,1}$ and $d_{y,N}$;

$$d_{y,\max} = d_{y,N} - d_{y,1} = \frac{v_y \tau (N - 1)}{1 - v_y \tau} \tag{7}$$

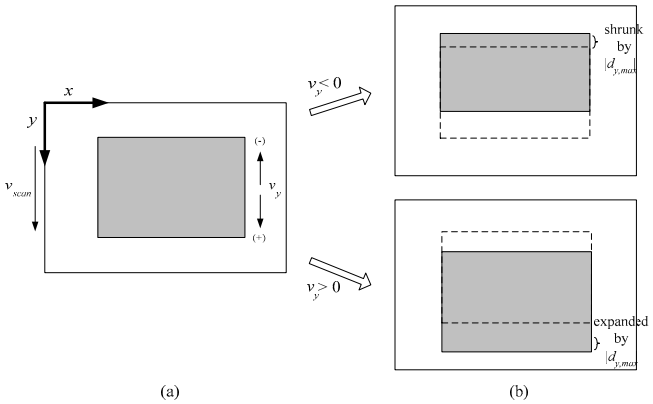


Fig.4 Distortion in a vertical motion. Dotted rectangle denotes the effective area, while shaded rectangles denote how it appears when $v_y = 0$. (a) Original image appears undistorted when $v_y = 0$. (b) Images in the RS system are vertically shrunk (top) when $v_y < 0$, and vertically expanded (bottom) when $v_y > 0$.

If $1 - v_y \tau > 0$, the sign of $d_{y,\max}$ is given by that of v_y. Vertical motion in the RS mechanism causes vertical distortion as shown in Fig. 4. When v_y is positive, the object is stretched by $d_{y,\max}$ and when v_y is negative, it is shrunk by $|d_{y,\max}|$.

C. Motion Vector Composition

For general motions with nonzero values for v_x and v_y, the distortion from the results of the previous sections. Eq. (2), (3), (6) and (7) are still valid under the same definitions. However, the skew angle needs to be rewritten as

$$\theta = \tan^{-1} \frac{d_{x,\max}}{N + d_{y,\max}} \tag{8}$$

The analysis up to now can explain the distortion patterns in Fig. 1. Because a vertical motion is dominant in the top and the middle case, the object is stretched or shrunk. In the bottom case where a horizontal motion is dominant, the object is skewed.

When we consider only rectilinear and global motions, the RS distortion can be represented by an affine transformation from non-distortion space (x, y, 1) to distortion space (x', y', 1) as shown in Fig.5. The image in the non-distortion space can be regarded as the result of the SS system. We can represent the transformation as

$$\begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} = \begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = A \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} \tag{9}$$

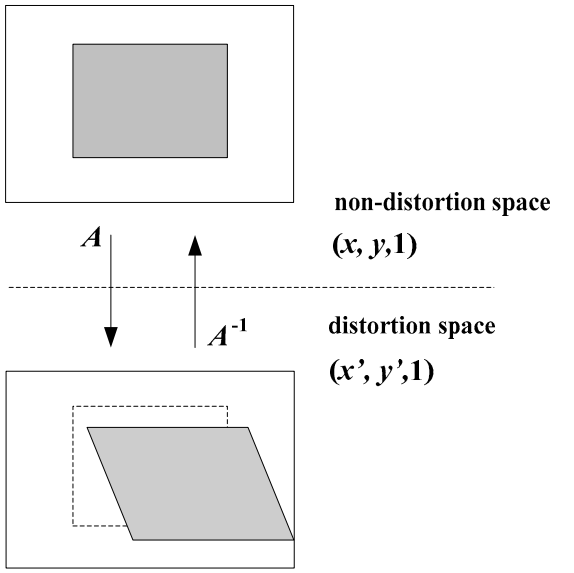


Fig.5 Distortion by rolling-shutter imager can be represented by an affine transformation

The same image as would be obtained by the SS system can also be obtained in the RS system by 'undoing' the distortion through the inverse transformation if the transformation matrix A can be found.

To find matrix A in Eq. (9), instead of substituting matching points between two spaces and solving simultaneous equations with respect to a_{ij}, we took advantage of well-known properties of the affine transformation. In Eq. (9), a_{11} and a_{22} reflect scaling factors with respect to x- and y-axis, respectively. Since no scaling takes place in the direction of x-axis, a_{11} is considered to be a unity. Vertical scaling factor a_{22} can be given by $(N + d_{y,max}) / N$ since the height of the sample is changed from N to $N + d_{y,max}$. On the other hand, a_{12} reflects a shearing factor with respect to x-axis which is given by $\cot\theta = (N+d_{y,max}) / d_{x,max}$ whereas a_{21} is zero since there is no shearing effect in the direction of y-axis. Thus A is given by

$$A = \begin{pmatrix} 1 & \dfrac{N + d_{y,\max}}{d_{x,\max}} & d_{x,1} \\ 0 & \dfrac{N + d_{y,\max}}{N} & d_{y,1} \\ 0 & 0 & 1 \end{pmatrix} \tag{10}$$

The variables can be evaluated by Eq. (3) and (7) along with known sensor parameters such as N and τ if motion vector (v_x, v_y) is acquired.

IV. Implementation Scheme to Reduce Rolling-Shutter Distortions

To reduce the complexity in the implementation, three assumptions were made, shown below along with the rationale for each assumption.

- Motions are rectilinear: Any general motion can be approximated into a combination of rectilinear ones for a sufficiently short period of time.
- Motion blur is less than a certain level: Motion blur, always accompanied by any motion, is affected by exposure time and motion velocity. However, for a short period of exposure time, motion blur can also be assumed to be negligible.
- There is only a global motion in an image: Distortions due to partial motion are less apparent than those due to a global motion. Therefore, partial motions are ignored.

Under these assumptions, we propose a post-processing routine as shown in Fig. 6. The routine receives RGB image as input and generates RGB output image. As a whole, the routine consists of *motion detection stage* and *transformation stage*. It operates in either *preview mode* or *capture mode*. In the preview mode, the sensor generates low-resolution video streams before a capture takes place. In the capture mode, high-resolution target image data are generated by the sensor after a capture signal is generated. Motion vectors, extracted from consecutive images by the motion detection stage in the preview mode, are used in the transformation stage to adjust the image. To reduce power consumption, only two image frames after each capture signal are used in the motion detection, which is accomplished by delaying the mode switching until two preview images are saved after the capture.

A. Motion Vector Detection Stage

Motion vector detection is an important issue in image processing and computer vision studies. Applications like video compression, video mosaic and video surveillance belong to its major application and they utilize block matching technique or optical flow method for the motion vector detection. On the other hand, image stabilization techniques [8], [9] are used for video camera to compensate for image deterioration due to unexpected swing of user's hands. Their motion vector detection and image stabilization method correspond to the first and second step of our approach shown in Fig. 6. To find global motion vectors, Oshima et al. [8] utilize a specialized gyro sensor and Kinugasa et al. [9] derive their motion vectors from consecutive images by projecting images into *x*- and *y*-axis and comparing the current projection with the previous projection. [9] is similar to our approach in utilizing consecutive images but its one-dimensional motion vector detection technique is less accurate than typical two-dimensional techniques.

Most of known methods to find motion vectors are based on consecutive images. The most straightforward way to find a motion vector from two consecutive images is to perform the exhaustive search for all possible relative motions between two frames so that the sum of errors between

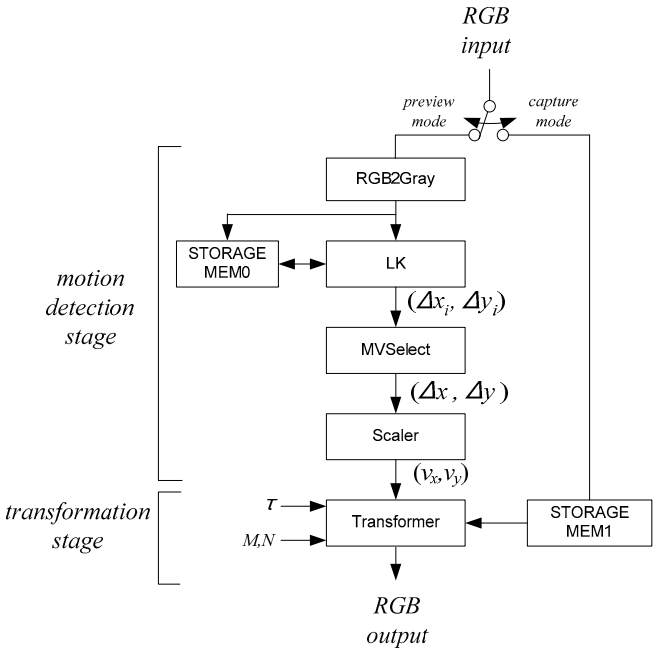


Fig.6 Block diagram of the proposed system

two images is minimal. The computation complexity becomes $O(M^2N^2)$ when the size of image is M x N.

Various methods to reduce the complexity were published [10]-[13]. The most popular one is Lucas-Kanade (LK) algorithm [12] on which our motion detection method is based. In the LK algorithm, for two image data $F(X)$ and $G(X)$ for $X = (x, y)$, error function $E(h)$ is defined as

$$E(h) = \sum_X [F(X + h) - G(X)]^2 \qquad (11)$$

where h is 2-D shift amount.

Motion vector detection between F and G is to find h which makes the error function minimal. Both sides of Eq. (11) are differentiated and equated to zero:

$$\begin{aligned} \frac{\partial E}{\partial h} &\approx \frac{\partial}{\partial h} \sum_X [F(X) + hF'(X) - G(X)]^2 \\ &= \sum_X 2F(X)[F(X) + hF'(X) - G(X)] \\ &= 0 \end{aligned} \qquad (12)$$

Eq. (12) can be solved with respect to h. $F(X)$ is then shifted by h and the same procedure is iterated until h reaches a target value. In this way, the complexity is reduced to $O(MN \log \sqrt{MN})$. Because the proposed routine is expected to be run on mobile devices of which computing power is not as powerful as stationary ones, it is needed to further reduce the complexity at the LK stage of the proposed routine.

In Fig. 6, the RGB2Gray block, the first stage of motion vector detection, performs data conversion from RGB to gray image to reduce the total processing time. Since the luminance component, Y, is dominant in YUV color space, it is possible to find the motion vector only from the gray image. We utilized the RGB-to-YUV conversion formula in [14].

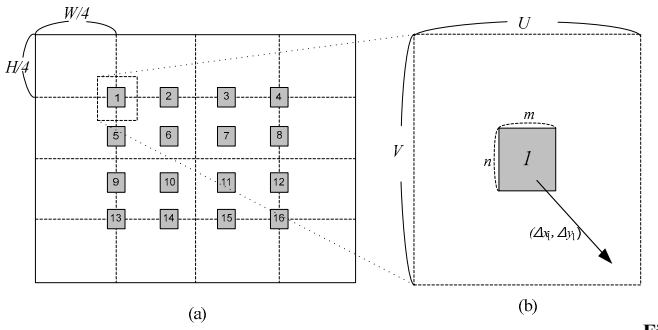


Fig. 7 Template windows and search window for the LK algorithm (a) 16 *m* x *n* template windows uniformly distributed around the center of the array (b) *U* x *V* search window of the first template window

A gray image from the RGB2Gray block is stored in STORAGE MEM 0 and is forwarded to the LK block at the same time. In the LK block, the LK algorithm is individually applied to each pair of template window and its

corresponding search window. As shown in Fig. 7 (a), the indexed $m \times n$ template windows are regularly placed around the center of the sensor array. From the 16 individual operations, 16 individual motion vectors are generated. Fig. 7 (b) shows the first template window and its corresponding search window whose size is U x V. As a result of LK on the i-th position, a motion vector $(\Delta x_i, \Delta y_i)$ is generated. To apply Eq. (11) to our case, $F(X)$ is set as a template window from the previous image and $G(X)$ is a search window from the current image. The size of search window $G(X)$ is defined so that every search window is included by the sensor area.

When the global motion exceeds a certain value in the transformation in Fig. 5, it is possible that some part of the effective area crosses over the sensor boundaries. To prevent this, the maximum velocity limits should be derived.

Consider a velocity vector (v_x, v_y) where $v_x > 0$ and $v_y > 0$. For the whole effective area to remain within the sensor boundary, the x- and y-coordinate of the bottom right corner of the effective area must be smaller than W and H, respectively.

$$\begin{aligned} & x_1 + M - 1 + d_{x,N} \\ & \quad = x_1 + M - 1 + (y_1 + N - 1) v_x \tau \\ & \quad \le W - 1 \end{aligned} \tag{13}$$

$$\begin{aligned} & y_1 + N - 1 + d_{y,N} \\ & \quad = y_1 + N - 1 + \frac{v_y \tau (y_1 + N - 1)}{1 - v_y \tau} \\ & \quad \le H - 1 \end{aligned} \tag{14}$$

Thus, the velocity limit of v_x and v_y, $v_{x,\max}$ and $v_{y,\max}$ can be derived by rewriting Eq. (13) and (14) with respect to v_x and v_y;

$$v_x \le \frac{W - M - x_1}{(y_1 + N - 1)\tau} = v_{x,\max} \tag{15}$$

$$v_y \le \frac{H - N - y_1}{(H - 1)\tau} = v_{y,\max} \tag{16}$$

Eq. (15), (16) can be generalized as follows to include all possible moving directions;

$$| v_x | \le v_{x,\max} \tag{17}$$

$$| v_y | \le v_{y,\max} \tag{18}$$

When a global motion happens, motion vectors extracted from all template windows are likely to have similar values. Even in case of a global motion, a vector may be erroneous if the size of a template window is small and/or set on a monotonous area. Generating the global motion vector, it is necessary to minimize the effect from this error. With 16 motion vectors given by the LK block, the MVselect block generates the global motion vector by performing the following procedure.

Procedure: PMVselect
function: select a global motion vector
input: local motion vectors from the template windows
output: a global motion vector

1) Convert the coordinate system of all the motion vectors from Cartesian to polar coordinates, i.e., $(\Delta x_i, \Delta y_i) \rightarrow (r_i, \theta_i)$ for all i.
2) Eliminate two motion vectors with the minimum and the maximum angle.
3) Eliminate two motion vectors with the minimum distance and the maximum distance.
4) Convert the remaining motion vectors back to Cartesian coordinates and calculate the medians of x- and y-components out of remaining 12 motion vectors.

First three steps in the PMVselect procedure get rid of error-prone motion vectors and then step 4) derive the final motion vector. We denote the global motion vector obtained from PMVselect by $(\Delta x, \Delta y)$. Since the unit of the vector is (`pixel`, `pixel`), the Scaler block divides each coordinate component by the period of an image frame to represent the motion vector in the unit of velocity. Hence, the final motion vector is represented by (v_x, v_y) where any vector component violating Eq. (17) or (18) is replaced by one of the boundary values. Using the procedures above, the complexity of the proposed algorithm can decrease to $O(16mn \log \sqrt{UV})$.

B. Transformation Stage

Images from the RS system belong to (x', y',1) space in Fig. 5. Instead of calculating A^{-1} for the inverse transformation from (x', y',1) to (x, y,1), the following procedure, PTransform, is executed in the Transformer block so that the integer mapping between two spaces is guaranteed. In the procedure, an interpolation approach can be adopted in step 2) instead of selecting the nearest integer, which, however, produces no appreciable difference in our situation. To execute PTransform, a captured image needs to be stored in FRAME MEM 1. We store not the entire image but part of it which covers all search windows.

Procedure: PTransform
function: rearrange each pixel
input: motion vector, coordinates of each pixel
output: rearranged pixel data

For each (x, y) of the effective area,
1) Calculate (x' y' 1)^T = A(x y 1)^T to have (x', y', 1).
2) If either u or v is not an integer, replace it with the nearest integer.
3) The pixel value at (x', y', 1) is delivered as output.

V. EXPERIMENTS

For the performance test of the proposed motion detector, consecutive image frames from a CIS-equipped video camera with VGA resolution were applied to the proposed motion detector and the results were compared to ones produced by the exhaustive search method.

For each test, ten consecutive image frames were taken with arbitrary camera motions. The images were given indices from 0 to 9 and motion vectors between the earliest images and the others were calculated by the motion detector implemented in software. The size of input images is 320x240 and other variables are set as m=10, n=10, U=80, V=80.

Fig. 8 shows selected test image frames (0th, 5th, 7th, and 9th) of the first example which were taken in an outdoor environment and their intermediate motion vectors based on 16 subwindows. From the intermediate motion vectors, the global motion vector for each input was calculated by procedure PMVselect.

The motion detection based on the exhaustive search method was also performed on the same input to demonstrate the correctness of the proposed approach. We set a center-oriented 120x90 template window on the 0th image, and scanned the target images pixel by pixel and calculated the error between the template window and its overlapped window on a target. The error is defined as sum of the square of all the elements of the matrix obtained by subtraction of two corresponding windows. The motion vector is determined from the position with the least error.

Table I summarizes the results of two approaches and it is shown that both results are close to each other. Fig. 9 shows that the maximum difference is at most one pixel. Table I also shows that the elapsed time to produce nine global motions by the proposed motion detector is only 0.054 seconds while that of the exhaustive search is 25.8 seconds, which demonstrates that the proposed algorithm can be adopted for low-power mobile applications.

For the second example, lower-quality image frames were taken in a low-luminance indoor environment. Fig. 10 shows its selected image frames and their intermediate vectors.

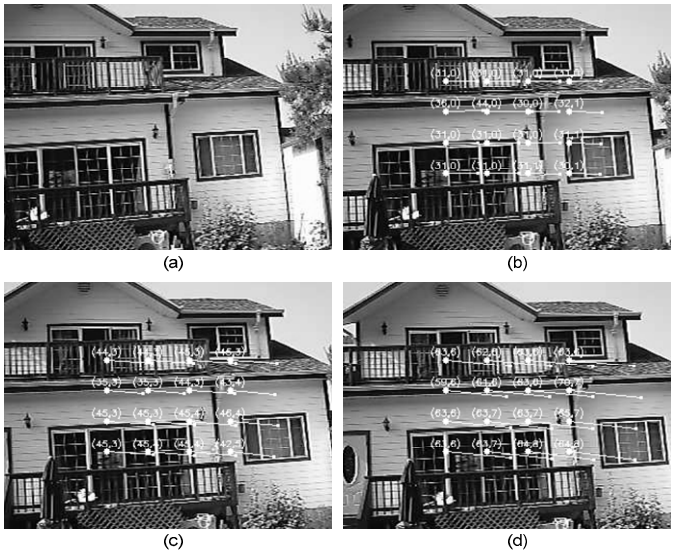


Fig. 8 Four selected images out of ten consecutive ones as inputs to the first example. For (b), (c), and (d), intermediate motion vectors oriented from the centers of 16 subwindows are calculated with respect to (a) and illustrated on their images. (a) 0th, (b) 5th, (c) 7th, and (d) 9th images

Table I Summary of the motion detection of the first example

			frame sequence number								
			1	2	3	4	5	6	7	8	9
proposed motion vector detector	subwindow index	0	(2, -1)	(7, -2)	(15, -1)	(22, 0)	(31, 0)	(38, 1)	(44, 3)	(53, 5)	(63, 6)
		1	(12, -2)	(12, -2)	(8, 0)	(22, 0)	(36, 0)	(46, 0)	(35, 3)	(55, 4)	(59, 6)
		2	(2, -1)	(7, -2)	(15, 0)	(22, 0)	(31, 0)	(38, 1)	(45, 3)	(53, 4)	(63, 6)
		3	(2, -1)	(7, -2)	(15, 0)	(22, 0)	(31, 0)	(38, 1)	(45, 3)	(53, 5)	(63, 6)
		4	(2, -1)	(7, -2)	(15, 0)	(23, 0)	(31, 0)	(38, 1)	(44, 3)	(53, 5)	(62, 6)
		5	(15, -1)	(22, -2)	(21, 0)	(35, 0)	(44, 0)	(52, 1)	(35, 3)	(55, 4)	(61, 6)
		6	(2, -1)	(7, -2)	(15, 0)	(22, 0)	(31, 0)	(38, 1)	(45, 3)	(53, 5)	(63, 7)
		7	(2, -1)	(7, -2)	(15, 0)	(22, 0)	(31, 0)	(38, 2)	(45, 4)	(53, 5)	(63, 7)
		8	(3, -1)	(7, -2)	(15, 0)	(23, 0)	(31, 0)	(38, 1)	(45, 3)	(53, 4)	(63, 6)
		9	(2, -1)	(7, -2)	(14, 0)	(22, 0)	(30, 0)	(38, 1)	(44, 3)	(52, 5)	(63, 6)
		10	(2, -1)	(7, -1)	(15, 0)	(22, 0)	(31, 0)	(38, 2)	(45, 4)	(53, 5)	(63, 7)
		11	(2, -1)	(7, -1)	(15, 0)	(23, 0)	(31, 1)	(38, 2)	(45, 4)	(53, 6)	(64, 8)
		12	(2, -1)	(7, -2)	(15, 0)	(23, 0)	(31, 0)	(39, 1)	(45, 3)	(53, 4)	(63, 6)
		13	(-6, 0)	(12, -1)	(17, 0)	(25, 1)	(32, 1)	(38, 2)	(43, 4)	(62, 5)	(70, 7)
		14	(2, -1)	(7, -1)	(15, 0)	(23, 1)	(31, 1)	(39, 2)	(46, 4)	(54, 5)	(65, 7)
		15	(2, -1)	(7, -1)	(11, 0)	(28, 1)	(30, 1)	(38, 2)	(42, 5)	(59, 6)	(64, 8)
	result motion		(2, -1)	(7, -2)	(15, 0)	(22,0)	(31, 0)	(38, 1)	(45, 3)	(53, 5)	(63, 7)
	elapsed time		0.054 (sec)								
exhast. search	result motion		(3, -1)	(7, -2)	(15, -1)	(23,1)	(31, 0)	(39, 2)	(45, 4)	(53, 5)	(63, 7)
	elapsed time		25.8 (sec)								

Comparison between the proposed approach and the exhaustive search is also illustrated in Fig. 11, where any remarkable degeneration in the accuracy of the produced motion vectors can not be observed.

To show the usefulness of the proposed rolling-shutter compensation system, we implemented a real-time evaluation system on a PC-based environment. The system utilized a web camera [15] which is one of the most common CIS equipment. The program was written in a Visual C++ and the

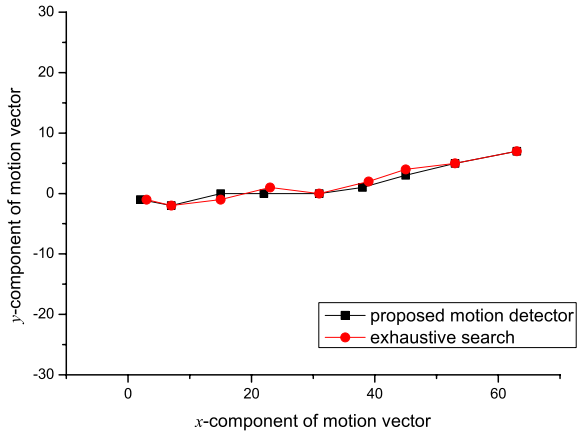


Fig. 9 Motion vector comparison between the proposed and the exhaustive search approach of the first example

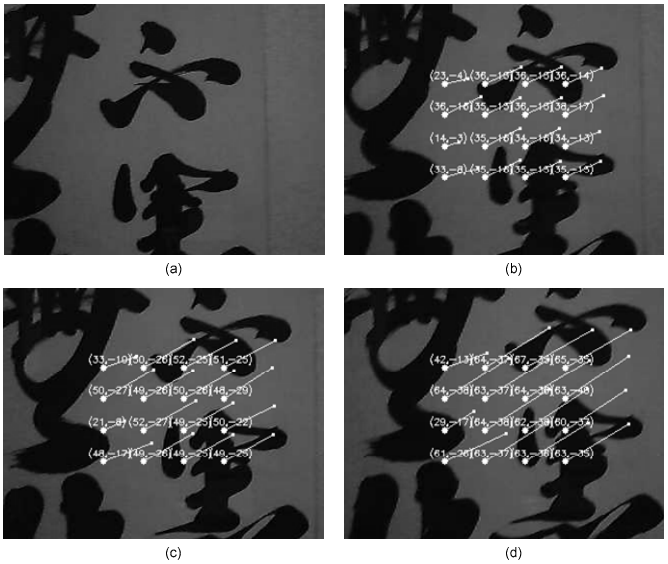


Fig. 10 Four selected images out of ten consecutive ones as the inputs of the second example (a) 0th, (b) 5th, (c) 7th, and (d) 9th images

interface between the web camera and the software was provided by OpenCV [16] which is one of the most popular toolkits for computer vision and image processing. We set the camera to the 640x480 mode and the effective area to 480x320. The maximum frame rate F_M and the black time b are set to 30 `frames/sec` and zero, respectively.

In (a) and (b) of Fig. 12, the distortions of the triangular and rectangular sign in the images (left) are restored by the correction. Although the motion in (c) is not a rectilinear motion but a circular motion, the vertical motion is so dominant that the correction performed vertically has produced a good result, i.e., the stretched letter 'G' are properly resized.

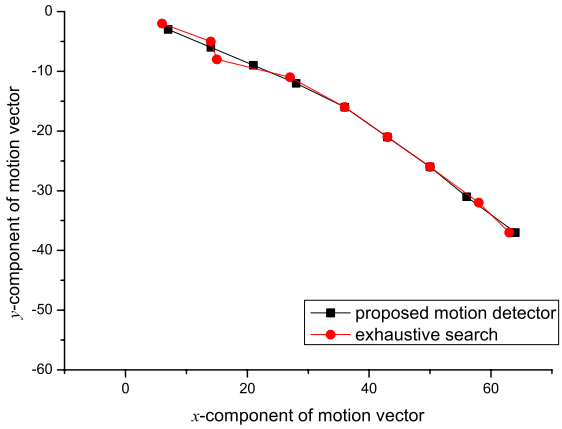


Fig. 11 Motion vector comparison between the proposed and the exhaustive search approach of the second example

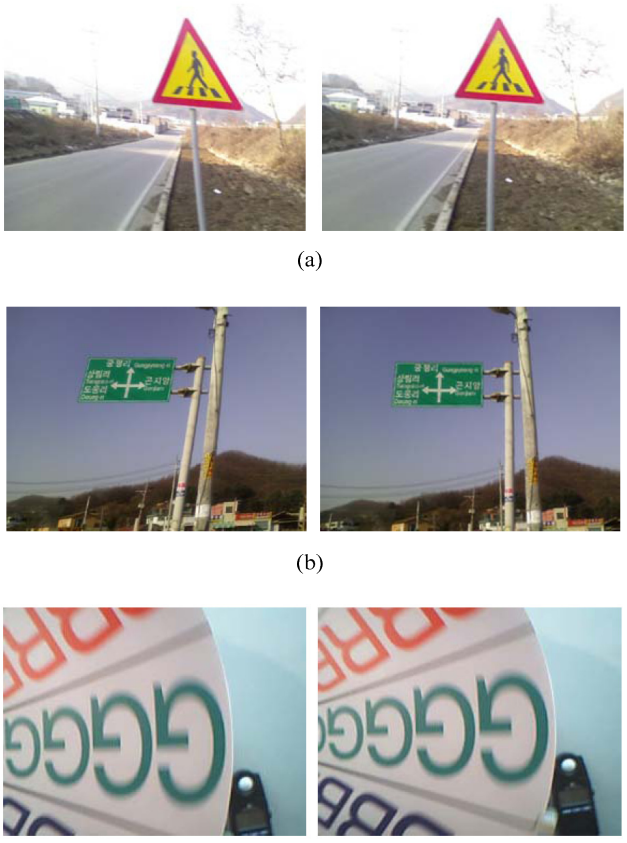


Fig. 12 Corrected rolling shutter distortions. (a) A notably distorted triangular sign (left) is corrected in the right image, (b) a distorted rectangular sign and background (left) are corrected in the right image, and (c) enlarged rotating panel of Fig.1 (left) is resized to the original (right) as a result of motion vector detection and correction

Fig. 13 shows another example where an object in the center moves while its background is stationary, i.e., the motion is not global. But, the correction was performed because the

motion vectors belonging to the moving object formed the majority, thus generating a global motion vector. Although the system ended up with some distortion in the background, the correction can be regarded still beneficial in that more dominant and outstanding part of the image was corrected.

Fig. 13 An example about a center-oriented motion: correcting distortion (left) caused by center-oriented motion can be considered as a feasible result (right) in spite of trivial distortion of its background.

VI. CONCLUSION

Image quality of CIS itself is usually considered to be inferior to that of CCD due to the rolling shutter distortion. In this paper, we proposed a post-processing technique to tackle the rolling shutter distortion known as one of the major weaknesses of CIS.

We presented a mathematical model of the RS distortion in terms of 2D velocity and induced an affine transformation between a non-distortion and a distortion space. The proposed method to correct the distortion can be implemented as hardware or software on recent mobile digital devices which integrate multiple functions with low power consumption. The proposed method is unique in that it utilizes the traditional pixel architecture of CIS so that the economical feasibility is maintained.

REFERENCES

[1] Albert J. P. Theuwissen et al., Solid-State Imaging with Charge-Coupled Devices, Kluwer Academic Publishers, pp. 53-83, 1995

[2] Abbas El Gamal et al., "A 10000 frames/s CMOS digital pixel sensor" , IEEE Journal of Solid-State Circuits, vol. 36, issue 12, pp. 2049-2059, 2001

[3] Christopher Geyer et al., "Geometric models of rolling-shutter cameras " , 2005, available online at http://arxiv.org/abs/cs/0503076

[4] Gerard Medioni et al., Emerging topics in computer vision, IMSC Press Multimedia Series, pp. 7-8, 2004

[5] Omar Ait-Aider et al., "Exploiting Rolling Shutter Distortions for Simultaneous Object Pose and Velocity Computation Using a Single View" , Fourth IEEE International Conference on Computer Vision Systems (ICVS' 06), p. 35, 2006

[6] Chia-Kai Liang et al., "Rolling shutter distortion correction", Proc. SPIE, vol. 5960, pp. 1315-1322, 2005

[7] Chia-Kai Liang et al., "The effect of digital image stabilization on coding performance", International Symposium on Intelligent Multimedia, Video and Speech Processing, pp. 402-405 , October 2004

[8] M. Oshima et al., "VHS camcorder with electronic image stabilizer", IEEE Trans. Consumer Electronics, vol. 35, issue 4, pp. 749-758, Nov. 1989

[9] T. Kinugasa et al., "Electronic image stabilizer for video camera use", IEEE Trans. Consumer Electronics, vol. 36, pp. 520-525, August 1990

[10] J. Bergen, P. Anandan, K. Hanna, R. Hingorani, "Hierarchical Model-Based Motion Estimation", ECCV, pp. 232-252, 1992

[11] D. Fleet, A. Jepson, "Computation of component image velocity from local phase information", International Journal of Computer Vision, vol. 5, pp. 77-104, 1990

[12] B. Lucas, T. Kanade, "An iterative image registration technique with an application to stereo vision", Proc. DARPA Image Understanding Workshop, pp. 121-130, 1981

[13] B. Horn, B. Schunck, "Determining optical flow", Artificial Intelligence, vol. 17, pp. 185-204, 1981

[14] Keith Jack, Video Demystified, LLH Technology Publishing, pp. 15-33, 2001

[15] http://www.microsoft.com/hardware/digitalcommunication/ProductDetails.aspx?pid=002

[16] http://www.intel.com/technology/computing/opencv

Jung-Bum Chun received the B.S degree in electronic engineering from Sogang University, Korea, in 1998, the M.S. degree in electrical engineering from Korea Advanced Institute of Science and Technology (KAIST), Daejeon, Korea, in 2000. He is also studying for the Ph.D. degree at KAIST. From May 2000 to October 2004, he worked for Paion Co., Ltd., Korea, where he developed high-speed Ethernet switching chipsets. From November 2004, he has been developing CIS at Mtekvision Co., Ltd. and ClairPixel Co., Ltd., Korea. Recently, he researchs on image processing algorithms specialized for CIS and their optimal SoC implementation.

Hunjoon Jung received the B.S. degree in physics and Ph., D. degree in materials science and engineering from Seoul National University, Seoul, Korea in 1986, and 2003, respectively. In January 1986, he joined LG Semiconductor Ltd., and worked on DRAM device and CCD design for nine years. He holds 26 patents on CCD. From 1995 to 1998, he was with Applied Materials Korea, and worked on the development of semiconductor manufacturing equipment and process. From 2003 to 2007, he was in charge of Advanced Technology Center of MtekVision Co., Ltd where he developed CMOS image sensor, LCD display driver and DRAM. Currently, he is CEO of ClairPixel Co., Ltd. which is developing novel CMOS image sensors and image signal processors.

Chong-Min Kyung (S'76–M'81–SM'99) received the B.S. degree in electronic engineering from Seoul National University, Korea, in 1975, and the M.S. and Ph.D. degrees in electrical engineering from Korea Advanced Institute of Science and Technology (KAIST), Daejeon, Korea, in 1977 and 1981, respectively. After graduation from KAIST, he worked at AT&T Bell Laboratories, Murray Hill, NJ, from April 1981 to January 1983, in the area of semiconductor device and process simulation. In February 1983, he joined the Department of Electrical Engineering, KAIST, where he is now a Professor. His current research interests include microprocessor/DSP architecture, chip design, and verification methodology. Dr. Kyung served as the Asian Representative in the International Conference on Computer-Aided Design (ICCAD) executive committee. He also served as Vice Chairman of the 1999 COOLChips II held in Kyoto, Japan, and as Cochair of the program committee of Asia and South Pacific Design Automation Conference (ASP-DAC) 2000. He received the Most Excellent Design Award, and Special Feature Award in the University Design Contest in the ASP-DAC, 1997 and 1998, respectively. He is a member of The National Academy of Engineering of Korea (NAEK) and Korean Academy of Science and Technology (KAST)