Analysis and Compensation of Rolling Shutter Effect

Chia-Kai Liang, *Student Member, IEEE*, Li-Wen Chang, and Homer H. Chen, *Fellow, IEEE*

***Abstract*—Due to the sequential-readout structure of complementary metal-oxide semiconductor image sensor array, each scanline of the acquired image is exposed at a different time, resulting in the so-called electronic rolling shutter that induces geometric image distortion when the object or the video camera moves during image capture. In this paper, we propose an image processing technique using a planar motion model to address the problem. Unlike previous methods that involve complex 3-D feature correspondences, a simple approach to the analysis of inter- and intraframe distortions is presented. The high-resolution velocity estimates used for restoring the image are obtained by global motion estimation, Bézier curve fitting, and local motion estimation without resort to correspondence identification. Experimental results demonstrate the effectiveness of the algorithm.**

***Index Terms*—Complementary metal-oxide semiconductor (CMOS) sensor array, motion analysis, rolling shutter.**

I. Introduction

THE DEMAND for image sensors is growing rapidly due to the increasing popularity of digital video cameras in surveillance, personal communications, consumer electronics, etc. Although the charge-coupled device (CCD) has been the dominant technology for image sensors, the complementary metal-oxide semiconductor (CMOS) technology is becoming a popular alternative because CMOS sensors can directly integrate with image processing or communication circuits, providing low cost, low power solutions for various applications.

However, due to the sequential-readout nature of CMOS sensor array, each row (scanline) of the sensor array is exposed at a different time, resulting in the so-called rolling shutter that induces geometric distortion to the image if the video camera or the object moves during image acquisition. As a result, the object may appear slanted, sheared, elongated, shrunk, or even arbitrarily deformed. An example of the rolling shutter effect is shown in Fig. 1, where the vertical edges of a static scene appear slanted to the right as the camera pans to the left across the scene. Such undesirable effect seriously degrades the visual quality of the image.

Fig. 1. Illustration of the rolling shutter effect. The Christmas tree and the vertical edges of the scene appear slanted as the camera pans across the scene.

We are primarily concerned with correcting the image deformation caused by rolling shutter by means of image processing. Although the rolling shutter effect of CMOS sensors is well known, little work has been found in the image processing literature that can effectively address the problem. A mathematical model for rolling-shutter cameras has been developed [1]–[3]; however, the estimation of model parameters requires 3-D correspondences and even object shapes that are difficult to obtain from the kind of images (Fig. 1) dealt with in this work. In contrast, the approach presented here is much simpler and more tractable.

The contributions of this paper are twofold. First, the modeling of the rolling shutter effect is accomplished without resort to 3-D analysis. The planar motion assumption of the model, which is commonly made in many image processing problems, greatly reduces the complexity of the problem. Second, the rolling shutter compensation algorithm provides scanline realignment at high resolution that even works for blurry image sequences with moderate depth range or large motion.

The rest of this paper is organized as follows. Section II explains how the rolling shutter is formed, and Section III reviews solutions for the rolling shutter problem. Section IV presents the modeling of the rolling shutter effect, and Section V describes the rolling shutter compensation algorithm. The setup and results of the experiment for testing the algorithm are presented in Section VI. The discussion is given in Section VII and the conclusion in Section VIII.

II. Rolling Shutter Effect

The internal structure of a common 3-transistor active photon sensor is shown in Fig. 2(a) [4]. The photodiode collects the

Manuscript received July, 23, 2007; revised March 25, 2008. This work was supported in part by National Science Council under contracts NSC 94-2213-E-002-070 and NSC 94-2752-E-002-006-PAE. C.-K. Liang was supported in part by the MiTac Technology Corporation. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Stanley J. Reeves.

C.-K. Liang is with the Graduate Institute of Communication Engineering, National Taiwan University, Taipei 10617, Taiwan, R.O.C. (e-mail: liangck@gmail.com).

L.-W. Chang was with the Department of Electrical Engineering, National Taiwan University, Taipei 10617, Taiwan, R.O.C. (e-mail: dddscy@gmail.com).

H. H. Chen is with the Graduate Institute of Communication Engineering, the Graduate Institute of Networking and Multimedia, and the Department of Electrical Engineering, National Taiwan University, Taipei 10617, Taiwan, R.O.C. (e-mail: homer@cc.ee.ntu.edu.tw).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TIP.2008.925384

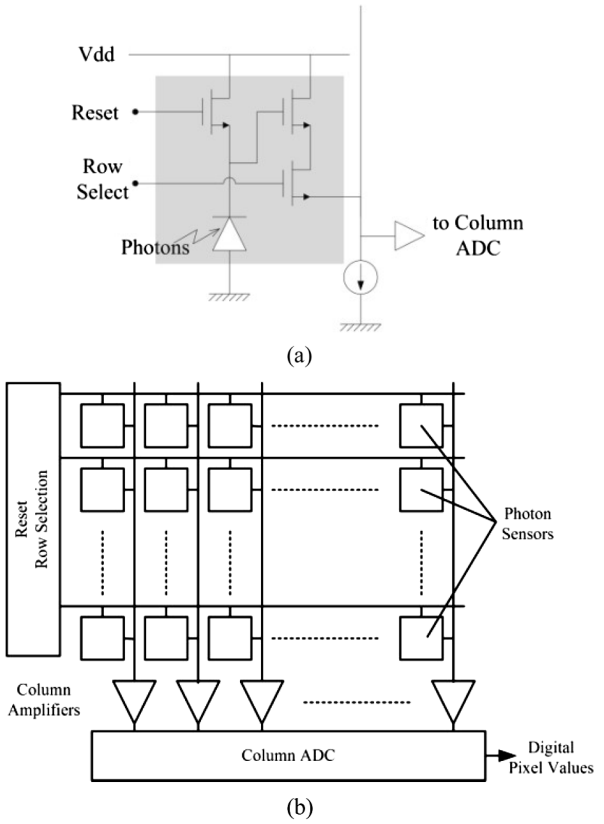


Fig. 2. (a) Schematics of a single CMOS photon sensor. (b) The architecture of the CMOS sensor array. There is only a row of column amplifiers and column ADCs for the 2-D sensor array.

photons and converts them to photocurrent. The charges are integrated in the photodiode capacitor, and the exposure time of the photocurrent integration is controlled by the *reset* signal. Up to this point, the CMOS sensor is similar to the CCD sensor.

However, the sequential-readout structure distinguishes the CMOSE sensor array from the CCD sensor array. In a typical CMOS sensor array architecture, as shown in Fig. 2(b), there is only one row of readout elements so that only one row of sensors is selected by the *row select* signal in each readout operation. The accumulated charges are transferred to the column amplifiers, where the signals are amplified. Then a row of analog-to-digital converters (ADCs) converts the signal to digital form. In this structure, the sampling time of each row is different, so is the reset time of integration.

The timing of readout and exposure is sketched in Fig. 3. Because there is only one row of readout circuits, the readout time of different rows cannot overlap. Also, the exposure of each row cannot begin until the readout of the corresponding scanline of the previous frame is completed. Since the exposure of each row is triggered at a different time, a rolling shutter is in effect and causes image deformation if the object or the camera moves during image capture.

The extent of the rolling shutter effect depends on the exposure time and the readout time. When the imaging chip has a standalone frame buffer, the readout process simply transfers the pixel data to the frame buffer. This operation can be done in a few clock cycles, allowing the total readout time per frame to be less than a few thousand cycles (on the order of millisecond

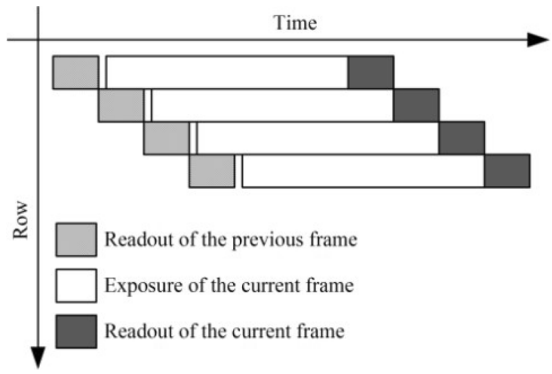


Fig. 3. Timing diagram of the CMOS sensor array shown in Fig. 2. The readout times of the scanlines do not overlap, and the exposure of each scanline can only begin after the readout of the corresponding scanline in the previous frame is completed. The small gap between the exposure and the readout is the time required for the reset of the photo sensor.

for a typical imaging chip) [5], [6]. The readout time in this case is much smaller than the exposure time; therefore, there is little rolling shutter effect unless the object moves extremely fast (more than one pixel in a single readout operation).

However, in cases where there is no on-chip frame buffer, which is a common design choice to save cost, the rolling shutter has a profound effect on image quality. Webcams, for instance, normally have no or very small local buffer and have to transfer via USB or Firewire the recorded image data to the host computer, where the image is processed. The readout time of such devices is limited by the transmission rate. In this case, the readout time is comparable to the exposure time and may dominate the whole imaging process, making the rolling shutter effect profoundly noticeable.

III. Related Work

The rolling shutter effect can be prevented by using a mechanical shutter. Alternatively, one can design a new circuit with a local sample-and-hold for each photo sensor so that the whole sensor array experiences a single exposure [7], [8]. However, this method reduces the fill factor of the photo sensor and degrades the efficiency of photo collection. As a result, the readout rate is still limited by the transmission rate.

Approaches that address the rolling shutter effect by image analysis have been proposed. Geyer *et al.* [1] derived a time-dependent perspective model for the rolling-shutter camera by assuming that the perspective transformation of each scanline is a function of the relative location and velocity between the object and the camera. The condition under which the rolling shutter effect becomes negligible was also derived. Ait-Aider *et al.* further applied the model to estimate the camera parameters and the object dynamics, but the shape of the object and the correspondences of the object between frames must be known *a priori* [2], [3].

Wilburn *et al.* [9] pointed out that the rolling shutter effect is not always undesirable and built a 2-D array of rolling-shutter cameras to achieve high-speed imaging. The final image was constructed by stitching the scanlines acquired by different cameras at the same exposure time. Assuming transitional and constant object motion, Sasaki and Nara [10] used several images

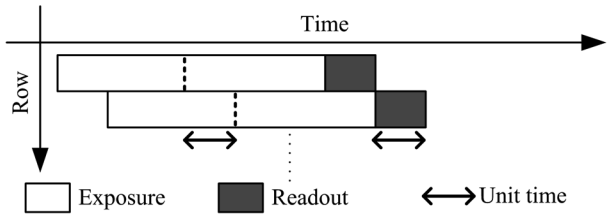


Fig. 4. Timing diagram. We assume each scanline is sampled at a specific instant in the exposure time. The interval between the sampling instants of two successive scanlines is defined as the unit time.

acquired by a rolling-shutter camera to generate a single clean image.

IV. Modeling of Rolling Shutter Effect

We focus on the scenario in which the camera undergoes with respect to the object a translational motion parallel (or nearly parallel) to the image plane. The planar motion model is based on two observations. First, the rolling shutter effect is noticeable when the camera or the object moves rapidly in a single direction. It is less noticeable for complex motions such as rotating and zooming. Second, the difference in image distortion is hardly noticeable when depth range of the objects in the scene is moderate. Such a planar motion model has been commonly adopted for many image processing problems, such as image deblurring, motion-compensated video coding, and video stabilization.

For simplicity, the image with rolling shutter effect is referred to as the distorted image throughout the remaining paper, whereas the image without rolling shutter effect is referred to as the normal image. With the planar motion model, the distorted image due to rolling shutter can be modeled as a transformation of the normal image solely dependent on the relative motion between the camera and the scene. Without loss of generality, we assume the scene is static and only the camera moves.

In the following, n denotes the frame number, $\mathbf{s} = (s_x, s_y)$ denotes the coordinates of a pixel in the distorted image, and $\mathbf{p} = (p_x, p_y)$ denotes the coordinates of a pixel in the normal image. Furthermore, let $\mathbf{v}(t) = (v_x(t), v_y(t))$ denote the camera motion at time t, and T the time interval between two successive images of the normal image sequence. For simplicity, we assume that the nth normal image is sampled at $(n + 0.5)T$ and the pixel $\mathbf{s}$ in the nth frame of the distorted image sequence is sampled at $nT+s_y$. As shown in Fig. 4, each scanline of a distorted image is sampled at a specific instant during the exposure, and the interval between the sampling instants of two successive scanlines is defined as the unit time.

Interframe and intraframe distortions are discussed separately. The intraframe distortion refers to the shape deformation of objects in a video frame, whereas the interframe distortion refers to the deformation of object shape between successive frames of an image sequence.

A. Interframe Distortion

Assume $\mathbf{p}_1$ in the $n-1$th frame and $\mathbf{p}_2$ in the nth frame of a normal image sequence correspond to the same scene point. Since the two frames are sampled at $(n-0.5)T$ and $(n+0.5)T$, respectively, the image displacement of the point can be described by

$$\mathbf{p}_2 = \mathbf{p}_1 + \int_{(n-0.5)T}^{(n+0.5)T} \mathbf{v}(t)\mathrm{d}t. \tag{1}$$

Likewise, the image displacement between a point $\mathbf{s}_1$ in the $n-1$th frame and the corresponding point $\mathbf{s}_2$ in the nth frame of a distorted image sequence can be described by

$$\mathbf{s}_2 = \mathbf{s}_1 + \int_{(n-1)T+s_{1y}}^{nT+s_{2y}} \mathbf{v}(t)\mathrm{d}t. \tag{2}$$

Note that because the sampling time of a scene point depends on the scanline on which it is projected, the integration interval is controlled by the vertical position of the scanline. If there is no camera motion between $(n-1)T + s_{1y}$ and $nT + s_{2y}$, $\mathbf{s}_2 = \mathbf{s}_1$ and the image displacement (or geometric distortion) is zero, there is no rolling shutter effect. This matches with intuition. On the other hand, if $v_y(t) = 0$ during the integration interval, the point remains on the same scanline, but its horizontal displacement is a function of $v_x(t)$. In general, $\mathbf{v}(t)$ can be an arbitrary function. Note that (2) cannot be solved because it is a self-dependent function of s_{2y}. One of the contributions of this paper is to make this equation tractable, as presented in Section V.

B. Intraframe Distortion

Let $\mathbf{p}_1$ and $\mathbf{p}_2$ be two points in the nth frame of a normal image sequence and η a time instant at which the camera is still. Denote the corresponding points of $\mathbf{p}_1$ and $\mathbf{p}_2$ at this time instant by $\mathbf{r}_1$ and $\mathbf{r}_2$, respectively. Note that $\mathbf{r}_1$ and $\mathbf{r}_2$ are auxiliary points for mathematical derivation. Generalizing (1) yields

$$\mathbf{p}_1 = \mathbf{r}_1 + \int_{\eta}^{(n+0.5)T} \mathbf{v}(t)\mathrm{d}t \tag{3}$$

$$\mathbf{p}_2 = \mathbf{r}_2 + \int_{\eta}^{(n+0.5)T} \mathbf{v}(t)\mathrm{d}t. \tag{4}$$

Subtracting (3) from (4), we have

$$\mathbf{p}_2 - \mathbf{p}_1 = \mathbf{r}_2 - \mathbf{r}_1. \tag{5}$$

Similarly, considering two points $\mathbf{s}_1$ and $\mathbf{s}_2$ in the nth frame of the distorted image sequence that correspond to $\mathbf{r}_1$ and $\mathbf{r}_2$, respectively, we have

$$\mathbf{s}_1 = \mathbf{r}_1 + \int_{\eta}^{nT+s_{1y}} \mathbf{v}(t)\mathrm{d}t \tag{6}$$

$$\mathbf{s}_2 = \mathbf{r}_2 + \int_{\eta}^{nT+s_{2y}} \mathbf{v}(t)\mathrm{d}t. \tag{7}$$

Subtracting (6) from (7) yields

$$\begin{aligned}\mathbf{s}_2 - \mathbf{s}_1 &= \mathbf{r}_2 - \mathbf{r}_1 + \int_{nT+s_{1y}}^{nT+s_{2y}} \mathbf{v}(t)\mathrm{d}t \\ &= \mathbf{p}_2 - \mathbf{p}_1 + \int_{nT+s_{1y}}^{nT+s_{2y}} \mathbf{v}(t)\mathrm{d}t.\end{aligned} \tag{8}$$

By rearranging the above equation, we have

$$\mathbf{p}_2 = \mathbf{p}_1 + \mathbf{s}_2 - \mathbf{s}_1 - \int_{nT+s_{1y}}^{nT+s_{2y}} \mathbf{v}(t)\mathrm{d}t \tag{9}$$

which has two implications. First, if the camera does not move during the time interval $(nT+s_{1y}, nT+s_{2y})$, $\mathbf{p}_2-\mathbf{p}_1 = \mathbf{s}_2-\mathbf{s}_1$ and, as we expect, there is no rolling shutter effect. Second, if $\mathbf{s}_2$ and $\mathbf{s}_1$ are on the same scanline, the integration interval in (9) is zero, and, thus, the distance between pixels of a scanline is not affected by the rolling shutter.

The analysis presented above forms the theoretical basis of the rolling shutter compensation algorithm described in the next section. Before moving on, we present another useful equation. In (2), the displacement from $\mathbf{s}_1$ to $\mathbf{s}_2$ is obtained by an integration of $\mathbf{v}(t)$. If $\mathbf{v}(t)$ is continuous, according to the mean value theorem, we can find some τ in the interval $((n-1)T + s_{1y}, nT + s_{2y})$ such that

$$\mathbf{v}(\tau) = \frac{1}{T + s_{2y} - s_{1y}}(\mathbf{s}_2 - \mathbf{s}_1). \tag{10}$$

This imposes a constraint on the velocity $\mathbf{v}(t)$ of a point between two successive frames.

V. Distortion Correction Algorithm

Undoing the rolling shutter effect is a challenging task. There are three primary problems to be addressed. First, from (9), we know that if $\mathbf{s}_1$ and $\mathbf{p}_1$ are identified and the velocity $\mathbf{v}(t)$ is known, the displacement between $\mathbf{s}_2$ and $\mathbf{p}_2$ can be derived and corrected. However, a pair of $\mathbf{s}_1$ and $\mathbf{p}_1$ need be identified first. Second, because the input sequence is geometrically distorted and motion blurred, velocity estimation by feature detection and matching is difficult. An alternative solution has to be sought. Third, the temporal sampling of $\mathbf{v}(t)$ must be at the scanline level, which is much higher than the frame rate. However, traditional motion analysis methods can only produce motion information at frame rate.

Most video sequences begin with a fixed scene and camera. If the camera is still at the beginning of a distorted image sequence so that the first frame is a normal image, the first problem can be solved. Specifically, when $\mathbf{v}(t) = 0$ for $0 \le t \le T$ so that $\mathbf{s} = \mathbf{p}$ for any point in the first frame, the center pixel $\mathbf{s}_m = (s_{\mathrm{mx}}, s_{\mathrm{my}})$ of the nth distorted frame is at the same location as its corresponding point $\mathbf{p}_\mathrm{m} = (p_{\mathrm{mx}}, p_{\mathrm{my}})$ of the nth normal image. This relationship can be derived from (1) and (2) by noting that

$$\begin{aligned}\mathbf{s}_m &= \mathbf{r} + \int_{s_{\mathrm{my}}}^{nT+s_{\mathrm{my}}} \mathbf{v}(t)\mathrm{d}t \\ &= \mathbf{r} + \int_{0.5T}^{nT+0.5T} \mathbf{v}(t)\mathrm{d}t = \mathbf{p}_m\end{aligned} \tag{11}$$

where $\mathbf{r}$ represents the location of $\mathbf{s}_\mathrm{m}$ (and $\mathbf{p}_\mathrm{m}$) in the first frame. We can choose the middle scanline of the image as the reference for compensation.

The second problem can be solved by determining the global motion of the entire frame first and estimating the velocity of each scanline from the resulting global motion. Note that it is easier to obtain robust global motion estimate than corresponding points. Also note that no identification of correspondences is needed in this approach.

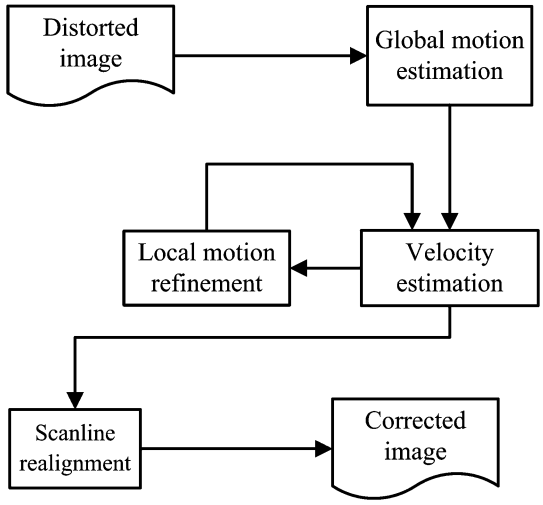


Fig. 5. Block diagram of the proposed algorithm.

The third problem is solved by using parametric curve fitting to interpolate the velocity and achieve fine-granularity velocity estimation. Unlike the deterministic interpolation, the parametric curve fitting can effectively generate smooth data and adapt to local samples [11].

The strength of the algorithm comes from the novel combination of several techniques. The overall flow chart of the proposed system is shown in Fig. 5. First, global motion estimation is performed per frame to obtain the initial velocity estimate. The resulting global motions are interpolated using Bézier curve to estimate the velocity per scanline. Then local motion refinement is performed to improve the velocity estimate. In this step, two constraints based on the analysis described in the previous section are imposed to detect outliers. Local motion refinement and velocity estimation are performed iteratively. Finally, the image is restored by realigning the scanlines.

Each component of the algorithm is detailed in the following. We use $\mathbf{v}'(t)$ to denote the estimated velocity and $\mathbf{v}(t)$ the true velocity.

A. Global Motion Estimation

The block-based motion clustering method described in [12] is adopted to compute the global motion vector frame by frame. Although a simple block matching instead of complex correspondence matching is used for global motion estimation, it is found that the overall system performance is not affected.

First, the current nth image is segmented into blocks (16×16 in our implementation). Second, for each block, a matched block in the previous frame with minimal sum of absolute difference is determined. The displacement between the current block and its matched block is the motion vector. Then, the 2-D histogram of the motion vectors of all blocks of the current frame is generated and low-pass filtered to reduce the estimation noise. Finally, the peak in the filtered histogram is chosen as the global motion vector $\mathbf{g}(n)$ of the current frame.

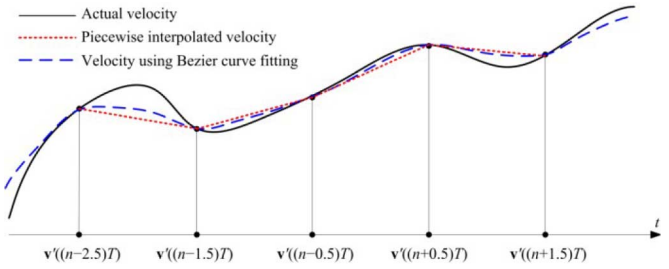


Fig. 6. Comparisons between the true velocity, the piecewise interpolated velocity, and the velocity obtained using curve fitting.

The exact relation between $\mathbf{g}(n)$ and $\mathbf{v}(t)$ is unknown. However, we observe empirically that $\mathbf{g}(n)$ of distorted image sequence is pretty close to the global motion vector of the normal image sequence. Therefore, from (1) we have

$$\mathbf{g}(n) \cong \int_{(n-0.5)T}^{(n+0.5)T} \mathbf{v}(t)\mathrm{d}t. \tag{12}$$

By setting $\tau = (n+0.5)T$ in (10), we get the initial estimate of the velocity of the middle scanline

$$\mathbf{v}'((n+0.5)T) = \mathbf{g}(n)/T. \tag{13}$$

B. Velocity Estimation

The global motion estimation generates a rough estimate of the velocity $\mathbf{v}'(t)$ at $t = (n+0.5)T$. However, the temporal resolution of the velocity required for the scanline is hundred times higher. For a VGA-size frame, for example, the velocity of each of the 640 scanlines of the frame has to be determined. Thus, an interpolation scheme is needed to meet the resolution requirement.

If piecewise linear interpolation is applied, discontinuities at frame instants would occur (Fig. 6) and result in *over compensation*. An illustration of such an annoying outcome that happens when a panning camera suddenly stops is shown in Fig. 7(d), where the Christmas tree, which is supposed to be upright, is tilted to the left. As a result, the Christmas tree in the compensated video sequence appears to be swinging left and right even though the camera never changes its moving direction when panning across the scene. Such overcompensation makes the video look worse than the rolling shutter effect itself.

Smooth parametric curves, specified by control points, are known for producing smooth representation of camera or object trajectory in film and computer graphics. We adopt this approach and choose Bézier curve to represent the velocity. A Bézier curve $\mathbf{q}(\lambda) = (q_x(\lambda), q_y(\lambda))$ is a 3-degree polynomial [11]. The x component of $\mathbf{q}(\lambda)$ is described by

$$q_x(\lambda) = (1-\lambda)^3 c_1 + 3\lambda(1-\lambda)^2 c_2 + 3\lambda^2(1-\lambda)c_3 + \lambda^3 c_4 \tag{14}$$

where $0 \le \lambda \le 1$ and c_1, c_2, c_3, and c_4 are the control points. The exact formula applies to the y component $q_y(i)$.

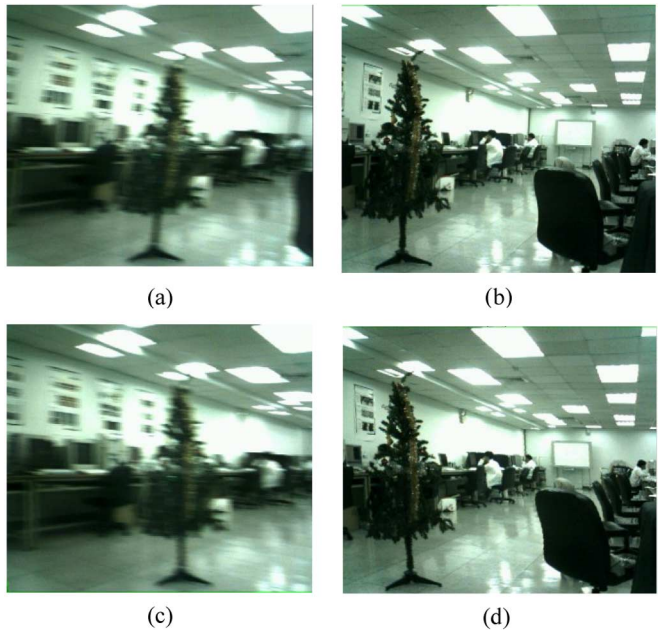


Fig. 7. Overcompensation. (a), (b) Two successive frames. (c), (d) Resulting frames compensated by using piecewise velocity interpolation. As a result of overcompensation, (d) the Christmas tree is tilted to the left.

The control points of the nth frame can be determined from four samples on the curve. We obtain four nearby velocity samples from (13) and solve for the control points by imposing the following conditions:

$$q_x(0) = v'_x((n-1.5)T) \tag{15}$$
$$q_x(1/3) = v'_x((n-0.5)T) \tag{16}$$
$$q_x(2/3) = v'_x((n+0.5)T) \tag{17}$$
$$q_x(1) = v'_x((n+1.5)T). \tag{18}$$

The same set of conditions is imposed on $q_y(i)$ except that v'_x is replaced by v'_y. Once the control points are obtained, the velocity of the ith scanline in the nth distorted image is computed by

$$\mathbf{v}'(nT+i) = \mathbf{q}\left(\frac{(i/T)+1.5}{3}\right). \tag{19}$$

This is an important equation since it can be applied to convert a sample on the Bézier curve to a velocity estimate and reversely a velocity estimate to a sample on the Bézier curve. The velocity estimate thus obtained is smooth and free of overcompensation.

C. Local Motion Refinement

Further improvement of the velocity estimate is needed because 1) not all blocks have true matchings in the global motion estimation due to limited field of view and 2) the Bézier curve is continuous and cannot handle the case where $\mathbf{v}(t)$ experiences a sudden change.

We adopt an iterative scheme in which the outliers are detected first and then local refinement of the velocity estimate

is performed. Consider a point $\mathbf{s}$ in the current frame. Its corresponding point in the previous frame must locate inside the frame. According to (10) and letting $\tau = nT + s_y$, the following two constraints can be established [13]:

$$0 \le s_x + \frac{v_x(nT+s_y)}{v_y(nT+s_y)+1}h \le w \tag{20}$$

$$0 \le s_y + \frac{v_y(nT+s_y)}{v_y(nT+s_y)+1}h \le h \tag{21}$$

where w and h, respectively, are the width and height of the video frame. When any of the constraints is violated, the true correspondence of $\mathbf{s}$ is outside the previous frame; thus, the velocity estimate $\mathbf{v}'(nT+s_y)$ is unreliable. In practice, because $\mathbf{v}(t)$ is unknown, we approximate (20) and (21) by using the estimated velocity $\mathbf{v}'(t)$ and adding the safety margins ε_x and ε_y

$$\varepsilon_x \le s_x + \frac{v'_x(nT+s_y)}{v'_y(nT+s_y)+1}h \le w - \varepsilon_x \tag{22}$$

$$\varepsilon_y \le s_y + \frac{v'_y(nT+s_y)}{v'_y(nT+s_y)+1}h \le h - \varepsilon_y. \tag{23}$$

These two constraints are used for outlier detection. If a point $\mathbf{s}_1$ and its velocity estimate satisfy (22) and (23), we perform local motion estimation to refine the velocity estimate, in a way similar to global motion estimation except that only a small region around $\mathbf{s}_1$ instead of the entire frame is considered. After the local motion $\mathbf{l}(\mathbf{s}_1,n) = (l_x(\mathbf{s}_1,n), l_y(\mathbf{s}_1,n))$ of $\mathbf{s}_1$ in the nth frame is found, the corresponding $\mathbf{s}_2$ in the previous frame is located using (2)

$$\mathbf{s}_2 = \mathbf{s}_1 - \mathbf{l}(\mathbf{s}_1,n) = \mathbf{s}_1 - \int_{(n-1)T+s_{2y}}^{nT+s_{1y}} \mathbf{v}(t)\mathrm{d}t. \tag{24}$$

Applying (24) and substituting $\tau = nT + s_{1y}$ into (10), we obtain the refined velocity of $\mathbf{s}_1$

$$\mathbf{v}'(nT+s_{1y}) = \frac{\mathbf{l}(\mathbf{s}_1,n)}{T-(s_{1y}-s_{2y})} = \frac{\mathbf{l}(\mathbf{s}_1,n)}{T-l_y(\mathbf{s}_1,n)}. \tag{25}$$

The local motion refinement and velocity estimation are performed iteratively. More precisely, if the residual in the local motion estimation is lower than a threshold, the velocity estimate obtained from (25) and converted through (19) is used to impose a new constraint on the control points, in addition to the four constraints (15)–(18), for new velocity estimation. Then, for each scanline, the closest four velocity samples are used to compute the control points in (14).

D. Scanline Realignment

Image correction is done by re-aligning the scanlines after the velocity estimate $\mathbf{v}'(t)$ is obtained. According to (8), the corresponding point $\mathbf{p}$ of $\mathbf{s}$ can be obtained by

$$\mathbf{p} = \mathbf{s} - \int_{(n+0.5)T}^{nT+s_y} \mathbf{v}'(t)\mathrm{d}t. \tag{26}$$

Because (26) does not depend on s_x, all pixels in the same scanline are moved by the same displacement. Since $\mathbf{v}'(t)$ is in floating point precision, $\mathbf{p}$ does not necessarily fall on the image sampling grid. In our algorithm, the intensity of $\mathbf{s}$ is assigned to the four samples nearest to $\mathbf{p}$, and each is weighed inversely proportional to its distance to $\mathbf{p}$. The intensity of the sample is normalized by the sum of the weights it receives. Finally, the missing pixels in the image are filled by averaging the nearby pixels. The scanline realignment algorithm described here is similar to the free-form deformation used in image metamorphosis [14].

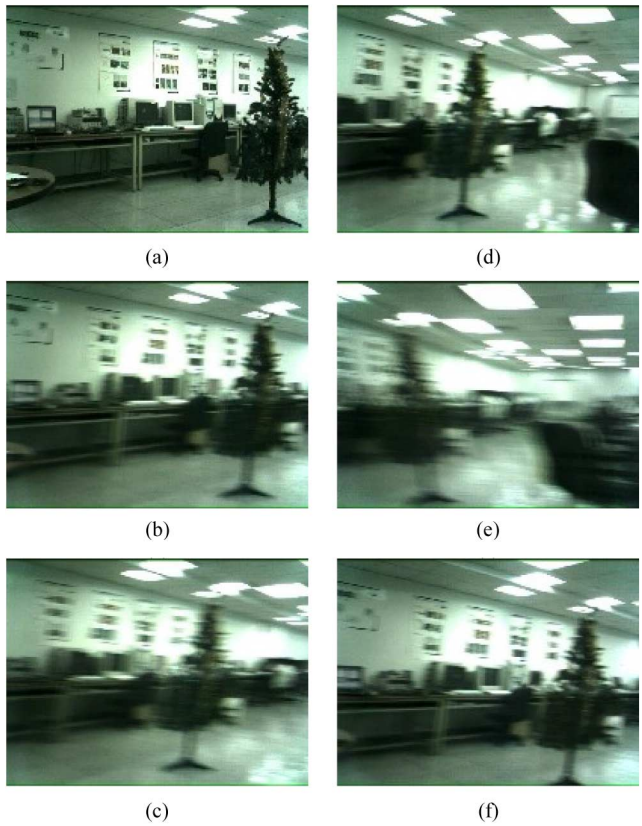


Fig. 8. Snapshots of the first test image sequence. From (a)–(f), the frame numbers are 1, 5, 6, 12, 16, and 38.

VI. Experimental Results

We capture the test image sequences using an OmniVision 1.3 MegaPixel CameraChip [15], which generates 640×480 video at 15 frames per second. There is no local frame buffer on the chip, so the captured data are streamed to a computer via USB 1.0 interface. To satisfy the condition of (11), the camera is fixed at the beginning of the test sequences. Two test sequences are shown in Figs. 8 and 9, where we can see that the objects are geometrically distorted due to the rolling shutter effect. The Christmas tree in Fig. 8 and the door in Fig. 9 are slanted, and the objects in Fig. 9 are vertically elongated. The images are seriously blurred due to fast camera motion; therefore, previous methods [2], [3] that require accurate correspondences are not applicable here even if the scene geometry is known.

The search range for global motion estimation is set to 160 pixels in both horizontal and vertical directions. For local motion refinement, the search range for each region is adaptively determined according to (22) and (23) and limited to 100 pixels.

(a) (b)

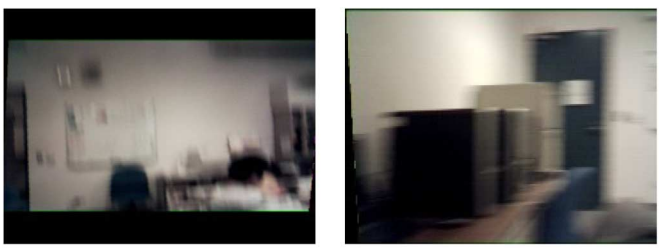
Fig. 9. Snapshots of the second test image sequence.

(a) (d)

(b) (e)

(c) (f)

Fig. 10. Corrected image sequence of Fig. 8.



(a) (b)

Fig. 11. Corrected image sequence of Fig. 9.

The number of iterations is set to 3. The output sequences are shown in Figs. 10 and 11, where the undefined pixels are left blank (black).

We can see that the geometric distortions in both sequences are properly compensated. The algorithm is able to restore the Christmas trees in Fig. 10 to the upright position even if the original image is so blurry. In Fig. 11, the size of the whiteboard on the wall is restored, and the edge of the door is made straight up. Note that the curved boundary of the black region in the corrected image is a result of the fact that the velocity estimation is performed per scanline. Finally, because of the adoption of Bézier curve interpolation, no overcompensation effect is found in the output.

VII. Discussion

Due to the use of a planar motion model, the algorithm is only applicable to scenes with moderate depth range. When the depth range is large, pixels of objects close to the camera have relatively larger displacements than pixels of farther objects. Therefore, moving all pixels on a scanline by the same displacement would not generate satisfactory results. For the same reason, the algorithm cannot handle scenes with multiple objects having different motions. These two problems can be remedied by decomposing the image sequence into layers using pixel- or segment-based motion analysis [16]. Each layer can then be processed separately by the algorithm, with user intervention if needed.

Another issue is related to the resolution of the velocity estimation. Our algorithm can successfully estimate large motion but may fail for small motion caused by hand shaking. This is because that the impulsive hand shake is of short duration so that only a few scanlines in the image are distorted. In such cases, the global motion estimation and the local motion refinement can hardly detect the hand shake. This remains an open issue for future research.

VIII. Conclusion

The relative motion between a CMOS camera and the scene causes a geometrical distortion known as the rolling shutter effect because CMOS image sensors transfer the pixel data on a line-by-line basis. This distortion becomes annoying if the read-out time per line is significant compared to the integration time. In this paper, we have described an image-domain analysis of the rolling shutter effect and developed a distortion correction algorithm to undo the rolling shutter effect. Instead of finding the exact correspondence between successive frames of the distorted image sequence, the algorithm uses global motion estimation, parametric curve fitting, and scanline realignment to compensate the distortion. The algorithm is simple and effective.

Acknowledgment

The authors would like to thank the anonymous reviewers for their constructive comments and suggestions.

References

[1] C. Geyer, M. Meingast, and S. Sastry, "Geometric models of rolling-shutter cameras," in *Proc. Omnidirectional Vision, Camera Networks and Non-classical Cameras*, Oct. 2005, pp. 12–19.

[2] O. Ait-Aider, N. Andreff, J. M. Lavest, and P. Martinet, "Exploiting rolling shutter distortions for simultaneous object pose and velocity computation using a single view," in *Proc. 4th IEEE Int. Conf. Computer Vision Systems*, Jan. 2006, pp. 35–41.

[3] O. Ait-Aider and A. Bartoli, "Kinematics from lines in a single rolling shutter image," in *Proc. Conf. Computer Vision and Pattern Recognition*, Minneapolis, MN, Jun. 2007, pp. 1–6.
[4] A. El Gamal and H. Eltoukhy, "CMOS image sensors," *IEEE Circuits and Devices Mag.*, vol. 21, no. 3, pp. 6–20, Mar. 2005.
[5] Z. Zhou, B. Pain, and E. R. Fossum, "Frame-transfer CMOS active pixel sensor with pixel binning," *IEEE Trans. Electron Devices*, vol. 44, no. 10, pp. 1764–1768, Oct. 1997.
[6] J. Nakamura, *Image Sensors and Signal Processing for Digital Still Cameras*. Boca Raton, FL: CRC, 2005.
[7] S. Lauxtermann, P. Schwider, P. Seitz, H. Bloss, J. Ernst, and H. Firla, "A high speed CMOS imager acquiring 5000 frames/sec,," in *Proc. Int. Electron Devices Meeting, Tech. Dig.*, Washington, DC, 1999, pp. 875–878.
[8] M. Wäny and G. Paul, "CMOS image sensor with NMOS-only global shutter and enhanced responsivity," *IEEE Trans. Electron Devices*, vol. 50, no. 1, pp. 57–62, Jan. 2003.
[9] B. Wilburn, N. Joshi, V. Vaish, E. Talvala, E. Antunez, A. Barth, A. Adams, M. Horowitz, and M. Levoy, "High performance imaging using large camera arrays," *ACM Trans. Graph*, vol. 24, no. 3, pp. 765–776, Jul. 2005.
[10] G. Sasaki and Y. Nara, "Image processor and camera system," U.S. Patent Application Publication, No. 2007/0120997 A1, May 31, 2007.
[11] R. H. Bartels, J. C. Beatty, and B. A. Barsky, *An Introduction to Splines for Use in Computer Graphics & Geometric Modeling*. San Francisco, CA: Morgan Kaufmann, 1987.
[12] H. H. Chen, C.-K. Liang, Y.-C. Peng, and H.-A. Chang, "Integration of digital stabilizer with video codec for digital video cameras," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 17, no. 7, pp. 801–813, Jul. 2007.
[13] L.-W. Chang, C.-K. Liang, and H. H. Chen, "Analysis and correction of rolling shutter distortion for CMOS image sensor arrays," in *Proc. Int. Symp. Communications*, Kaohsiung, Taiwan, R.O.C., Nov. 2005.
[14] S. Lee, K. Chwa, and S. Y. Shin, "Image metamorphosis using snakes and free-form deformations," in *Proc. 22nd Annu. Conf. Computer Graphics and interactive Techniques*, 1995, pp. 439–448.
[15] OmniVision Technologies, Inc. [Online]. Available: http://www.ovt.com
[16] J. Y. A. Wang and E. H. Adelson, "Representing moving images with layers," *IEEE Trans. Image Process.*, vol. 3, no. 5, pp. 625–638, Sep. 1994.
[17] C.-K. Liang, Y.-C. Peng, and H. H. Chen, "Rolling shutter distortion correction,," in *Proc. SPIE*, Beijing, China, Jul. 2005, vol. 5960, pp. 1315–1322.

Chia-Kai Liang (S'05) was born in Kaohsiung, Taiwan, R.O.C. He received the B.S. degree in electrical engineering from the National Taiwan University, Taipei, in 2004. He is currently pursuing the Ph.D. degree in the Graduate Institute of Communication Engineering, National Taiwan University.

His current research interests include image/video processing, video coding, and image-based rendering.

Li-Wen Chang was born in Kaohsiung, Taiwan, R.O.C. He received the B.S. degree in electrical engineering, with a minor in mathematics, from the National Taiwan University, Taipei, in June 2007.

He was an exchange student at University of Illinois at Urbana-Champaign, Urbana, in 2007.

Homer H. Chen (S'83-M'86-SM'01-F'03) received the Ph.D. degree in electrical and computer engineering from University of Illinois at Urbana-Champaign, Urbana.

Since August 2003, he has been with the College of Electrical Engineering and Computer Science, National Taiwan University, Taipei, Taiwan, R.O.C., where he is the Irving T. Ho Chair Professor. Prior to that, he held various R&D management and engineering positions with U.S. companies over a period of 17 years, including AT&T Bell Labs, Rockwell Science Center, iVast, and Digital Island. He was a U.S. delegate for ISO and ITU standards committees and contributed to the development of many new interactive multimedia technologies that are now part of the MPEG-4 and JPEG-2000 standards. His professional interests lie in the broad area of multimedia signal processing and communications.

Dr. Chen is an Associate Editor of the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY. He served as Associate Editor for the IEEE TRANSACTIONS ON IMAGE PROCESSING from 1992 to 1994, Guest Editor for IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY in 1999, and an Associate Editor for *Pattern Recognition* from 1989 to 1999.