Chapter 25
A Rectification Method for Removing Rolling Shutter Distortion

Gang Liu, Yufen Sun and Xianqiao Chen

Abstract We propose a rolling shutter video rectification method that can deal with both camera translation and rotation for videos obtained from unknown sources. As the exact distortions caused by rolling shutter are too complex, this method aims to remove the majority of the distortions. A 2D rotation model is used to approximately represent the motion of each frame. The parameters of this model are solved by minimizing the measurement constraints on point correspondences. To relax the restriction on the form of frame motion, the frame motion computation is performed sequentially. Experiments show that our method is comparable to the 3D models that require camera calibration.

Keywords CMOS image sensor · Rolling shutter distortion · Image rectification · Post-processing technique

25.1 Introduction

Nowadays, CMOS sensors are widely used in video cameras because of their low prices and the ability to integrate with image processing circuits. A CMOS camera commonly adopts rolling shutter to read and reset the rows of the sensor array sequentially, making the capture time of each row slightly after the capture time of the previous row. When camera or object moves during image capture, the scene imaged by each row presents relative motion, resulting in geometric distortion

G. Liu (✉) · Y. Sun · X. Chen
Intelligent Transportation System Research Center, Wuhan University of Technology, 430063 Wuhan, China
e-mail: liu_gang@whut.edu.cn

W. Lu et al. (eds.), *Proceedings of the 2012 International Conference on Information Technology and Software Engineering*, Lecture Notes in Electrical Engineering 210,
DOI: 10.1007/978-3-642-34528-9_25,

called rolling shutter distortion. These distortions degrade the quality of images and making objects in videos appear dynamic non-rigid deformation.

Removing rolling shutter distortion needs to compute the relative displacements between rows in a frame. This computational task is more difficult than video stabilization in which only the relative motions between frames are computed. Recently the problem of removing rolling shutter distortions from videos has been studied by many researchers [1–8]. These researches focused on the distortions caused only by camera motion. 2D image processing methods were proposed to process the distortions induced by smoothly varying translation parallel to the image plane [2, 3, 6, 7]. The more complex distortions caused by camera rotation was studied using 3D rotation models [4], with a requirement of camera calibration.

In this paper, we propose a rectification method that can handle both camera translation and camera rotation, without requiring any information about the camera. Since the general image transformation is very complex, we do not intent to estimate the motion of each pixel exactly, but only to compute approximate frame motions that can be used to remove the dominating distortion caused by rolling shutter. A 2D rotation model is used to represent the transformation between two successive frames. The parameters of the model are solved by minimizing the measurement constraints on point correspondences. Based on the rotation model, the distortion caused by rolling shutter can be revealed. Experiments show that our method is comparable to the 3D rotation model [4] that beat other 2D methods in most situations.

We introduce the related work in Sect. 25.2. Section 25.3 gives our 2D rotation model and analyzes why the model works for our task. Our rectification method is described in section four. Section five analyzes the experimental results. At last, section six concludes the paper.

25.2 Related Work

Mechanical methods have been designed to avoid rolling shutter distortion by using a mechanical global shutter or adopting a mechanical image stabilization system. However, the mechanical global shutter will reduce the fill factor of image sensors, and the mechanical image stabilization system is not equipped in most consumer cameras. Post-processing methods are thus required to rectify the distorted videos.

The most important thing in rectifying rolling shutter distortions is to find the relative pixel displacement during the image exposure interval. This displacement is equal to the temporal integral of the image motion between the time instants at which the pixels are imaged. If a camera undergoes a constant translation parallel to the image plane, the constant motion velocity of the image can be directly computed using the global motion between two successive frames [2, 3, 7]. Liang et al. proposed a method that can deal with smoothly varying translation parallel to the image plane [6]. The high-resolution motion velocity of frames is computed using global motion estimation and parametric curve fitting. These methods suppose the points on the same row have the same motion, and cannot be used to rectify distortion induced by camera rotation.

Forssén and Ringaby are the first to compute 3D camera motion for rolling shutter rectification [4, 8]. As the motion of image is the result of camera motion through perspective projection, their method has a clear physical meaning. They used spherical linear interpolation to interpolate the camera rotation from the knots represented by rotation parameters, and then solved the parameters using inter-frame correspondences. This method can effectively recover the rolling shutter images distorted by camera rotation. But it needs camera calibration, which is not convenient in some situations. To facilitate controlled evaluation of different algorithms, Forssén and Ringaby generated synthetic rolling shutter sequences that can be downloaded from [9].

When a camera is attached to a moving vehicle, the camera may vibrate at a higher frequency than the frame rate of the camera. Baker et al. studied this problem using translational model and affine model [1]. The camera jitter was solved by optimizing the measurement constraints of the correspondences sub-sampled from the optical flow fields.

Cho and Hong used affine matrix to represent the transformation between two distorted frames [2]. Their derivation was based on uniform frame translation.

Experiments showed that the video stabilization methods proposed by Liu et al. could reduce rolling shutter distortions [10, 11]. Though the theoretical explanation for why this happened was not clear.

25.3 An Image Motion Model

We first analyze the relationship between rolling shutter distortions and image motion. Then a 2D rotation model is introduced.

25.3.1 Rolling Shutter Distortions

To rectify a distorted rolling shutter frame, the middle row of the frame is normally chosen as the reference. As show in Fig. 25.1, the aim of the rectification is to obtain a frame as if it is imaged by a global shutter camera at the time instant when the middle row of the frame is imaged. We define the time interval between the sampling instants of two successive rows as the unit time. The displacement of a pixel p_i during the time interval between the instant r_i it is imaged and the reference instant r_m is:

$$d_i = \int_{r_i}^{r_m} v(t)dt, \tag{25.1}$$

where $v(t)$ represents the motion velocity of the pixel. If the inter-frame delay of the camera is zero, the time interval $r_m - r_i$ is at most half of the frame period.

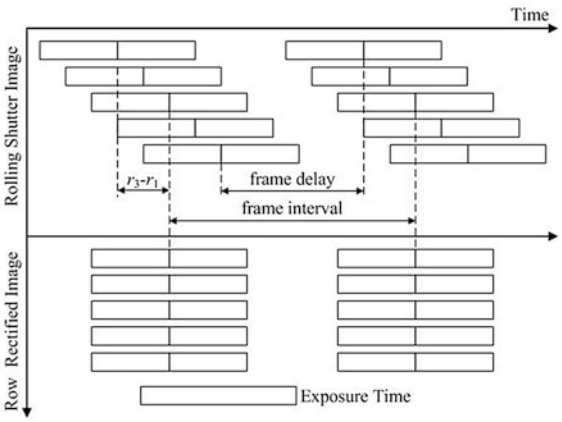

Fig. 25.1 The image rectification model

A nonzero value of inter-frame delay will decrease this rate further. So if the displacement of pixel p_i during the time interval between frames is small, the distortion d_i will be smaller, at most half of the former. Thus we can only consider large displacements of pixels to remove obvious distortions.

25.3.2 A 2D Rotation Model

For two successive frames captured by a global shutter camera, the major motion of each pixel is rotation around a point in the image plane or translation along a direction. If a frame rotates around a point **c** in the image plane through an angle θ, pixel **x** in this frame will move to a new position $\mathbf{x}'$, which is:

$$\mathbf{x}' = f(\mathbf{x}, \theta, c_x, c_y) = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \left(\begin{bmatrix} x \\ y \end{bmatrix} - \begin{bmatrix} c_x \\ c_y \end{bmatrix} \right) + \begin{bmatrix} c_x \\ c_y \end{bmatrix}. \tag{25.2}$$

This equation models in-plane rotation. If the motion is translation, the pixels can be thought as rotating around a far away point located on the direction that is perpendicular to the motion direction. Then Eq. (25.2) can be used to approximately model translation if we choose an appropriate rotation center.

25.4 Image Rectification

Suppose we have point correspondences **x** and $\mathbf{x}'$ in two successive frames, then:

$$\mathbf{x}' = \mathbf{x} + \int_{r}^{N+r'} v(t, x)dt \tag{25.3}$$

where r and r' are the row number of pixel $\mathbf{x}$ and $\mathbf{x}'$, N is the number of rows in a frame. In Eq. (25.3), we suppose the inter-frame delay is zero. If we have a set of data correspondences, the measurement constraint of the correspondences is:

$$E = \sum_{k=1}^{N} d(\mathbf{x}_k - \mathbf{o}_k)^2 + d(\mathbf{x}'_k - \mathbf{o}'_k)^2 \tag{25.4}$$

where $\mathbf{x}_k$ and $\mathbf{x}'_k$ are point correspondence, $\mathbf{o}_k$ and $\mathbf{o}'_k$ are relocated points. In Eq. (25.4), $\mathbf{o}'_k$ is computed by $\mathbf{o}_k$ and $v(t)$ using Eq. (25.3), $\mathbf{o}_k$ and $v(t)$ are parameters needed to be determined by minimizing the measurement constraint. If we don't add any constraint on $v(t)$, the parameters are hard to be solved even when we have enough correspondences.

25.4.1 Motion Computation

To compute the frame motion for image rectification, we first use a strong assumption to initialize the computation, and then use a weak assumption to finish the computation.

25.4.1.1 Initialization

If a frame uniformly rotates around a fixed center (c_x, c_y) during the capture interval of two successive frames, the rotation velocity v_r is:

$$v_r = \theta / N, \tag{25.5}$$

where θ represents the rotation angle of the frame in one frame interval. Then Eq. (25.2) can be rewritten as:

$$\mathbf{x}' = f(\mathbf{x}, (N + r' - r)v_r, c_x, c_y). \tag{25.6}$$

Using Eq. (25.6), θ, c_x and c_y can be easily solved by minimizing the measurement constraint. We use the sparse Levenberg-Marquardt algorithm given by Hartley and Zisserman [12] to solve this optimization problem.

Though the assumption of uniform rotation in two frame intervals seems strong, it can be satisfied by some frames in a video captured by a camera undergoing smoothly varying motion. To find such frames, we compute the measurement constraint for each pair of successive frames. The two frames giving the minimal value of E are considered to satisfy the assumption. They are chosen as the initial frames with their motion solved.

25.4.1.2 Sequential Computation

Now we can relax our assumption by assuming a frame uniformly rotates around a fixed center in only one frame interval. Based on the motion of the initial frames, we compute the motion of other frames in two directions, which is illustrated in Fig. 25.2.

For the frames behind the initial frame k, we first transform the frame k to the time instant when the last row was imaged:

$$\mathbf{x}^{new} = f(\mathbf{x}, ((N-r)/N)\theta(k), c_x(k), c_y(k)) \tag{25.7}$$

Then the distances between the points in correspondences are only caused by the motion of frame $k+1$. The relationship between the corresponding points $\mathbf{x}^{new}$ and $\mathbf{x}'$ is:

$$\mathbf{x}' = f(\mathbf{x}^{new}, (r'/N)\theta(k+1), c_x(k+1), c_y(k+1)) \tag{25.8}$$

Now we can get $\theta(k+1)$, $c_x(k+1)$ and $cy(k+1)$ by minimizing the equation (25.4).

To find the rotation of frames captured before frame k, we first rotate frame k to the instant before the first row was imaged:

$$\mathbf{x}^{new} = f(\mathbf{x}, (-r'/N)\theta(k), c_x(k), c_y(k)) \tag{25.9}$$

The relationship between $\mathbf{x}^{new}$ and the corresponding point $\mathbf{x}'$ in frame $k-1$ is:

$$\mathbf{x}' = f(\mathbf{x}^{new}, (-(N-r')/N)\theta(k-1), c_x(k-1), c_y(k-1)) \tag{25.10}$$

This process is going on until every frame motion is solved.

25.4.1.3 Finial Solution

When we compute the motion of frame $k+1$ or $k-1$ based on the motion of frame k, the error of the latter may decrease the quality of the computation. To handle this problem, for each frame we compare the solution computed in the initialization step with the solution acquired in the sequential computation step. The one with smaller value of E is chosen as the finial solution.

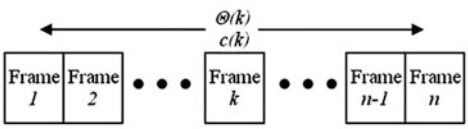

Fig. 25.2 Sequential computation of frame motion. The frame k is the initial frame with known motion

25.4.2 Frame Rectification

To remove the rolling shutter distortion in frame k, the corrected position of pixel $\mathbf{x}$ is:

$$\mathbf{x}^{cor} = f(\mathbf{x}, ((N/2 - r)/N)\theta(k), c_x(k), c_y(k)) \tag{25.11}$$

Because $\mathbf{x}^{cor}$ may not fall on the image sampling grid, we perform cubic interpolation to acquire the recovered image. The function griddata in Matlab is used to do this work.

25.5 Experimental Analysis

In our experiments, point correspondences are computed by the optical flow algorithm proposed by Sun et al. [13]. The flow fields over the whole frame are sampled uniformly to obtain the point correspondences. Thus the acquired correspondences cover the whole frame.

25.5.1 Accuracy Comparison

We use the synthetic dataset provided by Forssén and Ringaby [9] to compare our method with their 3D models [4]. These 3D models beat previous 2D rolling shutter rectification methods on this dataset.

We use the thresholded Euclidean colour distance to compute the rectification accuracy as in [4]. The threshold is also set to 0.3. Figure 25.3 shows the results on

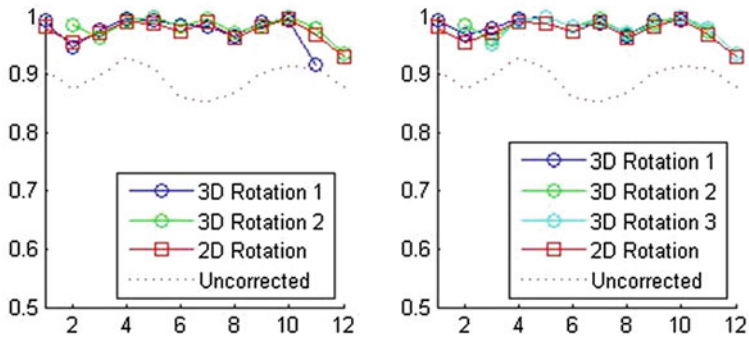

Fig. 25.3 Sequence rot1_B0. *Left* the 3D rotation model with 2-frame reconstruction window, our 2D rotation model, and uncorrected frames. *Right* the 3D rotation model with 3-frame reconstruction window, our 2D rotation model, and uncorrected frames

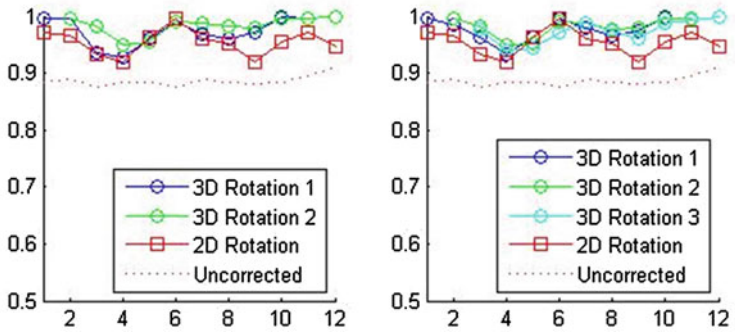

Fig. 25.4 Sequence rot2_B40. *Left* the 3D rotation model with 2-frame reconstruction window, our 2D rotation model, and uncorrected frames. *Right* the 3D rotation model with 3-frame reconstruction window, our 2D rotation model, and uncorrected frames

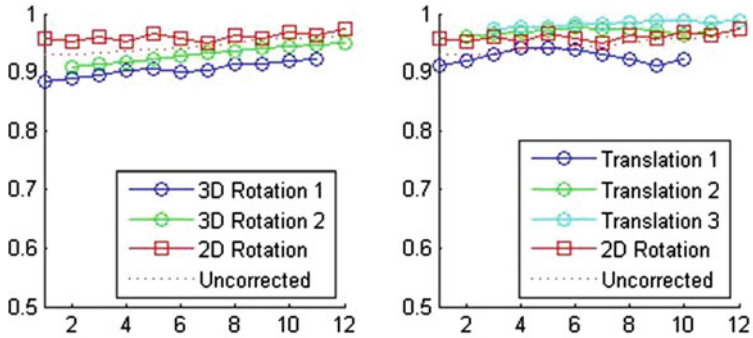

Fig. 25.5 Sequence trans1_B40. *Left* the 3D rotation model with 2-frame reconstruction window, our 2D rotation model, and uncorrected frames. *Right* the 3D translation model with 3-frame reconstruction windows, our 2D rotation model, and uncorrected frames

sequence rot1_B0 in the dataset. In this sequence, the camera rotates in a spiral fashion. Our method gets nearly the same result as the 3D rotation model. Figure 25.4 gives the results on sequence rot2_B40. This sequence simulates a complex camera rotation in 3D space. The performance of our 2D rotation model is a little worse than the 3D rotation model. In Fig. 25.5, we compare the 3D rotation model and the 3D translation model with our 2D model on sequence trans_B40, in which the camera makes a pure translation. Our model is better than the 3D rotation model, and is comparable to the 3D translation model. In Fig. 25.6, we compare the 3D rotation model and the 3D full model with our 2D model on sequence trans_rot1_B40. This sequence is generated by adding a translation to the sequence rot2_B40. The accuracy of our 2D model is similar to that of the 3D rotation model, and is much better than the 3D full model.

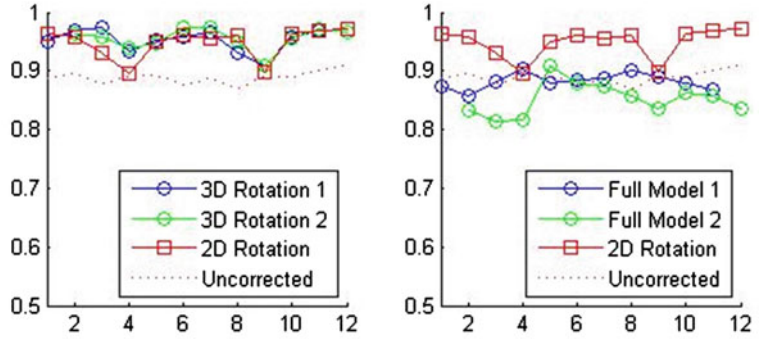

Fig. 25.6 Sequence trans_rot1_B40. *Left* the 3D rotation model with 2-frame reconstruction window, our 2D rotation model, and uncorrected frames. *Right* the 3D full model with 2-frame reconstruction window, our 2D rotation model, and uncorrected frames

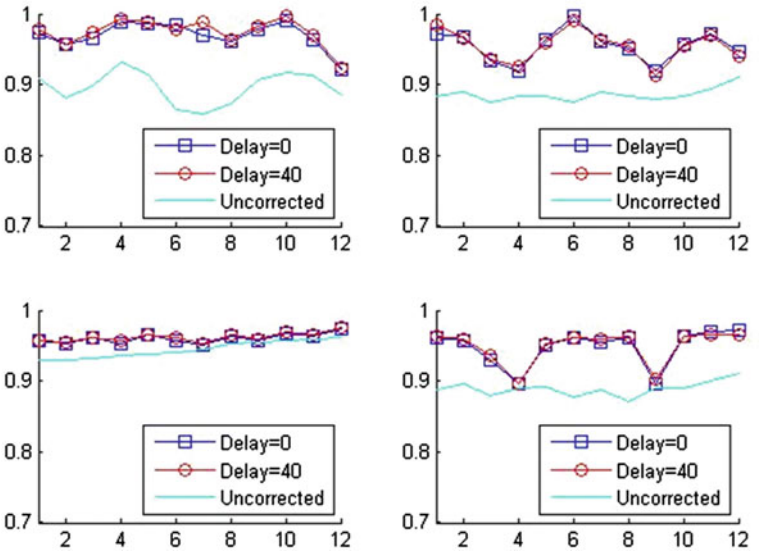

Fig. 25.7 Accuracy comparison with inter-frame delay = 0 vs. inter-frame delay = 40row. *Top left* Comparison on sequence rot1_B40. *Top right* Comparison on sequence rot2_B40. *Bottom left* Comparison on sequence trans1_B40. *Bottom right* Comparison on sequence trans_rot1_B40

From above figures we can see that our 2D model works well for both rotation and translation. Even for complex camera rotation, it is comparable with the 3D method that requires camera calibration.

25.5.2 The Inter-Frame Delay

In our 2D rotation model, we simply assume the inter-frame delay is zero. In this section, we show that our method is not sensitive to the value of inter-frame delay. Figure 25.7 illustrates the accuracy comparison between the results with inter-frame delay set to zero and the results with inter-frame delay set to the true value. It can be seen that the results are nearly the same. Thus our method does not need to compute the value of the inter-frame delay.

25.6 Conclusion

In this paper, we propose a 2D rectification method for removing rolling shutter distortion caused by camera motion. This method approximately models the frame motion as 2D rotation. The parameters of the model are solved by minimizing the measurement constraint on point correspondences. Experiments show the method can effectively remove distortions caused by camera rotation and translation. Without requiring any information of cameras, the method can be used to process videos obtained from unknown sources.

Acknowledgments This work was supported by the National Natural Science Foundation of China (51179146) and the Fundamental Research Funds for the Central Universities (2012-IV-041).

References

1. Baker S, Bennett E, Kang S, Szeliski R (2010) Removing rolling shutter wobble. In: IEEE conference on computer vision and pattern recognition, pp 2392–2399
2. Cho WH, Hong KS (2007) Affine motion based CMOS distortion analysis and CMOS digital image stabilization. IEEE Trans Consum Electron 54:833–841
3. Chun JB, Jung H, Kyung CM (2008) Suppressing rolling-shutter distortion of CMOS image sensors by motion vector detection. IEEE Trans Consum Electron 54:1479–1487
4. Forssén PE, Ringaby E (2010) Rectifying rolling shutter video from hand-held devices. In: IEEE conference on computer vision and pattern recognition, pp 507–514
5. Geyer C, Meingast M, Sastry S (2005) Geometric models for rolling-shutter cameras. In: International workshop on omni vision
6. Liang CK, Chang LW, Chen H (2008) Analysis and compensation of rolling shutter effect. IEEE TIP 17:1323–1330
7. Nicklin SP, Fisher RD, Middleton RH (2007) Rolling shutter image compensation. RoboCup 2006: Robot Soccer World Cup X 4434: 402–409
8. Ringaby E, Forssén PE (2012) Efficient video rectification and stabilisation for cell-phones. IJCV 96:335–352
9. Ringaby E Rolling shutter dataset with ground truth. (http://www.cvl.isy.liu.se/research/rs-dataset)
10. Liu F, Gleicher M, Jin H, Agarwala A (2009) Content-preserving warps for 3d video stabilization. In: ACM SIGGRAPH, p 44

11. Liu F, Gleicher M, Wang J, Jin H, Agarwala A (2011) Subspace video stabilization. ACM Trans Graph 30:1–10
12. Hartley R, Zisserman A (2011) Multiple view geometry in computer vision. Cambridge University Press, Cambridge
13. Sun D, Roth S, Black MJ (2010) Secrets of optical flow estimation and their principles. In: IEEE conference on computer vision and pattern recognition, pp 2432–2439