

A Robust Method for Strong Rolling Shutter Effects Correction Using Lines with Automatic Feature Selection

Yizhen Lao Omar Ait-Aider
Institut Pascal, Université Clermont Auvergne / CNRS
4 Avenue Blaise Pascal, 63178 Aubière Cedex, FRANCE
lyz91822@gmail.com omar.ait-aider@uca.fr

Abstract

We present a robust method which compensates RS distortions in a single image using a set of image curves, basing on the knowledge that they correspond to 3D straight lines. Unlike in existing work, no a priori knowledge about the line directions (e.g. Manhattan World assumption) is required. We first formulate a parametric equation for the projection of a 3D straight line viewed by a moving rolling shutter camera under a uniform motion model. Then we propose a method which efficiently estimates ego angular velocity separately from pose parameters, using at least 4 image curves. Moreover, we propose for the first time a RANSAC-like strategy to select image curves which really correspond to 3D straight lines and reject those corresponding to actual curves in 3D world. A comparative experimental study with both synthetic and real data from famous benchmarks shows that the proposed method outperforms all the existing techniques from the state-of-the-art.

1. Introduction

Rolling shutter cameras are widely-used for their low-cost and low consumption compared to global shutter (GS) ones. However, in RS acquisition mode, pixels are exposed sequentially row by row from the top to the bottom. Therefore, images captured by moving RS cameras will occur distortions (e.g. Wobble, Skew) called RS effects (shown in Fig. 1(a)). The aim from RS correction is to obtain images which not only are visually acceptable but also can serve as input data for computer vision applications such as image alignment or SfM.

1.1. Related Works and Motivation

The state-of-the-art works for RS image correction can be divided into three main classes:

Video-based methods: Methods of [11, 3, 10, 4, 20] try to

(a) RS distorted image (b) Detection by [13] (c) Correction by [13]

(d) Initial detection by our method (e) Result of automatic feature selection (f) Correction by our method

Figure 1: (a) An example of a distorted RS image. (b) Arc segments detected by [13] using LSD detector [18] where outliers are also considered in correction. In contrast, the automatic feature selection in our method successfully filters outliers among detected candidate curves (d) and obtains correctly fitted curves (e). Final corrections by [13] and our method are shown on (c) and (f).

recover the geometry between RS frames first, then to compensate RS effects by scanline realignment or RS-aware warping. These methods all require pairwise or more point correspondences obtained thanks to feature tracking and matching, which may be data and/or time consuming.

Gyroscopes-based methods: Methods of [7, 9, 12] utilize gyroscopes to measure camera ego-motion during acquisition and compensate RS effects directly. It is obvious that the strong dependency on external sensors is a strong limitation on the practical use of these methods.

Single-image-based methods: Most of the methods in this category are based on line features because straight lines are generally abundant in man made environments such as buildings interiors or urban cityspaces. Moreover, they are both easy to detect and robust to partial occlusion. Fi-

nally, straight lines offers a convenient way to introduce constraints on scene geometry. After line features had been explored in object motion estimation [1] and visual navigation [19], 3D straight lines have been also used for solving single RS image correction problem based on straightness [15] or vanishing direction [13] constraints. Rengaranjan et al. first developed a learning-based single RS correction method using CNN in [14]. Nevertheless, we point out that existing single-image-based methods suffer from following disadvantages:

(i) Methods of [15, 14] correct RS by taking human visually pleasant as standards without considering correctness of projection geometry. Thus, camera x and y axis ego-rotation are neglected during ego-motions estimation which may lead to geometrical inconsistencies.

(ii) Methods of [15, 13] require that the MW assumption is valid. This requires that the images feature at least two orthogonal vanishing directions. Beside the difficulty to find such image features, nonorthogonal 3D lines and 3D curves are also common in urban area. Since both them lack of outlier filter process, strong deformations occurs in the final correction (Fig. 1(b)(c)).

(iii) Nonlinear iterative solutions are used in [15, 13] which are time-consuming and suffer from the risk of local minima due to the absence of any specific initial guess estimation process.

In order to overcome disadvantages of the techniques from the state-of-the art, we present a method which enables us to compensate RS distortions using a set of image curves which correspond to 3D straight lines with free unknown directions. The method estimates linearly the camera ego-motion and then compensates image distortions according to the computed rotation. The presence of outlier curves which do not correspond to actual 3D straight lines is also addressed (Fig. 1(e)(f)).

1.2. Contributions and Paper Organization

After introducing the perspective projection model for RS cameras, we show that the parameterization of the projection of a 3D straight line leads to a first, second or third degree polynomial depending on the kinematic model considered during image acquisition. Then we show how to linearly extract ego angular-velocity basing on a rotational kinematic model with at least 4 curves. Finally, the image is corrected by compensating motion effects. The proposed method is integrated within a RANSAC-like framework which enables us to discard outlier curves which do not correspond to 3D straight lines from one hand, and to maximize the number of inlier curves which really correspond to 3D straight lines from the other hand. This step is crucial because it makes the method robust to noise and also fully automated. Our approach contributions can be summarized as follows:

- The first Linear solution for the rotational ego-motion estimation without pre-knowledge about directions of 3D straight lines or angles between these lines;
- The design of a practical automatic feature selection strategy to pick image curves which really correspond to 3D straight lines instead of 3D curves.

The rest of this paper organized as: Section. 2 presents RS projection model and details the parameterization of the projection of a 3D line into a RS camera under different kinematic models. How to linearly extract camera rotational velocity is explained in section. 3. The automatic feature selection algorithm is presented in section. 4. Section. 5, is dedicated to the evaluation of the proposed methods using both synthetic and real data sets.

2. Straight Line Projection with RS

2.1. RS Camera Model

In the static case, a RS camera is equivalent to a GS one. It follows the classical pinhole camera projection model defined by intrinsic parameters matrix K , rotation R and translation T between world and camera coordinate systems [5]:

$$s[m_i^{GS}, 1] = K[R \ T][P_i, 1] \quad (1)$$

where s indicates a scale factor, $P_i = [X, Y, Z]$ is a 3D point in the world coordinate system and $m_i^{GS} = [u_i, v_i]$ is its projection on the frame.

For a moving RS camera, during frame exposure, each row will be captured at a different pose. Therefore, for general camera motion model (with both angular and linear velocities), Eq. (1) becomes:

$$s[m_i^{RS}, 1] = K[R \ R_i \ T + \ T_i][P_i, 1] \quad (2)$$

where R_i and T_i are the rotation and the translation from the first row to the i -th row. Usually frame readout time for consumer cameras is short enough to make the assumption that the camera is under uniform motion during acquisition. Therefore, rotation and translation can be formulated with this assumption based on small rotate approximation of Rodrigues and linear velocity formulas:

$$R_i = I + v_i [\]_{\times} \quad T_i = v_i d \quad (3)$$

where I is the 3×3 identity matrix, d is the linear velocity while $[\]_{\times}$ indicates the antisymmetric matrix associated to angular velocity $\omega = [\omega_1, \omega_2, \omega_3]$.

2.2. 3D Line Representation

In this paper we adopt the convenient formulation used in [17] and which represents a 3D straight line in R^3 as a tuple $L = \langle R, (a, b) \rangle$ with 4 degrees of freedom (DoF) as illustrated in Fig. 2.

Table 1: Parametric representation of 3D straight line projection with different RS models

Camera model	Projection equation	Curve type	Parameters
GS camera	${}^{GS}F_1u + {}^{GS}F_2v + {}^{GS}F_3 = 0$	Straight line	R, t
Linear RS camera	${}^{Lin}F_1v^2 + {}^{Lin}F_2vu + {}^{Lin}F_3v + {}^{Lin}F_4u + {}^{Lin}F_5 = 0$	Hyperbolic curve	R, t, d
Rotate-only RS camera	${}^{Rot}F_1v^2 + {}^{Rot}F_2vu + {}^{Rot}F_3v + {}^{Rot}F_4u + {}^{Rot}F_5 = 0$	Hyperbolic curve	$R, t,$
Uniform RS camera	${}^{Unif}F_1v^3 + {}^{Unif}F_2v^2u + {}^{Unif}F_3v^2 + {}^{Unif}F_4vu + {}^{Unif}F_5v + {}^{Unif}F_6u + {}^{Unif}F_7 = 0$	Cubic curve	$R, t, d,$

Figure 2: **3D line representation.** The line can be treated as a line parallel to Z-axis passing through point $(a, b, 0)$ within XY -Plane (green line shown in the left figure) which is then rotated by R to a new position (Shown on the right part). Thus, the final straight line passes through point $R(ax + by)$, and is heading Rz .

2.3. 3D Line Projection with GS Camera

Assuming a calibrated camera, intrinsic matrix K is known. Schindler et al. prove that the projection of a 3D line into a GS camera image can be divided into three main steps [17]:

Transformation into camera coordinate frame. We denote a 3D line in the world coordinate system as $\langle R_w, (a_w, b_w) \rangle$ and the transformation between the camera coordinate frame and the world frame as R_c^w and t_c^w . The 3D straight line can be expressed in the camera coordinate system as:

$$R_c = R_c^w R_w \quad t_c = (t_x, t_y, t_z) = (R_w) \quad t_c^w \quad (4)$$

$$(a_c, b_c) = (a_w - t_x, b_w - t_y)$$

Perspective projection. The direction $m_{cip} = [m_x, m_y, m_z]$ of a straight line on the image denoted as $m_xu + m_yv + m_z = 0$ within a plane at $z = 1$ in the camera frame can be calculated by the cross product of Rz and $R_c(a_cx + b_cy)$:

$$m_{cip} = a_c R_{c2} - b_c R_{c1} \quad (5)$$

Where R_{c2} and R_{c1} are the second and first columns of R_c .

Image space projection. Image lines can be obtained as: $m_{ci} = K^{-1} m_{cip}$. Finally, we can write a projected 2D line in image as follows:

$${}^{GS}F_1u + {}^{GS}F_2v + {}^{GS}F_3 = 0 \quad (6)$$

2.4. 3D Line Projection with Uniform RS Model

Under the more realistic assumption of a uniform motion with both angular and translational velocities, the camera pose for v -th row can be denoted by Eq. (3) as:

$$R_c = ((I + [\quad]_x v)) R_w^c \quad R_w$$

$$t_c = (t_x, t_y, t_z) = (R_w) \quad t_c^w + dv \quad (7)$$

Finally we obtain a cubic curve:

$${}^{Unif}F_1v^3 + {}^{Unif}F_2v^2u + {}^{Unif}F_3v^2 + {}^{Unif}F_4vu + {}^{Unif}F_5v + {}^{Unif}F_6u + {}^{Unif}F_7 = 0 \quad (8)$$

Seven coefficients are determined by K , 3D line parameters, camera pose and kinematic parameters (d, \quad) .

From the uniform model in Eq. (8), one can derive two simpler models: linear RS model and rotate-only model which assumes pure translation and pure rotation during image acquisition. By either forcing the linear velocity d or the angular velocity \quad to be equal to 0 respectively, one can obtain a hyperbolic curve. The parameterizations of 3D line projection with different RS models are summarized in table. 1.

3. Ego-Motion from 2D Curves

3.1. Comparison of the Three RS Models

Some existing works argued that only angular-velocity plays a main role for hand-held and vehicle devices [16, 2, 15, 13, 8]. Here, we give a further quantitative analysis of both rotational and translational ego-motion effects on 3D line projection. Although the linear RS model will introduce a hyperbolic curve, however, its second order coefficients ${}^{Lin}F_1 = K_{22}^- (R_{w21}R_{w2} - R_{w22}R_{w1})d$, ${}^{Lin}F_2 = K_{11}^- (R_{w11}R_{w2} - R_{w12}R_{w1})d$ are provable much smaller compared to ${}^{Lin}F_3 = K_{22}^- (a_w R_{w22} - b_w R_{w21}) +$

$\frac{K_{31}^-}{K_{11}^-} \text{Lin} F_2 + \frac{K_{32}^-}{K_{22}^-} \text{Lin} F_1 + (R_{w31}R_{w2} - R_{w32}R_{w1})d$ and $\text{Lin} F_4 = K_{11}^- (a_w R_{w12} - b_w R_{w11})$ and can be ignored in practice. The simulated data experiment shown in Fig. 3 confirmed that even with a high linear speed, $\text{Lin} F_1$, $\text{Lin} F_2$ are relatively low, and projected curves (blue) are close to straight lines as for GS case (green). In practice, the effect of translational speed can be compensated by an increment on the rotational speed. Therefore, we chose to extract the angular velocity basing on the rotate-only RS model instead of the uniform model. This assumption holds because the translation during frame exposure is negligible in comparison with to the depth of the features to be used. Doing so, it becomes possible to compensate rolling shutter effects of the hole image independently from the depth associated to each pixel. This is the key of the single-view based RS correction.

3.2. Linear 4-Curves Algorithm

Now, we introduce a 4-curves linear solution called R4C to estimate ego-rotation basing on the RS rotate-only model. Denoting the 3D line structural parameters as $a_c R_{w2} - b_c R_{w1} = [s_1, s_2, s_3]$, for five hyperbolic coefficients of each curve, we can formulate a group of equations:

$$\begin{aligned} F_1 &= K_{22}^- (s_1^3 - s_3) \\ F_2 &= K_{11}^- (s_3^2 - s_2) \\ F_3 &= K_{22}^- s_2 + F_2 \frac{K_{31}^-}{K_{11}^-} + F_1 \frac{K_{32}^-}{K_{22}^-} + (s_2^2 - s_1) \\ F_4 &= K_{11}^- s_1 \\ F_5 &= K_{11}^- s_1 + K_{32}^- s_2 + s_3 \end{aligned} \quad (9)$$

where s_1 , s_2 and s_3 are different for each curve.

From Eq. (9), s_1 , s_2 , s_3 and s_4 can be substituted by s_1 and s_2 (more details are given in the supplemental materials). We can obtain a bivariate cubic polynomial. With new coefficients C_1 to C_8 which are only determined by matrix K and coefficients F_1 to F_5 . Now, by giving four curves, we have:

$$\begin{bmatrix} C_1^1 & \cdots & C_8^1 \\ \vdots & \ddots & \vdots \\ C_1^4 & \cdots & C_8^4 \end{bmatrix} \begin{bmatrix} s_1^3 & s_2^2 & s_1 & s_2^2 & s_1^2 & s_1 & s_2 & 1 \end{bmatrix} = 0 \quad (10)$$

By eliminating s_1^3 and s_2^2 , Eq. (10) becomes a two bi-variables quadratic polynomial equations:

$$\begin{bmatrix} T_1^1 & \cdots & T_6^1 \\ T_1^2 & \cdots & T_6^2 \end{bmatrix} \begin{bmatrix} s_1^2 & s_2^2 & s_1 & s_2 & s_1 & s_2 & 1 \end{bmatrix} = 0 \quad (11)$$

Where coefficients T are calculated by coefficients C in Eq-10. Again, we further substitute s_2 by s_1 and the 10 coefficients T in Eq. (11), then we obtain:

(a) (b) (c)

Figure 3: Projections of a 3D straight lines with different RS camera kinematics. (a) A simulated 3D straight line projected onto a 2D image as different forms of curves with no ego-motion (green), translation-only (blue), rotate-only (pink) and uniform ego-motion (yellow). Assuming the depth from the 3D straight line to camera as 1 unit length. Blue curves in (b) are the projection of a 3D line into a linear RS camera with linear velocities from 0.5 to 2.5 unit/s, while green line is for the GS case. The variations of $\text{Lin} F_1$, $\text{Lin} F_2$, $\text{Lin} F_3$ and $\text{Lin} F_4$ and constant value of $\text{GS} F_3$ are shown in (c).

$$(H_1, H_2, H_3, H_4, H_5) \begin{pmatrix} s_1^4 & s_1^3 & s_1^2 & s_1 & 1 \end{pmatrix} = 0 \quad (12)$$

Thus, Eq-11 turns into a bi-quadratic polynomial equation with one unknown (s_1). If more than 4 curves are available, parameter s_1 can be recovered by solving the non homogeneous linear system Eq. (12), after which s_2 will be recovered using Eq-10. Then we calculate s_3 based on Eq-9. Alternatively, one can solve the quartic and then pick one of the four solutions as explained in the supplemental material.

Finally, we compensate the effects of s_1 in order to correct RS image by performing a forward mapping (eliminating P with Eq-1 and Eq-2) to all pixels as:

$$m^{\text{GS}} = KR(v)^{-1} K^{-1} m^{\text{RS}} \quad (13)$$

where, the procedure will map original points m^{RS} to m^{GS} on global frame. $R(v)$ is calculated by using Eq. (3).

3.3. Degeneracies Analysis

Generally, ego-motion estimation basing on projected curves (namely on their coefficients F_1 to F_7 of Eq. (8)) leads to a unique solution $\{< R_w, (a_w, b_w) >, \}$. Nevertheless, some degenerate or singular configurations leads to ambiguous results on s_1 . We present three degenerate configurations (derivation details in supplemental material):

- **Case 1:** 3D line located within y-z-plane ($a_w = 0$, $b_w = x$, $R_w = (x, 0, 0)$) and arbitrary ego-rotation along x-axis ($\omega = (x, 0, 0)$).

Figure 4: Automatic actual 3D line selection. Both 3D straight lines and curves observed will be projected as 2D curves on RS cameras under ego-motion. We firstly fit all detected curve pixels to hyperbolic polynomials and discard curves with big fitting errors (red curves in step 1). In step 2, we randomly select putative sets of 4 curves (blue curves) and compute camera ego-motion for each set. Then we correct all image candidate curves basing on the obtained ego-motion and compute the global straightness score. After N samples, the sample with the smallest straightness score is chosen as best sample. After discarding curves whose corrected straightness exceeds a defined threshold, we obtain final set of inliers.

- **Case 2:** 3D line located within x-z-plane ($a_w = x, b_w = 0, R_w = (0, x, 0)$) and arbitrary ego-rotation along y-axis ($\theta = (0, x, 0)$).

- **Case 3:** 3D line parallel to x-axis ($a_w = x, b_w = x, R_w = (0, \sqrt{2}, 0)$) and arbitrary ego-rotation along y-axis ($\theta = (x, 0, 0)$).

However, the configurations occur rarely in practice with hand-held camera or with a camera embedded on a vehicle.

4. Automatic Selection of Actual 3D Lines

Since both 3D lines and curves will be rendered as 2D curves on image, the problem of how to automatically distinguish image curves corresponding to 3D lines from actual 3D curves arises. Method of [15] uses 3 degree polynomial fitting to reject obvious outliers, but not ambiguous ones. The method of [13] uses Huber M-estimator during joint estimation of motion parameters and vanishing directions to reject short Line Segments which are not sufficiently well oriented according to the vanishing directions. Unfortunately, some of these LS survive the rejection process because they may be aligned with one of the vanishing directions despite not satisfying MW assumption, thus participating to motion parameter computation. In this paper, we propose a 2-steps method (shown in Fig. 4) inspired from RANSAC technique. This selection process aims not only to reject features corresponding to non straight lines (outliers) but also to maximize the number of features corresponding to actual straight lines (inliers), thus increasing the accuracy and robustness of ego-motion calculation.

Step 1: The goal of this step is to generate a preliminary set of candidate curves. Edge curves are first detected using Canny detector and linked to close small gaps. Then short curves (less than 20 pixels in experiments) are discarded. Finally, curves which sufficiently fit hyperbolic polynomial are selected basing on RMSE fitting error score.

Step 2: The second step is a global consistency verification. It consists in the integration of the R4C method within a RANSAC-like framework:

- Repeat N times
 - (i) Select a random sample $S_i = \{C_1, C_2, C_3, C_4\}$ of 4 four curves C_j among the candidate curves.
 - (ii) Run R4C to calculate θ_i based on S_i .
 - (iii) Perform forward mapping Eq-13 to all curves pixels and calculate straightnesses of each curves after correction.
- Select the sample with maximum number of inliers (straightness of the corrected curve smaller than θ_{thr} pixels) as best solution over all samples.
- Refine the best solution by performing R4C again using the inliers.

The straightness of a curve is calculated by mean square of perpendicular offset of each curve pixel to its corresponding least-squares- fitting line. We set $\theta_{thr} = 1$ experimentally and the number of samples N is set automatically used method in [6]

5. Experiments

In this section, We compare our method (R4C with automatic feature selection) to existing works [15, 14, 13] using both synthetic and real data.

The experiments were conducted on a i5 CPU 2.8G Hz with 4G RAM. It took around 4.1s for curve detection and fitting, 1.9s for ego-motion estimation with automatic feature selection, and 0.1s for image correction with 240×320 size. The proposed method was implemented in MATLAB. A improvement can be expected using C++.

Figure 5: Comparison of the proposed method with [15, 13] under different configurations (varying angular velocities, translation velocity, outlier curves number and image noise). We use mean value of estimated rotation errors \bar{e}^{rotate} as a tool to quantitatively evaluate accuracy of ego-motion estimation.

5.1. Synthetic RS Image Experiments

Grid Scene: We simulated a grid scene where MW assumption holds on as required by [15, 13] (but not by our method). Images corresponding to random angular velocities were generated using the following virtual camera parameters: focal = 1 unit, resolution as 640×480 and scan speed = 7.5×10^{-5} s/row. Then values of ω were computed from deformed edges using R4C. While ground-truths are available, we evaluated ego-rotation accuracy using both visual checking and mean value of estimated rotation errors \bar{e}^{rotate} [14] of each image row (calculated using Eq-3). 100 values of \bar{e}^{rotate} were generated randomly in different directions. Since method in [14] which uses z-axis rotate model fails in most cases. We compared our method to two other single-image based RS corrections methods also using line features [15, 13].

The results in Fig. 6 (first row) show that our method and [15, 13] obtain correct results under gentle conditions ($d = 5$ unit/s, $\omega = 5$ during acquisition, RS image with average 0.5 pixel noise and no outlier curves). However, significant differences appear when RS effect becomes more important.

- **Accuracy vs Angular Velocity:** Experiments were carried out with ω varying from 0 to 30 degree/frame. Results in Fig. 5,6 show that the RS ego-motion estimation errors of [15] climb from 0 to 14 degrees while errors of [13] keep low under 15 deg/frame but dramatically increase with bigger ω . Inversely, R4C maintains the error under 1 deg.

Figure 6: Comparison of the proposed method with [13] and [15] under gentle condition (first column), large ω (second column), large d (third column) and outlier presence (fourth column).

- **Accuracy vs Translation Velocity:** Since all tested methods assume translation velocity is negligible during image exposure, we now verify accuracy of these three methods if RS translation velocity effect is significant. The translation velocity d was increased from 0 to 12 unit/s which is extremely high and rarely occurs in a real application context. The results in Fig. 5 shows that the three methods perform stable and achieve errors under 1.2 degree with big translation velocities.

- **Accuracy vs Outlier Curves:** In this experiment, we simulated 3D straight lines and added 3D curves as outliers. The number of outliers was increased from 0% to 50%. Results in Fig. 5,6 show that both [15, 13] fail in presence of outliers. In contrast, thanks to automatic feature selection, our method obtain correct correction in different settings.

- **Accuracy vs Pixel Noise:** We fixed the camera translational speed to 5 unit/s and the angular velocity to 5 deg/frame. We added a random Gaussian noise to projected curve points from 0 to 2 pixels. The results in Fig. 5 demonstrate that our approach is much more robust against increasing noises compared to [15, 13].

Complex Urban Scene: We evaluated performances of our method compared to [15, 14, 13] using synthetic RS images.

Our first experiment based on a public synthetic RS image dataset [3] which contains multiple RS video filming a house scene. Since ground-truth of camera ego-motion are known, we keep using \bar{e}^{rotate} to evaluate accuracy of corrections. The results in Fig. 7 shows that [13] fails since house roof and current lead violate MW assumption. Our method and [15, 14] obtain corrected images visually close to ground-truth image. However, both [15, 14] use simpli-

(a) Ground truth image	(b) RS image	(c) $\bar{e}^{\text{rotate}} = 2.28$ Correction by [15]
(d) $\bar{e}^{\text{rotate}} = 40.81$ Correction by [13]	(e) $\bar{e}^{\text{rotate}} = 3.59$ Correction by [14]	(f) $\bar{e}^{\text{rotate}} = 1.21$ Our method

Figure 7: Comparison of our method (f) with [15] (c), [14] (d) and [13] (e) on a synthetic RS image dataset [3]. The correction results are evaluated by using mean value of estimated rotation errors \bar{e}^{rotate} of each row compared to ground-truth.

fied ego-motion model which can not recover ego-rotation along all three x-y-z axis. In other words, visually acceptable correction does not ensure consistency of geometry. We also perform our method with different baselines using RS images from urban scene dataset [14] and results shown are in Fig. 8.

The quantitative evaluation results in Fig. 7 using \bar{e}^{rotate} demonstrate that our method not only offers visually pleasant corrected image, but also bales to recover images which better fit GS-based 3D geometry. The third and fourth images are veiled by tree branches which can be regarded as outliers. As a result, worse distortions are observed on corrected images of [15, 14] meanwhile our method and [14] obtain similar corrections, however small curvature remains on results of [14].

Since MW assumption holds on in the first RS image of Fig. 8, except [15], there are no significant curvatures left in the corrected images. However, [13] keeps effects of vertical shrinking while [14] and our method obtain better visual corrections. The second RS image shows a circular building with many 3D curves. The correction results demonstrate that our proposed method can successfully filter outliers while [15] and [13] fail. One can note that the corrected image of our method preserves curves belonging to the circular facade, meanwhile, [14] straightens every curve. Moreover, the curves obtained by our method fit better ellipse sections being the perspective projection of circles.

Figure 8: Visual comparison of the proposed method with baselines [15, 13, 14]. The red boxes highlights the superiority of our method.

5.2. Real RS Image Experiments

We conduct the first real images experiment on a RS video dataset [16]. Comparison of our frame-by-frame RS corrections with methods of [15, 13, 14] are shown in Fig. 9. We use the approach described in [13] to do a quantitative evaluation by counting mean value of the number of found inliers $|R_F|$ (point matches after estimating the fundamental matrix) between corrected frame pairs. The results show our method obtains the higher inlier number and demonstrate that it can better recover the consistency of projection geometry.

We also compared our method on a challenging complex urban dataset which was captured by a Logitech camera with strong RS effects. It can be seen that [15] obtain relatively acceptable corrections for the first image but fails for

(a) RS distorted image	(b) $\mathcal{R}_F \downarrow = 186.44$ [15]	(c) $\mathcal{R}_F \downarrow = 212.39$ [13]	(d) $\mathcal{R}_F \downarrow = 201.84$ [14]	(e) $\mathcal{R}_F \downarrow = 216.91$ Our method
------------------------	---	---	---	---

Figure 9: An example frame from a RS video [16] (a). The correction results by [15, 13, 14] and by our method are shown in (b)-(e). A quantitative evaluation using the mean number of found inliers $|\mathcal{R}_F|$ between corrected frame pairs are also shown below each corrected image.

Figure 10: Comparison of image correction results on two real RS images of a complex urban scene with strong RS effects against to methods of [15, 13, 14].

the second image due to the difficulty in grouping curves. Methods of [13, 14] fail in both cases because of the complexity of the scenes and the large angular velocities during acquisition. In contrast, we can see that our method obtains visually better corrections in both RS images.

6. Conclusions

We presented a novelty RS correction method which uses line features. Unlike existing methods, which uses iterative solutions and make MW assumption, our method R4C computes linearly the camera ego-motion using few image features. Besides, the method was integrated in a RANSAC-like framework which enables us reject outlier curves making image correction more robust and fully automated.

Extensive experiments demonstrated the robustness and the accuracy of the proposed method in variable complex urban scenes or under extreme filming conditions. Specifically, our method not only produces visually pleasant corrections, but also is able to preserve consistency of geom-

etry. Thus, the proposed method in this paper can serve for rolling shutter image correction as well as for pre-processing images in other computer vision applications such as feature matching and tracking or SfM.

Acknowledgement. This work has been sponsored by the French government research program "Investissements d'Avenir" through the IDEX-ISITE initiative 16-IDEX-0001 (CAP 20-25), the IMobS3 Laboratory of Excellence (ANR-10-LABX-16-01) and the RobotEx Equipment of Excellence (ANR-10-EQPX-44). This research was also financed by the European Union through the Regional Competitiveness and Employment program -2014-2020- (ERDF – AURA region) and by the AURA region.

References

- [1] O. Ait-Aider, A. Bartoli, and N. Andreff. Kinematics from lines in a single rolling shutter image. In *2007 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–6. IEEE, 2007.
- [2] G. Duchamp, O. Ait-Aider, E. Royer, and J.-M. Lavest. A rolling shutter compliant method for localisation and recon-

- struction. In *VISAPP (3)*, pages 277–284, 2015.
- [3] P.-E. Forssén and E. Ringaby. Rectifying rolling shutter video from hand-held devices. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 507–514. IEEE, 2010.
- [4] M. Grundmann, V. Kwatra, D. Castro, and I. Essa. Calibration-free rolling shutter removal. In *Computational Photography (ICCP), 2012 IEEE International Conference on*, pages 1–8. IEEE, 2012.
- [5] R. Hartley and A. Zisserman. *Multiple view geometry in computer vision*. Cambridge university press, 2003.
- [6] R. Hartley and A. Zisserman. *Multiple view geometry in computer vision*. Cambridge university press, 2003.
- [7] S. Hee Park and M. Levoy. Gyro-based multi-image deconvolution for removing handshake blur. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3366–3373, 2014.
- [8] E. Ito and T. Okatani. Self-calibration-based approach to critical motion sequences of rolling-shutter structure from motion. In *Computer Vision and Pattern Recognition (CVPR), 2017 IEEE Conference on*.
- [9] C. Jia and B. L. Evans. Probabilistic 3-d motion estimation for rolling shutter video rectification from visual and inertial measurements. In *MMSP*, pages 203–208, 2012.
- [10] Y.-G. Kim, V. R. Jayanthi, and I.-S. Kweon. System-on-chip solution of video stabilization for cmos image sensors in hand-held devices. *IEEE transactions on circuits and systems for video technology*, 21(10):1401–1414, 2011.
- [11] C.-K. Liang, L.-W. Chang, and H. H. Chen. Analysis and compensation of rolling shutter effect. *IEEE Transactions on Image Processing*, 17(8):1323–1330, 2008.
- [12] A. Patron-Perez, S. Lovegrove, and G. Sibley. A spline-based trajectory representation for sensor fusion and rolling shutter cameras. *International Journal of Computer Vision*, 113(3):208–219, 2015.
- [13] P. Purkait, C. Zach, and A. Leonardis. Rolling shutter correction in manhattan world. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 882–890, 2017.
- [14] V. Rengarajan, Y. Balaji, and A. Rajagopalan. Unrolling the shutter: Cnn to correct motion distortions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2291–2299, 2017.
- [15] V. Rengarajan, A. N. Rajagopalan, and R. Aravind. From bows to arrows: Rolling shutter rectification of urban scenes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2773–2781, 2016.
- [16] E. Ringaby and P.-E. Forssén. Efficient video rectification and stabilisation for cell-phones. *International Journal of Computer Vision*, 96(3):335–352, 2012.
- [17] G. Schindler, P. Krishnamurthy, and F. Dellaert. Line-based structure from motion for urban environments. In *3D Data Processing, Visualization, and Transmission, Third International Symposium on*, pages 846–853. IEEE, 2006.
- [18] R. G. Von Gioi, J. Jakubowicz, J.-M. Morel, and G. Randall. Lsd: A fast line segment detector with a false detection control. *IEEE transactions on pattern analysis and machine intelligence*, 32(4):722–732, 2010.
- [19] H. Yu and A. I. Mourikis. Vision-aided inertial navigation with line features and a rolling-shutter camera. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 892–899. IEEE, sep 2015.
- [20] B. Zhuang, L.-F. Cheong, and G. H. Lee. Rolling-shutter-aware differential sfm and image rectification. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 948–956, 2017.