Probabilistic 3-D Motion Estimation for Rolling Shutter Video Rectification from Visual and Inertial Measurements

Chao Jia [1] and Brian L. Evans [2]

Department of Electrical and Computer Engineering, The University of Texas at Austin
Austin, Texas 78712, USA
[1] cjia@utexas.edu
[2] bevans@ece.utexas.edu

***Abstract*—Video acquired by handheld CMOS cameras may suffer from rolling shutter artifacts. Rolling shutter artifacts, which are due to the rows in the image sensor array being exposed sequentially from top to bottom, increase with the speed of the relative motion between the scene and camera. To rectify these artifacts, one needs to recover the projection parameters for each row. In this paper, we propose a probabilistic method to estimate 3-D camera rotation by using video and inertial measurements on the handheld platform, such as a smart phone. Our contributions are (1) an efficient sensor fusion algorithm using an extended Kalman filter, and (2) a quality assessment method using vanishing point detection. Experiments indicate that the proposed sensor fusion algorithm produces a more accurate orientation estimate and better rectifies rolling shutter artifacts.**

I. Introduction

Handheld video cameras, especially in cell phones have become increasingly popular today because of their portability and low price. However, video sequences acquired by hand-held platforms usually suffer from annoying jitter due to camera shake. In addition, many hand-held video and image capture platforms use CMOS sensors instead of CCD sensors due to cheaper cost and on-chip processing. In a CMOS sensor camera, different rows in a frame are exposed sequentially from top to bottom. When there is fast relative motion between the scene and the video camera, a frame can be distorted since each row has its own projection parameters. This is known as the rolling shutter effects and the distortion usually includes skew and wobble [1], [2]. Rolling shutter effects can severely affect the visual quality and the follow-up processing/understanding of the video sequences.

Given a proper model, the camera motion can be estimated and then used to correct the rolling shutter effects (as shown in Fig. 1) and stabilize the video simultaneously. Video rectification and stabilization based on 3-D motion model usually performs better than 2-D models [3]. Traditional 3-D camera motion estimation methods rely on the video frames only, which is computationally expensive and not robust. Recently inertial measurement sensors such as gyroscopes and accelerometers, which can be found in many modern cell phones, have been used to estimate the camera motion directly without the help of video sequences [4], [5]. The estimation time is thus greatly shortened, which makes real-time rectification possible.

This research was supported by gift funding from TI.

Fig. 1. An example of a frame with rolling shutter effects (left) and the rectified frame (right).

However, the inertial measurement sensors usually suffer from measurement noise and bias, which leads to inaccurate motion estimates after long time periods. Errors also come from the discretization of the continuous-time dynamic motion system. More accurate motion estimates can be obtained when visual and inertial sensors are deployed together [6]. One way to fuse data from these two kinds of sensors is to use online Bayesian approaches such as extended Kalman filters (EKF). Unfortunately, existing algorithms assume a unique camera pose for each frame and therefore can fail when severe rolling shutter effects exist.

In this paper, we propose an EKF-based method to estimate 3-D camera motion using both visual and inertial measurements for each row in a video frame. In order to compare the quality of the rectified video using different algorithms, we propose a new quality assessment method using vanishing point detection that can be used when no ground truth is available. Compared with other algorithms that use only inertial measurement data, our algorithm results in higher motion estimation accuracy and better rectification quality.

II. Related Work

The readings of gyroscopes can be directly used to estimate camera rotation by using integration and interpolation. The camera translation, however, cannot be accurately estimated from inertial measurement sensors. The readings of accelerometers capture not only linear acceleration of cameras, but also gravity and acceleration caused by rotation. Besides, acceleration readings must be integrated twice to obtain the camera translation, which makes the estimation more prone to measurement noise. Even if we can obtain accurate camera translation, the video rectification and stabilization problem is still ill-posed since it is impossible to obtain depth information for every image pixel. Dense warping [3] and image-based rendering [7] have been applied to approximate the stabilization results based on sparse 3-D scene reconstruction. However, they are computationally prohibitive for many handheld platforms.

Fortunately, camera shake and rolling shutter effects are caused primarily by camera rotations. In fact, [4] and [8] have shown that taking only camera rotations into account is sufficient to produce satisfactory videos.

In our paper, we also use gyroscope readings. In the gyroscope-only method [4] the camera rotation is directly estimated by integrating the gyroscope readings (angular velocities). Another recent approach [5] uses both gyroscope and accelerometer readings to estimate the camera rotations based on EKF. The gyroscope readings are used as the control inputs in the dynamic motion model. The authors assume that users usually try to hold the camera in a steady position so the gravity is approximately the only source in the accelerometer measurements. Thus the accelerometer readings can be used as measurements of the camera rotation.

Our 3-D orientation estimation is also based on EKF, but our measurement model is quite different from [5]. We find that the linear acceleration of the camera and the acceleration caused by rotation are sometimes non-negligible. Thus we do not use the accelerometer readings as orientation measurements. Instead, we use the tracked feature points extracted from the video frames, which provide accurate geometric clue for the estimation of the camera motion. Based on the fact that matched feature points can be related by a homographic transformation under pure rotational motion, the relative rotation between consecutive frames can be measured [9].

Motion estimation based on visual and inertial measurement sensors have been extensively studied in the problem of simultaneous localization and mapping (SLAM) in robotics [10]. However, the rolling shutter camera model has never been considered in SLAM before. Our algorithm is the first EKF-based motion estimation method for rolling-shutter cameras that uses visual and inertial measurements. In our measurement model, tracked feature points in consecutive frames are only linked by the relative camera rotation between them. Therefore, our algorithm can be classified as a relative motion estimation method [11], [12].

III. Camera Model

For rolling shutter cameras, each row in a frame is exposed at a different time. Fig. 2 illustrates the image capture model of a rolling shutter camera, where t_r is the total readout time in each frame and t_{id} is the inter-frame idle time. Thus for an image point $\mathbf{u} = [u_0, u_1]^T$ in frame i, the exposure time is $t(\mathbf{u}, i) = t_i + t_r \times \frac{u_1}{h}$, where t_i is the timestamp of frame i and h is the total number of rows in each frame.

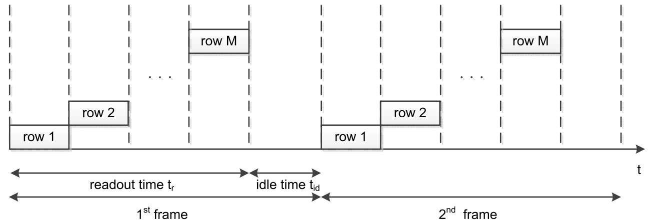

Fig. 2. Rolling shutter cameras sequentially expose rows. $t_r + t_{id} = \frac{1}{\textbf{frame per second}}$.

Assume the intrinsic camera matrix is $\mathbf{K}$, the sequences of rotation matrices and translation vectors of the camera are $\mathbf{R}(t)$ and $\mathbf{l}(t)$. A 3-D point $\mathbf{x}$ and its projection image $\mathbf{u}$ in frame i should satisfy the following equation:

$$\mathbf{u} \sim \mathbf{KR}(t(\mathbf{u}, i))(\mathbf{x} + \mathbf{l}(t(\mathbf{u}, i))) \tag{1}$$

where $\sim$ indicates equality up to scale.

Usually there is a constant delay t_d between the recorded timestamps of gyroscopes and videos. Thus using the timestamps of gyroscopes as reference, the exposure time equation should be modified as

$$t(\mathbf{u}, i) = t_i + t_d + t_r \times \frac{u_y}{h}. \tag{2}$$

When pure rotation is considered, the translation vector remains unchanged and thus the image of a certain scene point in one frame can be mapped to another frame through a 3×3 homography matrix

$$\mathbf{u}' \sim \mathbf{KR}(t(\mathbf{u}', i))\mathbf{R}^T(t(\mathbf{u}, j))\mathbf{K}^{-1}\mathbf{u} \tag{3}$$

where $\mathbf{u}'$ and $\mathbf{u}$ are the images in frame i and j respectively.

IV. Online Rotation Estimation

Our online motion estimation is based on EKF. Due to the special property of rolling shutter camera model and the pure rotation motion model, state definition and the structure of dynamical and measurement model need to be designed carefully.

A. State Vector and Dynamic Bayesian Network

The gyroscope in cell phone cameras usually has a higher sampling frequency (around 100 Hz) than the video frame rate, as illustrated in Fig. 3.

In Fig. 3, several gyroscope readings are grouped together since they are used to compute the camera rotations for the same frame during its corresponding exposure time. Note that due to the fact that the idle time t_{id} is large enough so that no pixels in frame i but only several pixels in frame $i+1$ are exposed after τ_{k+3}. Thus ω_{k+3} is relegated to group $i+1$. Further we assume that a certain 3-D feature point has its projection at $\mathbf{u}$ in frame i and $\mathbf{u}'$ in frame $i+1$. Without

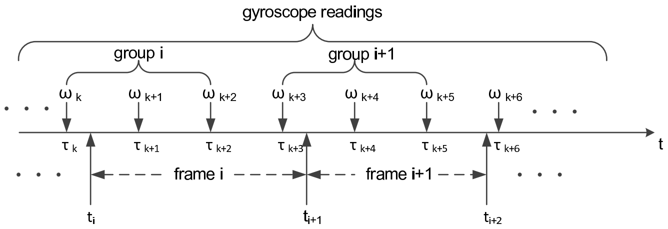

Fig. 3. The gyroscope readings and frame timestamps

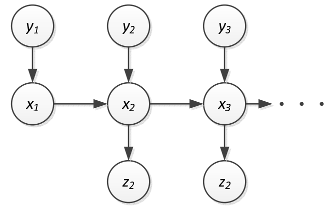

Fig. 4. The probabilistic graphical model of the online estimation.

loss of generality, assume $t(\mathbf{u}, i)$ is between τ_{k+1} and τ_{k+2}; $t(\mathbf{u}', i+1)$ is between τ_{k+4} and τ_{k+5}. Then the relative rotation matrix relates $\mathbf{u}'$ and $\mathbf{u}$ is:

$$\mathbf{R}(t(\mathbf{u}', i+1))\mathbf{R}^T(t(\mathbf{u}, i)) = \prod_{n=k+1}^{k+4} \Delta\mathbf{R}(\omega_n \Delta t_n) \quad (4)$$

where Δt_{k+1} to Δt_{k+4} are equal to $\tau_{k+2} - t(\mathbf{u}, i+1)$, $\tau_{k+3} - \tau_{k+2}$, $\tau_{k+4} - \tau_{k+3}$ and $t(\mathbf{u}', i+1) - \tau_{k+4}$ respectively. Each sub-relative rotation matrix can be computed by exponentiating the skew symmetric matrix formed from the angular velocity and its duration:

$$\Delta\mathbf{R}(\omega_n \Delta t_n) = \exp(\text{skew}(\omega_n)\Delta t_n). \quad (5)$$

Therefore, the two angular velocity groups i and $i+1$ are enough to represent the relative rotation matrix between any pair of matching feature points in frame i and $i+1$. Thus we define our state vector as

$$\mathbf{x}_i = [\omega(i-1,1), \ldots, \omega(i-1, N_{i-1}), \omega(i,1), \ldots, \omega(i, N_i)]^T \quad (6)$$

where N_i is the total number of angular velocity vectors in group i. The reason why we directly use angular velocity instead of the rotation unit quaternion (or axis-angle rotation representation) in the state vector is that: (a) The relative poses are actually updated independently, which makes using rotation representation in the state vector almost helpless. (b) As shown in (4), angular velocities can be directly used in computing the relative rotation matrix, while rotation representation needs more complicated spherical linear interpolation (SLERP). This fact matters since it will affect the complexity of computing the Jacobian matrices for EKF.

Given the definition of the state vector, Fig. 4 shows the dynamic Bayesian network that illustrates the EKF-based online estimation. Gyroscope readings $\mathbf{y}_i$ can be used as the control inputs. The feature points detected in frame i can be used as measurements $\mathbf{z}_i$ while assuming their matching points in frame $i-1$ are fixed parameters. Details of the motion and measurement models with EKF equations are shown as follows:

B. Dynamic Motion Model

Since in each state vector there are two groups of angular velocities for two consecutive frames, there will be overlap between $\mathbf{x}_i$ and $\mathbf{x}_{i+1}$. We can further rewrite $\mathbf{x}_i$ as $[\mathbf{x}_{i,1}; \mathbf{x}_{i,2}]$, where $\mathbf{x}_{i,1}$ and $\mathbf{x}_{i,2}$ represents the groups of angular velocities for frame $i-1$ and i respectively. Then given the gyroscope readings $\mathbf{y}_i$ (the measured angular velocity group $\mathbf{x}_{i,2}$) with Gaussian measurement noise $\mathbf{w}_i \sim \mathcal{N}(\mathbf{0}, Q)$, we can update the state as

$$\mathbf{x}_i = \begin{bmatrix} \mathbf{x}_{i,1} \\ \mathbf{x}_{i,2} \end{bmatrix} = \begin{bmatrix} \mathbf{x}_{i-1,2} \\ \mathbf{y}_i \end{bmatrix} + \begin{bmatrix} \mathbf{0} \\ \mathbf{w}_i \end{bmatrix}. \quad (7)$$

Note that the second group in the preceding state vector is copied to the first group in the current state vector. This is a linear motion model and if we represent it as $\mathbf{x}_i = f(\mathbf{x}_{i-1}, \mathbf{y}_i, \mathbf{w}_i)$, we have

$$A_i = \left.\frac{\partial f}{\partial \mathbf{x}}\right|_{\mathbf{x}_{i-1}} = \begin{bmatrix} \mathbf{0} & I \\ \mathbf{0} & \mathbf{0} \end{bmatrix}, W_i = \left.\frac{\partial f}{\partial \mathbf{w}}\right|_{\mathbf{w}_i} = \begin{bmatrix} \mathbf{0} \\ I \end{bmatrix}. \quad (8)$$

The initial state vector $\mathbf{x}_1$ is just a copy of the gyroscope readings since there is only one group of angular velocity in it.

C. Measurement Model

As we have mentioned, the measurement can be written as $\mathbf{z} = [\mathbf{u}_{i,1}, \mathbf{u}_{i,2}, \ldots, \mathbf{u}_{i,M}]^T$, where $\mathbf{u}_{i,j}$ is the 2-D coordinate value of the j th feature point in frame i. Assume its matching point in frame $i-1$ is $\mathbf{u}_{i-1,j}$ then according to (3) we have

$$\mathbf{u}_{i,j} = g\left(K\Delta R K^{-1} \begin{bmatrix} \mathbf{u}_{i-1,j} + \mathbf{v}_{i,j,1} \\ 1 \end{bmatrix}\right) + \mathbf{v}_{i,j,2} \quad (9)$$

where $g()$ is the function to convert a homogeneous vector into an inhomogeneous vector, $\mathbf{v}_{i,j,1}$ and $\mathbf{v}_{i,j,2}$ represent the Gaussian messurement noise in feature point detection for $\mathbf{u}_{i-1,j}$ and $\mathbf{u}_{i,j}$. Similar to the example shown in (4) the relative rotation matrix ΔR is expressed as

$$\prod_{k=1}^{N_{i-1}} \Delta R(\omega(i-1,k)\Delta t_{i-1,j,k}) \prod_{k=1}^{N_i} \Delta R(\omega(i,k)\Delta t_{i,j,k}) \quad (10)$$

where N_{i-1} and N_i are the number of angular velocities in group $i-1$ and i. The duration time Δt for each angular velocity can be computed in the same way as the example shown in (4). Note that some of them will be zero. The entire measurement model can be expressed as

$$\mathbf{z}_i = \begin{bmatrix} \mathbf{u}_{i,1} \\ \mathbf{u}_{i,2} \\ \vdots \\ \mathbf{u}_{i,M} \end{bmatrix} = \begin{bmatrix} h_1(\mathbf{x}_i, \mathbf{u}_{i-1,1}, \mathbf{v}_{i,1}) \\ h_2(\mathbf{x}_i, \mathbf{u}_{i-1,2}, \mathbf{v}_{i,2}) \\ \vdots \\ h_M(\mathbf{x}_i, \mathbf{u}_{i-1,M}, \mathbf{v}_{i,M}) \end{bmatrix} \tag{11}$$

where $h_j()$ is defined in (9). Note that the key idea is to consider $\{\mathbf{u}_{i-1,j}\}$ as known parameters with unknown noise and only use $\{\mathbf{u}_{i,j}\}$ as the measurements. In this way, the rotational homographic transformation can be directly used in the measurement model. Due to page limits, we omit the derivation (using the chain rule) of the closed-form Jacobian matrix of (11) used for EKF. Note that when computing the derivative with respect to the angular velocities, $\Delta\mathbf{R}(\omega\Delta t)$ in (5) can be linearly approximated as

$$\Delta\mathbf{R}(\omega\Delta t) = \begin{bmatrix} 1 & -\omega_z\Delta t & \omega_y\Delta t \\ \omega_z\Delta t & 1 & -\omega_x\Delta t \\ -\omega_y\Delta t & \omega_x\Delta t & 1 \end{bmatrix} \tag{12}$$

since Δt is very small.

The reason why this kind of relative rotational measurement model works is that the effect caused by the translation between two consecutive frames is negligible.

V. Rectification and Stabilization

Once we have run the EKF to estimate the angular velocities we can compute the rotation matrix at any time. To rectify the rolling shutter effect for any frame i, we fix a unique camera pose (rotation matrix) $\mathbf{R}_i$ which can be the pose either at the beginning or in the middle of the exposure time of that frame. Then any pixel $\mathbf{u}$ in the frame can be re-projected under this unique rotation matrix:

$$\mathbf{u}' \sim \mathbf{K}\mathbf{R}_i\mathbf{R}^T(t(\mathbf{u}, i))\mathbf{K}^{-1}\mathbf{u}. \tag{13}$$

If we want to further stabilize the video, we can apply a low-pass filter on the sequence of rotation matrices $\{\mathbf{R}_i\}$ to get a new matrix sequence $\{\mathbf{R}_i^{new}\}$. Then we can use similar re-projection method like (13) to stabilize the video. Specifically, we convert the sequence of rotation matrices to a sequence of Euler angles first and apply linear low pass filter on yaw, pitch and roll angle sequences respectively.

One problem about the rolling shutter rectification is that the transformation in (13) is non-invertible, so fast inverse interpolation method does not give the accurate result. However, after we compared it with different types of forward interpolation warping method, we found that inverse interpolation is still preferable as it is remarkably faster while only sacrificing a small amount of accuracy.

VI. Calibration and Synchronization

Before we run our algorithm to estimate the camera poses and rectify the video, we need to know the value of several parameters of the camera such as the readout time t_r and the intrinsic matrix $\mathbf{K}$. Also we need to synchronize the gyroscope timestamps and frame timestamps by finding the delay t_d between them.

The intrinsic matrix $\mathbf{K}$ can be estimated separately by some camera self-calibration algorithms such as [13] based on still pictures without the rolling shutter effects that only appear in videos. However, in this paper we follow the algorithm in [4] to estimate all the parameters at the same time. The calibration and synchronization algorithm in [4] is also based on the homographic transformation in (3). All the parameters are estimated using batch optimization which means that the objective function is the summation of all the re-projection errors throughout hundreds of consecutive frames. Note that the sequence of angular velocities is not an optimization variable in calibration. In other words, the gyroscope readings are trusted completely. Although this is not the assumption under our probabilistic motion estimation framework, the calibration result works pretty well. These parameters are only needed to be estimated once and will be fixed for further use.

VII. Experimental Results

In our experiments, we use a Google Nexus S Android cell phone that is equipped with a three-axis gyroscope and an accelerometer. To record the video and the inertial measurements at the same time, we use an application “Sensor Data Logger” developed by Cellbots [14]. All of the other processing is implemented in MATLAB. The feature points are tracked using the Kanade-Lucas-Tomasi (KLT) tracker [15]. The variances of the Gaussian noise in gyroscope readings and feature point detection are fixed as 0.005 and 0.3 respectively. We compare our algorithm with the methods in [4] and [5] that only use the inertial measurement sensors for 3-D rolling shutter rectification and video stabilization.

A. Estimation Accuracy of Camera Rotations

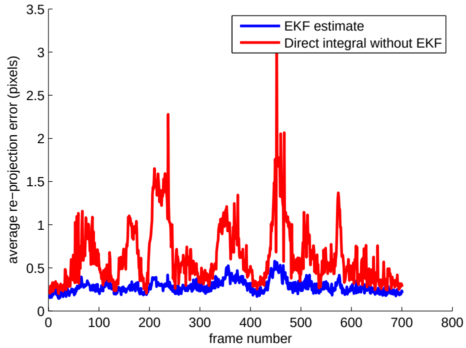

Fig. 5. The average re-projection error for each feature point in different frames.

In Fig. 5 we show the average re-projection error for each feature point in different frames with and without our EKF processing of the angular velocity. The re-projection error has been decreased significantly using the angular velocity that is estimated by EKF. However, it is unfair to claim that our algorithm is better since the re-projection error is what the EKF measurement model is based on.

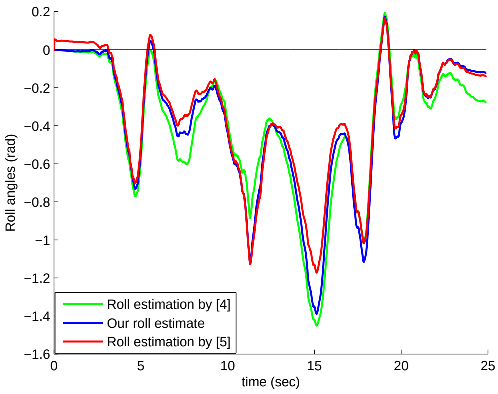

Fig. 6. The roll angle estimate by three algorithms.

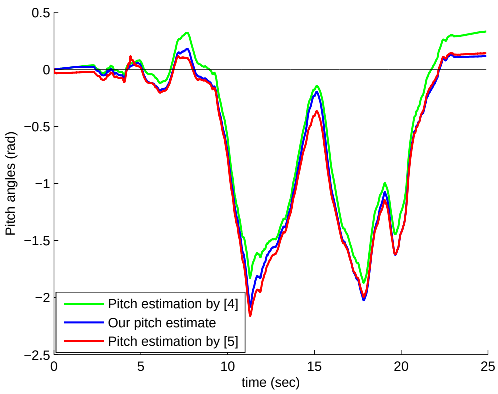

Fig. 7. The pitch angle estimate by three algorithms.

To show the accuracy of the rotation estimation with our algorithm, we put the cell phone still on a flat surface and then start recording video. We rotate the camera at will and finally put it back on the same flat surface. Theoretically there should be only rotation around the z-axis of the cell phone (the axis that points towards the outside of the front face of the screen). So if we set the initial position of the cell phone as the reference coordinate system, the pitch angle and roll angle will be back to zero at the end of the experiment. In Fig. 6 and Fig. 7 we can clearly find that the pitch and roll angles that are computed based on our algorithm converges closer to zero compared with the estimates in [4] and [5]. The method in [5] only performs slightly worse than ours in the sense of final estimate of pitch and roll angles, but occasionally there will be glitches in its rotation estimate since the gravity is not always the dominating source of the accelerometer measurements. The glitches can cause wavy distortion in the rectified frame, as shown in Fig. 8. This wavy distortion has never been found in the rectified frames using the rotation estimate from [4] or our method.

B. Evaluation of the Rectification Results

Fig. 9 is an example of rolling shutter effect rectification (with stabilization) using the rotation estimate from our algorithm. The difference among the rectified results with different motion estimation methods is very hard to tell visually. So we compare them numerically. Since there is no way to get the ground truth of the frame without rolling shutter effects, we cannot compute the difference between the ground truth and the rectified results. [8] uses 3-D animation software to create synthetic video sequences with rolling shutter effects to make the ground truth available. However, it is almost impossible for the synthetic data to contain real inertial measurements from cell phones accurately along with the video. Therefore the method in [8] cannot be used to compare our algorithm with [4] and [5].

Fig. 8. An example of the wavy distortion in the rectified frame using the rotation estimated by [5].

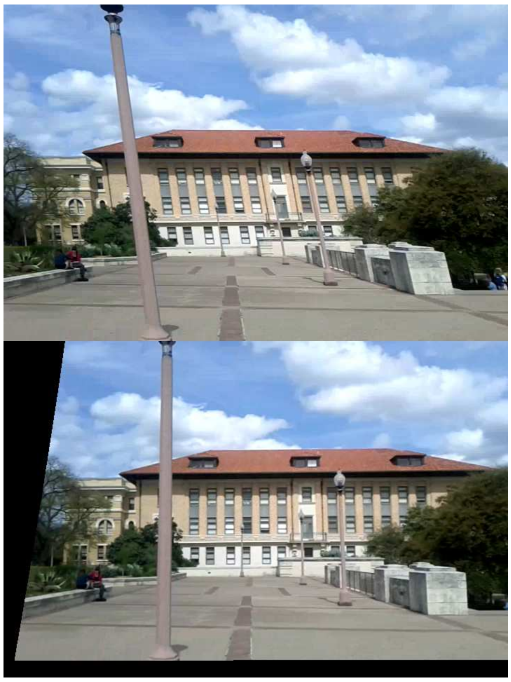

Fig. 9. Top: the original frame extracted from the video with rolling shutter effect. Bottom: the rectified frame (with stabilization) using the rotation estimate from our algorithm.

Fig. 10. Parallel lines in 3-D world extracted from the frames.

TABLE I
AVERAGE EUCLIDEAN DISTANCE FROM THE LINES TO THE VANISHING POINT (IN PIXEL)

Rectification method	**Video #1**	**Video #2**
No rectification (original)	3.500	2.800
Orientation estimated by [4]	1.820	2.150
Orientation estimated by [5]	1.628	1.387
Orientation estimated by proposed method	1.180	0.800

In this paper we make use of the fact that under perspective projection without rolling shutter problem parallel lines in 3-D world that are not parallel to the image plane should appear to converge to a unique vanishing point. For a rolling shutter video sequence, this property does not hold anymore since different rows have different projection time. We take several videos of buildings with clear parallel edges; then we extracted these parallel lines from the original videos and the rectified videos using different algorithms, as shown in Fig. 10. We find the estimated vanishing point for these lines that minimize the average square distance from each line to this point. The resulting minimum distance is used for comparison since it shows how well the parallel 3-D lines converge to the vanishing point after projection. Thus we can compare the geometric correctness of the rectified frames without ground truth. From Table I we can find that the proposed method outperforms the other two rectification methods.

C. Processing Time

On a 2.3GHz Intel i5 processor, our MATLAB implementation (without parallel processing) takes 41ms on average to estimate the angular velocities using EKF and compute the camera orientation for each frame (including the time for KLT feature tracking) with 60 feature points per frame. Considering the possible implementation improvement using more efficient programming languages and GPU-based implementation, it is very promising to implement rolling shutter effect rectification and video stabilization in real-time at 30fps on advanced cell phones. In fact, EKF-based SLAM has been implemented in real-time since as early as 2005. Our algorithm has less computational complexity than traditional SLAM since we do not have to estimate the 3-D coordinate values of the feature points at the same time.

VIII. CONCLUSION

In this paper we have proposed an algorithm that estimates the 3-D camera rotation online using both visual and inertial measurements based on a rolling shutter camera model. The algorithm can be directly used in rolling shutter effect rectification and video stabilization for video recording on cell phones. We have demonstrated in the experiments that our algorithm can result in a more accurate estimate of the camera rotations compared with the methods that use only inertial measurements. The more accurate rotation estimate can help us rectify the rolling shutter effects in video sequences better. In addition, we have proposed a new method to compare the results of rolling shutter rectification using vanishing point detection when no ground truth is available.

The main limitation of our method is that it relies on the quality of feature point detection. In low-light conditions the frames may be blurred due to the long exposure time, so it is harder to detect and track feature points precisely. The inertial measurement sensors, however, can keep the same measurement quality.

REFERENCES

[1] C.-K. Liang, C. L-W., and C. H., "Analysis and compensation of rolling shutter effect," *IEEE Trans. on Image Processing*, vol. 17, no. 8, pp. 1323–1330, August 2008.
[2] S. Baker, E. Bennett, K. S. B., and R. Szeliski, "Removing rolling shutter wobble," in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, June 2010.
[3] F. Liu, M. Gleicher, H. Jin, and A. Agarwala, "Content-preserving warps for 3d video stabilization," *ACM Trans. on Graphics*, vol. 28, no. 3, 2009.
[4] A. Karpenko, D. Jacobs, J. Baek, and M. Levoy, "Digital video stabilization and rolling shutter correction using gyroscopes," Stanford University, Tech. Rep., March 2011.
[5] G. Hanning, N. Forslöw, P.-E. Forssén, E. Ringaby, D. Törnqvist, and J. Callmer, "Stabilizing cell phone video using inertial measurement sensors," in *Proc. Intl. Workshop on Mobile Vision*, November 2011.
[6] D. Strelow and S. Singh, "Online motion estimation from image and inerital measurements," in *Proc. Workshop on Interation of Vision and Inertial Sensors*, June 2003.
[7] C. Buehler, M. Bosse, and L. McMillan, "Non-metric image-based rendering for video stabilization," in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, 2001, pp. 609–614.
[8] P.-E. Forssén and E. Ringaby, "Rectifying rolling shutter video from hand-held devices," in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, June 2010.
[9] C. Morimoto and R. Chellappa, "Fast 3D stabilization and mosaic construction," in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, June 1997.
[10] H. Durrant-Whyte and T. Bailey, "Simultaneous localisation and mapping (SLAM): Part I the essential algorithms," *Robotics & Automation Magazine*, June 2006.
[11] S. I. Roumeliotis and J. Burdick, "Stochastic cloning: A generalized framework for processing relative state measurements," in *Proc. IEEE Intl. Conf. on Robotics and Automation*, 2002, pp. 1788–1795.
[12] A. I. Mourikis and S. I. Roumeliotis, "On the treatment of relative-pose measurements for mobile robot localization," in *Proc. IEEE Intl. Conf. on Robotics and Automation*, 2006.
[13] Z. Zhang, "A flexible new technique for camera calibration," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 22, no. 11, pp. 1330–1334, 2000.
[14] Cellbots, "http://www.cellbots.com/."
[15] B. Lucas and T. Kanade, "An iterative image registration technique with application to stereo vision," in *Proc. Intl. Joint Conf. on Artificial Intelligence*, 1981, pp. 674–679.