Gyroscope-based Video Stabilisation With Auto-Calibration

Hannes Ovrén[1] and Per-Erik Forssén[1]

***Abstract*— We propose a technique for joint calibration of a wide-angle rolling shutter camera (e.g. a GoPro) and an externally mounted gyroscope. The calibrated parameters are time scaling and offset, relative pose between gyroscope and camera, and gyroscope bias. The parameters are found using non-linear least squares minimisation using the symmetric transfer error as cost function.**

The primary contribution is methods for robust initialisation of the relative pose and time offset, which are essential for convergence. We also introduce a robust error norm to handle outliers. This results in a technique that works with general video content and does not require any specific setup or calibration patterns.

We apply our method to stabilisation of videos recorded by a rolling shutter camera, with a rigidly attached gyroscope. After recording, the gyroscope and camera are jointly calibrated using the recorded video itself. The recorded video can then be stabilised using the calibrated parameters.

We evaluate the technique on video sequences with varying difficulty and motion frequency content. The experiments demonstrate that our method can be used to produce high quality stabilised videos even under difficult conditions, and that the proposed initialisation is shown to end up within the basin of attraction. We also show that a residual based on the symmetric transfer error is more accurate than residuals based on the recently proposed epipolar plane normal coplanarity constraint, and that the use of robust errors is a critical component to obtain an accurate calibration.

I. INTRODUCTION

This paper introduces a technique for joint calibration of a sports camera and an externally mounted gyroscope. We calibrate the time synchronisation (scaling and offset) and the relative pose between the two sensors, as well as the gyroscope measurement bias. The primary contribution is a robust initialisation of the relative pose and time offset parameters. The technique works with generic video and gyroscope sequences, and a recorded video can thus be used to first calibrate the setup, and then we can do a high quality stabilisation of the same video. See figure 1 for an example of input and output video frames.

Sports cameras (such as the GoPro series) are designed for documentation of first person sports events, such as cycling and mountaineering. Due to their small size they are also increasingly popular on small mobile robots like radio controlled cars and quadrotors. Sports cameras owe their good performance/size ratio to the use of an electronic rolling shutter [1], which needs to be considered in geometric computer vision, see e.g. [2]. Video stabilisation in post production is an option [3], [4], but it has known failure cases. For instance, the recently published Hyper-lapse[2] algorithm [6], produces summarising videos for GoPro video that look impressive when played at 10x speed, but frame-by-frame playback of e.g. walking sequences in the example videos reveals severe rolling shutter artefacts, and geometric errors near depth discontinuities. In contrast to post production approaches, the gyro based correction used here can correct for device rotations in *all* situations.

(a) Original (b) Stabilised

Fig. 1: Example of stabilised frames from the **RC-car** sequence (only the central part of frames are shown). The red lines show the same row and column in both frames, for reference. Note that the bent tree trunks in the input video have been corrected in the output, and that inter frame motion has been greatly reduced. See dataset webpage [7] for videos.

In the experiments we use an Arduino-based gyroscope logger that can easily be mounted together with a camera on small robot platforms. Compared to a gimbal solution, this type of solution is smaller, weighs much less, and requires very little power.

A. Related Work

Camera to IMU calibration is a well studied problem in the case of global shutter cameras, see e.g. [8] for a recent overview. Note however that the case of rolling shutter cameras requires more accurate time synchronisation, that can only be found by explicitly modelling a rolling shutter. We will thus focus this section on calibration of camera-IMU systems with rolling shutter cameras.

*Funded by the Swedish Research Council, project grant no. 2014-5928, and Swedish Foundation for Strategic Research (SSF) through grant IIS11-0081. ©2015 IEEE. Preprint from IEEE International Conference on Robotics and Automation 2015.

[1] Both authors are with the department of Electrical Engineering, Linköping University, Sweden hannes.ovren@liu.se

[2]Not to be confused with the recently released smartphone app Hyperlapse from Instagram, which is based on [5].

Calibration of rolling shutter readout time can be done using a flashing LED [9], [3], or using checkerboard calibration [10]. In [10] video of a checkerboard calibration pattern is recorded with a geometrically calibrated camera. Using tracks of points on the checkerboard pattern, a non-linear optimiser is used to find the camera trajectory relative to the checkerboard, as well as the unknown readout time.

Another related work is [11]. In this paper SLAM on a rolling shutter camera and IMU system is done, using a sliding window batch estimation of the continuous camera trajectory, while observing a calibration pattern. Errors in camera tracks, gyro and accelerometer measurements are optimised over. In the paper, the authors also investigate the use of their framework to estimate the relative pose, bias and the camera focal length. As the paper uses an expensive IMU with GPS-clock synchronisation, no time synchronisation is needed, and the rolling shutter readout is also pre-calibrated. Several of the limitations in [11] are addressed in [12], where an *Extended Kalman Filter* (EKF) is used to refine the cellphone-IMU calibration parameters. In addition to parameters used in [11], the radial and tangential distortion parameters of the (narrow angle) camera are also refined, as well as the time delay. EKF convergence is demonstrated when initialised with small errors in the state vector.

Most similar to our problem is [13]. Here a cellphone with a built-in gyroscope is calibrated using an EKF, and a novel rotation constraint for rolling shutter cameras. Rolling shutter readout time, relative pose, time delay and gyro bias are all optimised over, and the author has made his implementation available for download. Our tests of the author's implementation on the supplied data sequence, reveal that the method is very sensitive to initialisation. It diverges for small errors in relative pose, and time delay. Our method improves on this in that we provide a robust initialisation for both relative pose and time synchronisation. Another difference is that [13] also refine the linear intrinsics, f, c_x, and c_y, while we require that these, and the radial distortion are known beforehand. As [13] does not include radial distortion in their implementation it is unfortunately not possible to directly use their algorithm on sports cameras, but in the experiments, we test their residual in our batch optimisation framework.

The optimisation of the unknown parameters in a camera-IMU calibration is a non-convex problem, which requires an initialisation sufficiently close to the global minimum. The robust initialisation of the relative pose and the time scale and offset that we introduce would thus also allow the systems described in [13], [11], [12] to be used in more general conditions.

Initialisation of the relative pose and time synchronisation is also done in [14], but in a semi-automated fashion. Here a gyro sensor is attached to a Kinect sensor and used to rectify its depth maps. Sensor time synchronisation and relative pose are estimated in a semi-automated fashion, using generated rotations of the sensor package along two axes. The camera intrinsics, and the rolling shutter readout time are assumed to be known.

In summary, other authors have used optimisation to refine various subsets of the calibration parameters we estimate, while assuming other parameters to be known, or assuming specific recording conditions, see table I. However, none of the previously introduced methods include an automatic initialisation of relative pose and time synchronisation. Also, none of the previous methods have been shown to work on wide angle cameras. The proposed approach works also in cases when the sensors are not sharing the same clock, and with videos depicting generic scenes. This is not a trivial matter, as the use of natural landmarks instead of a calibration pattern requires a robust outlier rejection scheme to be embedded in the estimation.

method	readout	rel-pose	offset	sample rate	initialisation	generality
[10]	✔	N/A	N/A	N/A	✘	cpattern
[5]	✔	✘	✔	✘	✘	✔
[14]	✘	✔	✔	✔	✘	rotations
[11]	✘	✔	✘	✘	✘	cpattern
[13]	✔	✔	✔	✘	✘	✔
[12]	✔	✔	✔	✘	✘	✔
proposed	✘	✔	✔	✔	✔	✔

TABLE I: Related parameter estimation papers. Note that previous approaches do not perform automatic parameter initialisation. Note also that only [5], [13], [12] and the proposed method are free of specific patterns or motions and work under general conditions.

B. Structure

In section II we go through the background theory our method builds upon. Section III gives details on our optimisation framework, and section IV describes how to find good initial values for the optimisation. Finally section V describes our experiments, and section VI summarises the paper.

C. Notation

2D-vectors are written as lower-case, bold letters ($\mathbf{x}_k$), and 3D-vectors and matrices as upper-case bold letters ($\mathbf{X}_k$, $\mathbf{R}$).

We will often need to know if an entity belongs to the camera or gyroscope frame of reference. This will be marked with C or G as either super or sub script for that entity ($\mathbf{R}_G$, d_G, t^C) depending on which is most appropriate.

II. Theory

Our proposed method uses non-linear least squares optimisation to estimate the parameters. The aim of this section is to provide the required theory, while simultaneously introducing the cost functions used by the optimiser.

A. Rolling Shutter Geometry

Our rolling shutter camera to gyro calibration is based on feature tracking across short segments of video. For this we use the KLT tracker [15].

The tracker produces tracks on short intervals of video, which we call *slices*. A slice $S_{l,m}$ is computed from video frames $n \in [l, m] \subset \mathbb{N}$, and consists of K point tracks, $S_{l,m} = \{\mathcal{X}_k\}_{k=1}^{K}$. Each track $\mathcal{X}_k$ consists of a set of image

plane locations, $\mathcal{X}_k = \{\mathbf{x}_{k,n}\}_{n\in[l,m]}$, one for each of the video frames $n \in [l, m]$.

A successfully tracked 3D landmark $\mathbf{X}_k$ is related to the observed image points in track $\mathcal{X}_k$ as

$$\mathbf{x}_{k,n} \sim \mathbf{K} f(\mathbf{R}(t_{k,n})\mathbf{X}_k + \mathbf{p}(t_{k,n}), \mathbf{\Theta}) \,. \tag{1}$$

Here $\sim$ denotes equality after projection of the right operand, i.e. for vectors $\mathbf{x} \in \mathbb{R}^2$, $\mathbf{X} \in \mathbb{R}^3$ we have

$$\mathbf{x} \sim \mathbf{X} \quad \Leftrightarrow \quad \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} X_1/X_3 \\ X_2/X_3 \end{bmatrix} . \tag{2}$$

The function $f(\mathbf{X}, \mathbf{\Theta})$ in (1) is a lens-distortion function, operating on normalised image coordinates, using the parameter vector $\mathbf{\Theta}$. The matrix $\mathbf{K}$ is the internal camera calibration matrix. The camera orientation $\mathbf{R}$ and optical centre $\mathbf{p}$ are parameterised by a continuous time variable $t_{k,n}$, which corresponds to the time at which $\mathbf{x}_{k,n}$ was observed by the rolling shutter camera. As the readout is linear, the observation time is proportional to the image coordinate plus t_n, the frame start time. That is

$$t_{k,n} = t_n + r \cdot x_{k,n,\text{row}}/N_{\text{rows}} \,, \tag{3}$$

where r is the sensor readout time, $x_{k,n,\text{row}}$ is the image coordinate along the rolling shutter axis, and N_{rows} is the number of image lines along this axis.

As all rolling shutter rectification approaches neglect parallax effects, see e.g. [4], [16], we do likewise, and make the simplifying assumption that the optical centre $\mathbf{p}$ is stationary relative to the landmarks $\mathbf{X}_k$. This means that we can remove $\mathbf{p}$ from our equations, by choosing it as the origin of our 3D frame. In the experiments, we choose the lens model $f(\mathbf{X}, \mathbf{\Theta})$ as the three parameter FOV model which was introduced in [17].

B. Cost Function

Gyro based video rectification relies on the relative pose $\mathbf{R}_{CG}$ between camera and gyro being known, as well as the camera-gyro time delay, d_C, the gyro data rate, f_G, and the gyroscope bias $\mathbf{b}$. We propose to estimate these using non-linear batch optimisation where the following cost function is minimised

$$J(\mathbf{b}, f_G, d_C, \mathbf{R}_{CG}) = \mathbf{r}^T\mathbf{r} \,, \tag{4}$$

where $\mathbf{r}$ is a residual vector. The residual vector is constructed by stacking transfer errors based on individual correspondences. For a correspondence k, between frames l and m, the contribution consists of two errors with in total four elements, as we use a symmetric transfer error:

$$\mathbf{r} = \begin{bmatrix} \dots & \varepsilon_{k,l,m} & \varepsilon_{k,m,l} & \dots \end{bmatrix}^T . \tag{5}$$

The errors are defined as

$$\varepsilon_{k,l,m}^T = \mathbf{K} f(\mathbf{u}_{k,l}, \mathbf{\Theta}) - \mathbf{K} f(T_{l,m}(\mathbf{u}_{k,m}), \mathbf{\Theta}) \,. \tag{6}$$

Here $\mathbf{u}$ are normalized image coordinates, i.e.

$$\mathbf{u} = f^{-1}(\mathbf{K}^{-1}(\begin{smallmatrix}\mathbf{x}\\1\end{smallmatrix}), \mathbf{\Theta}) \;\text{ and }\; \mathbf{x} \sim \mathbf{K} f(\mathbf{u}, \mathbf{\Theta}) \,, \tag{7}$$

and $T_{l,m}()$ is the transfer function that transfers a point from frame m to frame l

$$T_{l,m}(\mathbf{u}_m) = \mathbf{R}(t_l)\mathbf{R}^T(t_m)\mathbf{u}_m \,. \tag{8}$$

As mentioned before, we assume that $\mathbf{K}$ and $\mathbf{\Theta}$ are known, and instead the sought parameters are to be found in the computation of the camera orientation trajectory $\mathbf{R}(t)$ from the gyro samples, as will be detailed in the next section.

Two more things about (8) should be mentioned before we proceed. As the camera centre $\mathbf{p}(t)$ from the projection equation (1) is absent from (8), parallax effects have been neglected. Points with high parallax are instead handled by our choice of error norm, see section III-B. Note also that the two terms in (5) correspond to transfer errors in images l and m respectively. This is thus the rolling shutter equivalent of the symmetric transfer error often used in homography estimation for global shutter cameras [18].

In [13], Jia and Evans propose a different residual than (6), based on a rolling shutter version of the epipolar plane normal coplanarity constraint [19]. For a triplet of correspondences this constraint contributes a single element to the residual vector [13]

$$\varepsilon_{1,2,3} = |\perp(\mathbf{u}_{1,l}, \mathbf{u}_{1,m}) \; \perp(\mathbf{u}_{2,l}, \mathbf{u}_{2,m}) \; \perp(\mathbf{u}_{3,l}, \mathbf{u}_{3,m})| \,, \tag{9}$$

where $|\cdot|$ is the matrix determinant, and $\perp(\mathbf{u}_l, \mathbf{u}_m)$ is a vector normal to the epipolar plane, formed by the correspondence and the two corresponding camera poses. It is computed as

$$\perp(\mathbf{u}_l, \mathbf{u}_m) = \mathbf{R}(t_l)^T\mathbf{u}_l \times \mathbf{R}(t_m)^T\mathbf{u}_m \,, \tag{10}$$

where $\times$ is the cross product operator.

Note that while (6) has four residuals for each correspondence, (9) has one residual for each group of three correspondences. We will compare these two residuals in the experiments.

C. Camera Orientation

One of the parameters to be estimated is the relative rotation between the camera and gyroscope, $\mathbf{R}_{CG}$. This rotation is defined by the relation

$$\mathbf{R}_C = \mathbf{R}_{CG}\mathbf{R}_G\mathbf{R}_{CG}^T \,. \tag{11}$$

where $\mathbf{R}_G$ and $\mathbf{R}_C$ is an orientation expressed in the reference frame of the gyroscope and camera, respectively. This can be realized by noting that $\mathbf{R}_G$ and $\mathbf{R}_C$ are operators that operate on points in the camera and gyro frames according to

$$\mathbf{p}'_c = \mathbf{R}_C\mathbf{p}_c \quad \text{and} \quad \mathbf{p}'_g = \mathbf{R}_G\mathbf{p}_g \,, \tag{12}$$

respectively. The transformation from gyro to camera allow us to relate the same points as

$$\mathbf{p}_c = \mathbf{R}_{CG}\mathbf{p}_g \quad \text{and} \quad \mathbf{p}'_c = \mathbf{R}_{CG}\mathbf{p}'_g \,, \tag{13}$$

which combined with (12) gives (11).

The relative orientation in (8) can thus be obtained as:

$$\mathbf{R}_C(t_{k,l})\mathbf{R}_C^T(t_{k,m}) = \mathbf{R}_{CG}\mathbf{R}_G(t_{k,l})\mathbf{R}_G^T(t_{k,m})\mathbf{R}_{CG}^T \,, \tag{14}$$

$$\text{or} \quad \Delta\mathbf{R}_C(t_{k,l}, t_{k,m}) = \mathbf{R}_{CG}\Delta\mathbf{R}_G(t_{k,l}, t_{k,m})\mathbf{R}_{CG}^T \,. \tag{15}$$

The orientation $\mathbf{R}_G(t)$ is in practise obtained by SO(3) integration of the gyro signal $\boldsymbol{\omega}_{\text{adj}}(t)$, using the unit quaternion integration method described in [20]. Before integration, the gyro signal has been adjusted by a bias correction according to

$$\boldsymbol{\omega}_{\text{adj}}(t) = \boldsymbol{\omega}(t) - \mathbf{b}\,, \tag{16}$$

where $\mathbf{b}$ is a three element vector to be estimated.

D. Camera Time

In the previous section we saw how $\Delta\mathbf{R}_G(t_{k,l}, t_{k,m})$ is converted to the camera frame. However, the time index of this sequence is still expressed in camera time frame. For unsynchronised clocks, conversion between time frames can be done using a scaling and an offset. When indexing the gyro sequence it is convenient to express this conversion with camera time t^C in seconds, and gyro time t^G in samples. This gives us the relation

$$t^G = f_G(t^C + d_C)\,, \tag{17}$$

where d_C is an offset in seconds and f_G is the gyro sample rate in Hz. By combining (17) with (3), the observation time of a particular image point $\mathbf{x}_{k,n}$ can be expressed in the gyro time frame as

$$t^G_{k,n} = f_G\left(t^C_n + r \cdot x_{k,n,\text{row}}/N_{\text{rows}} + d_C\right)\,. \tag{18}$$

As before r is the camera readout time in seconds, and t^C_n is the camera frame start time. The start time is computed from the frame index, n, as $t^C_n = n/f_C$ where f_C is the camera frame rate.

Note that the camera frame rate f_C is assumed to be known. The effect of this is just a matter of choosing a unit of time. Thus choosing it wrongly does not affect the performance, but it does mean that the found gyro sample rate f_G and offset d_C will be expressed relative to the chosen camera frame rate. However, it is important that the readout time r is calibrated against the chosen f_C.

E. Video Stabilisation

For video stabilisation we use the method described in [3], modified to use the FOV distortion model in [17].

A standard deviation of 20 was used for the Gaussian smoothing in the experiments. This value provided a good trade-off between a smooth camera path while still being able to handle large motions. See figure 1 and the dataset webpage [7] for sample output.

III. Optimisation

We have now defined all the parameters that need to be calibrated for, and as a summary we list them here again, and also count their degrees of freedom (DOF):

- The time scaling and offset, f_G, d_C, 2 DOF.
- The gyro to camera transformation $\mathbf{R}_{CG}$, parameterised as the axis-angle vector $\mathbf{r} = \alpha\hat{\mathbf{n}} \in \mathbb{R}^3$, 3 DOF.
- The gyro bias $\mathbf{b}$, 3 DOF.

In addition to these free parameters we also need to to know the following fixed parameters:

- Rolling shutter readout time, r.
- Internal camera calibration matrix, $\mathbf{K}$.
- Lens distortion parameters, $\boldsymbol{\Theta}$.

Subsets of these fixed parameters can be included in the optimisation, as was done in e.g. [5] for the readout and the focal length in $\mathbf{K}$. However, we have found that adding them will reduce the accuracy of the other parameters, and including all the fixed parameters results in a system with no unique solution.

We thus have in total, 8 DOF to determine by minimising the cost function (4). This optimisation problem has many local minima, and an appropriate initialisation of the optimiser is thus crucial for success.

A. Selecting Correspondences from the Video

As a recorded sequence may vary substantially in length, calibration of long sequences would be infeasible if all possible correspondences were used. In order to keep the computation time down we choose to use only parts of the data.

We divide the video into a large number of short frame intervals called slices, see section II. The slices are chosen randomly over the entire video sequence. Each slice has a random length (2-15 frames) and are spaced randomly from each other (2-15 frames).

To improve the quality of the tracks, we perform track-retrack [21], i.e. we track both forwards and backwards in the slices, and only keep those tracks that were successfully retracked to within 0.5 pixels of their initial positions.

We use the start and end point in each track to generate correspondences.

B. Robust Cost Function

While the used track-retrack scheme (see section III-A) will make sure that all the correspondences we have are stable, it does not mean that they belong to the same geometrical object, or satisfy the low-parallax assumption (see section II-B). Thus, our set of correspondences is likely to contain outliers.

In the global shutter case, outliers can be removed by e.g. RANSAC, on a frame global motion model, such as a homography or a fundamental matrix [18]. But this is not possible with a rolling shutter. Instead we will optionally replace the quadratic cost function (4) with a robust cost [22]

$$J(\mathbf{b}, f_G, d_C, \mathbf{R}_{CG}) = \psi(\mathbf{r})^T\psi(\mathbf{r})\,, \tag{19}$$

$$\text{where we use}\quad \psi(r_i) = \frac{r_i}{1 + |r_i|/c}\,. \tag{20}$$

Here r_i are individual elements in the residual vector, and c is a design parameter that can be used to scale the function. This results in an error norm similar to German-McClure [22], but with an additional scale parameter.

In the experiments we use the scale $c = 3$ for the residuals in (6), and $c = 10^{-5}$ for the constraint in (9), as they have a different magnitude.

C. Local Minimiser

The optimisation of the cost function can be done using local minimisers such as Gauss-Newton, Levenberg-Marquardt, and DogLeg [23]. We will use the minimiser `scipy.optimize.leastsq` in SciPy (version 0.14.0) [24] which uses the Levenberg-Marquardt algorithm. The minimiser is either fed the residual vector $\mathbf{r}$ directly, or if a robust norm is desired, the residual vector after applying (20).

IV. Initialisation

As described in section III-C we refine the calibration parameters using a local minimiser on a non-linear least-squares cost function. All local minimisers require a starting point sufficiently close to the global minimum. How to find this starting point is the topic of this section.

A. Gyro Sample Rate

The data sheet for the gyroscope should list its available output data rates. However, this value can be offset by a few percent. In our case the error with respect to the data sheet is 6%, giving a maximum rate of approximately 855 Hz instead of the listed 800 Hz.

By instead using timing information from the microcontroller which logs the gyroscope samples, we can get a more accurate estimate of f_G. A conservatve assumption is that the controller clock has an accuracy of 0.5% or better, which in our case translates to 4 Hz.

B. Time Offset

A classical approach to finding a time offset is to use signal correlation, and for camera to IMU calibration, correlation of gyro rates and estimated relative camera rotations have proven to be a robust approach [25]. Another option is to integrate the relative orientations and then use spatio-temporal ICP for alignment [26]. In the rolling shutter case, relative camera orientations are non-trivial to find, and a way to avoid estimating them is to instead use the optical flow magnitude, as proposed in [14]. We improve on this here, by adding a coarse to fine search, which speeds up the search by orders of magnitude.

In order to find the offset d_C we will search for the maximum correlation between the optical flow magnitude, $F(t)$, and the gyro magnitude $G(t) = \|\boldsymbol{\omega}(t)\|$. The optical flow magnitude, $F_n = F(t_n)$, at frame n is the mean pixel distance of a number of points that have been tracked from frame n to frame $n+1$.

However, in [14] the two logs were started by the same program, and thus a small chunk of the two signals could be extracted that was known to contain a generated movement. Here we only assume two streams of data, and thus need to correlate the entire sequences. In order to make this tractable, we make use of a coarse to fine approach.

First, we resample the flow magnitude $F(t)$, using an upsampling factor of f_G/f_C and linear interpolation, i.e.:

$$F_n = (1-w)F(k) + wF(k+1) \quad \text{where} \tag{21}$$

$$k = \lfloor nf_C/f_G \rfloor \quad \text{and} \quad w = nf_C/f_G - k\,. \tag{22}$$

We then successively subsample both F_n and G_n in octaves, by binning neighbouring samples, and find the offset τ in the coarsest scale as the shift with the highest normalised cross correlation. This shift is then refined, by trying neighbouring shifts in successively finer scales. Once τ at the finest scale has been found, we can compute a guess for the offset as $d_C = \tau/f_C$.

C. Gyro to Camera Transformation

With approximations of the time sync parameters, f_G and d_C, we can make an initial estimate of the gyro to camera transformation. This is possible since we now have a rough idea which gyroscope samples belong to which frames.

Using the generated set of slices, we can create corresponding pairs of rotation axes: one computed in the camera reference frame, and one in the gyro reference frame. The rotation axis for a slice $S_{l,m}$ in the gyroscope reference frame, $\hat{\mathbf{n}}^G_{l,m}$ is found as the rotation axis of the integrated relative orientation from middle row times of frame l and m.

To find the corresponding rotation axis in the camera reference frame, $\hat{\mathbf{n}}^C_{l,m}$, we use the point correspondences between the first and the last frame in $S_{l,m}$. First, the relative rotation is estimated using RANSAC on a pure rotation constraint on the normalised correspondences $\mathbf{u}_l = \mathbf{R}_{lm}\mathbf{u}_m$ (see (7)). RANSAC finds a solution to $\mathbf{R}_{lm}$, by successively drawing minimal samples of two correspondences [27], and finding candidate rotations with the Orthogonal Procrustes Problem (OPP) [28]. OPP requires three corresponding points, but the third can be generated using the cross-product on the first two points. Finally, the axis $\hat{\mathbf{n}}^C_{l,m}$ is extracted from $\mathbf{R}_{lm}$.

From the set of slices, we have now computed a set of rotation axes pairs $\hat{\mathbf{n}}^C_i \leftrightarrow \hat{\mathbf{n}}^G_i$, and using these we estimate the final gyro-to-camera transformation using RANSAC. In each iteration a candidate transformation is estimated from two rotation axes pairs using OPP, as $\hat{\mathbf{n}}^C_i = \mathbf{R}_{CG}\hat{\mathbf{n}}^G_i$. For evaluation we use the angle difference, θ_i, between the estimated camera rotation axis and the transformed gyroscope rotation axis:

$$\theta_i = \arccos((\hat{\mathbf{n}}^C_i)^T \mathbf{R}_{CG}\hat{\mathbf{n}}^G_i) \tag{23}$$

D. Gyro bias

The gyro bias is initialised with a zero vector $\mathbf{b} = \mathbf{0}$.

V. Experiments

The following experiments were carried out to test our method: 1) stability of parameter estimation when varying the number of correspondences, 2) stability of parameter estimation when varying the gyroscope sample rate, 3) sensitivity to errors in the initial values, 4) stability of parameter estimation depending on choice of residual function and robust error norm.

In the absence of ground truth data we will use a set of *reference parameters*. For each test sequence we performed optimisation using several different slice sets, and the optimised parameters were then used to produce a stabilised video. We carefully examined all the stabilised videos, and

Fig. 2: The three sequences used in the experiments. Top to bottom: **rotate**, **walk**, and **RC-car** sequences. Rightmost column shows normalised DFT plots of the corresponding gyro sequences (amplitude in radians as a function of frequency in Hz).

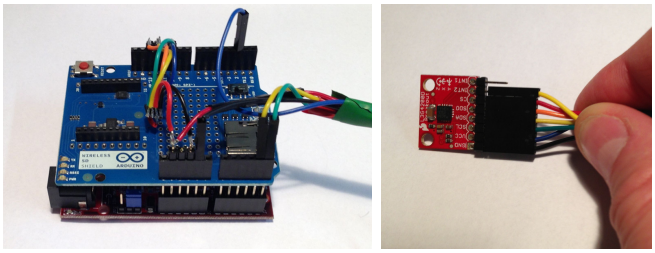

Fig. 3: Sensor logging plattform. Left: Uno32 Arduino-compatible board with flash memory and SD card reader. Right: L3G4200D triple-axis MEMS gyro.

the parameters that produced the best stabilised video were then chosen as reference for that test sequence. We argue that parameters that stabilises the video with good visual results should be close to the true parameters.

A. Sensor Logging Platform

Our gyroscope sample logs are recorded using an L3G4200D three-axis MEMS gyroscope from STMicroelectronics, which we attach rigidly to the video camera before recording. Our test sequences were captured using a gyroscope sample rate of approximately 855 Hz. The sensor platform is pictured in figure 3.

B. Test Sequences

In the experiments, we have used a GoPro HERO3+ Black Edition camera. The camera was set to record with HD resolution (1920×1080), at 29.97 fps. We calibrated the rolling shutter readout using the approach described in [3]. For the camera geometry calibration we used the checkerboard approach of Zhang's [29], with radial distortion modeled using the FOV model [17].

We use three video sequences with increasing level of difficulty: (1) **rotate**, a sequence recorded with the camera hand-held, and rotated roughly about its centre, (2) **walk**, a walking sequence where the camera has been pointed at various targets while walking, (3) **RC-car**, a sequence recorded with the camera mounted on a radio controlled (RC) model car, driving in rough terrain. Sample frames from the three sequences are shown in figure 2, together with the DFT of the gyro signal. As can be seen in the DFT-plots, the amplitude of the rotation is the highest in **walk**, followed by **rotate** and **RC-car**. However, what really makes a sequence challenging is the frequency content, and as can be seen, the **RC-car** sequence contains much more high frequency content.

C. Experiment 1: Varying number of tracks

The number of correspondences used by the optimiser have a direct influence on processing time, but should also affect the quality of the parameter estimate.

We examined how the parameter stability changed when using 400, 800, 1500, 3000, or 6000 correspondences. What we found is that there is a weak tendency of higher stability by using a higher number of correspondences, but there does not seem to be any gain in using more than 1500.

D. Experiment 2: Gyro sample rate

To examine how a lower sample rate affects the accuracy of the parameters we artificially reduced the sample rate of our gyroscope by subsampling the logged gyroscope signal with factors 2, 4, and 8. For each test sequence and subsample factor, five optimisations were made using different sets of slices.

Stability is examined by looking at the distribution of errors in the parameters, relative to the reference parameters. We show the error for the time parameters and relative pose. For the relative pose we show the absolute angular difference between the estimate and the reference. Bias is omitted since

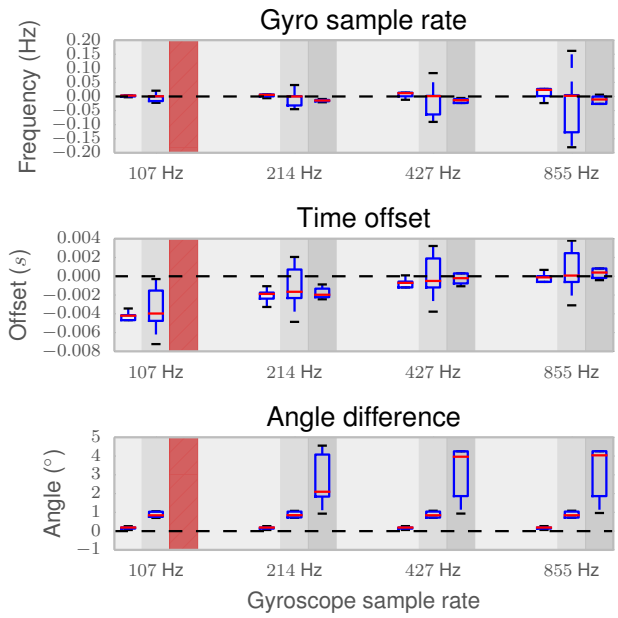

Fig. 4: Parameter convergence as a function of subsample rate on the gyroscope sequence which was used for the optimisation. For each interval, from left to right, with decreasing background brightness, is the three sequences: rotation, walk, and RC-car. The error is relative to the set of reference parameters for each sequence.

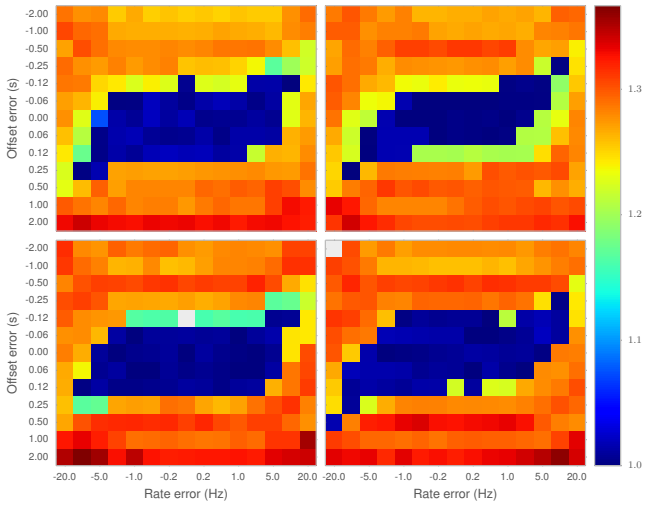

Fig. 5: Residuals after convergence for four different slice sets, as a function of an error in the initial value for the time parameters. The residual is normalised such that the minimum value is 1.

its influence is small compared to the other parameters, and also because it depends on the estimated relative pose.

As we can see in figure 4 the subsampling of the gyroscope data in general does not affect the result. The most notable exception is that the RC-car sequence failed when the subsample factor is 8 (107 Hz). At this sample rate, the initial time offset estimation fails.

E. Experiment 3: Sensitivity to Initialisation

Good initial values are important to make sure the optimisation converges to a good solution. In this experiment we examine the sensitivity to errors in the initialisation of the time parameters.

As a measure of convergence we chose to look at the norm of the residual vector, normalised by number of elements.

Figure 5 shows the normalised residual for four different sets of slices generated from the **walk** sequence. All four trials show similar basins of attractions, with convergence for errors in the gyro rate and time offset within ± 5 Hz and ± 0.1 seconds respectively.

The error for most microcontroller clocks fall well within this interval for the sample rate. The error for correlation-based estimation of the offset is less than a frame, which corresponds to 0.033 seconds in our case.

F. Experiment 4: Choice of Residual Function and Effect of Robust Norm

In section III-B we argued that using a robust error norm is important to mitigate the effect of outliers in the correspondences, and in section II-B we described two different residual functions. Figure 6 shows the parameter stability for each choice of residual function, and with robust error norm turned on or off. Like in the previous experiments we did five optimisations per sequence.

As expected, there is a strong case for using a robust error norm as the stability is greatly improved regardless of the residual function that was used.

Comparing the two residual functions we can see that our proposed residual results in more stable estimates than the one of Jia and Evans [13]. It should however be noted that while our original residuals have a clear geometric meaning that is independent of the correspondence, the residual function in (9) will change size depending on the correspondences chosen for a triplet. This means that choosing a constant value c for the scale of the robust error norm in (20) is much more difficult.

VI. Conclusions

We conclude that our initialisation scheme results in a starting point that is well within the basin of attraction for the cost function. We can also see that the use of a robust error norm is a critical component to obtain an accurate calibration. When comparing the Jia and Evans constraint (9) and the symmetric transfer constraint (6) we observe consistently better accuracy for the latter.

Our method currently requires known camera and lens distortion parameters, as well as readout time. It would obviously be useful if these parameters could also be included in the optimisation. Camera and lens distortion should be possible to include if a good enough initial estimate can be provided. The readout time, however, is problematic since it is coupled with the other time parameters. If it is also optimised for, the stability of the found solution is degraded. Thus we recommend that it is calibrated separately.

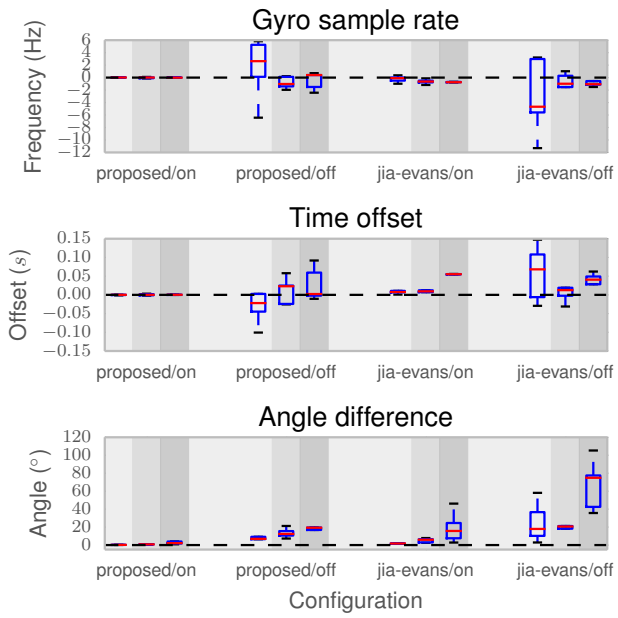

Fig. 6: Parameter convergence for different choices of residual function, and with robust error norm on and off. For each interval, from left to right, with decreasing background brightness, is the three sequences: rotation, walk, and RC-car. The error is relative to the set of reference parameters for each sequence.

Our method works well and reliably on both the **walk** and **rotation** test sequences. By well we mean that, when using reasonable conditions for the optimiser, we have so far always succeeded in creating a nicely stabilised video. For the **RC-car** sequence the method also succeeds in most cases, but we have occasionally observed cases when the output video is not stabilised correctly. Our hypothesis is that this is due to the much higher frequency content in the video, and that the random slice creation sometimes fails to generate sufficiently informative data. In the presence of high frequency motion even a very small error in the estimated sampling rate or time offset can cause negative interference due to phase errors. This results in large visual errors. A deterministic way to generate slices could help to avoid this issue, and is something we intend to investigate in future work.

When looking at resultant videos, 3D-structures look much more rigid than in the input (see dataset webpage [7]). It would be interesting to see whether structure-from-motion accuracy on GoPro video improves if the proposed approach is used.

References

[1] A. E. Gamal and H. Eltoukhy, "CMOS image sensors," *IEEE Circuits and Devices Magazine*, May/June 2005.

[2] O. Saurer, K. Köser, J.-Y. Bouguet, and M. Pollefeys, "Rolling shutter stereo," in *ICCV'13*, 2013.

[3] E. Ringaby and P.-E. Forssén, "Efficient video rectification and stabilisation for cell-phones," *International Journal of Computer Vision*, vol. 96, no. 3, pp. 335–352, February 2012.

[4] S. Baker, E. Bennett, S. B. Kang, and R. Szeliski, "Removing rolling shutter wobble," in *CVPR10*, IEEE Computer Society. San Francisco, USA: IEEE, June 2010.

[5] A. Karpenko, D. Jacobs, J. Baek, and M. Levoy, "Digital video stabilization and rolling shutter correction using gyroscopes," Stanford University Computer Science, Tech. Rep. CSTR 2011-03, September 2011.

[6] J. Kopf, M. F. Cohen, and R. Szeliski, "First-person hyper-lapse videos," in *SIGGRAPH Conference Proceedings*, 2014.

[7] H. Ovrén and P.-E. Forssén, "Gopro-gyro dataset," http://www.cvl.isy.liu.se/research/datasets/gopro-gyro-dataset/, February 2015.

[8] J. D. Hol, T. B. Schön, and F. Gustafsson, "Modeling and calibration of inertial and vision sensors," *International Journal of Robotics Research*, vol. 29, no. 2, pp. 231–244, February 2010.

[9] C. Geyer, M. Meingast, and S. Sastry, "Geometric models of rolling-shutter cameras," in *6th OmniVis WS*, 2005.

[10] L. Oth, P. Furgale, L. Kneip, and R. Siegwart, "Rolling shutter camera calibration," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR13)*, Portland, Oregon, June 2013, pp. 1360–1367.

[11] S. Lovegrove, A. Patron-Perez, and G. Sibely, "Spline fusion: A continuous-time representation for visual-inertial fusion with application to rolling shutter cameras," in *British Machine Vision Conference (BMVC)*. BMVA, September 2013.

[12] M. Li, X. Zheng, H. Yu, and A. I. Mourikis, "High-fidelity sensor modeling and online calibration in vision-aided inertial navigation," in *ICRA'14*, 2014.

[13] C. Jia and B. L. Evans, "Online calibration and synchronization of cellphone camera and gyroscope," in *IEEE Global Conference on Signal and Information Processing (GlobalSIP)*, December 2013.

[14] H. Ovrén, P.-E. Forssén, and D. Törnqvist, "Why would I want a gyroscope on my RGB-D sensor?" in *Proceedings of IEEE Winter Vision Meetings, Workshop on Robot Vision (WoRV13)*. Clearwater, FL, USA: IEEE, January 2013.

[15] J. Shi and C. Tomasi, "Good features to track," in *IEEE Conference on Computer Vision and Pattern Recognition, CVPR'94*, Seattle, June 1994.

[16] P.-E. Forssén and E. Ringaby, "Rectifying rolling shutter video from hand-held devices," in *CVPR'10*, 2010.

[17] F. Devernay and O. Faugeras, "Straight lines have to be straight: Automatic calibration and removal of distortion from scenes of structured environments," *Machine Vision and Applications*, vol. 13, no. 1, pp. 14–24, 2001.

[18] R. I. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2004.

[19] L. Kneip, R. Siegwart, and M. Pollefeys, "Finding the exact rotation between two images independently of the translation," in *European Conference on Computer Vision ECCV12*, 2012.

[20] D. Törnqvist, "Estimation and detection with applications to navigation," Ph.D. dissertation, Linköping University, 2008.

[21] J. Hedborg, P.-E. Forssén, and M. Felsberg, "Fast and accurate structure and motion estimation," in *ISVC09*, ser. Lecture Notes in Computer Science, vol. 5875, November 2009, pp. 211–222.

[22] Z. Zhang, "Parameter estimation techniques: A tutorial with application to conic fitting," *Journal of Image and Vision Computing*, vol. 15, no. 1, pp. 59–76, 1997.

[23] K. Madsen, H. B. Nielsen, and O. Tingleff, "Methods for non-linear least squares problems, 2nd ed." Technical University of Denmark, Tech. Rep., April 2004.

[24] E. Jones, T. Oliphant, P. Peterson, *et al.*, "SciPy: Open source scientific tools for Python," http://www.scipy.org/, 2001–.

[25] E. Mair, M. Fleps, M. Suppa, and D. Burschka, "Spatio-temporal initialisation for IMU to camera registration," in *IEEE Int. Conf. Robot. Biomimetics*, 2011.

[26] J. Kelly and G. S. Sukhatme, "A general framework for temporal calibration of multiple proprioceptive and exteroceptive sensors," in *ISER10*, 2010.

[27] M. Brown, R. Hartley, and D. Nistér, "Minimal solutions for panoramic stitching," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR07)*, 2007.

[28] G. H. Golub and C. F. van Loan, *Matrix Computations*. Baltimore, Maryland: Johns Hopkins University Press, 1983.

[29] Z. Zhang, "A flexible new technique for camera calibration," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 11, pp. 1330–1334, 2000.