Self-calibration of omnidirectional multi-cameras including synchronization and rolling shutter

Thanh-Tin Nguyen, Maxime Lhuillier

Self-calibration of omnidirectional multi-cameras including synchronization and rolling shutter

Thanh-Tin Nguyen and Maxime Lhuillier

Institut Pascal, UMR 6602 - CNRS/UCA/SIGMA, 63178 Aubière, France

The reference of this paper is: Thanh-Tin Nguyen and Maxime Lhuillier, Self-calibration of omnidirectional multi-cameras including synchronization and rolling shutter, Computer Vision and Image Understanding, 162:166-184, 2017.

This is the accepted manuscript version that is available at the webpage of the author. The published version (DOI: 10.1016/j.cviu.2017.08.010) is available at Elsevier via `https://doi.org/10.1016/j.cviu.2017.08.010`

Highlights

- Deal with consumer 360 cameras and spherical cameras without a privileged direction.
- Initialize the time offsets and intrinsic parameters using monocular structure-from-motion.
- Start multi-camera structure-from-motion with central and global shutter assumptions.
- Refine all parameters including time offsets and line delay thanks to a bundle adjustment.
- Experiment on long video sequences using helmet-held multi-cameras.

Abstract

360 degree and spherical cameras become popular and are convenient for applications like immersive videos. They are often built by fixing together several fisheye cameras pointing in different directions. However their complete self-calibration is not easy since the consumer fisheyes are rolling shutter cameras which can be unsynchronized. Our approach does not require a calibration pattern. First the multi-camera model is initialized thanks to assumptions that are suitable to an omnidirectional camera without a privileged direction: the cameras have the same setting and are roughly equiangular. Second a frame-accurate synchronization is estimated from the instantaneous angular velocities of each camera provided by monocular structure-from-motion. Third both inter-camera poses and intrinsic parameters are refined using multi-camera structure-from-motion and bundle adjustment. Last we introduce a bundle adjustment that estimates not only the usual parameters but also a sub-frame-accurate synchronization and the rolling shutter. We experiment using videos taken by consumer cameras mounted on a helmet and moving along trajectories of several hundreds of meters or kilometers, and compare our results to ground truth.

Keywords: Bundle adjustment, self-calibration, synchronization, rolling shutter, multi-camera, structure-from-motion.

1. Introduction

Multi-cameras built by fixing together several consumer cameras become popular thanks to their prices, high resolutions, growing applications including 360 videos (e.g. in YouTube), generation of virtual reality content [1, 2], 3D scene modeling [3]. However such a multi-camera has drawbacks.

First the synchronization of the videos can be a problem. In many cases like GoPro cameras [4], the manufacturer provides a wifi-based synchronization (the user starts all videos at once by a single click). However the resulting time offsets between videos are too inaccurate for applications: about 0.04s and sometimes above 0.1s in our experiments. Assume that a central multi-camera moves at 20km/h (e.g. biking in a city) and two cameras have a time offset equal to only 0.02s, then explain consequences on a 360 video obtained by video stitching. If we neglect this offset, the two videos are stitched as if

they have same camera centers at same frame number, although the distance between these centers is 0.11m (20/3.6*0.02). This generates artifacts in the 360 video due to foreground objects that are in the field-of-view (FoV) shared by the two cameras.

Secondly, the low price of a consumer camera implies that the camera is rolling shutter (RS). This means that two different lines of pixels of a frame are acquired at different instants. In a global shutter (GS) camera, all pixels have the same time. If we do a GS approximation of a RS camera, we assume that the camera poses are the same for all lines of a frame although they are not. This can degrade the quality of results in applications such as 3D reconstruction [5], similarly as an inaccurate synchronization degrades the quality of 360 videos.

Last the multi-camera is non-central, i.e. the baseline defined by the distance between the centers of two cameras is not zero. This is inadequate for applications that needs a central multi-camera such as 360 video (the smaller the baseline, the better the stitching quality). The user/manufacturer can reduce the baseline (and the multi-camera price) thanks to a small number of cameras. Here we use a DIY multi-camera composed of four Gopro Hero 3 enclosed in a cardboard such that the baseline is as small as possible. Since a small number of cameras also reduces the FoV shared by adjacent cameras, we avoid methods that rely on this shared FoV such as image matching between different monocular videos. A greater number of cameras can be used [6] to increase the shared FoV, but both price and baseline increase. We also experiment using a spherical camera [7] having only two large FoV (fisheye-like) images.

Our self-calibration takes into account these drawbacks (lack of synchronization, rolling shutter, almost central multi-camera) and does not require a calibration pattern. First the multi-camera model is initialized thanks to assumptions that are suitable to an omnidirectional camera without a privileged direction: the cameras have the same setting (frequency, image resolution, FoV) and are roughly equiangular. Second a frame-accurate synchronization is estimated from the instantaneous angular velocities of each camera provided by monocular structure-from-motion. Third both inter-camera poses and intrinsic parameters are refined using multi-camera structure-from-motion and bundle adjustment. Last we introduce a bundle adjustment that estimates not only the usual parameters but also the sub-frame-accurate synchronization and the rolling shutter. We experiment using videos taken by multi-cameras mounted on a helmet and moving along trajectories of several hundreds of meters or kilometer, then compare our self-calibration results with ground truth.

Sec. 2 briefly overviews previous work for each step of our method and presents our contributions. Several abbreviations are used in the paper: BA (bundle adjustment), SfM (structure-from-motion), GS (global shutter), RS (rolling shutter), FA (frame-accurate), SFA (sub-frame-accurate), IAV (instantaneous angular velocity), FoV (field-of-view), FpS (frame-per-second).

2. Previous work

2.1. Initializing the intrinsic parameters

The intrinsic parameters of a monocular perspective camera can be estimated without a calibration pattern using three steps [8]: projective reconstruction from the given images, self-calibration assuming that pixels are squares, and refinement using BA [9]. If the camera is an axially symmetric fisheye with an approximately known FoV angle, two radial distortion parameters can also be estimated [10] (this extends the one radial parameter case [11]). The initialization of these intrinsic parameters is not the paper topic. Here we initialize them assuming that the monocular cameras are roughly equiangular with an approximately known FoV. This is sufficient to experiment our contribution (synchronization and bundle adjustment) and we expect that a method like [10] improves the results.

2.2. Initializing the time offsets

Audio-based synchronization is possible if a distinct sound is available (e.g. a clap) and if the cameras do not have audio/video synchronization issues [1]. A survey of methods for video-based synchronization can be found in [12], but these require inter-camera matching or shared FoV or are designed for non-jointly moving cameras. In our case, we benefit by the assumption of jointly moving cameras but the shared FoV can be too small to automatically obtain a decent matching between two cameras. In [13], transformations are estimated between consecutive frames of every video instead of trying to match different videos. The estimated offset is the one that best "compares" the transformations between two videos. One intuitive example is the translation magnitude that is estimated from tracked features: the larger the translation in one video, the larger the translation in the other. However the transformations in [13] are heuristic (translation) or uncalibrated (homography/fundamental matrix) without radial distortion. Here we propose to compare the IAV estimated by a monocular SfM (Sec. 2.1), which does not have the above inconveniences.

2.3. Initializing the inter-camera poses

Once a 3D reconstruction is obtained for every camera (Sec. 2.1) and the time offsets are known (Sec. 2.2), the reconstructions are registered in the same coordinate system. In [14], a similarity transformation is robustly estimated between two reconstructions using a 3-point RANSAC algorithm and image matching for 3D points in different reconstructions. In [15], the relative pose between two cameras is directly estimated from the pose sequences of their two reconstructions (if the camera motion is not a pure translation). Averaging rotation (e.g. [16]) can also be used if there is a non constant relative pose between two reconstructions due to the drift of reconstruction(s). The initialization of the inter-camera poses is not the paper topic. Here we initialize them assuming that the multi-camera is roughly central with approximately known inter-camera poses: n cameras that are symmetrically mounted around a symmetry axis. This is enough to feed the bundle adjustment in our cases and [15, 14] can solve this step in all cases.

2.4. Global shutter multi-camera bundle adjustments

The BA in [17] refines the relative poses between the cameras in addition to the usual parameters (poses of the multi-camera and 3D points) by minimizing a reprojection error. However the reprojection error is in the undistorted space of the classical polynomial distortion model [18]. This is due to the fact that the forward-projection of this camera model does not have a closed-form. The BA in [19] deals with points at infinity, uses ray directions as observations, and transfers the uncertainty from the measure image space to the ray space. The refinement of intrinsic parameters is left as future work in [14, 17, 19].

Our multi-camera BA also refines intrinsic parameters (not only inter-camera poses and the other 3D parameters) and minimizes the reprojection error in the right space: the distorted space where the image points are detected. Under the standard assumption that the image noise due to point detection follows zero-mean normalized identical and independent Gaussian vectors, our BA is the Maximum Likelihood Estimator (this assumption is not true in the undistorted space, especially in case of large distortions between undistorted and distorted spaces).

2.5. Rolling shutter bundle adjustments

Previous monocular BAs estimate the RS assuming that the 3D points are known in a calibration pattern [20] or enforce a known RS coefficient [21, 22]. In the context of visual SLAM [23], GS BA is applied to RS (monocular) camera thanks to RS compensation: this method corrects beforehand the RS effects on the feature tracks by estimating instantaneous velocities of the camera. The previous multi-camera BAs estimate neither synchronization nor RS; only [24, 25] deal with known RS but need other sensors.

Every RS BA has a model of the camera trajectory, which provides the camera pose at each instant corresponding to each line of a frame, and which should have a moderated number of parameters to be estimated. In [21], one pose is estimated at each frame by BA and the poses between two consecutive frames are interpolated from the poses of these two frames. The BA in [22] adds extra parameters to avoid this linear interpolation assumption: it not only optimizes a pose but also rotational and translation speeds at every keyframe. In [20], a continuous-time trajectory model is used using B-splines and the BA optimizes the knots of the splines. The method chooses the number of knots and initializes their distribution along the trajectory sequence. In [24], the relative pose between an inter-frame pose and an optimized frame pose is provided by IMU at high frequency. In [25], rotational and translation speeds are also estimated at every frame (the FpS is only 4Hz) and the BA enforces a relative pose constraint using GPS/INS data. The visual-only RS approaches [23, 21, 22, 20] are experimented on few meters long camera trajectories. Our approach is also visual-only and deals with quite longer trajectories (hundreds of meters, kilometers) since it only estimates poses at keyframes.

2.6. Self-calibration and synchronization of sensors

In the context of a general multi-sensor, [26] simultaneously estimates the temporal and spatial registrations between sensors. In the experiments, the multi-sensor is composed of a camera and IMU. The best accuracy is obtained thanks to the use of all measurements at once, a continuous-time representation (a B-spline for IMU poses) and maximum likelihood estimation of the parameters (time offset, transformation between IMU and camera, IMU poses, and others). In [27], a camera-inertial multi-sensor is self-calibrated (synchronization, spatial registration, intrinsic parameters) by a sliding window visual odometry. Thanks to an adequate continuous-time motion parametrization, it also deals with RS cameras and has a better parametrization of the rotations. Indeed, it avoids the singularities of the global and minimal parametrization of rotations (e.g. in [26]), but assumes that the time between consecutive keyframes is uniform. Our work introduces a global minimal rotation parametrization and deals with non-uniform distribution of keyframes provided by standard SfM [28].

Recently, [29] synchronizes and self-calibrates consumer cameras using BA in a different context: assumptions are removed (rigidity on both multi-camera and scene), others are added (FoV shared by cameras, physics-based motion priors for moving objects), and the rolling shutter is not estimated.

2.7. Our contributions

Our multi-camera BA estimates the SFA synchronization and the line delay coefficient of the RS. Furthermore, this is done over long video datasets without additional sensors (hundreds of meters or kilometers). The previous work do not do this. In contrast to [17], our BA also estimates the intrinsic parameters and minimizes the reprojection errors in the original image space (not the rectified one) with the same polynomial distortion model [18, 30]. Another contribution is the FA synchronization that deals with cameras with small/empty shared FoV. As mentioned in Secs. 2.1 and Sec. 2.3, our initialization does not intend to compete with the accuracy and generality of previous initializations of intrinsic parameters and relative poses.

Contributions over our previous conference work [3, 31] are the following: check approximations [24] in the computation of the image projection that takes into account RS and synchronization, refine simultaneously calibration and synchronization/RS, deal with spherical cameras, more details on synchronization and BA (rotation parametrization, sparsity of solved system, derivatives of implicit reprojections errors). There are also new experiments on synchronization (comparison with ground truth, robustness of SFA refinement with respect to bad FA initialization), stability of both SFA synchronization and RS over time in long videos and with respect to keyframe sampling.

3. Overview of our algorithm

First the monocular camera model (we experiment the classical polynomial distortion model [30, 18, 17] and the unified camera model [32]) is initialized in Sec. 4 assuming that the fisheyes are roughly equiangular and using an approximate knowledge of their FoV angle.

Second we apply monocular SfM [28] and calibration refinement by BA for every camera. However SfM can fail for a video due to the combination of two difficulties: lack of texture and

approximate calibration. We assume that there is at least one textured enough video such that this is successful. Since the cameras have the same setting, we benefit by the refined intrinsic parameters by BA to redo the monocular SfM of the other videos. Thus a difficulty (approximate calibration) is reduced for the less textured videos and the risk of failure decreases.

Third the FA synchronization between all videos is obtained by using the method in Sec. 5. We skip few frames in each video such that the sequels of the videos are FA synchronized: from now frames with the same index are taken at the same time up to the inverse of the FpS.

Fourth a central multi-camera calibration is initialized from the estimated intrinsic monocular parameters and approximate inter-camera rotations (Sec. 2.3).

Fifth we apply multi-camera SfM [28] followed by multi-camera BA [17] by adding the intrinsic parameters as new estimated parameters. Up to now, we did three approximations: global shutter, central multi-camera, and zero sub-frame residual time offsets. Furthermore we only applied the SfMs (both monocular and multi-cameras) on the beginning of the videos to obtain initial synchronization and calibration (the 2k first frames in our experiments). Then the multi-camera SfM is applied a second time on the whole videos.

Last we apply the multi-camera BA in Sec. 6 for estimating the SFA synchronization and the line delay with usual parameters.

Sec. 7 explains how to efficiently compute non-closed form image projections and their derivatives involved in BA. The experiments and conclusion are in Secs. 8 and 9, respectively.

4. Equiangular initializations

Secs. 4.1 and 4.2 describe two monocular camera models and their initializations (before all SfM and BA computations). Both models involve the intrinsic parameter matrix $\mathtt{K}$ of a perspective camera: $\mathtt{K}$ has focal parameters f_x and f_y, principal point $\mathbf{z}_0$ and zero skew. The classical polynomial distortion model [30, 18, 17] is often used since its closed-form back-projection is useful for SfM tasks and epipolar geometry. It has several radial distortion parameters and can be applied to consumer cameras like Gopro [4]. The unified camera model [32] is also interesting since it deals with fisheyes having FoV larger than 180° (like those of spherical camera [7]) although it only has a single radial distortion parameter. The equiangular initialization can be adapted to other camera models.

4.1. Polynomial distortion model

4.1.1. Back-projection

The function from the distorted (i.e. original) image to the undistorted (i.e. rectified) image depends on radial distortion parameters k_i (tangential distortions are neglected). Let $\mathbf{z}_d$ and $\mathbf{z}_u$ be the distorted and undistorted coordinates of a pixel. Their normalized coordinates $\bar{\mathbf{z}}_d$ and $\bar{\mathbf{z}}_u$ meet

$$\mathtt{K}\begin{pmatrix}\bar{\mathbf{z}}_d\\1\end{pmatrix}=\begin{pmatrix}\mathbf{z}_d\\1\end{pmatrix}\text{ and }\mathtt{K}\begin{pmatrix}\bar{\mathbf{z}}_u\\1\end{pmatrix}=\begin{pmatrix}\mathbf{z}_u\\1\end{pmatrix}. \tag{1}$$

Let $\bar{r}_d=\|\bar{\mathbf{z}}_d\|$ be the normalized radial distance in the distorted image. The relation between distorted and undistorted coordinates is

$$\bar{\mathbf{z}}_u=(1+\sum_{i=1}^{n}k_i\bar{r}_d^{2i})\bar{\mathbf{z}}_d. \tag{2}$$

Lastly, the back-projected ray of pixel $\mathbf{z}_d$ has direction $\begin{pmatrix}\bar{\mathbf{z}}_u^\top & 1\end{pmatrix}^\top$ in the camera coordinate system.

4.1.2. Initialization

Here we initialize k_i, $\mathbf{z}_0$, f_x and f_y for an equiangular camera. The camera is equiangular if the angle μ between the principal direction $\begin{pmatrix}0 & 0 & 1\end{pmatrix}^\top$ and the back-projected ray is proportional to the (non-normalized) radial distance r_d in the distorted image. We have $r_d=\|\mathbf{z}_d-\mathbf{z}_0\|$ and $\tan\mu=\|\bar{\mathbf{z}}_u\|$. If the camera is equiangular, $f_x=f_y=f$ and there is a constant c such that $\mu=cr_d$. Thus $\mu=cf\bar{r}_d$. Since $\|\bar{\mathbf{z}}_u\|=\bar{r}_d(1+\sum_{i=1}^{n}k_i\bar{r}_d^{2i})$,

$$\tan(cf\bar{r}_d)=\tan\mu=\bar{r}_d+\sum_{i=1}^{n}k_i\bar{r}_d^{2i+1}. \tag{3}$$

Since tan is not a polynomial, Eq. 3 can not be exact. We use a Taylor's approximation

$$\tan\mu\approx\sum_{i=0}^{n}t_i\mu^{2i+1}=\mu+\frac{\mu^3}{3}+\frac{2\mu^5}{5}+\frac{17\mu^7}{315}+\cdots \tag{4}$$

and identify coefficients between Eqs 3 and 4. We obtain $cf=1$ using t_0 and $k_i=t_i$ if $i\geq 0$. In practice, we initialize $\mathbf{z}_0$ at the image center and compute $f=r_d/\mu$ for a pixel $\mathbf{z}_d$ at the center of an image border where the half-FoV μ is approximately known.

4.2. Unified camera model

4.2.1. Forward projection

Let $\mathbf{x}=\begin{pmatrix}x & y & z\end{pmatrix}^\top\in\mathbb{R}^3\setminus\{\mathbf{0}\}$ be a 3D point in the camera coordinate system. Let $\mathcal{S}$ be the unit sphere in $\mathbb{R}^3$ centered at $\mathbf{0}$ and let $\xi\in\mathbb{R}^+$. The projection $p(\mathbf{x})$ of $\mathbf{x}$ by this model is obtained as follows: first $\mathbf{x}$ is projected onto $\mathcal{S}$, then $\mathbf{x}/\|\mathbf{x}\|$ is projected onto the image plane by a perspective camera with the center $\begin{pmatrix}0 & 0 & -\xi\end{pmatrix}^\top$ and the intrinsic parameter matrix $\mathtt{K}$. Formerly,

$$p(\mathbf{x})=\pi(\mathtt{K}(\frac{\mathbf{x}}{\|\mathbf{x}\|}+\begin{pmatrix}0\\0\\\xi\end{pmatrix}))\text{ where }\pi(\begin{pmatrix}u\\v\\w\end{pmatrix})=\begin{pmatrix}u/w\\v/w\end{pmatrix}. \tag{5}$$

4.2.2. Initialization

Here we initialize ξ, $\mathbf{z}_0$, f_x and f_y for an equiangular camera. Let μ be the angle between the principal direction $\begin{pmatrix}0 & 0 & 1\end{pmatrix}^\top$ and the back-projected ray, which is a half-line started at $\mathbf{0}$ with the direction $\mathbf{x}/\|\mathbf{x}\|$. Appendix A shows that

$$f_x=f_y=f\Rightarrow\|p(\mathbf{x})-\mathbf{z}_0\|/f=\frac{\sin\mu}{\xi+\cos\mu}. \tag{6}$$

If the camera is equiangular, $f_x = f_y = f$ and there is a constant c such that $\mu = c\|p(\mathbf{x}) - \mathbf{z}_0\|$. Since $\frac{\sin\mu}{\xi+\cos\mu}$ is not linear in μ, we approximate it thanks to Taylor's expansions:

$$\begin{aligned}\frac{\sin\mu}{\xi+\cos\mu} &\approx \frac{\mu-\mu^3/6}{\xi+1-\mu^2/2} = \frac{\mu(1-\mu^2/6)}{(1+\xi)(1-\mu^2/(2\xi+2))} \\ &= \frac{\mu}{1+\xi}(1+\mu^2(\frac{1}{2\xi+2}-\frac{1}{6})+O(\mu^4)). \quad (7)\end{aligned}$$

We initialize $\xi = 2$ such that this approximation is linear in μ. Now we distinguish two cases for the initialization of $\mathbf{z}_0$ and f. If every pixel of the (rectangular) image has a back-projected ray, we initialize $\mathbf{z}_0$ at the image center and take a point $\mathbf{z}_1$ at the center of an image border where the half-FoV μ is approximately known. Otherwise, we assume that the pixels that have back-projected rays form a disk whose radius and center can be estimated. Then we initialize $\mathbf{z}_0$ by this center and take a point $\mathbf{z}_1$ at the disk boundary where the half-FoV μ is approximately known. In both cases, f is initialized by Eq. 6 using $p(\mathbf{x}) = \mathbf{z}_1$.

5. Synchronization initialization

The synchronization initialization is required by the multi-camera SfM-BA and has two steps. First Sec. 5.1 estimates instantaneous angular velocities thanks to monocular SfM-BA and global shutter approximation. Then time offsets are computed by correlation of IAVs of different cameras; Secs. 5.2 and 5.3 describe the two- and multi-camera cases respectively.

5.1. Instantaneous angular velocity (IAV)

Every monocular video is reconstructed such that every frame has a computed pose (both keyframes and non-keyframes, not only keyframes as in the paper remainder). Thus the keyframe-based SfM [28] is followed by pose calculations for the non-keyframes and by BA. In practice, it is sufficient to reconstruct few thousands of frames at the video beginning for the synchronization initialization. Let R_i^t be the rotation of the pose of the t-th frame in the reconstruction of the i-th video. The IAV θ_i^t at the t-th frame (of the i-th video) is approximated by the angle of rotation $\mathrm{R}_i^{t+1}(\mathrm{R}_i^t)^\top$, i.e.

$$\theta_i^t = \arccos((\mathrm{trace}(\mathrm{R}_i^{t+1}(\mathrm{R}_i^t)^\top) - 1)/2). \quad (8)$$

We omit the FpS coefficient since all cameras have the same. Intuitively, two frames of different but jointly moving cameras have same IAV if they are taken at the same time. This is shown in Appendix B by taking account the fact that the R_i^t are expressed in arbitrary coordinate systems due to the monocular SfM.

5.2. Synchronize two cameras

We compute an IAV table for every camera and find the time offset that maximizes the correlation (ZNCC) between two such tables (match two sub-tables with the same length in different tables). The time offset $o_{i,j}$ between the i-th and j-th cameras maximizes correlation $ZNCC_{i,j}$ between vectors θ_i^t and $\theta_j^{t+o_{i,j}}$.

We also introduce a simple SFA refinement method. The sub-frame offsets are estimated like sub-pixelic disparity using a quadratic fit [33]: first approximate the function from $o_{i,j}$ to $ZNCC_{i,j}$ using a quadratic polynomial defined by its 3 values at $o_{i,j}+\{-1,0,+1\}$; then estimate $\epsilon_{i,j}$ such that $o_{i,j}+\epsilon_{i,j}$ maximizes this polynomial. In contrast to the FA offsets $o_{i,j} \in \mathbb{Z}$, the SFA offsets $o_{i,j} + \epsilon_{i,j} \in \mathbb{R}$ are not used for the input of our BA.

5.3. Consistently synchronize more than two cameras

We remind that the goal of the FA synchronization is to skip s_i frames at the beginning of the i-th video such that the sequels of the videos are FA synchronized (this is required for multi-camera SfM). Thus $o_{i,j} = s_j - s_i$ for all $i \neq j$, which in turn imply that the sum of the offsets along every loop in the camera graph should be zero (e.g. we should have $o_{0,1} + o_{1,2} + o_{2,0} = 0$ for loop $0 \to 1 \to 2 \to 0$). However such a sum can be non-zero since the offsets are estimated independently.

There are several ways to deal with this loop constraint. First only compute offsets $o_{0,i}$. But this solution privileges a camera. Second compute all offsets $o_{i,j}$, generate candidate offsets around $o_{i,j}$ for every pair (i, j), and select the candidate offsets that maximizes $\sum_{i\neq j} ZNCC_{i,j}$ such that the sum of candidate offsets along every loop is zero. We implement an intermediate and simple solution where every camera has the same importance assuming that the cameras are symmetrically mounted around a symmetry axis: we only consider the spatial adjacency of the n cameras, i.e. we only compute offsets $o_{0,1}, o_{1,2}, \cdots o_{n-2,n-1}, o_{n-1,0}$ (instead of all $o_{i,j}$) and only use the loop $0 \to 1 \to \cdots n-1 \to 0$ (instead of all loops) in the scheme above. In practice, we found that it is sufficient to generate candidate offsets that differ from the initial ones by $+1$ or 0 or -1. In the remainder of the paper, we use the notation $o_{i,j}$ for offsets that meet the loop constraint.

6. Bundle adjustment for RS and synchronization

This is the last step of our method and it requires the multi-camera initialization described in Sec. 6.1. Sec. 6.2 presents our continuous-time parametrization of the multi-camera motion: it is defined by the composition of a function M from a time interval to $\mathbb{R}^3 \times \mathbb{R}^k$ and a function $\mathcal{R}$ from $\mathbb{R}^k$ to the set of rotations in $\mathbb{R}^3$. Sec. 6.3 describes a keyframe of the multi-camera, where every line has a time that depends on the camera that captures the line, its y-coordinate and the line delay. Sec. 6.4 approximates $M(t)$ at a time t from the few $M(t_i)$ corresponding to the beginnings of the keyframes; this is useful to moderate the number of parameters estimated by BA. Sec. 6.5 provides a simple method to compute the reprojection error minimized by BA. Last Secs. 6.6 and 6.7 are more technical: we choose $\mathcal{R}$ in the former and detail the sparse structure of the linear system solved by BA in the latter.

6.1. Initialization

First we assume that the monocular videos are FA synchronized by removing few frames at their beginning (Sec. 5). Then we define the i-th frame of the multi-camera by a concatenation of sub-images, every of them is the i-th frame of a monocular camera. From now on, we use word *frame* for "frame of

the multi-camera" and the *video* is the sequence defined by all these frames. Last we use a standard SfM based on keyframe sub-sampling of the video (Appendix G) and local BA [28] assuming GS. We remind that the *keyframes* are the only frames whose poses are refined by the BAs (this is useful for both time computation and accuracy).

6.2. Parametrization of the multi-camera trajectory

Let $\mathcal{R}$ be a C^1 continuous and surjective function that maps $\mathbb{R}^k$ to the set of the 3D rotations (typical values are $k \in \{3, 4\}$). We assume that there is a C^3 continuous function $M : \mathbb{R} \to \mathbb{R}^3 \times \mathbb{R}^k$ that parametrizes the motion of the multi-camera. More precisely, $M(t)^T = \begin{pmatrix} T_M(t)^T & E_M(t)^T \end{pmatrix}$ where $t \in \mathbb{R}$ is the time, $T_M(t) \in \mathbb{R}^3$ is the translation and $\mathcal{R}(E_M(t))$ is the rotation of the multi-camera pose. The columns of $\mathcal{R}(E_M(t))$ and $T_M(t)$ are the vectors of the multi-camera coordinate system expressed in world coordinates. The choice of $\mathcal{R}$ (including E_M and k) is detailed in Sec. 6.6 for the paper clarity.

Thanks to these notations and assumptions, we will approximate $M(t)$ by using values of M taken at few times $t_1, \cdots, t_m$. Then our model of the camera trajectory not only provides the multi-camera pose at each instant corresponding to each line of a frame, but it also has a moderated number of parameters to be estimated by BA: the vector concatenating all $M(t_i)$, which has dimension $m(3 + k)$. Sec. 6.3 defines t_i and Sec. 6.4 describes our approximations of $M(t)$ by using the $M(t_i)$.

6.3. Time, RS and synchronization parameters

The i-th keyframe is an image composed of sub-images taken by the monocular cameras. Every line of every sub-image is taken at its own time, which is described now. The 0-th line of the 0-th sub-image in the i-th keyframe is taken at time t_i, assuming that the time exposure of a line is instantaneous [5]. Thus $t_{i+1} - t_i$ is a multiple of the inverse of the FpS. Since the cameras are RS, the line delay τ is such that the y-th line of the 0-th sub-image in the i-th keyframe is taken at time $t_i + y\tau$. Let $\Delta_j \in \mathbb{R}$ be the sub-frame residual time offset between the j-th video and the 0-th video. Then the 0-th line of the j-th sub-image in the i-th keyframe is taken at time $t_i + \Delta_j$. Since we assume that all cameras have the same FpS and same (and constant) τ, the y-th line of the j-th sub-image in the i-th keyframe is taken at time $t_i + \Delta_j + y\tau$. Fig. 1 illustrates the trajectory $M(t)$ of a multi-camera defined by four monocular rolling shutter cameras having non-zero time offsets Δ_j.

6.4. Approximations for the multi-camera trajectory

Let $\Delta = \max_i(t_{i+1} - t_i)$ and shortened notation $\mathbf{m}_i = M(t_i)$. Thanks to the C^3 continuity of M and Taylor's expansions of M at t_i, we explicit two approximations $M_1(t)$ and $M_2(t)$ of $M(t)$ in the neighborhood of t_i as functions of $\mathbf{m}_{i-1}$, $\mathbf{m}_i$ and $\mathbf{m}_{i+1}$. These approximations have remainders expressed in terms of Δ and $|t - t_i|$. By neglecting these remainders, we compute $M(t)$ for the y-th line of the j-th camera/sub-image in the i-th keyframe using $t = t_i + \Delta_j + y\tau$ (Sec. 6.3) during our BA.

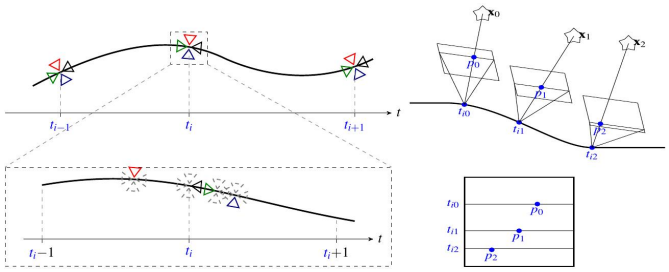

Figure 1: Time continuous trajectory of a multi-camera. Left: four monocular cameras at time t_i, which have non-zero time offsets. Right: a rolling shutter monocular camera, which moves and sees points at several times/lines in a single frame.

6.4.1. Linear approximation M_1 of M

We have Taylor's linear expansion

$$M(t) = \mathbf{m}_i + (t - t_i)M'(t_i) + O(|t - t_i|^2) \tag{9}$$

and express the derivative $M'(t_i)$ as a function of $\mathbf{m}_{i-1}$, $\mathbf{m}_i$ and $\mathbf{m}_{i+1}$. Let reals $a > 0$ and $b > 0$, vectors $\mathbf{x}, \mathbf{y}, \mathbf{z}$ in $\mathbb{R}^{k+3}$, function D_1 such that

$$D_1(\mathbf{x}, \mathbf{y}, \mathbf{z}, a, b) = \frac{b\mathbf{z}}{a(a+b)} - \frac{a\mathbf{x}}{b(a+b)} + \frac{(a-b)\mathbf{y}}{ab}, \tag{10}$$

and shortened notation

$$D_1^i = D_1(\mathbf{m}_{i-1}, \mathbf{m}_i, \mathbf{m}_{i+1}, t_{i+1} - t_i, t_i - t_{i-1}). \tag{11}$$

Appendix C shows that $M'(t_i) = D_1^i + O(\Delta^2)$. We obtain

$$M_1(t) = \mathbf{m}_i + (t - t_i)D_1^i \text{ if } t \approx t_i. \tag{12}$$

If $i = 0$ (similarly if $i = m$), we use $D_1^0 = \frac{\mathbf{m}_1 - \mathbf{m}_0}{t_1 - t_0}$.

6.4.2. Quadratic approximation M_2 of M

Similarly, we have Taylor's quadratic expansion of M at t_i and express the derivative $M''(t_i)$ as a function of $\mathbf{m}_{i-1}$, $\mathbf{m}_i$ and $\mathbf{m}_{i+1}$. Let a, b, $\mathbf{x}$, $\mathbf{y}$, $\mathbf{z}$ as in Sec. 6.4.1, function D_2 such that

$$D_2(\mathbf{x}, \mathbf{y}, \mathbf{z}, a, b) = \frac{2\mathbf{z}}{a(a+b)} + \frac{2\mathbf{x}}{b(a+b)} - \frac{2\mathbf{y}}{ab}, \tag{13}$$

and shortened notation

$$D_2^i = D_2(\mathbf{m}_{i-1}, \mathbf{m}_i, \mathbf{m}_{i+1}, t_{i+1} - t_i, t_i - t_{i-1}). \tag{14}$$

Appendix C shows that $M''(t_i) = D_2^i + O(\Delta)$. We obtain

$$M_2(t) = \mathbf{m}_i + (t - t_i)D_1^i + \frac{(t - t_i)^2}{2}D_2^i \text{ if } t \approx t_i. \tag{15}$$

If $i = 0$ (similarly if $i = m$), we use $D_1^0 = \frac{\mathbf{m}_1 - \mathbf{m}_0}{t_1 - t_0}$ and $D_2^0 = 0$.

6.5. Reprojection error of the multi-camera

Since our BA minimizes the sum of squared modulus of reprojection error for every inlier, this section describes the computation of a reprojection error for a 3D point $\mathbf{x} \in \mathbb{R}^3$ (in world coordinates) and its inlier observation $\tilde{\mathbf{p}} \in \mathbb{R}^2$ in the j-th sub-image of the i-th keyframe.

First we introduce notations. Let $\mathbf{p} \in \mathbb{R}^2$ be the projection of $\mathbf{x}$ in the j-th sub-image of the i-th keyframe. The reprojection error is $\mathbf{p} - \tilde{\mathbf{p}}$. Let $(\mathrm{R}_j, \mathbf{t}_j)$ be the pose of the j-th camera in the multi-camera frame. Let $p_j : \mathbb{R}^3 \setminus \{\mathbf{0}\} \to \mathbb{R}^2$ be the projection function of the j-th camera. We assume that $p_j, \mathrm{R}_j, \mathbf{t}_j$ are constant. The acquisition times of $\mathbf{p} = (x, y)$ and $\tilde{\mathbf{p}} = (\tilde{x}, \tilde{y})$ are

$$t_{\mathbf{p}} = t_i + \Delta_j + y\tau \text{ and } t_{\tilde{\mathbf{p}}} = t_i + \Delta_j + \tilde{y}\tau. \tag{16}$$

Second we detail the relation between $\mathbf{p}$ and $\mathbf{x}$. Both $E_M(t_{\mathbf{p}})$ and $T_M(t_{\mathbf{p}})$ (i.e. $M(t_{\mathbf{p}})$ in Sec. 6.2) are defined by one equation chosen among Eq. 12 and Eq. 15 using the index i of the keyframe and $t = t_{\mathbf{p}}$. The coordinates of $\mathbf{x}$ in the multi-camera coordinate system is

$$\mathbf{x}_M = \mathcal{R}(E_M(t_{\mathbf{p}}))^\top (\mathbf{x} - T_M(t_{\mathbf{p}})). \tag{17}$$

The coordinates of $\mathbf{x}$ in the j-th camera coordinate system and the projection of $\mathbf{x}$ are

$$\mathbf{x}_j = \mathrm{R}_j^\top (\mathbf{x}_M - \mathbf{t}_j) \text{ and } \mathbf{p} = p_j(\mathbf{x}_j). \tag{18}$$

We see that $\mathbf{p}$ needs the computation of $\mathbf{x}_M$, which in turn needs the computation of (the y coordinate of) $\mathbf{p}$.

Such a problem is solved thanks to an approximation in [24]: $t_{\mathbf{p}}$ is replaced by $t_{\tilde{\mathbf{p}}}$ in Eq. 17, i.e. we assume that the multi-camera pose is the same at times $t_{\tilde{\mathbf{p}}}$ and $t_{\mathbf{p}}$. We think that this is acceptable since $|t_{\tilde{\mathbf{p}}} - t_{\mathbf{p}}| \leq \tau \|\mathbf{p} - \tilde{\mathbf{p}}\|$ and the magnitude order of τ is 10^{-5} s/pixel and $\tilde{\mathbf{p}}$ is an inlier (i.e. $\|\mathbf{p} - \tilde{\mathbf{p}}\| \leq 4$ pixels). Sec. 7.3 presents another solution without this approximation.

6.6. Parametrization of rotations

Sec. 6.6.1 lists useful properties of $\mathcal{R}$ (reminder: $\mathcal{R}$ is a function introduced in Sec. 6.2 which maps $\mathbb{R}^k$ to the set of the 3D rotations). Then Sec. 6.6.2 explains our choice of $\mathcal{R}$ to meet the properties in Sec. 6.6.1.

6.6.1. Details on $\mathcal{R}$ properties

First we note that $\mathcal{R}$ is a global parametrization used for the whole camera trajectory (we do not use local parametrizations, i.e. different parametrizations for different keyframes). Second the C^1 continuity of $\mathcal{R}$ is needed by BA for the derivative computations of the reprojection errors. Third we follow [20] by using a minimal (non-redundant) parametrization $\mathcal{R}$ of the rotations to limit the number of estimated parameters. Thus $k = 3$.

Fourth BA needs another property. According to Secs. 6.2 and 6.3, $\mathbf{m}_i = M(t_i) = \begin{pmatrix} T_M(t_i)^\top & E_M(t_i)^\top \end{pmatrix}^\top \in \mathbb{R}^6$ is one of the parameter vectors estimated by BA such that $(T_M(t_i), \mathcal{R}(E_M(t_i)))$ is the pose (of the first line) of the i-th keyframe. Since the set of all rotations in a neighborhood of a current estimate of rotation $\mathcal{R}(E_M(t_i))$ should be reachable by the parametrization $\mathcal{R}$ during every BA iteration [9], the jacobian $\partial\mathcal{R}$ of $\mathcal{R}$ should be rank 3 at $E_M(t_i)$. In other words, $E_M(t_i)$ should not be a *singularity* of $\mathcal{R}$.

Unfortunately, every 3D parametrization $\mathcal{R}$ of the rotation set has singularities [34]. Thus we choose $\mathcal{R}$ in Sec. 6.6.2 such that all its singularities are far from the multi-camera motion that we want to refine using BA.

6.6.2. Choice of a minimal parametrization $\mathcal{R}$

First we consider $\mathcal{R}$ candidates and describe constraints that they induce on a class of multi-camera motions: all yaw motions are possible but pitch and roll are small. Such motions are very common for a helmet-held multi-camera and an user exploring the environment without special objective like grasping at object on the ground (and also for a car-fixed multi-camera).

Even the popular exponential map $\omega \mapsto \exp([\omega]_\times)$ has singularities: they form concentric spheres with center $\mathbf{0}$ and radii that are multiples of 2π [35]. This can be seen thanks to the equivalent angle-axis $(\theta, \mathbf{n})$ representation where $\|\mathbf{n}\| = 1$ and $\omega = \theta\mathbf{n}$. Thus the range of the angle θ is equal to 4π for every axis $\mathbf{n}$. If we choose $\mathcal{R}(\omega) = \exp([\omega]_\times)$ as in [20] and would like to avoid the singularities, the multi-camera should avoid multiple turns on the left (or right) around buildings and avoid straight trajectory segments where $\|\omega\| \approx 2\pi k$ and $k \in \mathbb{Z}^*$.

We also detail the case of Euler's parametrization

$$\mathcal{E}(\alpha, \beta, \gamma) = R_z(\gamma) R_y(\beta) R_x(\alpha) \tag{19}$$

where $R_x(\alpha)$, $R_y(\beta)$ and $R_z(\gamma)$ are the rotations about respective axes $\begin{pmatrix} 1 & 0 & 0 \end{pmatrix}^\top, \begin{pmatrix} 0 & 1 & 0 \end{pmatrix}^\top, \begin{pmatrix} 0 & 0 & 1 \end{pmatrix}^\top$ and with respective angles α, β, γ. The singularities (α, β, γ) of $\mathcal{E}$ form parallel and equidistant planes of equations $\beta = \pi/2 + p\pi$ such that $p \in \mathbb{Z}$ [34]. If we choose $\mathcal{R} = \mathcal{E}$ and the coordinate systems (both world and multi-camera) are such that $\forall i, \mathcal{R}(E_M(t_i)) \approx R_x(\alpha_i)$, we are far from the singularities. If the coordinate systems are such that $\forall i, \mathcal{R}(E_M(t_i)) \approx R_y(\pi/2) R_x(\alpha_i)$, we are close to the singularities.

Last we choose $\mathcal{R}$ inspired by the Euler's case above. Let

$$\mathcal{R}(\alpha, \beta, \gamma) = \mathtt{A} R_z(\gamma) R_y(\beta) R_x(\alpha) \mathtt{B} \tag{20}$$

where rotations $\mathtt{A}$ and $\mathtt{B}$ do not depend on (α, β, γ). We estimate $\mathtt{A}$ and $\mathtt{B}$ such that β is close to 0 for all keyframe rotations of the multi-camera trajectory before the BA in Sec. 6 (technical details in Appendix D). Now the camera motion in our class is far from all singularities. Note that the local Euler parametrization, that is used in BA [9], is a special case of this parametrization.

6.7. Sparsity of the reduced camera system (RCS)

In this section, we explicit the sparsity of the RCS that is solved by BA [9]. This is important for efficient computations.

6.7.1. Notations and global structure of the RCS

The m vectors $\mathbf{m}_i = M(t_i)$ are the parameters of the multi-camera trajectory (Sec. 6.2), they meet $\mathbf{m}_i \in \mathbb{R}^6$ (Sec. 6.6.1), and we define $\mathbf{M} = \begin{pmatrix} \mathbf{m}_1^\top & \cdots & \mathbf{m}_m^\top \end{pmatrix}^\top \in \mathbb{R}^{6m}$. Let $\mathbf{m}' \in \mathbb{R}^{m'}$ be the other optimized camera parameters among intrinsic parameters, camera poses in multi-camera coordinates, line delay and time offsets. Since these other optimized parameters are the sames at all keyframes, $m' \ll 6m$. For example, $m = 1000$ and $m' \leq 4{+}4{*}15 = 64$ if the multi-camera has four Gopro cameras: there are line delay τ, time offsets Δ_1, Δ_2, Δ_3 (Sec. 6.3), and every camera has parameters $f_x, f_y, u_0, v_0, k_1 \cdots k_5$ (Sec. 4.1.1) and 6D pose in multi-camera coordinates. Let $\mathbf{X}$ be the vector that concatenates points $\mathbf{x}_l \in \mathbb{R}^3$ in world coordinates. The

Figure 2: Shape of the standard (left) and our (right) $\mathtt{Z}$ for a video sequence with 1573 keyframes and closed loops and same inliers.

projection function of $\mathbf{x}_l$ in the i-th keyframe is concisely written $\varphi(\mathbf{m}_{i-1}, \mathbf{m}_i, \mathbf{m}_{i+1}, \mathbf{m}', \mathbf{x}_l)$ for both approximations in Sec. 6.4 (omit $\mathbf{m}_{i-1}$ if $i = 1$ and omit $\mathbf{m}_{i+1}$ if $i = m$). Let

$$\begin{aligned} \mathtt{H} &= \sum \begin{pmatrix} \frac{\partial\varphi}{\partial\mathbf{M}} & \frac{\partial\varphi}{\partial\mathbf{m}'} & \frac{\partial\varphi}{\partial\mathbf{X}} \end{pmatrix}^\top \begin{pmatrix} \frac{\partial\varphi}{\partial\mathbf{M}} & \frac{\partial\varphi}{\partial\mathbf{m}'} & \frac{\partial\varphi}{\partial\mathbf{X}} \end{pmatrix} \\ &= \begin{pmatrix} \mathtt{U} & \mathtt{U}' & \mathtt{W} \\ (\mathtt{U}')^\top & \mathtt{U}'' & \mathtt{W}' \\ \mathtt{W}^\top & (\mathtt{W}')^\top & \mathtt{V} \end{pmatrix}. \end{aligned} \tag{21}$$

Here $\mathtt{H}$ is the approximated hessian of the cost function minimized by BA. It is defined as a sum for all 2D inliers (detailed notations are omitted). The RCS is

$$\begin{pmatrix} \mathtt{U} & \mathtt{U}' \\ (\mathtt{U}')^\top & \mathtt{U}'' \end{pmatrix} - \begin{pmatrix} \mathtt{W} \\ \mathtt{W}' \end{pmatrix} \mathtt{V}^{-1} \begin{pmatrix} \mathtt{W} \\ \mathtt{W}' \end{pmatrix}^\top = \begin{pmatrix} \mathtt{Z} & \mathtt{Z}' \\ (\mathtt{Z}')^\top & \mathtt{Z}'' \end{pmatrix}. \tag{22}$$

We have $\mathtt{Z} = \mathtt{U} - \mathtt{W}\mathtt{V}^{-1}\mathtt{W}^\top \in \mathbb{R}^{6m\times 6m}$, $\mathtt{Z}' \in \mathbb{R}^{6m\times m'}$ and $\mathtt{Z}'' \in \mathbb{R}^{m'\times m'}$. Since $m' \ll 6m$, $\mathtt{Z}$ is the preponderant block in the RCS and we only focus on the $\mathtt{Z}$ sparsity.

6.7.2. Sparsity of $\mathtt{Z}$

Here we represent $\mathtt{Z}$ by a shape included in $\mathbb{Z}^2$, i.e. a set of pixels in an image such that every pixel corresponds to a non-zero 6×6-block of $\mathtt{Z}$. Then we show in Appendix E that the shape of our $\mathtt{Z}$ (which involves SFA and RS using projection function $\varphi(\mathbf{m}_{i-1}, \mathbf{m}_i, \mathbf{m}_{i+1}, \mathbf{m}', \mathbf{x}_l)$) is included in a dilation of the shape of the standard $\mathtt{Z}$ (which involves FA and GS using projection function $\varphi(\mathbf{m}_i, \mathbf{m}', \mathbf{x}_l)$) by $\{-1, 0, +1\}^2$. We remind that this dilation is an operation morphology that expands a shape by one pixel in both dimensions and both directions. Thus our RCS is slightly less sparse that the standard RCS (in practice it is very similar according to the example in Fig. 2).

In the case where the loops are not closed in the video and the track length is bounded by l, the standard $\mathtt{Z}$ is a 6×6-block-wise band matrix with bandwidth l (Sec. A6.7.1 in [8]) and our $\mathtt{Z}$ is a 6×6-block-wise band matrix with bandwidth $l + 1$.

7. Non-closed-form image projections

Sec. 7 computes the projection $\mathbf{p}$ and its derivatives with respect to a vector θ of parameters optimized by BA, although $\mathbf{p}$ does not have a closed-form expression from θ. This is needed by BA and occurs in two cases in this paper. The general case is described in Sec. 7.1 assuming that $\mathbf{p}$ and θ meet an equation that implicitly defines $\mathbf{p}$ for a given value of θ. Then Secs. 7.2 and 7.3 apply the general case in two cases: the projection of the polynomial distortion model (reminder: Sec. 4.1.1 only computes the back-projection) and the exact projection using a continuous-time trajectory model (reminder: Sec. 6.5 only computes an approximate projection). This problem and its solution are similar to those of a general non-central catadioptric camera (Appendix D in [36]).

7.1. General case

We know an approximate value $\tilde{\mathbf{p}}$ of $\mathbf{p}$ ($\tilde{\mathbf{p}}$ is an inlier detected in an image), a C^1 continuous function $g(\mathbf{z}, \theta)$ from $\mathbb{R}^2 \times \mathbb{R}^p$ to $\mathbb{R}^2$ such that $\mathbf{p}$ is the solution $\mathbf{z}$ of $g(\mathbf{z}, \theta) = \mathbf{0}$, and the current value θ_0 of θ (provided by initialization or previous iteration of BA). First $\mathbf{p}$ is estimated by non-linear least-squares minimizing $\mathbf{z} \mapsto \|g(\mathbf{z}, \theta_0)\|^2$. In practice, we use the iterative Gauss-Newton's method starting from $\mathbf{z} = \tilde{\mathbf{p}}$ with no more than 5 iterations (Newton's method can also be used). Then the implicit function Theorem implies that we locally have a C^1 continuous function ψ such that $\mathbf{p} = \psi(\theta)$ if $\det \frac{\partial g}{\partial \mathbf{z}} \neq 0$. By differentiating $g(\psi(\theta), \theta) = \mathbf{0}$ using the Chain rule, we obtain

$$\frac{\partial \mathbf{p}}{\partial \theta} = \frac{\partial \psi}{\partial \theta} = -(\frac{\partial g}{\partial \mathbf{z}})^{-1} \frac{\partial g}{\partial \theta}. \tag{23}$$

7.2. Case 1: polynomial distortion model

Here we focus on the projection $\mathbf{p} = p_j(\mathbf{x}_j)$ in Eq. 18 using the camera model in Sec. 4.1.1 and assuming that the 3D point $\mathbf{x}_j$ is known (in camera coordinates). First we define θ and g by

$$\theta = (f_x, f_y, \mathbf{z}_0, k_1, k_2, \cdots, k_n, \mathbf{x}_j), \mathbf{z}_u = \pi(\mathrm{K}\mathbf{x}_j), \tag{24}$$

$$\begin{pmatrix} u \\ v \end{pmatrix} = \mathbf{z}, \begin{pmatrix} u_0 \\ v_0 \end{pmatrix} = \mathbf{z}_0, \bar{r}^2 = \frac{(u-u_0)^2}{f_x^2} + \frac{(v-v_0)^2}{f_y^2}, \tag{25}$$

$$g(\mathbf{z}, \theta) = (1 + \sum_{i=1}^{n} k_i \bar{r}^{2i})(\mathbf{z} - \mathbf{z}_0) - \mathbf{z}_u + \mathbf{z}_0. \tag{26}$$

Then we show that $g(\mathbf{p}, \theta) = \mathbf{0}$. Since $\begin{pmatrix} \bar{\mathbf{z}}_u^\top & 1 \end{pmatrix}^\top$ in Sec. 4.1.1 and $\mathbf{x}_j$ are colinear, $\mathbf{z}_u$ is the same in Sec. 4.1.1 and Eq. 24. Furthermore, $\mathbf{p} = \mathbf{z} = \mathbf{z}_d$ implies that $\bar{r}_d$ (in Sec. 4.1.1) and $\bar{r}$ are the same. We obtain $g(\mathbf{p}, \theta) = \mathbf{0}$ by multiplying Eq. 2 on the left by $\begin{pmatrix} f_x & 0 \\ 0 & f_y \end{pmatrix}$. Last we apply Sec. 7.1: find $\mathbf{p}$ by minimizing $\mathbf{z} \mapsto \|g(\mathbf{z}, \theta_0)\|^2$ and use Eq. 23 for $\mathbf{p}$ derivatives:

$$\frac{\partial \mathbf{p}}{\partial k_i} = -(\frac{\partial g}{\partial \mathbf{z}})^{-1} \bar{r}^{2i} (\mathbf{z} - \mathbf{z}_0), \tag{27}$$

$$\frac{\partial \mathbf{p}}{\partial \mathbf{x}_j} = (\frac{\partial g}{\partial \mathbf{z}})^{-1} \frac{\partial \mathbf{z}_u}{\partial \mathbf{x}_j}. \tag{28}$$

Using a similar function g, Appendix F shows that

$$\begin{pmatrix} \frac{\partial \mathbf{p}}{\partial f_x} & \frac{\partial \mathbf{p}}{\partial f_y} & \frac{\partial \mathbf{p}}{\partial u_0} & \frac{\partial \mathbf{p}}{\partial v_0} \end{pmatrix} = \begin{pmatrix} \frac{u-u_0}{f_x} & 0 & 1 & 0 \\ 0 & \frac{v-v_0}{f_y} & 0 & 1 \end{pmatrix}. \tag{29}$$

Eqs. 27, 28 and 29 need $\mathbf{z} = \mathbf{p}$ and $\theta = \theta_0$.

We note that the derivative computations in Eq. 28 are easy from those of a standard perspective camera (i.e. $\frac{\partial \mathbf{z}_u}{\partial \mathbf{x}_j}$): multiply on the left by $(\frac{\partial g}{\partial \mathbf{z}})^{-1}$. This also holds for derivatives with respect to the parameters defining $\mathbf{x}_j$ (Eq. 18) thanks to the Chain rule.

7.3. Case 2: exact calculation for RS and synchronization

Here we focus on the projection $\mathbf{p}$ by the j-th camera in the i-th keyframe of the 3D point $\mathbf{x}$ in world coordinates without the approximation in Sec. 6.5. Let $p(\theta_j, \mathbf{m}, \mathbf{x})$ be the projection of $\mathbf{x}$ where $\mathbf{m} = M(t)$ is the parameter of the multi-camera pose (reminder: $M(t)$ is introduced in Sec. 6.2) and the vector θ_j concatenates the intrinsic/distortion parameters and the pose of the j-th camera in the multi-camera coordinate system. The acquisition time of $\mathbf{p}$ is

$$t(\Delta_j, \tau, \mathbf{p}) = t_i + \Delta_j + \tau \begin{pmatrix} 0 & 1 \end{pmatrix} \mathbf{p}. \tag{30}$$

and we use notation $M(\mathbf{m}_{i-1}, \mathbf{m}_i, \mathbf{m}_{i+1}, t)$ for the chosen approximation (Eq. 12 or Eq. 15). Thus we have

$$\mathbf{p} = p(\theta_j, M(\mathbf{m}_{i-1}, \mathbf{m}_i, \mathbf{m}_{i+1}, t(\Delta_j, \tau, \mathbf{p})), \mathbf{x}). \tag{31}$$

Now we define θ and g by

$$\theta = (\theta_j, \mathbf{m}_{i-1}, \mathbf{m}_i, \mathbf{m}_{i+1}, \Delta_j, \tau, \mathbf{x}), \tag{32}$$

$$g(\mathbf{z}, \theta) = p(\theta_j, M(\mathbf{m}_{i-1}, \mathbf{m}_i, \mathbf{m}_{i+1}, t(\Delta_j, \tau, \mathbf{z})), \mathbf{x}) - \mathbf{z}. \tag{33}$$

We see that $g(\mathbf{p}, \theta) = \mathbf{0}$. Then we apply Sec. 7.1: find $\mathbf{p}$ by minimizing $\mathbf{z} \mapsto \|g(\mathbf{z}, \theta_0)\|^2$ and use Eq. 23 for the $\mathbf{p}$ derivatives.

If we use the linear trajectory approximation M_1 (Eq. 12), we have a simple expression

$$\frac{\partial g}{\partial \mathbf{z}} = \frac{\partial p}{\partial \mathbf{m}} \frac{\partial M}{\partial t} \frac{\partial t(\Delta_j, \tau, \mathbf{z})}{\partial \mathbf{z}} - \mathtt{I}_2 \tag{34}$$

$$= \tau \frac{\partial p}{\partial \mathbf{m}} D^i_1 \begin{pmatrix} 0 & 1 \end{pmatrix} - \mathtt{I}_2 \in \mathbb{R}^{2\times 2}. \tag{35}$$

We note that the derivative computations without approximation (Eq. 23) can be deduced from those with approximation (i.e. $\frac{\partial \mathbf{p}}{\partial \theta} = \frac{\partial g}{\partial \theta}(\tilde{\mathbf{p}}, \theta_0)$): replace $\tilde{\mathbf{p}}$ by $\mathbf{p}$ in the derivative by θ and multiply it on the left by $-(\frac{\partial g}{\partial \mathbf{z}})^{-1}$.

8. Experiments

8.1. Datasets

Secs. 8.1.1 and 8.1.2 present cameras and video sequences that are used in the experiments, respectively. Tab. 1 summarizes our dataset (both cameras and videos).

8.1.1. Cameras

The consumer multi-cameras are modeled by several rigidly mounted monocular cameras and the user fixes them on a helmet. We assume that all calibrations parameters (time offsets Δ_j, line delay τ, intrinsics, radial distortion, relative poses) are constant during a video acquisition. The camera gain is not fixed and evolves independently for every camera.

First there are 360 cameras composed of four GoPro Hero 3 cameras [4], that are started by a single click on a wifi remote. The user can choose the relative poses of the cameras: they are enclosed in a cardboard for small baseline or are fixed by using the housings provided with the cameras (larger baseline).

Second there is a spherical camera modeled by two opposite fisheyes (no relative pose choice) that are synchronized. The Ricoh Theta S multi-camera [7] has a very small baseline thanks to the use of a prism mirror in front of a monocular camera (its FoV is split in two equal parts, each of them sees more than a half-sphere as a real fisheye does).

Figure 3: Cameras (four Gopro Hero 3 in a cardboard) and images for BC1, WT and BC2. The rolling shutter always goes from right to left, the image motion goes toward left on the two left columns and goes toward right on the two right columns.

We also experiment on a professional multi-camera (Point-Grey Ladybug 2 [37]) since its ground truth is provided by the manufacturer (as a table of rays) and also for experimenting on an ideal multi-camera with global shutter and perfect synchronization. Except in a synthetic case, the other cameras have incomplete ground truth (a strobe always provides τ).

8.1.2. Videos

There are three real multi-camera videos taken under various conditions using four GoPro cameras (BC1: bike riding in a city, WT: walking in a town, FH: paragliding flying at very low height above a hill). WT is taken in the early morning during summer to avoid moving car and people, but the lighting is low. BC1 has sunny lighting (with contre-jours) and most cars are parked. WT and BC1 have the same calibration setting, FH has larger baseline and lower FpS and better angular resolution. Fig. 3 shows images and cameras of BC1 and WT, Fig. 4 shows images and cameras of FH.

BikeCity2 (BC2) is generated by ray-tracing of a synthetic urban scene having real textures and by moving the camera

Name (short name)	Camera	f	r (mr)	b (cm)	τ (μs)	$f\Delta_j$	l (m)	fr	FoV	kfr	#Tracks	$\|\|\beta_i\|\|_\infty$
BikeCity1 (BC1)	4*Gopro 3	100	1.56	7.5	9.10	?	2500	50.4k	90	2047	343k	0.223
WalkTown (WT)	4*Gopro 3	100	1.56	7.5	9.10	?	900	70.3k	90	1363	240k	0.268
FlyHill (FH)	4*Gopro 3	48	1.06	18	11.3	?	1250	8.6k	90	627	432k	0.494
BikeCity2 (BC2)	4*Gopro 3	100	1.56	7.5	9.12	i/4	615	12.5k	90	225	51k	0.074
CarCity (CC)	Ladybug 2	15	1.90	6	0	0	2500	7.7k	72	891	282k	0.068
WalkUniv (WU)	Theta S	30	3.85	1.5	-32.1	0	1260	29.4k	200	1287	154k	0.129

Table 1: Datasets: FpS f, angular resolution r (millirad.), diameter b of multi-camera centers, line delay τ (ground truth), time offset $f\Delta_j$ (ground truth), approximate trajectory length l, numbers of frames fr and keyframes kfr, FoV angle used for initialization, maximum of angles $|\beta_i|$ (radians) of our parametrization in Sec. 6.6.2.

Figure 4: Cameras (four Gopro Hero 3 in their housings) and images for FH. The rolling shutter always goes from top to bottom, the image motion goes forward/toward right/backward/toward left.

Figure 5: Spherical camera (Ricoh Theta S) and image for WU. Both rolling shutter and image motion go from bottom to top (τ is negative, the bottom line is the oldest).

along a trajectory that mimics that of BC1 (the "pose noise", i.e. the relative pose between consecutive frames, are similar in both videos). We obtain a video for each camera by compressing the output images using ffmpeg and options "-c:v libx264 -preset slow -crf 18". BC2 has ground truth: $f\Delta_1 = 0.25$, $f\Delta_2 = 0.5$, $f\Delta_3 = 0.75$ and similar τ as BC1 (reminder: if $f\Delta_j = 1$ and f is the FpS, Δ_j is the time between two consecutive frames). Fig. 3 also shows images of BC2.

There is also video WU (walking in the campus of the UCA university) using a spherical camera: the Ricoh Theta S (Fig. 5).

Last CarCity is taken by the Ladybug 2 and has a similar trajectory as BC1. However, it is mounted on a car (using a mast) and is about 4 meters above the ground, as shown in Fig. 6. The images are uncompressed (all others are videos compressed using H.264).

Figure 6: Cameras (PointGrey Ladybug 2) and images for CC. The cameras are global shutter. The camera pointing toward the sky is not used in experiments.

8.2. *Main notations*

We use shortened notations:

- #2D= number of 2D inliers
- GT= ground truth,
- f= FpS
- RMS= RMS error in pixels in the original (distorted) image space,
- BA= bundle adjustment.
- method gs.sfa is the SFA refinement in Sec. 5.2
- y_{max} is the number of lines of a monocular image (y_{max} = 768 for CC, y_{max} = 1440 for FH, y_{max} = 960 for BC1,WT,BC2 and WU).

Our BA is named by a combination of several notations that describes the estimated parameters:

- C (central approx.) estimates all rotations R_j and fixes all translations $\mathbf{t}_j = \mathbf{0}$ (reminder: $(\mathrm{R}_j, \mathbf{t}_j)$ is the pose of the j-th camera in the multi-camera frame)
- NC (non-central) estimates all $(\mathrm{R}_j, \mathbf{t}_j)$
- RS (rolling shutter) estimates the line delay τ
- GS (global shutter) fixes $\tau = 0$
- SFA (sub-frame accurate) estimates all time offsets Δ_j

- FA (frame accurate) fixes all $\Delta_j = 0$
- INT (intrinsics) estimates all intrinsic parameters: $(f_x, f_y, u_0, v_0, \xi)$ or $(f_x, f_y, u_0, v_0, k_1 \cdots k_5)$ depending on the camera model chosen in Sec. 4.2.1 or Sec. 4.1.1 respectively; every camera has its own parameters.

Thus GS.NC.SFA.INT (or gs.nc.sfa.int) is a BA that fixes $\tau = 0$ and estimates simultaneously all $\Delta_j, \mathrm{R}_j, \mathbf{t}_j$ and intrinsic parameters and keyframe poses $\mathbf{m}_i$ and 3D points. The threshold for the inlier selection is set to 4 pixels in all videos. Every BA has three inlier updates, each one is followed by a Levenberg-Marquardt minimization for these inliers. A succession of two BAs is possible, e.g. gs.c.fa.int+rs.c.sfa.

The error $e(\Delta)$ is the sum of the absolute errors of all $f\Delta_j$. The error $e(\tau)$ is the relative error of τ. We also define the error of the estimated multi-camera calibration by a single number d, which is the RMS for all multi-camera pixels of the angle between rays of the two calibrations (the estimated one and the GT one) that back-project the same pixel. There are two reasons to do this. First the accuracy is only needed for the ray directions in applications (SfM, video stitching, 3d modeling of a scene) in the central case. Second parameters can compensate themselves if their estimations are biased (e.g. the rotation/principal point near-ambiguity for one view [38]). Now we detail the computation of d. Since the rays of the estimated calibration and the rays of the GT calibration can be expressed in different coordinate systems, we estimate a registration between both coordinate systems before computing angles between rays. The registration is defined by a rotation R, that maps one ray set to the other ray set (ignoring translation of ray origins). More precisely, R is the minimizer of $e(\mathrm{R}) = \sum_{i=1}^{N} \|\mathbf{r}_i^{gt} - \mathrm{R}\mathbf{r}_i^{est}\|^2$ where $\mathbf{r}_i^{est}$ (respectively, $\mathbf{r}_i^{gt}$) is the ray direction of the i-th pixel by the estimated (respectively, GT) multi-camera calibration. Our distance is $d = \sqrt{e(\mathrm{R})/N}$ where N is the number of (sampled) rays in a multi-camera image. Note that d is expressed in radians if $d \ll 1$; we always convert it in pixels by dividing it by the angular resolution r in Tab. 1.

8.3. Frame-accurate synchronization

Here we experiment the FA synchronization summarized in Sec. 3 and detailed in Sec. 5.

Fig. 7 draws the IAV (defined in Eq. 8) for consecutive frames taken in a rectilinear segment of the trajectory and the correlation function (that maps FA offset candidate $o_{0,1}$ to $ZNCC_{0,1}$) for cameras 0 and 1. This is done for biking (BC1), walking (WT) and car+mast (CC). There are similar variations of the IAV for different cameras and a single maximum of the ZNCC except for WT. Two consecutive offsets of WT have very similar greatest ZNCC values and the other ZNCC values are below, which suggest a half-frame residual time offset. These examples can convince the reader that we have enough information in the IAV to obtain a FA synchronization (at least if the cameras are helmet-held or mounted on a car thanks to a mast).

Tab. 2 shows the FA time offsets for all sequences. First we examine $o_{j,j+1}$ for the four Gopro Hero 3 cameras in BC1,

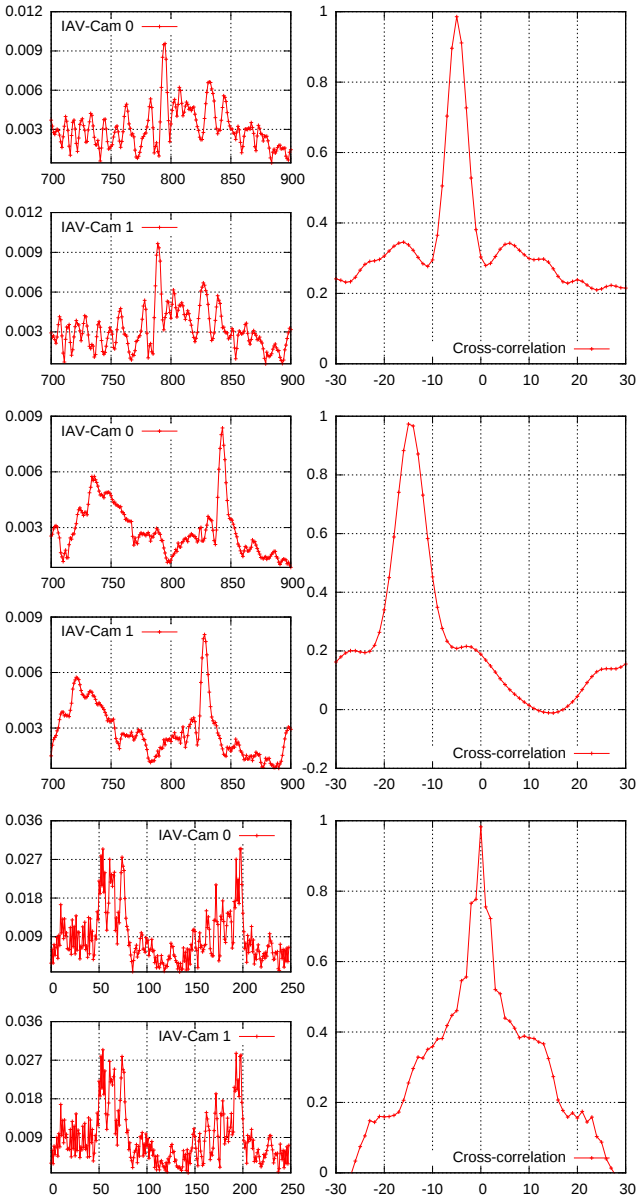

Figure 7: IAV for two cameras (left) and their correlation curves (right) for rectilinear trajectory segments of trajectories of BC1 (top), WT (middle) and CC (bottom). In the left, we have frame numbers (x-axes) and IAVs in radian (y-axes). In the right, we have offset candidates $o_{0,1} \in \mathbb{Z}$ (x-axes) and its correlation $ZNCC_{0,1} \in [-1, 1]$ (y-axes).

	$o_{0,1}$	$o_{1,2}$	$o_{2,3}$	$o_{3,4}$	$Zncc_1$	$Zncc_2$
BC1	-5	3	4	na	3.912	3.884
WT	-15	-1	14	na	3.919	3.918
FH	-1	1	-2	na	3.991	3.983
BC2	0	0	0	na	3.915	3.907
CC	0	0	0	0	4.987	4.347
WU	0	na	na	na	0.993	0.677

Table 2: FA time offsets $o_{j,j+1}$ with loop constraint. $Zncc_1$ is the greatest sum of the ZNCCs of the n computed time offsets, and $Zncc_2$ is the second greatest ZNCC (thus $-n \leq Zncc_i \leq +n$). We remind that $o_{j,j+1}$ counts a signed number of frames between the j-th and the $j+1$-th videos (it is "na" if the multi-camera has less than $j+2$ cameras, i.e. if $n < j+2$).

WT and FH. Different experiments have different $o_{j,j+1}$ although they are taken by the same cameras. Thus synchronization should be done at every experiment. Furthermore, the wifi-based synchronization of the Gopro is not very accurate: about 0.04s and sometimes above 0.1s (reminder: their FpS is 100Hz or 48Hz). Second we check that the $o_{j,j+1}$ are FA accurate when their ground truths are known (BC2, CC and WU).

8.4. Intrinsic parameters using GS.X.FA.INT

Before exploring BA for rolling shutter and synchronization in the next Sections, Sec. 8.4 experiments BA using two approximations: global shutter and zero sub-frame residual time offsets. In other words, we investigate the BA in [17] with two modifications: estimating the intrinsic parameters and minimizing the reprojection errors in the right image space. Sec. 8.4.1 compares error minimizations in the original and rectified image spaces using the polynomial distortion model. Sec. 8.4.2 compares results obtained without or with the central approximation.

We remind that the BA input is obtained as summarized in Sec. 3: initialization of multi-camera calibration on the video beginning (SfM and GS.C.FA.INT applied to the 2k first frames) followed by multi-camera SfM applied to the whole video. Note that the FoV angles used for equiangular initialization (Sec. 4) are in the FoV column of Tab. 1.

8.4.1. Polynomial distortion model: original vs. rectified errors

Tab. 3 compares the calibrations obtained by minimizing the reprojection errors in original (i.e. distorted) and rectified (i.e. undistorted) image spaces using GS.C.FA.INT applied to the 2k first frames of the CC video (the beginning of our method). Column "init" gives details on the initialization of the calibration: "72" means that we use an initial FoV angle equals to $360/5°$ for monocular camera and "pat" means that we use the calibration estimated using a planar calibration pattern [30]. Although the numbers of 2D inliers are similar, the calibration error d of the distorted case is quite better (about 6 times smaller) than that of the rectified one. Such a difference can be explained as follows: there are large distortions between rectified and distorted images, the BAs minimize errors in different image spaces (rectified and distorted), and the distorted space is the right one to obtain a Maximum Likelihood Estimator (more details at the end of Sec. 2.4). Tab. 3 also provides the numbers of 2D inliers and RMS for GS.FA that enforces calibration "pat" during BA; our RMS and inliers are slightly better but the calibration error d of "pat" is the best. In the paper remainder, we always use the reprojection error in the original space.

Error	init	Method	#2D	RMS	d
Rectified	72	gs.c.fa.int	213335	1.216	9.575
	pat	gs.fa	213015	1.225	1.023
Distorted	72	gs.c.fa.int	213495	0.932	1.683
	pat	gs.fa	213108	0.946	1.023

Table 3: Comparing accuracy of gs.fa.X using rectified and distorted reprojection errors on 2k first frames of CC (reminder: d is in pixels).

Method	d	RMS	#2D	d	RMS	#2D
gs.c.fa	3.379	0.728	204k	1.685	0.938	965k
gs.c.fa.int	2.018	0.723	204k	1.173	0.938	965k
gs.nc.fa	3.397	0.727	204k	1.684	0.938	965k
gs.nc.fa.int	1.417	0.723	204k	1.313	0.937	965k

Table 4: Accuracies of gs.fa.X for BC2 (left) and CC (right).

8.4.2. Central vs. non-central

Tab. 4 compares calibration error d (and RMS and 2D inliers) obtained using GS.C.FA.INT, GS.NC.FA.INT, GS.C.FA and GS.NC.FA applied to videos BC2 and CC. First GS.C.FA.INT provides a better (smaller) d than GS.C.FA since the intrinsic parameters (INT) are estimated from a longer video. The comparison between GS.NC.FA.INT and GS.NC.FA is similar. Second we compare GS.C.FA and GS.NC.FA and see that the non-central refinement (NC) changes almost nothing if INT is not refined. If INT is refined, the NC refinement improves the BC2 calibration but it does not improve (even degrades) the CC calibration. We interpret this result as follows: the central approximation is more tenable for CC than for BC2, since CC has a larger ratio between camera-scene distance and baseline than BC2. We also note that the RMS and 2D inliers are similar in all cases.

Tab. 5 provides accuracies of the intrinsic parameters of the first BC2 camera using GS.C.FA.INT and GS.NC.FA.INT. The absolute errors of f_x, f_y, u_0, v_0 are about 2 pixels or less; the relative errors of k_1 is good and those of k_i are bad if $i > 2$. The NC-values of f_x, f_y and u_0 are slightly better than those of C.

8.5. Rolling shutter and sub-frame accurate synchronization

First Sec. 8.5.1 provides all estimation errors of several BAs for videos BC2 and CC, that have complete ground truth. Second Sec. 8.5.2 provides SFA time offsets, line delay, and top view of reconstruction for every video using RS.C.SFA.INT.

8.5.1. Accuracies

Tab. 6 provides the errors $e(\Delta)$, $e(\tau)$ and d (Sec. 8.2) for several BAs estimating both SFA time offsets Δ_j and line delay τ (Sec. 6). We compare GS.C.FA.INT+RS.C.SFA and

		gs.c.fa.int		gs.nc.fa.int	
	G.T.	value	error	value	error
f_x	580.773	582.809	2.037	581.296	0.523
f_y	581.266	582.844	1.578	582.605	1.339
u_0	640.827	640.989	0.162	640.978	0.151
v_0	469.056	471.305	2.249	471.540	2.284
k_1	0.368	0.368	7e-4	0.368	6e-4
k_2	0.067	0.063	0.061	0.062	0.062
k_3	0.013	0.026	1.001	0.025	0.970
k_4	0.002	-0.013	6.463	-0.013	6.378
k_5	0.013	0.018	0.434	0.018	0.420

Table 5: Accuracies of intrinsic parameters of the first camera (BC2) using gs.X.fa.int. Reminder: we use absolute errors in pixels for f_x, f_y, u_0, v_0 and relative errors for the k_is.

Methods applied to BC2	$e(\Delta)$	$e(\tau)$	d
gs.c.fa.int+rs.c.sfa	0.057	14.6%	1.970
rs.c.sfa.int	0.097	2.7%	1.476
gs.nc.fa.int+rs.nc.sfa	0.051	12.2%	1.312
rs.nc.sfa.int	0.111	3.7%	0.366
gs.sfa (in Sec. 5.2)	0.215	na	na
Methods applied to CC	$e(\Delta)$	$y_{max}f\tau$	d
gs.c.fa.int+rs.c.sfa	0.052	-0.0052	1.176
rs.c.sfa.int	0.055	-0.0069	1.167
gs.nc.fa.int+rs.nc.sfa	0.034	-0.0020	1.322
rs.nc.sfa.int	0.039	0.0086	1.313
gs.sfa (in Sec. 5.2)	6e-3	na	na

Table 6: Accuracies of rs.sfa.X.(int) for BC2 and for CC.

	$f\Delta_1$	$f\Delta_2$	$f\Delta_3$	$y_{max}f\tau$	GT	$e(\tau)$
BC1	-0.334	-0.153	0.132	0.8755	0.8736	0.2%
WT	-0.583	-0.320	-0.795	1.013	0.8736	16.0%
FH	0.287	0.203	-0.326	0.8372	0.7810	7.2%
BC2	0.246	0.546	0.797	0.8989	0.8755	2.7%
CC	-0.017	-0.013	-0.006	-0.0069	0	nan
WU	0.001	na	na	-0.8882	-0.9244	3.9%

Table 7: SFA time offsets and line delay accuracy for all datasets using rs.c.sfa.int. Here GT is the ground truth of $y_{max}f\tau$.

RS.C.SFA.INT, i.e. we compare separate and simultaneous estimations of INT and RS.SFA parameters. We also compare these central BAs and their non-central versions, and the SFA synchronization without BA in Sec. 5.2.

First we experiment on the only RS sequence that has complete ground truth: BC2. We see that the simultaneous estimation of INT and RS.SFA has quite smaller $e(\tau)$ and smaller d than separate estimations (both C and NC BAs). However the separate case has a twice smaller $e(\Delta)$ than the simultaneous case, which in turn is more than twice smaller than that of the SFA refinement without BA. We remind that $e(\Delta)$ cumulates absolute errors of SFA synchronization: a value of 0.1 (for the simultaneous case) means that the mean SFA sync. error of n cameras is only $0.1/(n-1)$. The NC BAs also greatly reduce d.

Second we experiment on CC, which is the only real sequence with complete ground truth. Since it is GS, the relative error $e(\tau)$ is not a number and we replace it by $y_{max}f\tau$ (the smaller absolute value, the best result). We see that all BAs provides small $|y_{max}f\tau|$ compared to that of consumer RS cameras: we obtain values in $[0.002, 0.009]$, which are small compared to typical values in $[0.8, 0.9]$ of consumer RS cameras. Furthermore, all $e(\Delta)$ are smaller than 0.055 for five cameras; d increases and $e(\Delta)$ decreases by the NC BAs. The SFA refinement without BA provides the smallest $e(\Delta)$.

8.5.2. Time offsets, line delays and reconstruction

Tab. 7 shows time offsets $f\Delta_j$, normalized line delay $y_{max}f\tau$ and error $e(\tau)$ for all videos by applying RS.C.SFA.INT. We see that error $e(\tau)$ is less than 7.2% except in the WT case. In the WT case, τ is over-estimated ($y_{max}f\tau$ is even greater than its theoretical maximum value 1) and has large error $e(\tau)$ equal to 16%. In contrast to this, $e(\tau)$ in the BC1 case looks lucky. In fact, the τ value in a single experiment should be moderated since it depends on the keyframe choice (this will be experimented in Sec. 8.6). Note that a negative τ (for WU) simply means that the time of the y-th line increases when y decreases.

Last Figs. 8 and 9 show a top view of the RS.C.SFA.INT reconstructions (both keyframes locations and 3D point cloud). In the WU case, we observe a non-negligible drift since the beginning and end of the trajectory should be the same (the drift is less noticeable in the other examples). There are several reasons: we do not enforce loop closure, the incremental multi-camera SfM by local BA [28] is done using an intermediate calibration computed from only 2k first frames, and the final BA (RS.C.SFA.INT) does not remove the drift. We redo the incremental SfM using the final multi-camera calibration (computed from the whole sequence by RS.C.SFA.INT) and see that an important part of drift is removed. This suggests that the final multi-camera calibration is better than the intermediate one.

8.6. Stability with respect to keyframe sampling

Now we experiment the stability of our results (SFA synchronization, line delay and calibration) with respect to moderated changes of the keyframe sampling. The keyframe sampling is tuned by a single threshold N_3, which is a lower bound for the number of matches between three consecutive keyframes (more details in Appendix G). For every value $N_3 \in \{400, 425, 450, 475, 500\}$, we apply multi-camera SfM based on keyframe sampling followed by RS.C.SFA.INT and then discuss the results. The initial multi-camera calibration and FA synchronization are the same for all N_3 and are computed from the video beginning as in the other experiments.

The left of Tab. 8 shows estimation errors $e(\Delta)$, $e(\tau)$ and d. There are 206 keyframes if $N_3 = 400$ and 248 keyframes if $N_3 = 500$. The variations of errors are important: from single to double for $e(\Delta)$, from single to quadruple for $e(\tau)$, and about 30% for d. We provide an explanation in Sec. 8.6.1 and a correction of the results in Sec. 8.6.2. Tab. 9 shows time offsets $f\Delta_j$ and error $e(\tau)$ for the longest sequence BC1. The variation

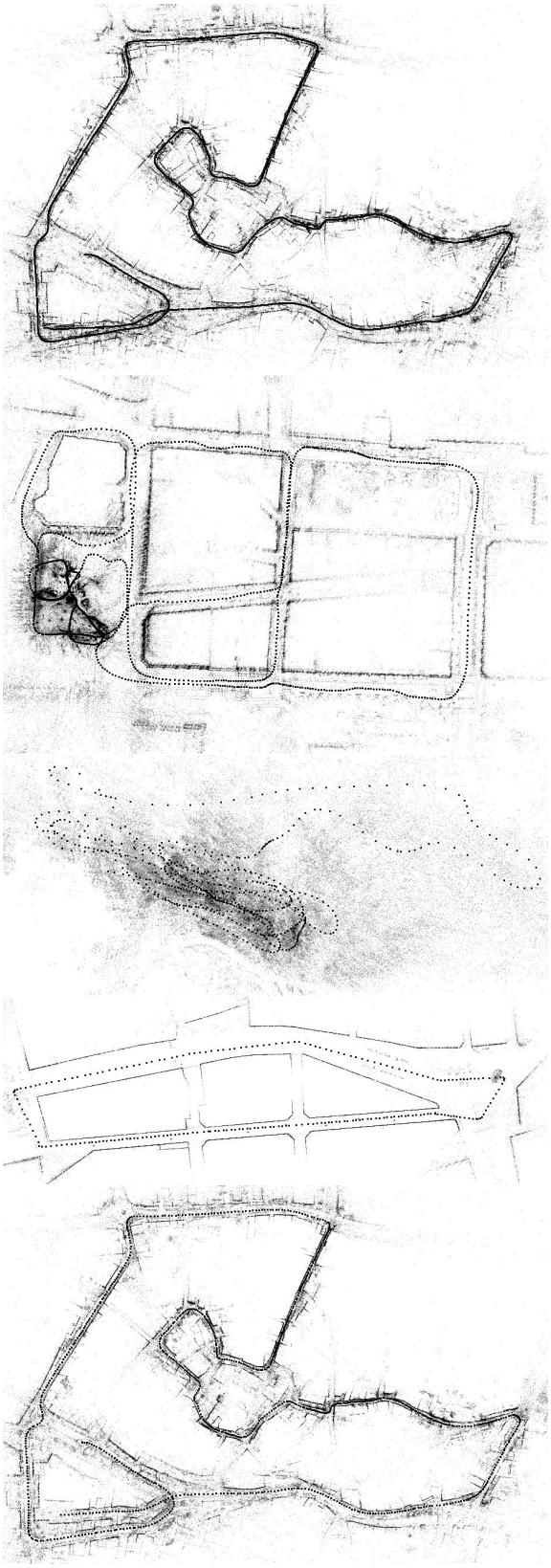

Figure 8: Top views of RS.C.SFA.INT reconstructions of BC1, WT, FH, BC2 and CC (from top to bottom) without loop closure. The input videos of BC1, WT and FH are taken by four Gopro cameras mounted on a helmet. The FH trajectory has a lot of sharp S turns.

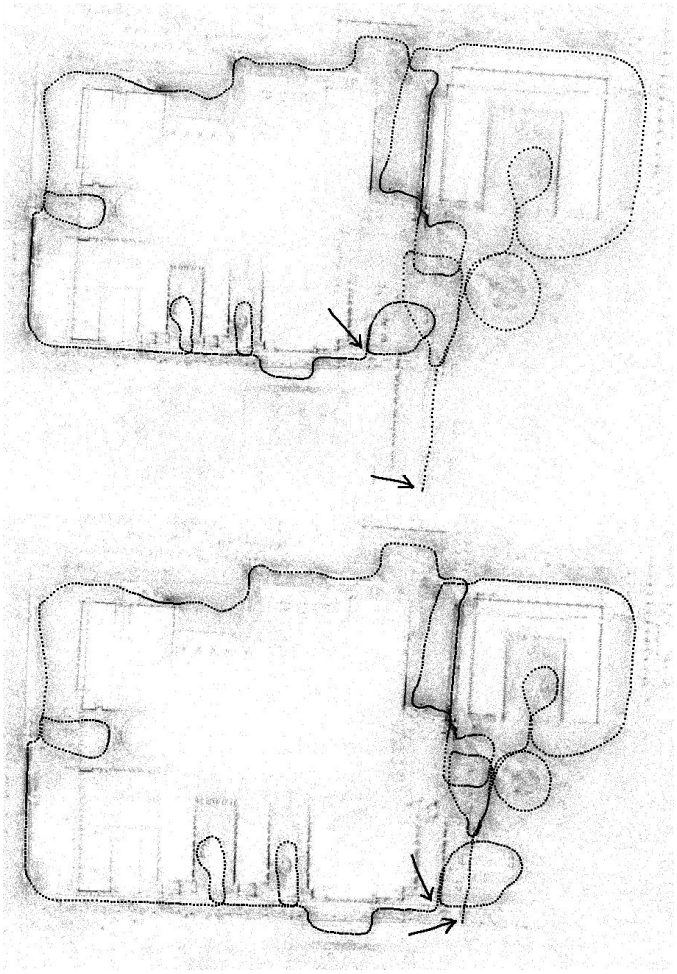

Figure 9: Top views of WU reconstructions without loop closure. The input video is taken by the Ricoh Theta S mounted on a helmet. The drift is between the two arrows. Top: result of RS.C.SFA.INT. Bottom: incremental SfM [28] that is redone using the calibration estimated by RS.C.SFA.INT.

	rs.c.sfa.int			rs.c.sfa.int+rs.c.sfa.int.h		
N_3	$e(\Delta)$	$e(\tau)$	d	$e(\Delta)$	$e(\tau)$	d
400	0.089	1.9%	1.466	0.091	2.4%	1.458
425	0.131	3.0%	1.570	0.123	0.8%	1.563
450	0.097	2.7%	1.476	0.110	2.8%	1.541
475	0.191	7.4%	1.867	0.136	5.5%	1.710
500	0.199	2.7%	1.664	0.101	0.4%	1.499
mean	0.141	3.5%	1.609	0.112	2.4%	1.554
max/min	2.23	4.0	1.27	1.49	12.3	1.17

Table 8: Accuracy stability with respect to keyframe sampling for BC2 using rs.c.sfa.int (left) and rs.c.sfa.int+rs.c.sfa.int.h (right) with $h = 70\%$.

N_3	kfr	$f\Delta_1$	$f\Delta_2$	$f\Delta_3$	$y_{max}f\tau$	$e(\tau)$
400	1813	-0.341	-0.171	0.131	0.8920	2.1%
425	1929	-0.337	-0.155	0.131	0.8882	1.6%
450	2047	-0.334	-0.153	0.132	0.8755	0.2%
475	2166	-0.366	-0.156	0.134	0.9399	7.5%
500	2256	-0.337	-0.141	0.130	0.8786	0.5%
mean	2042	-0.343	-0.155	0.132	0.8948	2.4%
mx-mn	443	0.032	0.030	0.004	0.0644	7.3%

Table 9: Stabilities of time offsets and line delay with respect to keyframe sampling for BC1 using rs.c.sfa.int. The number of keyframe is kfr.

of $e(\tau)$ are also important; the variation of $f\Delta_j$ are less than 0.032.

8.6.1. Analysis

We remind that the reprojection errors in the i-th keyframe are computed using the approximation in Eq. 12 of the multi-camera trajectory $M(t)$ where $t \approx t_i$: $M(t)$ is a linear combination of $\mathbf{m}_{i-1}$, $\mathbf{m}_i$ and $\mathbf{m}_{i+1}$, and M is linear in time $t - t_i$. The better this approximation, the smaller the reprojection errors in the i-th keyframe.

At first glance, the estimation errors decrease if N_3 increases: if N_3 increases, the keyframe density increases, thus the accuracy of these approximations is better (the remainders $O(\Delta)$ and $O(\Delta^2)$ in Sec. 6.4 decrease), the reprojection errors decrease and last the estimation errors decreases). However, the errors in Tab. 8 are not decreasing series but look noisy.

Here is a second explanation. The true value of $M(t)$ near t_i can be different to its approximated value for some i, e.g. if the true speed vector $M'(t_i)$ is different to the speed vector D_1^i computed using the multi-camera poses of neighboring keyframes $i-1$ and $i+1$ in Eq. 11. Then a keyframe with bad approximation has high reprojection errors and act as an outlier perturbing the BA. The estimation errors in the left of Tab. 8 depend on the set of this kind of outliers, which in turn depends on N_3.

8.6.2. Correction

The idea is simple: if the i-th keyframe has high reprojection errors, we redefine its approximation by $M(t) = \mathbf{m}_i + (t - t_i)\mathbf{d}_i$ if $t \approx t_i$ thanks to a new velocity parameter $\mathbf{d}_i \in \mathbb{R}^6$ that is estimated by BA like $\mathbf{m}_i$. Then the camera motion is not constrained by keyframes $i-1$ and $i+1$ if $t \approx t_i$, and we expect that the resulting reprojection errors of the i-th keyframe decrease such that the i-th keyframe does no act as an outlier of the BA.

Name	$\#\mathbf{d}_i$	$f\Delta_1$	$f\Delta_2$	$f\Delta_3$	$y_{max}f\tau$	$e(\tau)$
BC1	141	-0.349	-0.152	0.112	0.9177	5.0%
WT	126	-0.541	-0.322	-0.789	0.9139	4.6%
FH	130	0.284	0.208	-0.329	0.8435	8.0%
BC2	15	0.261	0.548	0.801	0.9001	2.8%
CC	3	-0.004	-0.015	-0.011	-0.0009	nan
WU	131	-0.001	na	na	-0.8772	5.1%

Table 10: SFA time offsets and line delay accuracy for all datasets using rs.c.sfa.int+rs.c.sfa.int.h with $h = 70\%$ (to be compared with Tab. 7). The number of keyframes with additional velocity parameter is $\#\mathbf{d}_i$.

In practice, we start from a current estimation obtained by RS.C.SFA.INT and introduce an user defined percentage h. Let c_h be the h-fractile over all reprojection errors. For every keyframe, we compute the RMS of its own reprojection errors. If this RMS is greater than c_h, the keyframe has an additional velocity parameter $\mathbf{d}_i$ as above (otherwise it does not have). We name RS.C.SFA.INT.h the new BA obtained by modifying RS.C.SFA.INT like this.

Tab. 8 provides estimation errors $e(\Delta)$, $e(\tau)$, d of both RS.C.SFA.INT and RS.C.SFA.INT+RS.C.SFA.INT.h using $h = 70\%$. Thanks to the correction, all errors are improved in the following sense: both mean and maximum of every error are reduced, the variations of $e(\Delta)$ and d are damped (the variations of $e(\tau)$ expressed using ratio max/min are not damped due to a small error 0.4% for $N_3 = 500$).

Last Tab. 10 shows time offsets $f\Delta_j$, normalized line delay $y_{max}f\tau$ and error $e(\tau)$ for all videos by applying RS.C.SFA.INT+RS.C.SFA.INT.h. We see that all errors $e(\tau)$ have the same magnitude order (in interval $[2.8, 8]$). This contrasts to Tab. 7 using $N_3 = 450$, where $e(\tau)$ is small for BC1 and large for WT.

8.7. Robustness with respect to FA synchronization

We would like to know whether an error in the FA synchronization can be corrected by SFA synchronization. Such an error can have two reasons: the IAV variations are insufficient in the video beginning for FA accurate synchronization, or a consumer camera skips frame(s) for any technical reasons. We simulate such an error by skipping x frames of camera 1 (reminder: camera 0 is the first one), then we use multi-camera SfM followed by RS.C.SFA.INT and compare the results for $x \in \{0, 1, 2, 3\}$.

Tab. 11 shows errors $e(\Delta)$, $e(\tau)$, d and time offsets $f\Delta_j$ estimated for BC2. We see that $e(\Delta)$ and d increases moderately if $x = 1$, $e(\Delta)$ is multiplied by 2.5 and $e(\tau)$ by 3.9 if $x = 2$, and all errors increase a lot if $x = 3$.

8.8. Variations of τ and Δ_j in a long sequence

Here we examine the variations of τ and Δ_j in a long sequence. We split the GS.C.FA.INT reconstruction of BC1 into six segments of 300 keyframes (segments 0-299, 300-599, etc) and independently apply RS.C.SFA.INT to every segment. Tab. 12 shows the results.

x	$f\Delta_1$	$f\Delta_2$	$f\Delta_3$	$e(\Delta)$	$e(\tau)$	d
0	0.246	0.546	0.797	0.097	2.7%	1.476
1	1.342	0.505	0.769	0.117	1.0%	1.743
2	2.026	0.489	0.760	0.245	10.5%	1.298
3	2.472	0.532	0.781	0.841	10.4%	3.244

Table 11: Accuracies of rs.c.sfa.int applied to BC2 if we skip x additional frame(s) of camera 1. The ideal result meets $f\Delta_1 = x + 0.25$.

s	$f\Delta_1$	$f\Delta_2$	$f\Delta_3$	$y_{max}f\tau$	$e(\tau)$
0	-0.443	-0.180	0.014	1.0570	20.1%
1	-0.417	-0.152	0.014	0.9347	6.9%
2	-0.306	-0.073	0.028	0.7678	12.2%
3	-0.308	-0.163	0.115	0.9225	5.6%
4	-0.271	-0.157	0.178	0.9056	3.7%
5	-0.299	-0.105	0.199	0.8394	3.9%
mean	-0.341	-0.138	0.091	0.9045	8.7%
max-min	0.172	0.107	0.185	0.2892	16.4%

Table 12: Stabilities of time offsets and line delay over time in long sequence BC1 using rs.c.sfa.int. The s-th video segment is taken between keyframes $300s$ and $300(s+1)-1$.

The variations of $f\Delta_j$ are moderated (less than 0.2) and the $f\Delta_j$ globally increase over time, i.e. when s increases. At first glance, we could expect that we can detect a frame skipped by a camera (if any) by an increase/decrease of 1 as in Sec. 8.7. However this is not the case. Since all cameras are in the same manufacturing series and have the same setting, all cameras skip similar numbers of frames (if any) in a segment, which in turn would imply that we do not observe large time offset perturbations. Furthermore we could interpret the slow increase of the $f\Delta_j$s as follows: the FpS of camera 0 is slightly lower than those of the other cameras.

The variations of τ are important (especially in the first half part of BC1) and τ globally decreases over time. The variations of τ are reduced if we take a larger video segment size, e.g. we divide by two the $y_{max}f\tau$ range (max-min in Tab. 12) with 500 keyframes per segment.

8.9. Other experiments

All previous experiments are done using the linear approximation of $M(t)$ in Eq. 12 and using the approximation $t_{\mathbf{p}} = t_{\tilde{\mathbf{p}}}$ in Eq. 17. Tab. 13 shows the results for all videos using the quadratic approximation of $M(t)$ in Eq. 15 or using Sec. 7.3 (i.e. without the approximation $t_{\mathbf{p}} = t_{\tilde{\mathbf{p}}}$ in Eq. 17). We do not observe significant improvements of $e(\tau)$ compared to those in Tab. 7 except for FH using Eq. 15. By recomputing errors $e(\Delta)$ and d for BC2 and CC using these two changes, we obtain very similar results as in Tab. 6: $e(\Delta)$ difference is less than 0.004 and d difference is less than 0.01.

Name	$f\Delta_1$	$f\Delta_2$	$f\Delta_3$	$y_{max}f\tau$	$e(\tau)$
rs.c.sfa.int using Eq. 15 (instead of Eq. 12)					
BC1	-0.337	-0.157	0.127	0.8624	1.3%
WT	-0.580	-0.323	-0.791	0.9974	14.2%
FH	0.284	0.201	-0.326	0.8252	5.7%
BC2	0.249	0.551	0.798	0.9020	3.0%
CC	-0.017	-0.013	-0.005	-0.0082	nan
WU	-2e-4	na	na	-0.8896	3.8%
rs.c.sfa.int using Sec. 7.3 (without approx. $t_{\mathbf{p}} = t_{\tilde{\mathbf{p}}}$ in Eq. 17)					
BC1	-0.334	-0.153	0.131	0.8758	0.25%
WT	-0.581	-0.320	-0.793	1.0061	15.2%
FH	0.287	0.203	-0.326	0.8381	7.3%
BC2	0.245	0.547	0.796	0.9028	3.1%
CC	-0.017	-0.014	-0.005	-0.0096	nan
WU	0.001	na	na	-0.8689	6.0%

Table 13: SFA time offsets and line delay accuracy for all datasets using rs.c.sfa.int (to be compared with Tab. 7).

9. Conclusion

This article introduces the first self-calibration method for a multi-camera moving in an scene, that simultaneously estimates intrinsic parameters, inter-camera poses, time offsets and line delay in addition to the usual parameters (3D points and multi-camera poses). We start by a rough calibration assuming that the multi-camera is central and omnidirectional without privileged direction. Then we estimate frame-accurate time offsets using monocular structure-from-motion and bundle adjustment (SfM and BA) without assumption on the field-of-view shared by adjacent cameras. Last we apply multi-camera SfM and BA twice: using simple and complicated camera models. The former forces to 0 line delay, sub-frame residual time offsets and baseline between cameras; then the former initializes the latter.

We experiment in a context that we believe useful for applications (3D modeling and 360 videos): several consumer cameras or a spherical camera mounted on a helmet and moving along long trajectories by walking and biking (among others). Long trajectories are useful for calibration accuracy and are allowed since our BA only refines the keyframes provided by SfM. We compare central and non-central results, provide accuracy for calibration/line delay/time offsets with respect to ground truth, examine the influence of the tuning of keyframe selection, show variations of time offsets in a long sequence, experiment and compare different approximations (for time continuous camera trajectory and for reprojection errors).

However the method has a limitation: the image deformations due to rolling shutter should be moderated for SfM (before the final BA that refines all parameters including line delay) since SfM uses the global shutter approximation. Furthermore several improvements and future work are possible. First the initialization, which is not the paper topic, can be improved thanks to previous work for both intrinsic parameters and inter-camera poses. Second a preprocessing should select segment(s) in the video where we safely apply SfM. Third variants of the method can be experimented, e.g. by using non-minimal parametrization of rotations, alternative keyframe selections and other camera models. Last, we should examine the improvements in applications provided by our non-zero line delay and sub-frame-accurate time offsets.

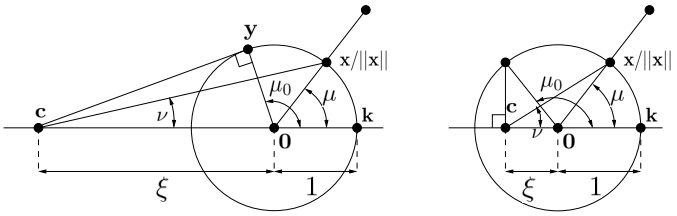

Figure A.10: Notations in two cases: $\xi > 1$ (left) and $0 < \xi < 1$ (right).

Acknowledgments

Thanks to CNRS, Université Clermont Auvergne and Institut Pascal for funding Maxime Lhuillier and Thanh-Tin Nguyen.

Appendix A. Properties of the unified camera model

We remind that this model is described in Sec. 4.2 using a perspective camera. Let $\mathbf{k} = \begin{pmatrix}0 & 0 & 1\end{pmatrix}^\top$, $\mathbf{c} = \begin{pmatrix}0 & 0 & -\xi\end{pmatrix}^\top$ and 3D point $\mathbf{x}$. Let ν be the angle between $\mathbf{k}$ and a ray of the perspective camera (a half-line started at $\mathbf{c}$ and including $\mathbf{x}/\|\mathbf{x}\|$). Let μ be the angle between $\mathbf{k}$ and a ray of the unified camera (a half-line started at $\mathbf{0}$ with direction $\mathbf{x}/\|\mathbf{x}\|$).

Appendix A.1. Theoretical field-of-view

Fig. A.10 shows notations ν, μ, μ_0 (a value of μ), $\mathbf{x}/\|\mathbf{x}\|$, $\mathbf{c}$ and $\mathbf{k}$ in two cases: $\xi > 1$ and $0 < \xi < 1$. In both cases, μ_0 is the maximum value of μ that ensures that there is only one back-projected ray direction corresponding to the projection $p(\mathbf{x})$ by the unified camera model. The angle μ_0 is the half-angle of the *theoretical* FoV of this model, i.e. by ignoring the bounded size of the image and the projection of the camera itself. The FoV in the paper core ignores nothing and is included in the theoretical FoV.

If $\xi = 0$, the unified camera model is the standard perspective camera model with the camera center $\mathbf{c} = 0$. Since $\mathbf{x}/\|\mathbf{x}\|$ is in front of the perspective camera, the theoretical FoV is the half-space $z > 0$ and $\mu_0 = \pi/2$.

If $0 < \xi < 1$, $\mathbf{c}$ is inside the unit sphere $\mathcal{S}$. Since $\mathbf{x}/\|\mathbf{x}\|$ is in front of the perspective camera, the theoretical FoV is the half-space $z > -\xi$ and $\cos\mu_0 = -\xi$.

If $\xi > 1$, $\mathbf{c}$ is outside $\mathcal{S}$ and $\mathcal{S}$ is entirely in front of the perspective camera. The projection of $\mathcal{S}$ by p is an ellipse and its interior. Let C be the cone that is tangent to $\mathcal{S}$ with apex $\mathbf{c}$, i.e. the union of every line $(\mathbf{cy})$ that intersects $\mathcal{S}$ at a single point $\mathbf{y}$. We have $\mathbf{y}^\top(\mathbf{c}-\mathbf{y}) = 0$ and $\mathbf{k}^\top\mathbf{y} < 0$ and $\cos\mu_0 = \mathbf{k}^\top\mathbf{y} = 1/(-\xi)$. For example, $\xi = 2$ (in Sec. 4.2.2) implies that $\mu \leq \mu_0 = 2\pi/3$.

Appendix A.2. Angle of back-projected ray

There is a relation between ν and μ in all cases. Using notation $\begin{pmatrix}x_1 & x_2 & x_3\end{pmatrix}^\top = \mathbf{x}/\|\mathbf{x}\|$, we have $\cos\mu = x_3$ and $\sin\mu = \sqrt{x_1^2 + x_2^2}$. Since $\mathbf{x}/\|\mathbf{x}\|$ is in front of the perspective camera, $\xi + x_3 > 0$ and $\tan\nu = \sqrt{x_1^2 + x_2^2}/(\xi + x_3)$. Thus

$$\tan\nu = \frac{\sin\mu}{\xi + \cos\mu}. \tag{A.1}$$

Since $p(\mathbf{x}) = (f_x x_1/(\xi + x_3) + u_0, f_y x_2/(\xi + x_3) + v_0)$ and $p(\mathbf{k}) = (u_0, v_0) = \mathbf{z}_0$,

$$f_x = f_y = f \Rightarrow \|p(\mathbf{x}) - \mathbf{z}_0\|/f = \tan\nu = \frac{\sin\mu}{\xi + \cos\mu}. \tag{A.2}$$

Appendix B. IAV property

Here we show that the IAV in Eq. 8 is the same for two frames of different but jointly moving cameras if they are taken at the same time. We remind properties of a change of basis in $\mathbb{R}^3$ expressed by rotation matrices: the columns of $\mathtt{R}_{A,B}$ are vectors of B expressed using coordinates in A, we have $\mathtt{R}_{A,B}^\top = \mathtt{R}_{B,A}$ and $\mathtt{R}_{A,B} = \mathtt{R}_{A,C}\mathtt{R}_{C,B}$.

Since the monocular SfMs are not done in the same coordinate system, the notation $\mathtt{R}_i^t$ used in Eq. 8 is ambiguous. Here we write instead $\mathtt{R}_{w_i,i}^t$ where w_i is the (world) basis where is reconstructed the i-th video (vectors of the i-th camera at frame t expressed using coordinates in w_i). Furthermore, we note that $\mathtt{R}_{w_i,w_j}$ and $\mathtt{R}_{i,j}$ do not depend on frame numbers (the former is obvious, the latter is due to the fact that the cameras are rigidly mounted). If the t_i-th frame of the i-th camera and the t_j-th frame of the j-th camera are taken at the same time,

$$\mathtt{R}_{w_j,w_i}\mathtt{R}_{w_i,i}^{t_i} = \mathtt{R}_{w_j,j}^{t_j}\mathtt{R}_{j,i}. \tag{B.1}$$

Since the cameras have the same FpS, we also have

$$\mathtt{R}_{w_j,w_i}\mathtt{R}_{w_i,i}^{t_i+1} = \mathtt{R}_{w_j,j}^{t_j+1}\mathtt{R}_{j,i}. \tag{B.2}$$

Thanks to Eqs. B.1 and B.2, we obtain

$$\mathtt{R}_{w_j,j}^{t_j+1}(\mathtt{R}_{w_j,j}^{t_j})^\top = \mathtt{R}_{w_j,w_i}\mathtt{R}_{w_i,i}^{t_i+1}(\mathtt{R}_{w_i,i}^{t_i})^\top\mathtt{R}_{w_j,w_i}^\top. \tag{B.3}$$

Since $\mathrm{trace}(\mathtt{XY}) = \mathrm{trace}(\mathtt{YX})$, we obtain

$$\mathrm{trace}(\mathtt{R}_{w_j,j}^{t_j+1}(\mathtt{R}_{w_j,j}^{t_j})^\top) = \mathrm{trace}(\mathtt{R}_{w_i,i}^{t_i+1}(\mathtt{R}_{w_i,i}^{t_i})^\top). \tag{B.4}$$

Thus $\theta_j^{t_j} = \theta_i^{t_i}$ according to Eq. 8.

Appendix C. Derivatives of the camera motion $M(t)$

The notations used in Appendix C.1 and Appendix C.2 are defined in Secs. 6.4.1 and 6.4.2, respectively.

Appendix C.1. Approximation of $M'(t_i)$

First we show that

$$M'(t) = D_1(M(t-b), M(t), M(t+a), a, b) + O(a^2 + b^2) \tag{C.1}$$

if $a > 0$ and $b > 0$. Since M is C^3 continuous,

$$M(t+a) = M(t) + aM'(t) + \frac{a^2}{2}M''(t) + O(a^3) \tag{C.2}$$

$$M(t-b) = M(t) - bM'(t) + \frac{b^2}{2}M''(t) + O(b^3). \tag{C.3}$$

We eliminate $M''(t)$ by summing $\frac{b}{a}$(Eq. C.2) -$\frac{a}{b}$(Eq. C.3):

$$\frac{b}{a}M(t+a) - \frac{a}{b}M(t-b) = (\frac{b}{a} - \frac{a}{b})M(t)$$

$$+(b+a)M'(t) + bO(a^2) + aO(b^2). \quad \text{(C.4)}$$

Since $a > 0$ and $b > 0$,

$$M'(t) = \frac{1}{a+b}(\frac{b}{a}M(t+a) - \frac{a}{b}M(t-b) + (\frac{a}{b} - \frac{b}{a})M(t)) + O(a^2+b^2). \quad \text{(C.5)}$$

Second we use Eq. C.1 with $t = t_i$, $a = t_{i+1} - t_i$, $b = t_i - t_{i-1}$, $\Delta = \max_i(t_{i+1} - t_i)$, and obtain

$$M'(t_i) = D_1(M(t_{i-1}), M(t_i), M(t_{i+1}), t_{i+1} - t_i, t_i - t_{i-1}) + O(\Delta^2). \quad \text{(C.6)}$$

Appendix C.2. Approximation of $M''(t_i)$

First we show that

$$M''(t) = D_2(M(t-b), M(t), M(t+a), a, b) + O(a+b) \quad \text{(C.7)}$$

We eliminate $M'(t)$ by summing b(Eq. C.2)+a(Eq. C.3):

$$bM(t+a) + aM(t-b) = (a+b)M(t) + \frac{ab}{2}(a+b)M''(t) + O(ba^3 + ab^3). \quad \text{(C.8)}$$

Since $a > 0$ and $b > 0$,

$$M''(t) = \frac{2M(t+a)}{a(a+b)} + \frac{2M(t-b)}{b(a+b)} - \frac{2M(t)}{ab} + O(a+b). \quad \text{(C.9)}$$

Second we use Eq. C.7 with $t = t_i$, $a = t_{i+1} - t_i$, $b = t_i - t_{i-1}$, $\Delta = \max_i(t_{i+1} - t_i)$, and obtain

$$M''(t_i) = D_2(M(t_{i-1}), M(t_i), M(t_{i+1}), t_{i+1} - t_i, t_i - t_{i-1}) + O(\Delta). \quad \text{(C.10)}$$

Appendix D. Estimation of rotations A and B

Here we not only compute $\mathtt{A}$ and $\mathtt{B}$ in the definition of our rotation parametrization $\mathcal{R}$ (Eq. 20), but also initialize $E_M(t_i)$. Let $\mathtt{R}_i^0$ be the rotation of the i-th keyframe estimated by GS BA (before the final BA in Sec. 6). Using the definitions of E_M and t_i in Secs. 6.2 and 6.3, $E_M(t_i)$ is initialized such that $\mathcal{R}(E_M(t_i)) = \mathtt{R}_i^0$. We also check that $E_M(t_i)$ is far from the singularities of $\mathcal{R}$.

Let $\mathbf{k} = \begin{pmatrix}0 & 0 & 1\end{pmatrix}^T$. Since the multi-camera trajectory has small pitch and roll, there are rotations $\mathtt{A}$ and $\mathtt{B}$ such that $\forall i, \mathtt{A}^{-1}\mathtt{R}_i^0\mathtt{B}^{-1}$ is almost a rotation around $\mathbf{k}$ (details in Appendix D.1). Let angles $(\alpha_i, \beta_i, \gamma_i)$ be such that $\mathcal{E}(\alpha_i, \beta_i, \gamma_i) = \mathtt{A}^{-1}\mathtt{R}_i^0\mathtt{B}^{-1}$ and $\mathcal{E}$ is defined in Eq. 19 and β_i is close to 0 (details in Appendix D.2). We initialize $E_M(t_i) = \begin{pmatrix}\alpha_i & \beta_i & \gamma_i\end{pmatrix}^\top$. Thanks to Eqs. 20 and 19, we obtain

$$\mathcal{R}(E_M(t_i)) = \mathcal{R}(\alpha_i, \beta_i, \gamma_i) = \mathtt{A}\mathcal{E}(\alpha_i, \beta_i, \gamma_i)\mathtt{B} = \mathtt{R}_i^0. \quad \text{(D.1)}$$

According to Sec. 6.6.2, $E_M(t_i)$ is a singularity of $\mathcal{E}$ iff $\beta_i \in \pi/2 + \pi\mathbb{Z}$. Since $\mathcal{R}$ has the same singularities as $\mathcal{E}$ (proof in Appendix D.3) and β_i is close to 0, $E_M(t_i)$ is far from the singularities of $\mathcal{R}$.

Appendix D.1. Technical Details: Estimate $\mathtt{A}$ and $\mathtt{B}$

Let $R(\mathbf{v}, \theta)$ be the rotation with axis $\mathbf{v}$ and angle θ. Since the multi-camera trajectory has small pitch and roll (Sec. 6.6.2), all $(\mathtt{R}_i^0)^T\mathtt{R}_j^0$ are roughly rotations sharing a same axis $\mathbf{v} \in \mathbb{R}^3$. Thus there are rotation $\mathtt{R}$ and angle θ_i such that $\mathtt{R}_i^0 \approx \mathtt{R}R(\mathbf{v}, \theta_i)$. For all i and j, $(\mathtt{R}_i^0)^T\mathtt{R}_j^0 \approx R(\mathbf{v}, \theta_j - \theta_i)$. Let $\mathbf{v}_{i,j}$ be the axis of $(\mathtt{R}_i^0)^T\mathtt{R}_j^0$. First we search $\mathbf{v}$ as the most colinear vector to all $\mathbf{v}_{i,j}$, i.e. $\mathbf{v}$ maximizes $\sum_{i,j}(\mathbf{v}_{i,j}^T\mathbf{v})^2$. Thus $\mathbf{v}$ is the eigen vector of the largest eigen value of the symmetric matrix $\sum_{i,j}\mathbf{v}_{i,j}\mathbf{v}_{i,j}^T$. Second we estimate rotation $\tilde{\mathtt{R}}$ such that $\tilde{\mathtt{R}}\mathtt{R}_i^0 \approx R(\mathbf{v}, \theta_i')$. Since $\mathtt{R}_i^0 \approx \mathtt{R}R(\mathbf{v}, \theta_i)$, $\mathtt{R}_i^0\mathbf{v} \approx \mathtt{R}\mathbf{v}$. Let $\tilde{\mathbf{v}} = \sum_i \mathtt{R}_i^0\mathbf{v}/\|\sum_i \mathtt{R}_i^0\mathbf{v}\|$. Thus $\tilde{\mathbf{v}} \approx \mathtt{R}\mathbf{v} \approx \mathtt{R}_i^0\mathbf{v}$. Let $\tilde{\mathtt{R}}$ be a rotation such that $\tilde{\mathtt{R}}\tilde{\mathbf{v}} = \mathbf{v}$. Since $\tilde{\mathtt{R}}\mathtt{R}_i^0\mathbf{v} \approx \tilde{\mathtt{R}}\tilde{\mathbf{v}} = \mathbf{v}$, $\tilde{\mathtt{R}}\mathtt{R}_i^0 \approx R(\mathbf{v}, \theta_i')$. Third we estimate $\mathtt{A}$ and $\mathtt{B}$. Let $\mathtt{R}'$ be a rotation such that $\mathtt{R}'\mathbf{v} = \mathbf{k}$. We obtain $\mathtt{R}'\tilde{\mathtt{R}}\mathtt{R}_i^0\mathtt{R}'^T \approx R(\mathbf{k}, \gamma_i)$. Thus $\mathtt{A}^{-1} = \mathtt{R}'\tilde{\mathtt{R}}$ and $\mathtt{B}^{-1} = \mathtt{R}'^T$.

Appendix D.2. Estimate $(\alpha_i, \beta_i, \gamma_i)$

Since $\mathcal{E}$ is surjective on the set of 3D rotations, the angles α_i, β_i and γ_i exist. Furthermore, they are defined up 2π multiples. We choose β_i that has the smallest $|\beta_i|$. Since $\mathcal{E}(\alpha_i, \beta_i, \gamma_i) \approx R(\mathbf{k}, \gamma_i)$, β_i is close to 0. We also remind that $E_M(t)$ is continuous (Sec. 6.2) and $|t_i - t_{i+1}|$ is small thanks to the keyframe sampling. Thus the γ_i series is chosen such that $|\gamma_i - \gamma_{i-1}|$ is as small as possible, and we do similarly for α_i ($|\beta_i - \beta_{i-1}|$ is also small).

Appendix D.3. $\mathcal{R}$ and $\mathcal{E}$ have the same singularities

Here we show that $\ker\partial\mathcal{R} = \ker\partial(\mathtt{A}\mathcal{R}\mathtt{B})$ if $\mathtt{A}$ and $\mathtt{B}$ are two invertible 3×3 matrices. Let $\mathbf{x} \in \ker\partial\mathcal{R}$, $\mathcal{R}_{ij}$ be the coefficients of $\mathcal{R}$, and $\partial\mathcal{R}_{ij}$ be the gradient of $\mathcal{R}_{ij}$ with respect to parameters (α, β, γ). Thus $\partial\mathcal{R}_{ij}.\mathbf{x} = 0$ and

$$\begin{aligned}(\partial(\mathtt{A}\mathcal{R}\mathtt{B})_{ij}).\mathbf{x} &= (\partial(\sum_{k,l}\mathtt{A}_{ik}\mathcal{R}_{kl}\mathtt{B}_{lj})).\mathbf{x} \\ &= \sum_{k,l}\mathtt{A}_{ik}\mathtt{B}_{lj}(\partial\mathcal{R}_{kl}).\mathbf{x} = 0. \end{aligned} \quad \text{(D.2)}$$

We see that $\partial(\mathtt{A}\mathcal{R}\mathtt{B}).\mathbf{x} = 0$, i.e. $\ker\partial\mathcal{R} \subseteq \ker\partial(\mathtt{A}\mathcal{R}\mathtt{B})$. Since $\mathtt{A}$ and $\mathtt{B}$ are invertible, we use this inclusion (replace $\mathcal{R}$ by $\mathtt{A}\mathcal{R}\mathtt{B}$, replace $\mathtt{A}$ by $\mathtt{A}^{-1}$, replace $\mathtt{B}$ by $\mathtt{B}^{-1}$) and obtain

$$\ker\partial(\mathtt{A}\mathcal{R}\mathtt{B}) \subseteq \ker\partial(\mathtt{A}^{-1}(\mathtt{A}\mathcal{R}\mathtt{B})\mathtt{B}^{-1}) = \ker\partial\mathcal{R}. \quad \text{(D.3)}$$

Appendix E. Sparsity of Z in the RCS (Eq. 22)

Appendix E.1. Notations and prerequisites

If L_1 and L_2 are two lists of integers, we use notations

$$\begin{aligned}L_1 + L_2 &= \{i_1 + i_2, i_1 \in L_1, i_2 \in L_2\}, &\text{(E.1)}\\ L_1 \times L_2 &= \{(i_1, i_2), i_1 \in L_1, i_2 \in L_2\}, &\text{(E.2)}\\ (L_1)^2 &= L_1 \times L_1. &\text{(E.3)}\end{aligned}$$

We also use Eq. E.1 if L_1 and L_2 are two lists of integer pairs. If a matrix $\mathtt{A}$ is partitioned by blocks $\mathtt{A}_i$ (horizontally or vertically), we define an index list

$$L(\mathtt{A}) = \{i, \mathtt{A}_i \neq 0\}. \quad \text{(E.4)}$$

If a matrix $\mathtt{A}$ is partitioned by blocks $\mathtt{A}_{i,j}$ (both horizontally and vertically), we define a list of index pairs

$$L(\mathtt{A}) = \{(i,j), \mathtt{A}_{i,j} \neq 0\}. \tag{E.5}$$

If matrices $\mathtt{A}$ and $\mathtt{B}$ are horizontally partitioned by blocks $\mathtt{A}_i$ and $\mathtt{B}_j$ respectively, $\mathtt{A}^\top\mathtt{B}$ is partitioned by blocks $\mathtt{A}_i^\top\mathtt{B}_j$ and for these blocks we have

$$L(\mathtt{A}^\top\mathtt{B}) \subseteq L(\mathtt{A}) \times L(\mathtt{B}). \tag{E.6}$$

If $\mathtt{C} = \sum_i \mathtt{C}_i$ and all $\mathtt{C}_i$ have the same block partition,

$$L(\mathtt{C}) \subseteq \cup_i L(\mathtt{C}_i). \tag{E.7}$$

If all blocks considered by L have the same size $a \times b$, we can write $L_{a\times b}$ instead of L.

In practice, it is improbable that a product or sum of non-zero matrices is zero. Thus we replace the inclusions above by equalities in our proof, i.e. we use

$$L(\mathtt{A}^\top\mathtt{B}) = L(\mathtt{A}) \times L(\mathtt{B}) \text{ and } L(\mathtt{C}) = \cup_i L(\mathtt{C}_i). \tag{E.8}$$

Let $\mathtt{Z}^g$ and $\mathtt{Z}^r$ be the $6m \times 6m$ top-left blocks of the RCS in the standard case and our case (more details in Sec. 6.7.2), respectively. In the next section, we show that

$$L_{6\times6}(\mathtt{Z}^r) = L_{6\times6}(\mathtt{Z}^g) + \{-1,0,1\}^2. \tag{E.9}$$

In these expressions and the following ones, we implicitly omit integers that are below 1 and above m (e.g. we omit $i-1$ if $i = 1$ and omit $i+1$ if $i = m$).

Appendix E.2. Proof of Eq. E.9

The image projection function of the l-th 3D point in the i-th multi-camera pose is $\varphi^g_{i,l}$ in the standard case and $\varphi^r_{i,l}$ in our case. According to Sec. 6.7.2, we have

$$\varphi^g_{i,l} = \varphi(\mathbf{m}_i, \mathbf{m}', \mathbf{x}_l) \text{ and } \varphi^r_{i,l} = \varphi(\mathbf{m}_{i-1}, \mathbf{m}_i, \mathbf{m}_{i+1}, \mathbf{m}', \mathbf{x}_l) \tag{E.10}$$

Thus $\frac{\partial\varphi^g_{i,l}}{\partial\mathbf{m}_{i'}} \neq 0$ iff $i' = i$, and $\frac{\partial\varphi^r_{i,l}}{\partial\mathbf{m}_{i'}} \neq 0$ iff $i' \in \{i-1, i, i+1\}$. We rewrite this using the notations in Appendix E.1:

$$L_{2\times6}(\frac{\partial\varphi^g_{i,l}}{\partial\mathbf{M}}) = \{i\} \text{ and } L_{2\times6}(\frac{\partial\varphi^r_{i,l}}{\partial\mathbf{M}}) = \{i-1, i, i+1\}. \tag{E.11}$$

Notations $\varphi_{i,l}$ and $\mathtt{Z}, \mathtt{U}\cdots$ are used in expressions that hold for both "r" and "g" upper-indices added to these notations. Let V_l be the list of keyframe indices where the l-th point is inlier. According to Eqs. 21 and 22, we have $\mathtt{Z} = \mathtt{U} - \mathtt{W}\mathtt{V}^{-1}\mathtt{W}^\top$ where

$$\mathtt{U} = \sum_{i\in V_l} (\frac{\partial\varphi_{i,l}}{\partial\mathbf{M}})^\top \frac{\partial\varphi_{i,l}}{\partial\mathbf{M}}, \mathtt{W} = \sum_{i\in V_l} (\frac{\partial\varphi_{i,l}}{\partial\mathbf{M}})^\top \frac{\partial\varphi_{i,l}}{\partial\mathbf{X}}, \tag{E.12}$$

$$\mathtt{V} = \sum_{i\in V_l} (\frac{\partial\varphi_{i,l}}{\partial\mathbf{X}})^\top \frac{\partial\varphi_{i,l}}{\partial\mathbf{X}}. \tag{E.13}$$

Since $L_{6\times6}(\mathtt{U}) = \cup_{i\in V_l}(L_{2\times6}(\frac{\partial\varphi_{i,l}}{\partial\mathbf{M}}))^2$, we have

$$\begin{aligned} L_{6\times6}(\mathtt{U}^g) &= \cup_i\{(i,i)\} \text{ and} && \text{(E.14)}\\ L_{6\times6}(\mathtt{U}^r) &= \cup_i\{i-1,i,i+1\}^2 && \text{(E.15)}\\ &= \{-1,0,+1\}^2 + \cup_i\{(i,i)\}. && \text{(E.16)} \end{aligned}$$

Furthermore, $\mathtt{W}$ in Eq. 21 is horizontally partitioned in blocks $\mathtt{W}_l = \sum_{i\in V_l}(\frac{\partial\varphi_{i,l}}{\partial\mathbf{M}})^\top \frac{\partial\varphi_{i,l}}{\partial\mathbf{x}_l}$. Since

$$\begin{aligned} L_{6\times3}(\mathtt{W}_l) &= \cup_{i\in V_l} L_{6\times3}((\frac{\partial\varphi_{i,l}}{\partial\mathbf{M}})^\top \frac{\partial\varphi_{i,l}}{\partial\mathbf{x}_l}) && \text{(E.17)}\\ &= \cup_{i\in V_l} L_{2\times6}(\frac{\partial\varphi_{i,l}}{\partial\mathbf{M}}), && \text{(E.18)} \end{aligned}$$

we have

$$L_{6\times3}(\mathtt{W}^g_l) = V_l \text{ and } L_{6\times3}(\mathtt{W}^r_l) = V_l + \{-1,0,+1\}. \tag{E.19}$$

Since $\mathtt{V}$ in Eq. 21 is block-wise diagonal with invertible blocks $\mathtt{V}_l = \sum_{i\in V_l}(\frac{\partial\varphi_{i,l}}{\partial\mathbf{x}_l})^\top \frac{\partial\varphi_{i,l}}{\partial\mathbf{x}_l}$ and $\mathtt{W}\mathtt{V}^{-1}\mathtt{W}^\top = \sum_l \mathtt{W}_l\mathtt{V}_l^{-1}\mathtt{W}_l^\top$, we have $L_{6\times6}(\mathtt{W}\mathtt{V}^{-1}(\mathtt{W})^\top) = \cup_l(L_{6\times3}(\mathtt{W}_l))^2$. Thus

$$\begin{aligned} L_{6\times6}(\mathtt{W}^g\mathtt{V}^{-1}(\mathtt{W}^g)^\top) &= \cup_l(V_l)^2 \text{ and} && \text{(E.20)}\\ L_{6\times6}(\mathtt{W}^r\mathtt{V}^{-1}(\mathtt{W}^r)^\top) &= \cup_l(V_l + \{-1,0,+1\})^2 && \text{(E.21)}\\ &= \{-1,0,+1\}^2 + \cup_l(V_l)^2. && \text{(E.22)} \end{aligned}$$

Thus

$$\begin{aligned} L_{6\times6}(\mathtt{Z}^g) &= \cup_i\{(i,i)\} \cup_l (V_l)^2 \text{ and} && \text{(E.23)}\\ L_{6\times6}(\mathtt{Z}^r) &= \{-1,0,+1\}^2 + (\cup_i\{(i,i)\} \cup_l (V_l)^2). && \text{(E.24)} \end{aligned}$$

We obtain Eq. E.9.

Appendix F. Proof of Eq. 29

Let

$$\theta = (f_x, f_y, u_0, v_0, k_1, k_2, \cdots, k_n, \mathbf{x}_j), \tag{F.1}$$

$$\mathbf{z} = \begin{pmatrix} u \\ v \end{pmatrix}, \bar{\mathbf{z}} = \begin{pmatrix} \frac{u-u_0}{f_x} & \frac{v-v_0}{f_y} \end{pmatrix}^\top, \bar{\mathbf{z}}_u = \pi(\mathbf{x}_j), \tag{F.2}$$

$$g(\mathbf{z},\theta) = (1 + \sum_{i=1}^{n} k_i\|\bar{\mathbf{z}}\|^{2i})\bar{\mathbf{z}} - \bar{\mathbf{z}}_u. \tag{F.3}$$

Thanks to Eqs. 1 and 2 and since $\mathbf{p} = \mathbf{z}_d$, we have $g(\mathbf{p},\theta) = 0$. First we note that

$$\begin{aligned} \frac{\partial\bar{\mathbf{z}}}{\partial(u_0,v_0)} &= -\begin{pmatrix} 1/f_x & 0 \\ 0 & 1/f_y \end{pmatrix} = -\frac{\partial\bar{\mathbf{z}}}{\partial\mathbf{z}}, && \text{(F.4)}\\ \frac{\partial\bar{\mathbf{z}}}{\partial(f_x,f_y)} &= \frac{\partial\bar{\mathbf{z}}}{\partial\mathbf{z}}\begin{pmatrix} \frac{u_0-u}{f_x} & 0 \\ 0 & \frac{v_0-v}{f_y} \end{pmatrix}. && \text{(F.5)} \end{aligned}$$

Thus

$$\begin{aligned} \frac{\partial g}{\partial\mathbf{z}} &= \frac{\partial g}{\partial\bar{\mathbf{z}}}\frac{\partial\bar{\mathbf{z}}}{\partial\mathbf{z}}, && \text{(F.6)}\\ \frac{\partial g}{\partial(u_0,v_0)} &= \frac{\partial g}{\partial\bar{\mathbf{z}}}\frac{\partial\bar{\mathbf{z}}}{\partial(u_0,v_0)} = -\frac{\partial g}{\partial\bar{\mathbf{z}}}\frac{\partial\bar{\mathbf{z}}}{\partial\mathbf{z}} = -\frac{\partial g}{\partial\mathbf{z}}, && \text{(F.7)}\\ \frac{\partial g}{\partial(f_x,f_y)} &= \frac{\partial g}{\partial\bar{\mathbf{z}}}\frac{\partial\bar{\mathbf{z}}}{\partial(f_x,f_y)} = \frac{\partial g}{\partial\mathbf{z}}\begin{pmatrix} \frac{u_0-u}{f_x} & 0 \\ 0 & \frac{v_0-v}{f_y} \end{pmatrix}. && \text{(F.8)} \end{aligned}$$

Last we obtain Eq. 29 using Eqs. F.7, F.8 and 23.

Appendix G. Keyframe sub-sampling of the videos

The keyframe sub-sampling is that in Sec. 2.3 of [39] with an improvement if the multi-camera motion is slow. It is based on counting the number of Harris points matched using ZNCC correlation between the current frame and its two preceding keyframes. Every frame is considered in increasing order of its index. First the current frame is rejected if its image motion compared to that of the last keyframe is small, i.e. if 70% of its matches have a 2D motion less than 5 pixels. Second, a non-rejected current frame is selected as a keyframe if (1) it has at least N_2 matches with the previous keyframe and (2) it has at least N_3 matches with the two previous keyframes and (3) the next frame does not meet (1) or (2). We always use $N_3 = N_2/2$, use $N_3 = 450$ for the multi-frame of four Gopro cameras at 100 FpS, $N_3 = 1200$ for four Gopro cameras at 48 FpS, $N_3 = 625$ for Ladybug 2 and $N_3 = 300$ for Theta S.

References

[1] Videostitch, http://www.video-stitch.com, last accessed on 2017/02/16.
[2] Kolor, http://www.kolor.com, last accessed on 2017/02/16.
[3] M. Lhuillier, T. Nguyen, Synchronization and self-calibration for helmet-held consumer cameras, applications to immersive 3d modeling and 360 videos, in: 3DV'15.
[4] Gopro, https://gopro.com/, last accessed on 2017/02/16.
[5] C. Geyer, M. Meingast, S. Sastry, Geometric models of rolling-shutter cameras, in: OMNIVIS'05.
[6] 360heros, http://www.360rize.com/, last accessed on 2017/02/16.
[7] Theta s, https://theta360.com/, last accessed on 2017/02/16.
[8] R. Hartley, A. Zisserman, Multiple view geometry in computer vision, Second Edition, Cambridge University Press, 2004.
[9] B. Triggs, P. McLauchlan, R. Hartley, A. Fitzgibbon, Bundle adjustment – a modern synthesis, in: Vision Algorithms: Theory and Practice, 2000.
[10] B. Micusik, T. Pajdla, Structure from motion with wide circular field of view cameras, PAMI 28 (7).
[11] A. Fitzgibbon, Simultaneous linear estimation of multiple view geometry and lens distortion, in: CVPR'01.
[12] T. Gaspar, P. Oliveira, P. Favaro, Synchronization of two independently moving cameras without feature correspondences, in: ECCV'14.
[13] L. Spencer, M. Shah, Temporal synchronization from camera motion, in: ACCV'04.
[14] G. Carrera, A. Angeli, A. Davison, SLAM-based extrinsic calibration of a multi-camera rig, in: ICRA'11.
[15] S. Esquivel, F. Woelk, R. Koch, Calibration of a multi-camera rig from non-overlapping views, in: DAGM'07.
[16] Y. Dai, J. Trumpf, H. Li, N. Barnes, R. Hartley, Rotation averaging with application to camera-rig calibration, in: ACCV'09.
[17] P. Lebraly, E. Royer, O. Ait-Aider, C. Deymier, M.Dhome, Fast calibration of embedded non-overlapping cameras, in: ICRA'11.
[18] P. Sturm, S. Ramalingam, J. Tardif, S. Gasparini, J. Barreto, Camera models and fundamental concepts used in geometric computer vision, Foundations and Trends in Computer Graphics and Vision 6 (1).
[19] J. Schneider, W. Forstner, Bundle adjustment and system calibration with points at infinity for omnidirectional cameras, Tech. Rep. TR-IGG-P-2013-1, Institute of Geodesy and Geoinformation, University of Bonn (2013).
[20] L. Oth, P. Furgale, L. Kneip, R. Siegwart, Rolling shutter camera calibration, in: CVPR'13.
[21] J. Hedborg, P. Forseen, M. Felsberg, R. Ringaby, Rolling shutter bundle adjustment, in: CVPR'12.
[22] G. Duchamp, O. Ait-Aider, E. Royer, J. Lavest, Multiple view 3D reconstruction with rolling shutter cameras, in: VISIGRAPP'15.
[23] G. Klein, D. Murray, Parallel tracking and mapping on a camera phone, in: ISMAR'09.
[24] B. Klingner, D. Martin, J. Roseborough, Street view motion-from-structure-from-motion, in: ICCV'13.
[25] O. Saurer, M. Pollefeys, G. Lee, Sparse to dense 3d reconstruction from rolling shutter images, in: CVPR'16.
[26] P. Furgale, J. Rehder, R. Siegwart, Unified temporal and spatial calibration for multi-sensor systems, in: IROS'13.
[27] S. Lovegrove, A. Patron-Perez, G. Sibley, Spline fusion: a continuous-time representation for visual-intertial fusion with application to rolling shutter cameras, in: BMVC'13.
[28] E. Mouragnon, M. Lhuillier, M. Dhome, F. Dekeyser, P. Sayd, Generic and real-time structure from motion, in: BMVC'07.
[29] M. Vo, S. Narasimhan, Y. Sheikh, Spatiotemporal bundle adjustment for dynamic 3d reconstruction, in: CVPR'16.
[30] J. Lavest, M. Viala, M. Dhome, Do we really need an accurate calibration pattern to achieve a reliable camera calibration ?, in: ECCV'98.
[31] T. Nguyen, M. Lhuillier, Adding synchronization and rolling shutter in multi-camera bundle adjustment, in: BMVC'16.
[32] C. Geyer, K. Daniilidis, A unifying theory for central panoramic systems and practical implications, in: ECCV'00.
[33] R. Szeliski, D. Scharstein, Symmetric sub-pixelic stereo matching, in: ECCV'02.
[34] P. Singla, D.Mortari, J.L.Junkins, How to avoid singularity when using Euler angles ?, in: AAS Space Flight Mechanics Conference, 2004.
[35] F. S. Grassia, Practical parametrization of rotations using the exponential map, Journal of graphics tools 3 (3).
[36] M. Lhuillier, Automatic scene structure and camera motion using a catadioptric system, CVIU 109 (2).
[37] Ladybug2, http://www.ptgrey.com, last accessed on 2017/02/16.
[38] L. Agapito, E. Heyman, I. Reid, Self-calibration of rotating and zooming cameras, IJCV 45 (2).
[39] E. Royer, M. Lhuillier, M. Dhome, J. Lavest, Monocular vision for mobile robot localization and autonomous navigation, IJCV 74 (3).