Synchronization and Self-Calibration for Helmet-Held Consumer Cameras, Applications to Immersive 3D Modeling and 360 Video

Maxime Lhuillier and Thanh-Tin Nguyen
Institut Pascal, 4 av. Blaise Pascal, 63178 Aubière Cedex, France
Maxime.Lhuillier at free.fr

Abstract

This paper presents the first 3D reconstruction system using unsynchronized and helmet-held consumer cameras, without the use of a calibration pattern. Our assumptions are easy to meet in practice: the cameras have the same setting (frequency, image resolution, field-of-view, roughly equiangular). First, the time offsets between cameras are estimated without accurate calibration as input. Second, both inter-camera rotations and intrinsic parameters are refined using structure-from-motion and bundle adjustment. We experiment both synchronization and self-calibration on four GoPro cameras mounted on a helmet, such that the resulting multi-camera is assumed to be central and provides a 360 degree field-of-view in the horizontal plane. A surface is also estimated from a multi-camera video acquired by walking in a city.

1. Introduction

The automatic 3D modeling of a scene from an image sequence is a well studied topic. Today most available softwares reconstruct objects or "small" parts of environments due to the small field-of-view of the camera, unless aerial images are taken at well selected viewpoints (*e.g.* thanks to UAV). Flexible and helmet-held systems to reconstruct complete (even immersive) environments from ground imagery are still lacking. In this paper, we propose to mount several consumer cameras on a helmet; the user then moves at a low speed (walking) in an environment and takes the videos; lastly a surface approximating the viewed scene is reconstructed based on assumptions about the cameras that are easy to be meet. The reconstruction stage includes synchronization and self-calibration of the multi-camera (the main topics of the paper), as well as structure-from-motion and surface reconstruction from a cloud of 3D points estimated from the images.

Figure 1. Top: four GoPro Hero3 cameras enclosed in a cardboard box. Bottom: resulting images of our DIY multi-camera.

1.1. Assumptions about the Cameras

The user starts all videos at once by a single click on a wifi remote. However this synchronization is too inaccurate for applications like immersive video and our 3D scene modeling. Furthermore, we assume that the cameras are roughly equiangular with an approximately known field-of-view and orientation (rotations) in the multi-camera coordinate system. In our context, this knowledge is easy to obtain since the cameras are the same and mounted such that a 360 video [1, 2] can be generated by stitching: n cameras that are symmetrically mounted around a symmetry axis. In practice we use the same four cameras that are enclosed in a cardboard box as shown in Fig. 1. This can be seen as a DIY version of the Ladybug [3] multi-camera, but the price of our multi-camera is one magnitude smaller than that of the Ladybug. The camera configuration is chosen such that the distance between optical centers is as short as possible (we do a central approximation). A greater number of cameras can be used [4] to increase the common field-of-view of adjacent cameras and then to make easier image matching and blending between two adjacent cameras (this is useful for video stitching). However both the price and distance increase. In our case, the common field-of-view is too small for us to automatically obtain a decent matching using available tools (we tried Hugin [5], VideoStitch [2] and VisualSFM [6]).

2. Previous Work

2.1. Initializing the Intrinsic Parameters

There is an automatic method to estimate the intrinsic parameters of a single camera through structure-from-motion (SfM): projective reconstruction from an image sequence, auto-calibration and refinement using bundle adjustment [14, 28] (BA). The radial distortion can also be included in intrinsic parameters that are estimated [11, 21]. A more detailed overview about uncalibrated SfM and auto-calibration is outside the scope of this paper.

2.2. Time Offset Estimation

In [24], transformations are estimated between consecutive frames of every video instead of trying to match different videos. The estimated offset is then that which best "compares" the transformations between two videos. One example in [24] is the translation magnitude that is estimated from tracked features. Intuitively, the larger the translation in one video, the larger the translation in the other. The comparison is done using cross-correlation.

Such approaches are frame-accurate (the offset is an integer) and need neither common field-of-view nor matching between different cameras. This is interesting in our case where matching is difficult between cameras. A survey of methods for video synchronization can be found in [13], but these require inter-camera matching or common field-of-view or are designed for non jointly moving cameras.

2.3. Initializing the Inter-Camera Rotations

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 frame. In [8], 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 [10], 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.* [9]) can also be used if there is a non constant relative pose between two reconstructions due to the drift of reconstruction(s).

2.4. Refinements for Multi-Camera

A BA is introduced in [17] to refine the relative poses between the cameras. The successive poses of the multi-camera and 3D points are also refined by minimizing a reprojection error. But the intrinsic parameters are not refined and the reprojection error is in the undistorted space of the classical polynomial distortion model [25]. The BA in [23] 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 [8, 23].

Work [12] suggests that time offset and spatial transformations (*e.g.* rotations) between multiple sensors can be jointly estimated using a continuous state representation. However there are only two sensors in experiments: a calibrated camera and IMU. Such a method would also estimate the time offset with sub-frame accuracy.

3. Contributions

3.1. Initializations

First we do not intend to compete with the accuracy of the initialization methods for intrinsic parameters (Sec. 2.1) and inter-camera rotations (Sec. 2.3). In this paper, SfM is started using calibration initialized by approximate values. We think that this is useful to experiment our contribution (synchronization and BA) with respect to inaccurate input.

Here we describe two cases: single camera and multi-camera. Since our cameras are roughly equiangular and we have approximate knowledge of their field-of-view, their intrinsic parameters (including radial distortion) are approximately known. Standard SfM is then applied for every camera using these parameters and a global BA refines the result. Once the time offsets are estimated and since we have approximate knowledge of the intrinsic parameters and relative/inter-camera rotations, an approximate calibration of the multi-camera is known. We then apply SfM for the multi-camera and a global BA refines the result.

3.2. Synchronization

Second we estimate the time offset using a method that is less heuristic than the translation-based method (Sec. 2.2). Let $\mathtt{R}_t$ be the rotation at frame t estimated by monocular SfM for a camera (Sec. 2.1). The instantaneous angular velocity is the angle in $[0, \pi]$ of rotation $\mathtt{R}_t^{-1}\mathtt{R}_{t+1}$. This angle is the same for all cameras at the same time. Indeed, $\mathtt{R}_t^{-1}\mathtt{R}_{t+1}$ does not change if we replace $\mathtt{R}_t$ by $\mathtt{R}\mathtt{R}_t$ and $\mathtt{R}_{t+1}$ by $\mathtt{R}\mathtt{R}_{t+1}$ where $\mathtt{R}$ is the (constant) relative rotation between two cameras. Then we compute the table of angles for every camera and find the time offset that maximizes the correlation (ZNCC) between two such tables (try to match two sub-tables with the same length in different tables).

For $n > 2$ cameras, we first estimate the best offset for every pair of adjacent cameras. Since the camera adjacencies can form loops (they do in our setup) and the offsets are estimated independently, the sum of offsets along a loop can be non-zero although it should be zero. In this case, we also try offsets around the best one (their difference is in $\{-k, \cdots -2, -1, 1, 2, \cdots + k\}$, where $k = 1$ is enough in our experiments). The final offsets are those whose sum along loop is zero and that maximize the sum of ZNCC.

We expect this calculation to have low ambiguity thanks

to small head motions and since the cameras are helmet-held. In experiment (Sec. 5.2), both translation- and angle-based methods are compared. In the translation method, the angle of a frame is replaced by the mean of translation moduli of feature tracks (Harris matched by ZNCC).

3.3. Bundle Adjustment

Third, we improve a BA [17] in two ways, which is based on the classical polynomial distortion model [25]. Our BA also refines intrinsic parameters (not only inter-camera rotations 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, the result returned by BA is a Maximum Likelihood Estimator (this assumption is not true in the undistorted space). Sec. 4 details how to initialize the BA and how to compute the non-closed form forward-projection of the classical polynomial distortion model. In Sec. 5.3, we experiment on the BA started from inaccuracies of intrinsic parameters and inter-camera rotations. We also compare our results to others that are estimated using a calibration pattern, and compare results without and with inter-camera matches (Sec. 5.5) that are obtained using image pre-rectification and our calibration estimation.

3.4. Applications: 3D Modeling and 360 Video

Fourth, Sec. 5.4 shows a 3D model of an environment (a part of a city) computed by a 3D reconstruction system that uses our synchronization and BA methods. The input multi-camera video is taken by four GoPro Hero3 cameras mounted on a helmet. As far as we are aware, this is the first time such an experiment has been conducted. Lastly, there are **joint videos** that show the surface and 360 videos.

4. Polynomial Distortion Model

First Sec. 4.1 provides the equations of the classical polynomial distortion model [25], which is often used since its closed-form back-projection is useful for SfM tasks and epipolar geometry. Sec. 4.2 then details how to obtain the model parameters from a given field-of-view and assuming that the camera is equiangular. This is useful to initialize BA. Last, Sec. 4.3 details how to efficiently compute the forward-projection and its derivatives, which are not closed-form. They are required by BA for minimizing errors in the right image space (Sec. 3.3).

4.1. Back-Projection

Let $\mathtt{K}$ be the intrinsic parameter matrix of a perspective camera. $\mathtt{K}$ has focal parameters f_x and f_y, principal point $\mathbf{z}_0$ and zero skew. The mapping from the distorted space to the undistorted space is closed-form and depends on radial distortion parameters k_i (tangential distortions are neglected). Let $\mathbf{z}_d$ and $\mathbf{z}_u$ be the distorted (original) and undistorted (rectified) coordinates of a pixel. Their normalized coordinates $\bar{\mathbf{z}}_d$ and $\bar{\mathbf{z}}_u$ meet $\mathtt{K}\begin{pmatrix}\bar{\mathbf{z}}_d^\top & 1\end{pmatrix}^\top = \begin{pmatrix}\mathbf{z}_d^\top & 1\end{pmatrix}^\top$ and $\mathtt{K}\begin{pmatrix}\bar{\mathbf{z}}_u^\top & 1\end{pmatrix}^\top = \begin{pmatrix}\mathbf{z}_u^\top & 1\end{pmatrix}^\top$. Let $\bar{r}_d = ||\bar{\mathbf{z}}_d||$ be the normalized radial distance in the distorted image. The link between distorted and undistorted coordinates is $\bar{\mathbf{z}}_u = (1 + \sum_{i=1}^n k_i \bar{r}_d^{2i})\bar{\mathbf{z}}_d$. 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 frame. Note that the distortion center is the principal point.

4.2. Equiangular Initialization

A camera is equiangular if angle θ between the principal axis 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 $\tan\theta = \bar{r}_u$ where $\bar{r}_u = ||\bar{\mathbf{z}}_u||$.

If the camera is equiangular, there is a constant c such that $\theta = c r_d$ and the pixels are squares (*i.e.* $f_x = f_y = f$). Thus $\theta = cf\bar{r}_d$. Since $\bar{r}_u = \bar{r}_d(1 + \sum_{i=1}^n k_i \bar{r}_d^{2i})$,

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

Since $\tan$ is not polynomial, Eq. 1 can not be exact. We use Taylor approximation

$$\tan\theta \approx \sum_{i=0}^{n} t_i \theta^{2i+1} = \theta + \frac{\theta^3}{3} + \frac{2\theta^5}{5} + \frac{17\theta^7}{315} + \cdots \quad (2)$$

and identify coefficients between Eqs 1 and 2. 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 $f = r_d/\theta$ for pixel $\mathbf{z}_d$ at the center of an image border where the half-field-of-view θ is approximately known. Note that the resulting field-of-view is not exactly as desired due to the Taylor approximation.

4.3. Forward Projection and its Derivatives

The forward projection and its derivatives are required by the Levenberg-Marquardt routine involved in BA. The computations from 3D to undistorted coordinates are not detailed since they are standard. The mapping from undistorted to distorted coordinates is implicitly defined by $g(\mathbf{z}_d, \mathbf{z}_u, \mathbf{z}_0, \mathbf{k}) = 0$ where $\mathbf{k} = (f_x, f_y, k_1, k_2, \cdots, k_n)$ and function $g : \mathbb{R}^{n+8} \mapsto \mathbb{R}^2$ meets

$$g(\mathbf{z}_d, \mathbf{z}_u, \mathbf{z}_0, \mathbf{k}) = (1 + \sum_{i=1}^{n} k_i \bar{r}_d^{2i})(\mathbf{z}_d - \mathbf{z}_0) - \mathbf{z}_u + \mathbf{z}_0. \quad (3)$$

In more detail, the implicit function Theorem implies that locally we have a $\mathcal{C}^1$ continuous function φ such that $\mathbf{z}_d = \varphi(\mathbf{z}_u, \mathbf{z}_0, \mathbf{k})$ if $\det \frac{\partial g}{\partial \mathbf{z}_d} \neq 0$.

First $\mathbf{z}_d$ is estimated from $\mathbf{z}_u$, $\mathbf{z}_0$ and $\mathbf{k}$ by non linear least-square minimizing $\mathbf{z}_d \mapsto ||g(\mathbf{z}_d)||^2$. In our BA context, we would like to minimize reprojection error $||\mathbf{z}_d - \tilde{\mathbf{z}}_d||$ where $\tilde{\mathbf{z}}_d$ is an inlier point detected in an image. Thus $\tilde{\mathbf{z}}_d$ is known and close to $\mathbf{z}_d$ (in our experiments, $||\mathbf{z}_d - \tilde{\mathbf{z}}_d|| < 2$ pixels). We use an iterative method starting from $\tilde{\mathbf{z}}_d$ to estimate $\mathbf{z}_d$. In practice, the Gauss-Newton method is acceptable with no more than 5 iterations.

Second, $\frac{\partial \mathbf{z}_d}{\partial k_i}$ should be estimated to fill the hessian in BA. We differentiate Eq. 3 with respect to k_i and obtain

$$\frac{\partial \mathbf{z}_d}{\partial k_i} = -(\frac{\partial g}{\partial \mathbf{z}_d})^{-1} \frac{\partial g}{\partial k_i}. \quad (4)$$

The derivatives of $\mathbf{z}_d$ with respect to $f_x, f_y, \mathbf{z}_0$ and $\mathbf{z}_u$ are similar. The other derivatives (with respect to relative Euler angles/rotation, translation etc) are computed by the chain rule involving derivative $\frac{\partial \mathbf{z}_d}{\partial \mathbf{z}_u}$.

5. Experiments

5.1. Input

The multi-camera is defined by four GoPro Hero3 cameras (Fig. 1) with the same setting; they are numbered from 0 to 3. They are mounted on the helmet such that 0 (respectively, 1,2 and 3) is pointing forward left, (respectively, forward right, backward right, and backward left). The videos are recorded while the user walks the streets of a city and under trees in a garden for a duration of 703s. The images are a little dark since the acquisition was done in the early morning to avoid cars and pedestrians. The camera gain is not fixed and evolves independently for every camera. A high frame-rate (100 fps) is used to allow accurate synchronization. The resolution of a GoPro is 1280×960, but the image sizes are reduced by 50% to accelerate all computations. Here we neglect rolling shutter and non-central effects. We also assume that the calibration (both intrinsic and inter-camera rotations) does not change over time.

5.2. Synchronization

Several synchronization methods are experimented on the first 2000 frames of the four videos (a walk lasting 20s).

5.2.1 Notations

Let A_c be the name of the angle method based on standard SfM (based on local BA [22] and refined by global BA) using calibration c. The 9 parameters of c are f_x, f_y, $\mathbf{z}_0$ and k_i where $1 \leq i \leq 5$.

If $c = pat$, the cameras are accurately calibrated using a planar calibration pattern [16]. If $c = 90$, c is equiangular and its field-of-view is 90 degrees in the horizontal direction of Fig. 1 (Sec. 4.2). If $c = 90r$, every camera has the same process: first apply SfM using $c = 90$ on key-frames of the video; then refine c (assuming that c is constant in the sequence) using a global BA; lastly, use SfM once again for all frames with the refined and final c.

Method	$o_{0,1}$	$o_{1,2}$	$o_{2,3}$	$o_{3,0}$	Li	Lr
A_{90}	-15	-1	14	1	-1	3.8e-3
A_{90r}	-15	-1	14	1	-1	-2.1e-3
A_{pat}	-15	-1	14	1	-1	-2.0e-3
T_f	-12	0	12	0	0	0.10
T_b	-15	0	14	0	-1	-1.45
S	-18	-2	17	2	-1	-0.46

Table 1. Time offsets without loop constraint. Li (Lr, respectively) is the sum of four integer (real, respectively) offsets.

Let T_f be the name of the translation method such that the mean of translation moduli is computed in the complete frame of a camera. Let T_b be a variant of T_f: the mean is computed in a small area including the common field-of-view with other cameras (most left and right columns of images in Fig. 1). We expect T_b to be better than T_f if the translation magnitude evolves similarly in different cameras, especially in the common field-of-view. We also try a sound-based synchronization method S, where sound replaces angle/translation in the tables.

There are four offsets $o_{0,1}$, $o_{1,2}$, $o_{2,3}$, $o_{3,0}$ between adjacent cameras that are computed by one of the six methods above. Let $L = o_{0,1} + o_{1,2} + o_{2,3} + o_{3,0}$. Since we have a loop of camera adjacencies (Fig. 1), we should have $L = 0$ by enforcing a loop constraint as in Sec. 3.2.

In fact we distinguish Li, the sum of integer offsets $o_{i,i+1}$, and Lr the sum of real offsets $o_{i,i+1} + \epsilon_{i,i+1}$. We estimate sub-frame offsets like sub-pixelic disparity using a quadratic fit [26]: first approximate the mapping from $o_{i,i+1}$ to ZNCC using a quadratic polynomial defined by its 3 values at $o_{i,i+1} + \{-1, 0, +1\}$; then $o_{i,i+1} + \epsilon_{i,i+1}$ maximizes this polynomial. Note that Lr is not used by computations in our paper, it is used as a confidence measure.

5.2.2 Experiments

The estimated offsets without and with loop constraint are provided in Tab. 1 and Tab. 2, respectively. Note that an offset equal to ± 15 between two cameras means that the difference between the starting dates of their videos is 0.15 s. This is non negligible. We see that the integer offsets computed by A_c do not depend on the attempted calibration c, although the difference between two SfM results with different c is not negligible (*e.g.* the SfM with $c = 90$ has a final RMS which is 1.22 greater than that of the SfM with $c = 90r$; the latter has a number of 2D inliers 5% greater than that of the former). The better the c value, the better (smaller) the $|Lr|$ in Tab. 1. The better the c value, the better (larger) the ZNCC score in Tab. 2. In comparison,

Method	$o_{0,1}$	$o_{1,2}$	$o_{2,3}$	$o_{3,0}$	$Zncc_1$	$Zncc_2$
A_{90}	-15	-1	14	2	3.854	3.795
A_{90r}	-15	-1	14	2	3.941	3.880
A_{pat}	-15	-1	14	2	3.941	3.880
T_f	-12	0	12	0	2.610	2.421
T_b	-15	0	14	1	3.319	3.221
S	-18	-2	18	2	1.68	1.63

Table 2. Time offsets with loop constraint. $Zncc_1$ is the greatest sum of the four ZNCC of the four computed time offsets, $Zncc_2$ is the second greatest ZNCC. We have $-4 \leq Zncc_i \leq 4$.

the $|Lr|$ and ZNCC scores of methods S, T_f and T_b are the worst. The results of T_f and T_b are not surprising since the translation-based methods are essentially heuristic. Here is one reason for the results of the sound-based method S: some cameras, including our GoPro Hero3, provide poor Audio/Video synchronization (according to the user guide of [2]). A **joint video** (https://youtu.be/8bZM20g1p0U) shows the videos without and with our synchronization.

5.3. Multi-Camera Bundle Adjustment

Thanks to Sec. 5.2, the four videos are frame-accurately synchronized by skipping s_i frames at the beginning of the i-th video ($s_i - s_{i+1} = o_{i,i+1}$ provided by A_{90r}). Now we examine the BA result with respect to the initialization.

5.3.1 Notations

Let $B_{c,R}$ be the name of the following method: multi-camera SfM using calibration c and inter-camera rotations R, followed by our multi-camera BA. We define $c \in \{90, 90r, pat\}$ as in Sec. 5.2. Writing $R = \pi/2$ means that the inter-camera rotations between adjacent cameras are exactly the rotation with angle $\pi/2$ around the z-axis of the multi-camera coordinate frames. According to Fig. 1, this approximates the true inter-camera rotations of our setup. We also try other R that are perturbations of $R = \pi/2$, for example $R = \pi/2 + 6$ means that we multiply every inter-camera rotation by a 6 degree rotation with random axis.

The multi-camera BA in $B_{c,R}$ refines all inter-camera rotations (4*3 relative Euler angles), all intrinsic parameters c (4*(4+5) parameters for $\mathtt{K}$ and k_i), all 3D points and multi-camera poses in world coordinates. Thus both inter-camera rotation and c are assumed to be constant in every video.

Methods $B^{=}_{c,R}$ and $B^{const}_{c,R}$ are the same as $B_{c,R}$ up to the c refinement: $B^{=}_{c,R}$ enforces the same c for the 4 cameras and $B^{const}_{c,R}$ does not refine c.

5.3.2 How to Compare two Calibrations

We compute a distance d between the multi-camera calibrations estimated by $B_{c,R}$ and $B^{const}_{pat,\pi/2}$. The latter is assumed to be the most accurate since its $\mathtt{K}$ and k_i parameters are estimated using a calibration pattern.

Distance d is based on the angle between rays of the two calibrations that have the same pixel. There are two reasons for this. First the accuracy is only needed for the ray directions in our 3D modeling application, since these directions are directly used by SfM. Second parameters can compensate themselves if their estimations are biased (*e.g.* the rotation/principal point near-ambiguity for one view [7]).

Since two estimated calibrations can have different multi-camera coordinate systems, their sets of rays must be registered in the same coordinate system. The registration is defined by rotation $\mathtt{R}$, which maps one ray set to another (no translation since the calibrations are central in our paper). $\mathtt{R}$ is robustly estimated by RANSAC applied to pairs of matched rays in different calibrations, and by minimizing $e(\mathtt{R}) = \sum_{i=1}^{N} ||\mathbf{r}_i^1 - \mathtt{R}\mathbf{r}_i^0||^2$ where both ray directions $\mathbf{r}_i^1$ and $\mathbf{r}_i^0$ have the same pixel. Our distance is $d = \sqrt{e(\mathtt{R})/N}$ where N is the number of (sampled) rays in a multi-camera image. Note that d is expressed in radians if $d \ll 1$.

Method	d	RMS	#3Dpts	#2Dpts
$B^{=}_{90,\pi/2}$	0.36	0.511	16376	76837
$B_{90,\pi/2}$	0.34	0.507	16365	76842
$B_{90r,\pi/2}$	0.31	0.498	16346	76677
$B^{const}_{pat,\pi/2}$	0	0.507	16303	76430
$B_{90r,\pi/2+6}$	0.32	0.502	16347	76794
$B_{90r,\pi/2+10}$	0.34	0.504	16342	76397
$B_{90r,\pi/2+14}$	0.39	0.498	16027	71192
$B_{90r,\pi/2+18}$	6.70	0.553	11687	52750

Table 3. Results of multi-camera SfM followed by our BA using several initializations: angular-based distance (in degrees) between $B_{c,R}$ and $B^{const}_{pat,\pi/2}$, RMS for reprojection errors in pixels, numbers of reconstructed 3D points and 2D inlier points.

5.3.3 Experiments

Tab. 3 provides results for several initial calibrations. In all cases, SfM generates the same set of matched interest points and the same number of key-frames. We obtain 41 key-frames by SfM running on the first 2000 frames.

We first examine the inter-camera rotation initializations where $R = \pi/2$ (top of Tab. 3). According to d, the calibration of $B_{90r,\pi/2}$ is better than that of $B_{90,\pi/2}$. This result is expected since both methods are the same and the former has a better initialization. Furthermore, the calibration of $B_{90,\pi/2}$ is better than that of $B^{=}_{90,\pi/2}$. In other words, there is an accuracy loss if we assume that the four cameras have exactly the same c (although they have the same setting).

We note that all these methods have similar reprojection errors (RMS), their numbers of reconstructed 3D points and

c	90	$90r$	pat	$B_{90r,\pi/2}$
f_x	305.57	[289.8,291.4]	[290.1,290.9]	[289.9,292.5]
f_y	305.57	[288.5,291.4]	[290.1,290.9]	[289.8,291.7]
u_0	320.00	[318.6,324.7]	[319.7,325.8]	[318.0,325.6]
v_0	240.00	[233.9,242.0]	[234.2,240.9]	[235.2,240.3]
k_1	0.3333	[.3580,.3739]	[.3689,.3746]	[.3675,.3792]
k_2	0.4000	[.0475,.0991]	[.0335,.0669]	[.0297,.0638]
k_3	0.0539	[-.035,.0518]	[.0041,.0611]	[-.007,.0699]
k_4	0.0218	[-.033,.0406]	[-3e-4,.0173]	[-.033,.0180]
k_5	0.0088	[.0015,.0239]	[.0067,.0197]	[.0099,.0248]

Table 4. Intrinsic parameters of the four cameras in several cases.

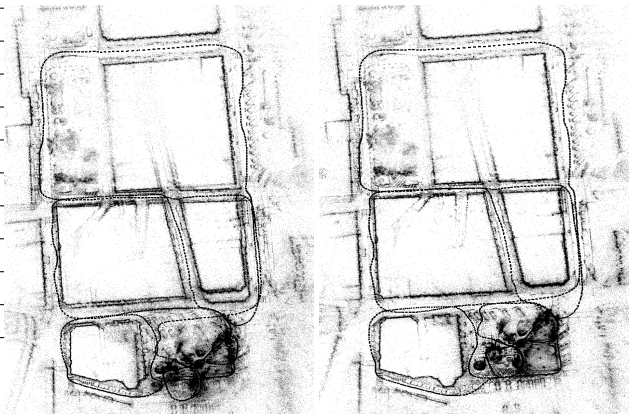

Figure 2. Top views of the SfM result without (left) and with (right) automatic loop detection and closure: 1834 key-frame poses (small black squares) and 710k points (gray-black points) reconstructed from 70k $4 \times 640 \times 480$ images. The garden area is on the bottom right corner and is surrounded by a few parked cars.

2D inliers are roughly the sames. In the whole paper, RMS is about 0.5 pixel (the Harris point detector has pixelic accuracy and every detected point is classified as inlier for BA if its reprojection error is less than 2 pixels).

Second, we examine the intrinsic parameter initializations $c = 90r$ (bottom of Tab. 3). The smaller the $R = \pi/2 + x$, the better the d value (as expected). BA provides bad results if the R perturbation is too high (+14 and +18). In the other cases, d is about 0.33 degrees.

Tab. 4 provides values for the intrinsic parameters of the four cameras in several cases: $c = 90$ (equiangular, Sec. 4.2), $c = 90r$ (camera-wise refinement in Sec. 5.2), $c = pat$ (using calibration pattern, [16]), and $c = B_{90r,\pi/2}$ (*i.e.* multi-camera-wise refinement provided by $B_{90r,\pi/2}$, Sec. 5.3). We see that $90r$, pat and $B_{90r,\pi/2}$ have similar $\mathtt{K}$ and k_1 parameters for all cameras, but k_i varies a lot if $i \geq 2$. According to $c = pat$, the four cameras have similar $(f_x, f_y, k_1, \mathbf{z}_0)$, their k_2 have the same magnitude order, but their k_i varies a lot if $3 \leq i \leq 5$.

5.4. 3D Modeling

First we apply SfM to the whole sequence with the multi-camera calibration provided by $B_{90r,\pi/2}$ in Sec. 5.3. We use SfM [22] based on key-frame sub-sampling, central generic camera model, and local BA. SfM selects 1834 key-frames from 70k frames of the multi-camera, and reconstructs 710k points. Loops are then detected and closed using a method similar to [15, 27] in 18 min. on a Latitude E5510 Laptop (i5 CPU M560 @ 2.67GHz). Fig. 2 compares the SfM results without and with automatic loop detection and closure. The low drift without loop computation confirms that the calibration is good.

Second, the sparse cloud of points is completed for surface reconstruction. For every key-frame, we detect Canny curves and Harris points and match them to those of the previous key-frame using [18] and the epipolar constraint. A Canny point is ignored if the angle between its curve tangent and the epipolar line at this point is less than $\pi/4$. Both Canny and Harris points are reconstructed by ray intersection. To limit the point cloud size, we only retain one reconstructed Canny pixel over four consecutive ones in a curve. 2.7M 3D points are computed in 42 min.

Third a surface reconstruction method ([19] improved using [20]) is applied to the point cloud. The cloud is filtered beforehand. A point is rejected if its apical angle is less than 10 degrees. Furthermore, points are rejected if they are below the ground or in the sky. Let $\mathbf{v}$ be an estimate of the vertical direction. Let V_j be the set of points that are detected and inlier in the j-th key-frame. We reject point $\mathbf{p}$ if there is key-frame j such that $\mathbf{p} \in V_j$ and altitude $\mathbf{p}.\mathbf{v}$ is one of the ten smallest or ten largest values in $\{\mathbf{q}.\mathbf{v}, \mathbf{q} \in V_j\}$. The resulting surface has 1.8M triangles and is computed in 2 min. (time without texturing).

Fig. 3 shows a global view of the reconstructed surface and Fig. 4 shows local views. A **joint video** (http://youtu.be/5r46SEBvz5w) shows walkthroughs in the output surface. There are several reasons for surface errors. Since we use ground imagery taken by a pedestrian, there is a lack of points and bad accuracy near the tops of buildings. Thus the main errors are in this area, and they are more visible in Fig. 3 than in the video (the viewing conditions are different: a global and top view in Fig. 3, local views close to the camera trajectory in the video). Another reason for errors is the aperture problem: the edges that are parallel to a street (*e.g.* those of a building top) can not be reconstructed if the street is parallel to the pedestrian motion.

5.5. 360 Video

Here we compare the calibrations estimated by $B_{90r,\pi/2}$ from several sets of image matches. The calibration quality is evaluated for the 360 video application, *i.e.* how good is the stitching of the four videos using the calibration ?

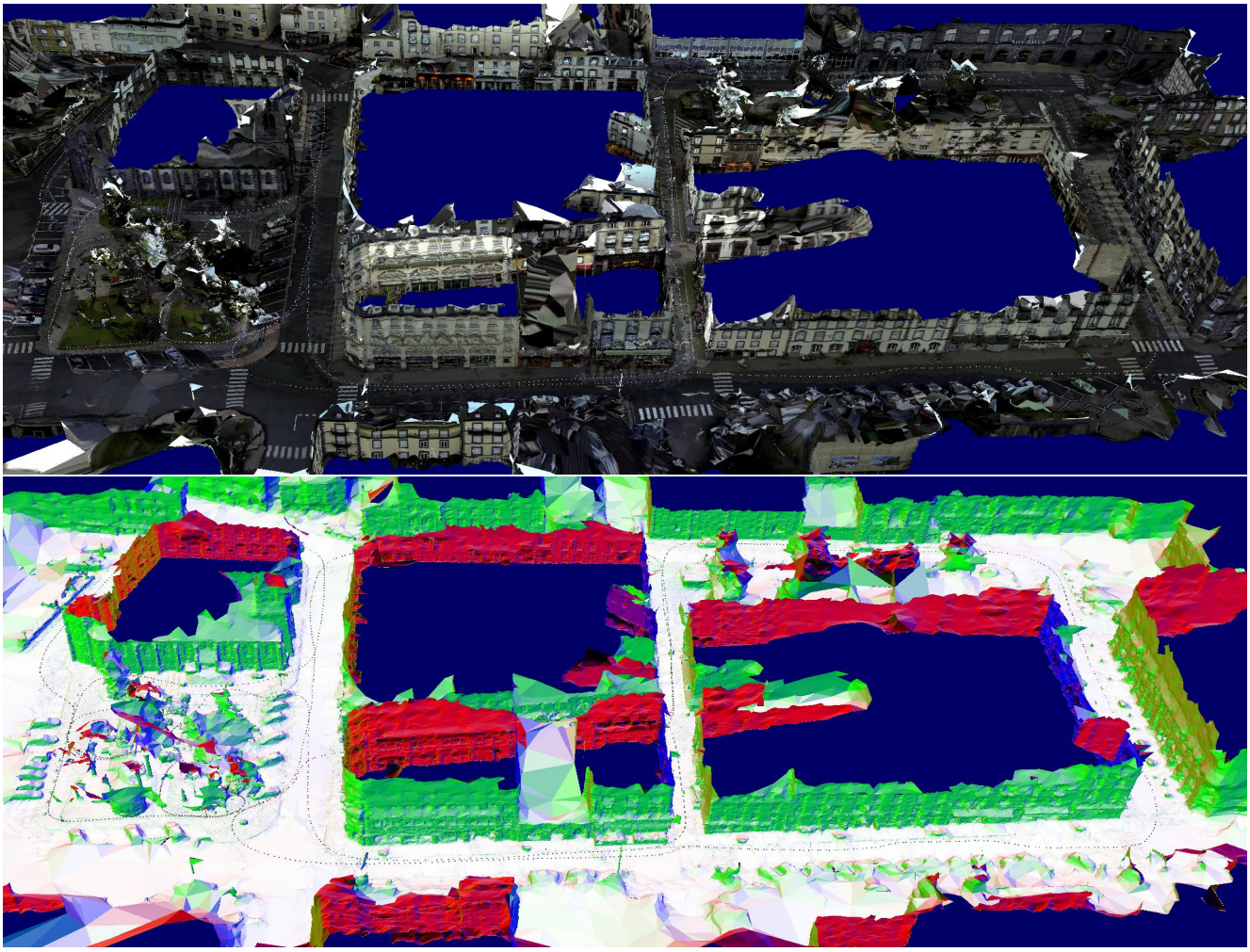

Figure 3. Global view of the 3D model (textures and triangle normals) computed from 2.7M points. The triangles in the sky are removed. Top: the key-frame poses of the multi-camera are also drawn using small white points. Bottom: the normals are encoded by colors.

5.5.1 Notations

The first set is $s = 2k$ as in Secs. 5.2 and 5.3: both SfM and BA use the 2000 first frames of the video. The second set is $s = all$, *i.e.* all frames of the video are used. In more detail, SfM first uses the calibration above (provided by $B_{90r,\pi/2}$ and $s = 2k$) on the whole multi-camera video, and BA then refines all parameters including the calibration.

Set $s = all^*$ is $s = all$ plus additional inter-camera matches. These matches connect reconstructed tracks in different cameras, *i.e.* they merge 3D points. They are computed after the SfM step and before BA. Their matching is made easier by prewarping onto a cylinder of the images of every camera; a standard ZNCC-based correlation (as in SfM) is then applied. In practice, we try to match points of the i-th camera in the t-th key-frame with points of the $(i+1)$-th camera (indices modulo 4) in the $(t+\alpha)$-key-frame where $\alpha \in \{-8, -7, \cdots +7, +8\}$.

5.5.2 Image-Based Evaluation

An image-based method is used to evaluate a central calibration. With images from the 4 cameras at the same time and a calibration, we warp the 4 images on a cylinder and accumulate the gray level discrepancies over the cylinder pixels where two images are warped. A discrepancy is the absolute value of the differences between two luminances. This process is done several times in the video (once per 10s of video). The result is the mean of the discrepancies.

5.5.3 Experiments

Fig. 5 shows an example of a cylindrical image without and with its discrepancies (Fig. 5 also shows that the common field-of-view between adjacent cameras is small). The means of discrepancies are 14.51 for $s = 2k$, 13.97 for $s = all$, and 13.63 for $s = all^*$ (gray levels in $[0, 255]$). Thus

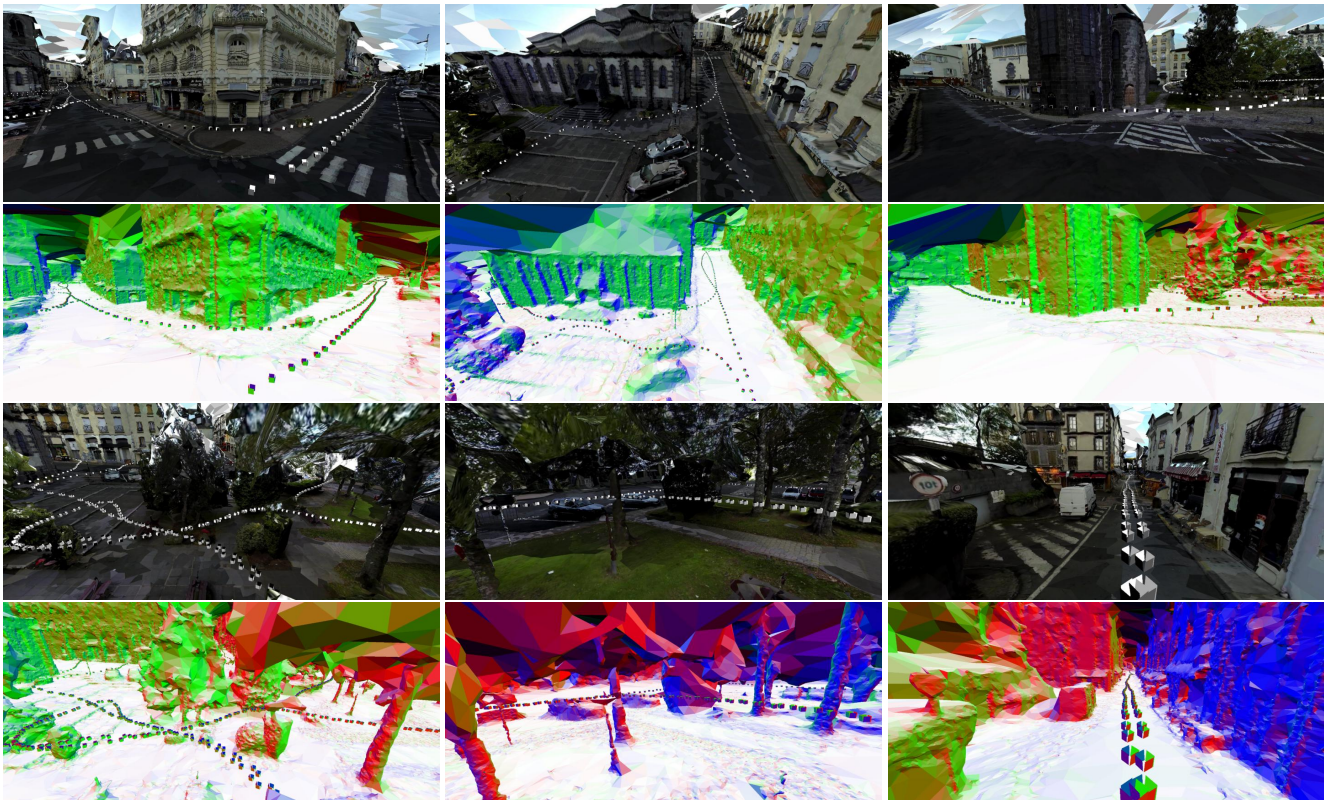

Figure 4. Local views of the 3D model (textures and triangle normals). The normals are encoded by colors (ground: white, vertical: red-green-blue, sky: black). Every key-frame pose is represented by four squares (one square per camera).

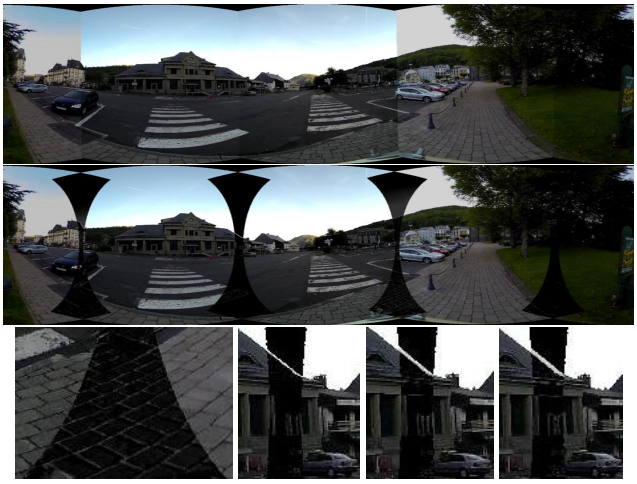

Figure 5. Cylinders without (top) and with (middle) discrepancies in areas shared by two original images in case s=all*. Bottom: local views of discrepancies if s=all*, s=all*, s=all and s=2k, respectively. The lower the discrepancy, the darker the gray level.

the improvements of all^* over all and of all over $2k$ are globally small. A **joint video** shows input frames and 360 videos using $s = all^*$ (https://youtu.be/8bZM20g1p0U).

The time of one successful BA iteration is 29s if $s = all$ (710k 3D points and 1.8k multi-camera 6D poses and 4*(3+4+5) calibration parameters are refined, 3.4M 2D inliers). Our implementation uses sparse implementation of Hessian and Cholesky factorization of the reduced camera system [28]. If $s = all^*$, there are 11k, 16k, 13k, and 16k matches between cameras 0-1, 1-2, 2-3, and 3-0, resp.

Lastly, we make two observations in the cylindrical images. First, there are some local and non-negligible improvements like those at the bottom-right of Fig. 5. Second, a textured ground in the close vicinity of the cameras is a frequent reason for high discrepancy (one example can be found in the bottom-left corner of Fig. 5). Indeed, the central approximation is inaccurate in this area.

5.6. Limitations and Further Experiments

Our synchronization method (A_{90r} in Sec. 5.2) is done using monocular SfM for every camera at the beginning of the dataset. Thus it can fail due to the lack of texture in a camera during this period. Nevertheless, the robustness with respect to low texture is improved thanks to (1) a modification of A_{90r}, (2) our assumption that the same setup is used for all cameras and (3) the fact that SfM robustness is better for accurate calibration. First apply monocular SfM

and calibration refinement for one video that has a lot of texture, then compute monocular SfM for the other (low textured) videos with this refined calibration. Similarly, it is straightforward to do the multi-camera refinement ($B_{c,r}$ in Sec. 5.3) for a low textured dataset by computing the multi-camera calibration on a more textured dataset, then assume that the calibration is similar between the two datasets.

In general, the number of inliers decreases in keyframes where the assumptions are not meet and SfM can fail for this reason. Shocks can generate changes of inter-camera rotations and rolling shutter effects due to fast image motion. However, we think that shocks due to walking are damped by the pedestrian. Another dataset has low framerate (30 fps) and fast head rotations. This is more difficult than our paper's dataset: the synchronization is less accurate and the rolling shutter effects are greater. To avoid failures of the synchronization and self-calibration, we increase the number of inliers (multiply by 2 the inlier threshold) and select a video segment for SfM with slow image motion.

6. Conclusion

This paper focuses on the synchronization and self-calibration of a DIY multi-camera system mounted on a helmet. It presents large-scale 3D scene modeling using such a setup for the first time. Thanks to the central approximation, 360 videos are also generated (although the common field-of-view between adjacent cameras is small).

We start with structure-from-motion, then the instantaneous angular velocity (IAV) is known and the initial calibration is refined by bundle adjustment (BA). Synchronization methods are compared including ours which is based on IAV. We extend previous BAs designed for multi-cameras: both intrinsic parameters and inter-camera rotations are refined, and the minimized errors are the true reprojection errors (although the forward-projection is not closed-form with the classical polynomial distortion model). We also experiment our BA for several calibration parametrizations and several sets of tracked points in images.

Future work includes a BA that simultaneously refines geometry and sub-frame synchronization, a detailed comparison of central and non-central self-calibrations for multi-cameras like ours that are almost central, investigations on rolling shutter and non-constant calibration, and improving the robustness with respect to low texture.

References

[1] http://www.kolor.com. 1

[2] http://www.video-stitch.com. 1, 5

[3] http://www.ptgrey.com. 1

[4] http://www.360heros.com/. 1

[5] http://hugin.sourceforge.net. 1

[6] http://ccwu.me/vsfm. 1

[7] L. Agapito, E. Heyman, and I. Reid. Self-calibration of rotating and zooming cameras. *IJCV*, 45(2), 2001. 5

[8] G. Carrera, A. Angeli, and A. Davison. SLAM-based extrinsic calibration of a multi-camera rig. In *ICRA'11*. 2

[9] Y. Dai, J. Trumpf, H. Li, N. Barnes, and R. Hartley. Rotation averaging with application to camera-rig calibration. In *ACCV'09*. 2

[10] S. Esquivel, F. Woelk, and R. Koch. Calibration of a multi-camera rig from non-overlapping views. In *DAGM'07*. 2

[11] A. Fitzgibbon. Simultaneous linear estimation of multiple view geometry and lens distortion. In *CVPR'01*. 2

[12] P. Furgale, J. Rehder, and R. Siegwart. Unified temporal and spatial calibration for multi-sensor systems. In *IROS'13*. 2

[13] T. Gaspar, P. Oliveira, and P. Favaro. Synchronization of two independently moving cameras without feature correspondences. In *ECCV'14*. 2

[14] R. Hartley and A. Zisserman. *Multiple view geometry in computer vision*. Cambridge University Press, 2000. 2

[15] M. Klopschitz, C. Zach, A. Irschara, and D. Schmalstieg. Generalized detection and merging of loop closures for video sequences. In *3DPVT'08*. 6

[16] J. Lavest, M. Viala, and M. Dhome. Do we really need an accurate calibration pattern to achieve a reliable camera calibration ? In *ECCV'98*. 4, 6

[17] P. Lebraly, E. Royer, O. Ait-Aider, C. Deymier, and M.Dhome. Fast calibration of embedded non-overlapping cameras. In *ICRA'11*. 2, 3

[18] M. Lhuillier and L. Quan. Match propagation for image-based modeling and rendering. *PAMI*, 24(8), 2002. 6

[19] M. Lhuillier and S. Yu. Manifold surface reconstruction of an environment from sparse structure-from-motion data. *CVIU*, 117(11), 2013. 6

[20] V. Litvinov and M. Lhuillier. Incremental solid modeling from sparse structure-from-motion data with improved visual artifact removal. In *ICPR'14*. 6

[21] B. Micusik and T. Pajdla. Structure from motion with wide circular field of view cameras. *PAMI*, 28(7), 2006. 2

[22] E. Mouragnon, M. Lhuillier, M. Dhome, F. Dekeyser, and P. Sayd. Generic and real-time structure from motion. In *BMVC'07*. 4, 6

[23] J. Schneider and W. Forstner. Bundle adjustment and system calibration with points at infinity for omnidirectional cameras. Technical Report TR-IGG-P-2013-1, Institute of Geodesy and Geoinformation, University of Bonn, 2013. 2

[24] L. Spencer and M. Shah. Temporal synchronization from camera motion. In *ACCV'04*. 2

[25] P. Sturm, S. Ramalingam, J. Tardif, S. Gasparini, and J. Barreto. Camera models and fundamental concepts used in geometric computer vision. *Foundations and Trends in Computer Graphics and Vision*, 6(1), 2011. 2, 3

[26] R. Szeliski and D. Scharstein. Symmetric sub-pixelic stereo matching. In *ECCV'02*. 4

[27] A. Torii, M. Havlena, and T. Pajdla. From google street view to 3d city models. In *ICCV Workshop'09*. 6

[28] B. Triggs, P. McLauchlan, R. Hartley, and A. Fitzgibbon. Bundle adjustment – a modern synthesis. In *Vision Algorithms: Theory and Practice*, 2000. 2, 8