Synchronisation et auto-étalonnage de plusieurs caméras fixées sur un casque

Thanh-Tin Nguyen Maxime Lhuillier

Institut Pascal, UMR 6602 CNRS/UBP/IFMA, Campus des Cézeaux, 63178 Aubière Cedex
prénom.nom@univ-bpclermont.fr

Résumé

Cet article propose des méthodes de synchronisation et d'auto-étalonnage pour plusieurs caméras grand public fixées sur un casque, sans utiliser de mire ou de cible. L'hypothèse principale est que toutes les caméras ont la même configuration (fréquence, résolution d'image, angle de champs de vue, et approximativement équiangulaires). On synchronise d'abord en utilisant le fait que toutes les caméras ont le même mouvement de rotation au même instant. Ensuite, tous les paramètres extrinsèques et intrinsèques sont raffinés par Structure-from-Motion et un ajustement de faisceaux. Nos expériences incluent des évaluations quantitatives et des vidéos urbaines acquises avec une multi-caméra fait maison (4 caméras GoPro fixées sur un casque), de sorte que l'on obtienne un champs de vue de 360 degrés dans le plan horizontal.

Mots Clef

Synchronisation, auto-étalonnage, ajustement de faisceaux multi-caméra.

Abstract

This paper proposes methods that synchronize and self-calibrate helmet-held consumer cameras, without the use of a calibration pattern. The main assumption is that the cameras have the same setting (frequency, image resolution, field-of-view, roughly equiangular). First the synchronization is done using the fact that all cameras have the same rotation motion at the same time. Second, all extrinsic and intrinsic parameters are refined using Structure-from-Motion and bundle adjustment. Our experiments include both quantitative evaluation and urban videos acquired by a DIY multi-camera (4 GoPro cameras mounted on a helmet) that provides a 360 degree field-of-view in the horizontal plane.

Keywords

Synchronization, self-calibration, multi-camera bundle adjustment.

1 Introduction

La modélisation 3D automatique d'environnements à partir de séquences d'images est un sujet déjà bien étudié. A ce jour, il y a plusieurs logiciels qui reconstruisent des objets ou des petits environnements en raison du faible champs de vue de caméra, sauf si les images sont aériennes. Des systèmes flexibles pour reconstruire des environnements complets (voire immersifs) à partir d'images terrestres sont toujours inexistants. Dans cet article, des vidéos sont prises par des caméras fixées sur un casque, en se déplaçant à faible vitesse dans un environnement urbain. En pratique, nous utilisons 4 caméras GoPro enfermées dans une boîte en carton et fixées sur un casque (Fig. 1). On se concentre ici sur les étapes de synchronisation et d'auto-étalonnage de la multi-caméra obtenue. Même si une télécommande Wifi permet de démarrer toutes les vidéos en même temps, cette synchronisation n'est pas

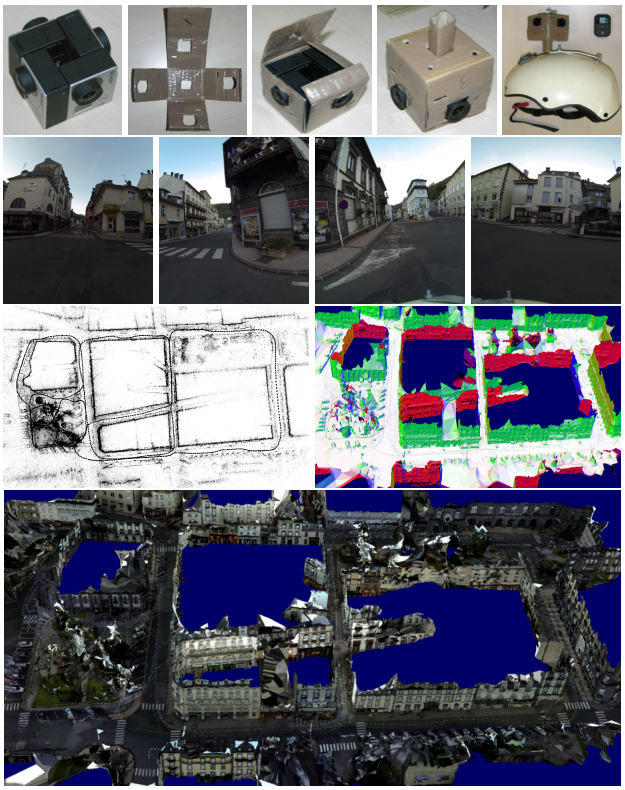

FIGURE 1 – De haut en bas : 4 caméras GoPro Hero3 enfermées dans une boîte en carton et fixées sur un casque, images capturées par notre multi-caméra, nuage de points et surface reconstruits avec 4 vidéos acquises pendant 703s de marche dans un village. La surface reconstruite est montrée dans la vidéo http://youtu.be/5r46SEBvz5w.

assez précise pour des applications de modélisation 3D (l'écart entre deux vidéos peut dépasser 0.1s). De plus, nous supposons que toutes les caméras ont la même configuration (fréquence, résolution, angle de champs de vue et approximativement équiangulaire) facile à obtenir en pratique. On peut mettre plus de caméras [1, 3] afin d'augmenter les recouvrements de champs de vues (ce qui facilite la mise en correspondance entre caméras), mais le prix augmente. Dans notre cas, ces recouvrements sont trop petits pour obtenir automatiquement une mise en correspondance entre caméras en utilisant [2, 5, 6].

1.1 Travaux antérieurs

Synchronisation. L'article [10] propose un état de l'art sur la synchronisation de vidéos, mais ces méthodes nécessitent une mise en correspondances entre caméras ou un champs de vue (partiellement) recouvrant, ou sont conçues pour un système de caméras non rigide. Certaines méthodes ont une précision "sous-trame" : le décalage temporel entre deux vidéos est un nombre d'images non entier (un réel). Plusieurs méthodes exploitent les points d'intérêt observés dans les vidéos et alignent des vidéos en se basant sur des caractéristiques visuelles.
Une méthode voisine de la notre [20] évite une mise en correspondance entre caméras : elle calcule pour chaque vidéo une suite de transformations entre images successives, et le décalage temporel est celui qui corrèle le mieux les suites de deux vidéos. Un exemple intuitif est celui où la transformation est la translation 2D calculée avec un suivi de points : plus la translation est grande dans une vidéo, plus la translation est grande dans une autre.

Auto-étalonnage de caméras. Dans le cas d'une caméra (monoculaire), il y a une approche classique [11, 23] pour estimer les paramètres intrinsèques. Les étapes sont : reconstruction projective, auto-calibration (ex : en supposant que les pixels sont carrés), et ajustement de faisceaux (AF). La distortion radiale se calcule aussi [17].
Dans le cas d'une multi-caméra, il faut d'abord une estimation initiale des poses entre caméras (ie rotations et translations relatives entre caméras). On peut les calculer à partir de reconstructions obtenues séparement pour chaque caméras, et si elles sont synchronisées. Il y a deux types de méthodes pour cela : des recalages basés sur les points 3D [8] ou sur les poses [9]. Dans le premier cas, les points sont mis en correspondance grâce à leur projections dans deux vidéos différentes et par l'estimation robuste d'une similitude. Dans le second cas, le mouvement ne dois pas être une translation pure.
Il faut enfin raffiner la géométrie multi-caméra avec un AF. L'AF de [13] raffine les poses relatives entre caméras, en plus des poses multi-caméras et des points 3D, en minimisant une erreur de reprojection dans l'espace image rectifié. L'AF de [19] traite le cas de points à l'infini et utilise les directions de rayons comme observations. Le raffinement de paramètres intrinsèques (focale, distortion, ...) ne sont suggérés qu'en travaux futurs dans [13, 19].

1.2 Contributions

Une version précédente de cet article a été publiée en anglais [14]. Les contributions ne portent pas sur l'initialisation des paramètres intrinsèques (partie 1.1) mais sur la synchronisation et l'AF. Ici on applique des Structure-from-Motion (SfM) mono et multi-caméras [18] avec des paramètres intrinsèques approximativement connus, qui sont ensuite raffinés par AF. On a une estimation initiale des paramètres intrinsèques de chaque caméra car on connait approximativement leur champs de vue et elles sont approximativement équiangulaires. On a aussi une estimation initiale des poses relatives entre caméras (mêmes centres, rotation d'angle $2\pi/k$ entre deux caméras adjacentes, avec $k = 4$ caméras d'après Fig. 1).

Synchronisation. On n'utilise pas directement les points d'intérêts mais les poses successives de chaque caméra calculées par SfM monoculaire. Comme les caméras sont rigidement liées les unes aux autres, elles ont le même mouvement de rotation au même instant. On calcule donc une vitesse de rotation au cours du temps pour chaque caméra, et on cherche les décalages qui les font le mieux correspondre. La transformation utilisée dans [20] n'est pas une rotation mais une translation (heuristique) ou homographie/matrice fondamentale sans distortion radiale.

Ajustement de faisceaux. Nous améliorons de deux façons l'AF dans [13] qui est basé sur le modèle de distortion polynomial classique [21, 12] pour une caméra. Notre AF raffine les paramètres intrinsèques, pas seulement les paramètres extrinsèques entre caméras, et les autres paramètres 3D (poses multi-cameras et points 3D). Il minimise l'erreur de reprojection dans l'espace image original (ie l'espace "distordu" où les points d'images sont détectés), pas dans l'espace image rectifié. Sous l'hypothèse courante que les bruits d'images sont de moyennes nulles, indépendants, et ont des distributions normales et identiques, le résultat obtenu par notre AF est un estimateur au maximum de vraisemblance (cette hypothèse n'est pas vraie dans l'espace rectifié).

2 Synchronisation

Soit $\mathtt{R}^t_{w,i}$ la matrice de passage du repère monde (w) au repère de la i-ième caméra pour sa t-ième image, ie la rotation de la caméra i à l'image t calculée par le SfM monoculaire. Soit $\mathtt{R}_{j,i}$ la matrice de passage du repère j au repère i (ie les coordonnées du repère de la caméra i dans celui de j, qui sont constantes). Supposons que l'image t de la caméra i et l'image t' de la caméra j sont prises au même instant. On obtient $\mathtt{R}_{w',w}\mathtt{R}^t_{w,i} = \mathtt{R}^{t'}_{w',j}\mathtt{R}_{j,i}$.
On a aussi $\mathtt{R}_{w',w}\mathtt{R}^{t+1}_{w,i} = \mathtt{R}^{t'+1}_{w',j}\mathtt{R}_{j,i}$ car les fréquences des vidéos sont identiques. Donc on a $\mathtt{R}^{t'+1}_{w',j}(\mathtt{R}^{t'}_{w',j})^\top = \mathtt{R}_{w',w}\mathtt{R}^{t+1}_{w,i}(\mathtt{R}^t_{w,i})^\top\mathtt{R}^\top_{w',w}$. L'angle de $\mathtt{R}^{t+1}_{w,i}(\mathtt{R}^t_{w,i})^\top$ est $\theta^t_i = \arccos((\mathrm{trace}(\mathtt{R}^{t+1}_{w,i}(\mathtt{R}^t_{w,i})^\top) - 1)/2)$.
Toutes les caméras ont donc ce même angle au même instant (c'est la vitesse angulaire instantanée de la multi-

caméra). On synchronise alors 2 caméras de la manière suivante : on calcule la table des θ_i^t pour chaque caméra i, puis on recherche le décalage entier o (offset) entre tables qui maximise un score de corrélation (ZNCC) des tables.
On peut raffiner o à une précision "sous-trame" de la même façon que l'on raffine une disparité entière à une précision sous-pixelique en stéréo [22]. La fonction f qui à $\epsilon \in [-1,+1]$ associe le score de corrélation pour le décalage $o+\epsilon$ est connue en $\epsilon \in \{-1,0,1\}$. On approxime f par un polynôme de degrés 2 connaissant ses valeurs en $\{-1,0,1\}$, et ϵ maximise ce polynôme. Soit $\tilde{o} = o + \epsilon$.
En pratique, on a plus de 2 caméras et on estime les décalages entiers $o_{0,1}, o_{1,2}, o_{2,3}, o_{3,0}$ dans le cas de 4 caméras entre caméras adjacentes i et $i+1$ (Fig. 1). Soit $L = o_{0,1} + o_{1,2} + o_{2,3} + o_{3,0}$. L'objectif de la synchronisation est de passer les s_i premières images de la i-ième vidéo, de sorte que les vidéos soient synchronisées pour le SfM et l'AF multi-caméra. On a donc 4 relations $s_i - s_{i+1} = o_{i,i+1}$ (indices modulo 4), et ceci n'est possible que si $L = 0$. Comme les décalages sont estimés de façon indépendente, on peut avoir $L \neq 0$. Dans ce cas, on cherche à remplacer chaque décalage o par un autre dans $\{o-k, \cdots o-1, o, o+1 \cdots o+k\}$ de sorte à maximiser la somme des corrélations (pour chaque o) sous la contrainte $L = 0$ ($k = 1$ suffit en pratique, et on ne modifie pas $\tilde{o}$).

3 Auto-étalonnage

La partie 3.1 décrit le modèle de distortion polynomial classique pour une caméra [21, 12]. Celui ci est souvent utilisé pour les calculs de SfM et de géométrie épipolaire (pour apparier), car sa projection inverse est explicite. La partie 3.2 explique comment initialiser les paramètres intrinsèques si la caméra est en gros équiangulaire à champs de vue connu. La partie 3.3 explique comment calculer la fonction de projection et ses dérivées (non explicites). Ceci est nécessaire pour l'AF multi-caméra.

3.1 Projection inverse

Les paramètres intrinsèques d'une caméra sont les distances focales (f_x, f_y), le point principal $\mathbf{p} = (p_x, p_y)$, et les coefficients k_i de distortion radiale. Soient $\mathbf{z}_d = (x_d, y_d)$ et $\mathbf{z}_u = (x_u, y_u)$ les coordonnées d'un pixel dans l'image originale ("distordue") et rectifiée ("non-distordue"). Les coordonnées normalisées sont $\overline{\mathbf{z}}_d = ((x_d - p_x)/f_x, (y_d - p_y)/f_y)$ et $\overline{\mathbf{z}}_u = ((x_u - p_x)/f_x, (y_u - p_y)/f_y)$. En notant $\overline{r}_d = ||\overline{\mathbf{z}}_d||$, on a $\overline{\mathbf{z}}_u = \left(1 + \sum_{i=1}^n k_i \overline{r}_d^{2i}\right) \overline{\mathbf{z}}_d$. Le rayon correspondant au pixel $\mathbf{z}_d$ a pour direction $\begin{pmatrix} \overline{\mathbf{z}}_u^\top & 1 \end{pmatrix}^\top$ dans le repère caméra.

3.2 Initialisation en caméra équiangulaire

Une caméra est équiangulaire si l'angle θ entre l'axe principal et le rayon de projection inverse est proportionnel à la distance radiale non-normalisée $r_d = ||\mathbf{z}_d - \mathbf{p}||$ dans l'image originale. Dans ce cas les pixels sont carrés (ie $f_x = f_y = f$) et on a une constante c telle que $\theta = cr_d = cf\overline{r}_d$. De plus, $\tan\theta = \overline{r}_u$ avec $\overline{r}_u = ||\overline{\mathbf{z}}_u|| = \overline{r}_d \left(1 + \sum_{i=1}^n k_i \overline{r}_d^{2i}\right)$. Nous obtenons donc

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

En outre, la série de Taylor de $\tan$ est

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

En identifiant les coefficients des équations 1 et 2, on obtient $cf = 1$ et $k_i = t_i$. En pratique, nous initialisons $\mathbf{p}$ au centre de l'image et prenons $f = r_d/\theta$ pour un pixel $\mathbf{z}_d$ au centre d'un bord d'image où le demi-champ de vue θ est connu approximativement.

3.3 Projection et ses dérivées partielles

Notre AF minimise la somme des $||\mathbf{z}_d - \tilde{\mathbf{z}}_d||^2$ avec $\tilde{\mathbf{z}}_d$ un point détecté dans l'image originale. On a besoin de calculer $\mathbf{z}_d$ à partir du point 3D qui lui correspond.
La projection du point 3D sur $\mathbf{z}_u$ étant classique, on ne détaille que la fonction associant $\mathbf{z}_d$ à $\mathbf{z}_u$. Cette fonction est définie implicitement par $g : \mathbb{R}^{n+8} \mapsto \mathbb{R}^2$, $g(\mathbf{z}_d, \mathbf{z}_u, \mathbf{p}, \mathbf{k}) = 0$ avec $\mathbf{k} = (f_x, f_y, k_1, ..., k_n)$ et

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

En effet, d'après le théorème des fonctions implicites, il existe localement une fonction φ de classe $\mathcal{C}^1$ telle que $\mathbf{z}_d = \varphi(\mathbf{z}_u, \mathbf{p}, \mathbf{k})$ si $\det \frac{\partial g}{\partial \mathbf{z}_d} \neq 0$.
On estime d'abord $\mathbf{z}_d$ à partir de $\mathbf{z}_u$, $\mathbf{p}$ et $\mathbf{k}$ en minimisant $\mathbf{z}_d \rightarrow ||g(\mathbf{z}_d)||^2$ par moindres carrés non linéaires (l'estimation initiale étant $\tilde{\mathbf{z}}_d$). On calcule ensuite les dérivées partielles de φ nécessaires à l'AF, comme

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

ainsi que les autres (par rapport à f_x, f_y, $\mathbf{p}$, pose multi-caméra, pose caméra dans le repère multi-caméra, point 3D) avec la formule de dérivation des fonctions composées.

3.4 Comment évaluer la calibration ?

On calcule l'erreur de chaque paramètre intrinsèque en comparant avec la calibration calculée avec une mire [12], qui sert de vérité terrain : les erreurs absolues sur f_x, f_y, p_x, p_y en pixels et les erreurs relatives sur k_i (on fait la moyenne de ces erreurs sur toutes les caméras).
On aimerait aussi, avec un unique nombre, évaluer l'erreur de notre calibration multi-caméra avec celle de la vérité terrain. On utilise une distance d basée sur les angles entre les directions de rayons de calibrations correspondant à un même pixel. Il y a plusieurs raisons pour faire cela. D'abord on a seulement besoin de la direction des rayons pour le calcul de SfM central [18]. Ensuite on utilise la vérité terrain fournie par le constructeur de la multi-caméra

(Ladybug [4]), qui est une table de rayons. Enfin, des paramètres peuvent se compenser s'ils sont biaisés (ex : l'ambiguité rotation-point principal [7] d'une caméra).
Il faut aussi tenir compte du fait que les repères multi-caméras ne sont pas les mêmes. On estime pour cela la rotation $\mathtt{R}$ qui permet de passer d'un repère à l'autre. On minimise $e(\mathtt{R}) = \sum_{i=1}^{n} ||\mathbf{r}_i^{vt} - \mathtt{R}\mathbf{r}_i^{est}||^2$, avec $\mathbf{r}_i^{est}$ et $\mathbf{r}_i^{vt}$ qui sont les directions des rayons correspondants au même pixel pour les calibrations estimée et vérité terrain. Puis on définit $d = \sqrt{e(\mathtt{R})/n}$ avec n le nombre de rayons (échantillonnés) dans l'image multi-caméra.

4 Expérimentation

4.1 Contexte

Nous utilisons deux multi-caméras, les deux sont non-centrales, ont un champs de vue de 360^o dans le plan horizontal, et se déplacent dans des environnements urbains. La première (Ladybug [4]) est parfaitement calibrée et synchronisée. Elle est utilisée pour l'évaluation quantitative. Elle a 5 caméras global shutter prenant des images 1024×768 à 15 Hz (il y en a une 6-ième pointant vers le haut, mais on ne l'utilise pas dans l'article). Elle est fixée sur une plate-forme de voiture et est placée à environ 4m du sol. La seconde est composée de 4 caméras GoPro Hero 3 (bas coût), qui sont rolling shutter, fixées sur un casque, et prennent des images 1280×960 à 100 Hz. Dans tous les calculs, on utilise les images divisées par 2.

Synchronisation. On expérimente notre méthode de synchronisation A_c pour plusieurs calibrations c. Si $c = pat$, les caméras sont calibrées avec une mire [12]. Si $c = pg$, on utilise la calibration Ladybug du constructeur [4] (une table de rayons). Une caméra a 9 paramètres de calibration $f_x, f_y, \mathbf{p}$ et k_i avec $1 \leq i \leq 5$ comme [12, 13] (sauf si $c = pg$). Si $c = 90r$, on part d'une calibration équiangulaire que l'on raffine. D'abord on effectue le SfM [18] sur une des vidéos monoculaire avec une calibration équiangulaire à champs de vue égal à 90 degrés. Ensuite cette calibration est raffinée par AF global (d'où le "r"). Pour les autres vidéos, on refait le SfM [18] avec la calibration raffinée. Jusque là, seules des images clefs sont reconstruites, on effectue alors un calcul de pose pour toutes les images. En pratique, on ne calcule que les 2000 premières images de chaque vidéo. Enfin, on calcule les décalages comme dans la partie 2.

AF multi-caméra. On expérimente notre AF $B_{c,p}^l$ pour plusieurs initialisations c et hypothèses p et longueurs l de vidéos. L'AF optimise les points 3D, les poses successives du système multi-caméra, et étalonne intrinsèquement et extrinsèquement les caméras. Si $c = 90r$, les paramètres intrinsèques *initiaux* sont ceux calculés par la synchronisation (les 9 mêmes paramètres pour toutes les caméras). Le nombre de paramètres intrinsèques *estimés* par $B_{c,p}^l$ est lui égal à $9k$: chacune des k caméras a ses propres paramètres intrinsèques (bien qu'elles aient le même réglage). Sinon $c = pat$ et l'AF ne raffine pas les paramètres intrinsèques. On initialise aussi la pose des caméras dans le repère multi-caméra : les rotations sont d'angles multiples de $2\pi/k$ autour de l'axe oz ($k \in \{4, 5\}$) et les centres en $(0, 0, 0)$. La multi-caméra est centrale si $p = c$, sinon on relache cette contrainte et $p = nc$. La longueur l est en nombre d'images (par défaut, $l = 2000$).

4.2 Vérité terrain - LadyBug

Les décalages estimés entre caméras adjacentes sont dans la table 1 (sans contrainte de boucle $L = 0$) et la table 2 (avec contrainte $L = 0$). La table 1 montre que A_c donne les mêmes décalages entiers, et que les décalages sous-trames sont bons : très proches de la vérité terrain qui est 0 (l'erreur est toujours inférieure à 4e-3).

A_c	$o_{0,1}$	$o_{1,2}$	$o_{2,3}$	$o_{3,4}$	$o_{4,0}$	L
(*)	0	0	0	0	0	0
pg	-32e-4	13e-4	3e-4	-6e-4	20e-4	-2e-4
pat	-16e-4	12e-4	-8e-4	-26e-4	39e-4	1e-4
$90r$	-20e-4	6e-4	-1e-4	-10e-4	26e-4	1e-4

TABLE 1 – Décalages estimés sans contrainte $L = 0$ (Ladybug). En haut : décalages entiers identiques pour tout les c. En bas : décalages sous-trames.

A_c	$o_{0,1}$	$o_{1,2}$	$o_{2,3}$	$o_{3,4}$	$o_{4,0}$	ZNCC
pg	0	0	0	0	0	4.964
	0	1	-1	0	0	4.489
pat	0	0	0	0	0	4.987
	0	0	0	-1	1	4.516
$90r$	0	0	0	0	0	4.985
	0	0	0	-1	1	4.512

TABLE 2 – Décalages avec contrainte $L = 0$ (Ladybug) pour les deux meilleurs scores ZNCC ($-5 \leq$ ZNCC ≤ 5).

Une fois les décalages entiers calculés (tous nuls avec $L = 0$), on applique le SfM et l'AF multi-caméras pour estimer la calibration multi-caméra. La table 3 montre l'erreur de calibration d (partie 3.4) et la table 4 montre les erreurs sur les paramètres intrinsèques, pour $B_{c,p}$, en optimisant dans l'espace image rectifié ou original. Si $c = 90r$,les erreurs sont meilleurs (plus petites) pour l'espace original (et légèrement mieux si $c = pat$). Sur ce jeux de données, l'approximation centrale semble justifiée car l'hypothèse non-centrale n'apporte rien.

4.3 4 caméras GoPro sur un casque

Les décalages estimés entre caméras adjacentes sont dans la table 5 (sans contrainte de boucle $L = 0$) et la table 6 (avec contrainte $L = 0$). On remarque qu'il y a un décalage ± 15 non négligeable entre deux caméras (0.15 secondes). De plus, la partie fractionnaire du décalage sous-trame de la caméra 0 par rapport à la caméra 1 (ou 3) est presque 0.5. Ceci explique que l'on obtienne deux

E.	Méth.	#3Dpts	#2Dpts	RMS	d
R.	$B_{90r,c}$	115499	432806	1.381	1.537
	$B_{pat,c}$	115492	432731	1.415	0.149
	$B_{90r,nc}$	115501	432687	1.382	1.604
	$B_{pat,nc}$	115489	432732	1.415	0.149
D.	$B_{90r,c}$	115594	433042	1.024	0.203
	$B_{pat,c}$	115596	433163	1.029	0.143
	$B_{90r,nc}$	115590	433036	1.024	0.238
	$B_{pat,nc}$	115594	433170	1.029	0.143

TABLE 3 – Résultats de l'AF multi-caméra (Ladybug) : nombre de points 3D reconstruits, de points 2D inliers et RMS d'erreurs de reprojections en pixels, distance angulaire d (en degrés) entre $B_{c,p}$ et la calibration constructeur. Espace (E.) : rectifié (R.) et distordu (D.).

		R.		D.	
	$i = pat_0$	$B_{90r,c}$	$B_{90r,nc}$	$B_{90r,c}$	$B_{90r,nc}$
f_x	545.791	24.489	23.286	2.450	1.760
f_y	545.774	2.765	3.368	0.810	0.728
p_x	510.697	1.259	1.236	1.165	1.185
p_y	386.952	2.919	3.010	0.886	1.144
k_1	0.408	0.347	0.352	0.035	0.033
k_2	0.130	13.441	13.746	1.252	1.309
k_3	0.363	2.914	2.974	0.443	0.452
k_4	-0.338	2.943	3.004	0.645	0.648
k_5	0.307	1.365	1.400	0.420	0.414

TABLE 4 – Vérité terrain et erreurs (Ladybug) sur les paramètres intrinsèques estimés par $B_{c,p}$ pour les cas central et non-central. Espace : rectifié (R.) et distordu (D.).

synchronisations à décalages entiers équivalentes au sens de la corrélation ZNCC ; cette corrélation est très bonne et similaire pour toutes les calibrations initiales c. Comme ici on a deux synchronisations possibles, la table 7 montre les différences que cela produit sur le résultat de l'AF multi-caméra au niveau des paramètres intrinsèques mais aussi de la distance angulaire d entre les deux calibrations multi-caméras estimées (une par synchronisation). Ces résultats montrent que les calibrations multi-caméras sont très semblables (d est inférieur à 0.07 degrés, et même 0.02 degrés si la séquence entière est utilisée dans l'AF). Les paramètres intrinsèques f_x, f_y, p_x, p_y sont les mêmes au pixel près, les coefficients de distortions k_i peuvent eux beaucoup varier (on définit ici l'erreur relative entre deux estimations k_i et k'_i par $2|k_i - k'_i|/(|k_i| + |k'_i|)$).

A_c	$o_{0,1}$	$o_{1,2}$	$o_{2,3}$	$o_{3,0}$	L
pat	-15	-1	14	1	-1
	-14.517	-0.957	14.031	1.426	-0.017
$90r$	-15	-1	14	1	-1
	-14.537	-0.939	14.019	1.440	-0.017

TABLE 5 – Décalages entiers et sous-trames (4 GoPro).

A_c	$o_{0,1}$	$o_{1,2}$	$o_{2,3}$	$o_{3,0}$	ZNCC
pat	-14	-1	14	1	3.916
	-15	-1	14	2	3.913
$90r$	-14	-1	14	1	3.917
	-15	-1	14	2	3.912

TABLE 6 – Décalages entiers avec contrainte $L = 0$ (4 GoPro), pour les deux meilleurs scores ZNCC ($-4 \leq$ ZNCC ≤ 4).

On choisit maintenant une des deux synchronisations à décalages entiers, et la table 8 compare les résultats de $B_{c,p}$ en terme de RMS et d'erreur angulaire d (entre $B_{90r,p}$ et $B_{pat,p}$). Dans tous les cas, on a des nombres de points 3D et inliers 2D similaires. On vérifie que minimiser dans l'espace distordu donne un résultat plus précis que minimiser dans l'espace rectifié. De plus, on note ici que l'on gagne en précision si on ne fait pas l'approximation centrale, contrairement au cas Ladybug (partie 4.2). On explique cela de la façon suivante. On sait que l'approximation centrale est d'autant moins tenable que le rapport entre les distances scène-caméra et caméra-caméra diminue, et ici ce rapport est plus petit pour les vidéos des 4 GoPro que par celles de Ladybug (les 4 GoPro se déplacent dans des rues plus étroites).

	$i = pat_0$	$B^{2k}_{90r,c}$	$B^{2k}_{90r,nc}$	$B^{70k}_{90r,c}$	$B^{70k}_{90r,nc}$
f_x	581.813	0.978	0.660	0.167	0.231
f_y	581.879	0.547	0.540	0.246	0.239
p_x	640.603	0.706	0.667	0.093	0.143
p_y	468.603	0.193	0.312	0.118	0.345
k_1	0.375	0.009	0.008	0.002	0.002
k_2	0.051	0.564	0.233	0.037	0.038
k_3	0.025	0.654	0.998	0.067	0.036
k_4	0.003	0.485	0.958	0.173	0.040
k_5	0.010	0.321	0.287	0.032	0.007
d		0.066	0.052	0.018	0.020

TABLE 7 – Différences entre deux synchronisations en termes de paramètres intrinsèques estimés par $B^l_{c,r}$ et distance angulaire d multi-caméra (4 GoPro), pour les séquences courtes (2k) et longues (70k), et pour les hypothèses centrale et non centrale.

Modélisation 3D d'un environnement urbain. Dans une version précédante [14] de cet article, on a utilisé l'une des synchronisations en table 6 et la méthode $B^{2k}_{90r,c}$ pour calculer les paramètres intrinsèques. Ceux-ci ont ensuite été utilisés pour calculer un nuage de points épars puis une surface texturée sur l'ensemble des vidéos (4*70k images). On résume ici très brièvement le calcul. On applique un SfM [18], puis les boucles sont détectées et fermées automatiquement. On a en tout 2.7M points 3D (points de Harris et points répartis sur des contours de Cany) pour 1834 images clefs. Un filtrage sur les points est ensuite appliqués (rejets celon les angles aperture, des points trop

E.	Méthode	#3Dpts	#2Dpts	RMS	d
R.	$B_{90r,c}$	15684	71462	1.215	0.539
	$B_{pat,c}$	15683	71410	1.312	0
	$B_{90r,nc}$	15684	71457	1.206	0.408
	$B_{pat,nc}$	15684	71437	1.288	0
D.	$B_{90r,c}$	15685	71487	0.876	0.376
	$B_{pat,c}$	15685	71450	0.911	0
	$B_{90r,nc}$	15685	71490	0.869	0.299
	$B_{pat,nc}$	15685	71462	0.898	0

TABLE 8 – Résultats de $B_{c,p}$ pour les séquences courtes (2k) : distance angulaire d (en degrés) entre $B_{90r,x}$ et $B_{pat,x}$, nombre de points 3D reconstruits, de points 2D inliers, et RMS d'erreurs de reprojections en pixels.

bas ou trop hauts). Enfin on applique une méthode de reconstruction de surface (une version de [15] améliorée avec [16]). La vidéo http://youtu.be/5r46SEBvz5w et la Fig. 1 illustrent les résultats (plus de détails dans [14]).

5 Conclusion

On a proposé une méthode flexible pour la synchronisation et l'auto-étalonnage (extrinsèque et intrinsèque) d'une multi-caméra, dont les caméras sont grand publics et fixées sur un casque. La méthode est applicable même lorsque les champs de vues recouvrants sont négligeables, et permet, pour la première fois, la modélisation 3D d'environnements avec ce genre de matériel.

On initialise d'abord les paramètres intrinsèques avec un champs de vue approximativement connu et l'hypothèse équiangulaire. On calcule ensuite les vitesses angulaires instantanées par Structure-from-Motion monoculaire et un ajustement de faisceaux. Les vidéos sont synchronisées par un alignement de leurs vitesses angulaires.

On améliore ensuite de deux façons un ajustement de faisceaux multi-caméra existant et basé sur un modèle de distortion polynomial classique, dont la fonction de projection n'est pas explicite : on raffine en plus les paramètres intrinsèques, et on minimise les erreurs de reprojections dans l'espace image original (de mesure/détection), pas dans l'espace image rectifié.

Dans les expériences, les étapes de synchronisation et auto-étalonnage sont évaluées quantitativement avec des vérités terrains. On compare aussi les calibrations obtenues entre minimisations dans les espaces images original et rectifié, avec ou sans l'approximation centrale de multi-caméra.

Des travaux futurs possibles incluent un ajustement de faisceaux raffinant simultanément la synchronisation et la géometrie, des investigations sur les caméras rolling-shutter et une calibration non-constante.

Références

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

[2] Hugin. http://hugin.sourceforge.net.

[3] Kolor. http://www.kolor.com.

[4] Ptgrey. http://www.ptgrey.com.

[5] Video-stitch. http://video-stitch.com.

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

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

[8] G. Carrera, A. Angeli, and A. Davison. Slam-based automatic extrinsic calibration of a multi-camera rig. In *ICRA 2011*.

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

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

[11] R. Hartley and A. Zisserman. *Multiple view geometry in computer vision*. C.U.P., 2004.

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

[13] P. Lebraly, E. Royer, O. Ait-Aider, C. Deymier, and M. Dhome. Fast calibration of embedded non-overlapping cameras. In *ICRA*, 2011.

[14] M. Lhuillier and T.-T. Nguyen. Synchronization and self-calibration for helmet-held consumer cameras, applications to immersive 3d modeling and 360 video. In *International Conference on 3D Vision, 2015*.

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

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

[17] B. Micusik and T. Pajdla. Structure from motion with wide circular field of view cameras. *PAMI*, 2006.

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

[19] J. Schneider and W. Förstner. Bundle adjustment and system calibration with points at infinity for omnidirectional cameras. *Technical Report -IGG-UB-2013*.

[20] L. Spencer and M. Shah. Temporal synchronization from camera motion. In *ACCV*, 2004.

[21] P. Sturm, S. Ramalingam, J.-P. Tardif, S. Gasparini, and J. Barreto. Camera models and fundamental concepts used in geometric computer vision. *Found. Trends. Comput. Graph. Vis.*, 2011.

[22] R. Szeliski and D. Scharstein. Symmetric sub-pixelic stereo matching. In *ECCV*, 2002.

[23] B. Triggs, P. F. McLauchlan, R. I. Hartley, and A. W. Fitzgibbon. Bundle adjustment-a modern synthesis. In *Vision algorithms : theory and practice*. 2000.