SIAM J. Imaging Sciences
Vol. 9, No. 3, pp. 869–900

Synchronization of Independently Moving Cameras via Motion Recovery*

Tiago Gaspar†, Paulo Oliveira†, and Paolo Favaro‡

Abstract. This work addresses the video synchronization problem, which consists in finding the temporal alignment between sequences of images acquired with unsynchronized cameras. This problem has been addressed before, under the assumptions that the cameras are static or jointly moving and that there are correspondences between the features visible in the different sequences. There are some methods in the literature that managed to drop one of these assumptions, but none of them was successful in getting rid of both assumptions simultaneously. In this work, we introduce a new strategy that synchronizes cameras that are allowed to move freely even when there are no correspondences between the features that are visible in the different sequences. Our approach consists in tracking features on two rigid objects that move independently on the scene and use the relative motion between them as a clue for the synchronization. New synchronization algorithms for static or jointly moving cameras that see (possibly) different parts of a common rigidly moving object are also presented. Even though the emphasis of this work is essentially on the theoretical contribution of the proposed methods, rather than on an exhaustive experimental validation, several proof of concept experiments conducted with both real and synthetic data are presented. In the case of static or jointly moving cameras, comparisons with a state-of-the-art approach are also provided.

Key words. video synchronization, multicamera systems, feature-based methods, nonmatching features, parameter estimation, structure from motion

AMS subject classifications. 68T45, 62H35

DOI. 10.1137/15M1035367

1. Introduction.

The recent advances in modern computing and in the quality of imaging sensors have contributed to the proliferation of computer vision-based systems. In particular, multicamera configurations have proved very useful, as they can be used in a wide range of contexts. When the scene is dynamic, as is often the case, merging information coming from the videos acquired with the different cameras requires that all of them are synchronized, i.e., that the correspondence between the frames of the several videos is known; see Figure 1. The existence of a method that solves the video synchronization problem is thus essential in areas such as three-dimensional (3D) reconstruction, human action recognition, calibration of multiple cameras, or dynamic depth estimation; see examples in [30], [32], [23], and [34], respectively.

*Received by the editors August 14, 2015; accepted for publication (in revised form) April 21, 2016; published electronically July 7, 2016. This work was partially supported by FCT, through IDMEC, under LAETA, project UID/EMS/50022/2013.
http://www.siam.org/journals/siims/9-3/M103536.html
†Instituto Superior Técnico, Universidade de Lisboa, Lisbon 1049-001, Portugal (tgaspar@isr.ist.utl.pt, pjcro@isr.ist.utl.pt).
‡Institut für Informatik und Angewandte Mathematik, Universität Bern, Bern CH-3012, Switzerland (favaro@iam.unibe.ch).

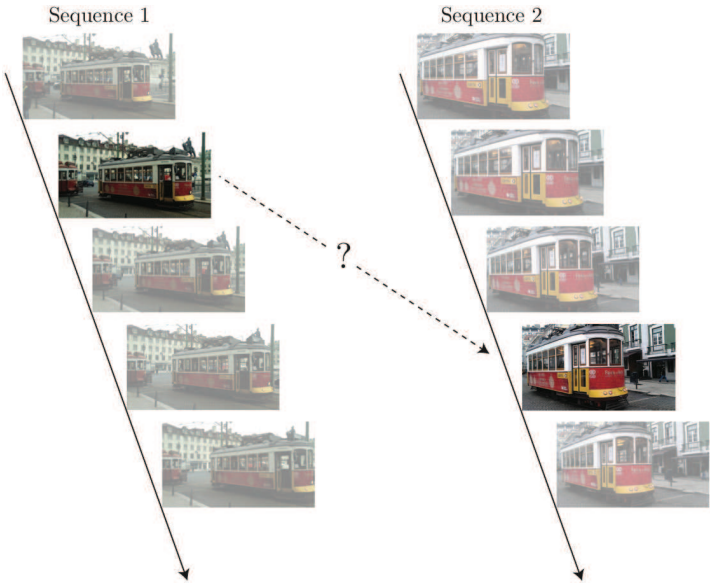


Figure 1. *Video synchronization problem.*

In professional applications, it is possible to synchronize several cameras using proper hardware, which is typically very expensive. A different approach consists in synchronizing videos manually, but this process is both tedious and difficult. This is the reason why the problem of synchronizing two videos, usually acquired by cameras with unknown relative intercamera extrinsic parameters, has received a lot of attention. Throughout this document, the synchronization of two cameras is addressed, but the methods that are proposed can be easily extended for more cameras.

Previous work. The video synchronization problem that has aroused more interest in the scientific community considers that the cameras are static and that there exist correspondences between the features observed in the two videos. Two of the directions of work that have been pursued to solve this problem are presented by Tresadern and Reid in [27] and by Caspi and Irani in [5], in what they called the "feature-based sequence alignment" approach. In the first case, the time offset is found by searching for the minimum of the relative magnitude of the fourth singular value of the "measurement matrix" introduced by Tomasi and Kanade in [26]. The second strategy aligns video sequences, in time and space, when the two sequences are related by a homography or by the projective epipolar geometry. To overcome the requirement that the cameras are static or jointly moving, i.e., that the relative intercamera extrinsic parameters do not change, and that correspondences between the features observed in the two videos exist, some research has been done on algorithms that drop one of these assumptions.

In [5], Caspi and Irani present a second method, the "direct intensity-based sequence alignment," which exploits spatio-temporal brightness variations within each sequence. This approach can handle complex scenes and drops the need for having feature correspondences across the two sequences, but it still requires that the cameras are static or jointly moving and that they see the same scene. In [30], Wolf and Zomet propose a strategy that builds on

the idea that every 3D point tracked in one sequence results from a linear combination of the 3D points tracked in the other sequence. This approach copes with articulated objects but still requires that the cameras are static or moving jointly.

There are also some works that can deal with independently moving cameras, but at the cost of requiring the existence of correspondences between features tracked in the two video sequences. Tuytelaars and Van Gool were the first to address the problem of automatic video synchronization for independently moving cameras and general 3D scenes; see [29]. Their method is based on the fact that although points on the scene may move nonrigidly, they can be considered as a rigid configuration for frames taken at the same time instant. Using this idea, they reformulate the video synchronization problem in terms of checking the rigidity of at least five nonrigidly moving points, matched and tracked throughout the two sequences. Another example is the work by Meyer et al. [18], which consists of a two-step approach. First, an algorithm that estimates a frame-accurate offset by analyzing the motion trajectories observed in the images and by matching their characteristic time patterns is used. After this step, subframe-accurate results are obtained by estimating a fundamental matrix between the two cameras, using a correspondence of nine nonrigidly moving points. Even though the motion of these points is not rigid, the correspondence between their projection into two images acquired by two cameras is perfectly explained by a fundamental matrix if such images are taken at the same time instant. The motion of the cameras and the motion of the tracked object are assumed to be linear between consecutive time instants.

As can be seen, video synchronization has been addressed from different perspectives. However, to the best of our knowledge, the most general and complex case, which arises when the cameras move independently and the parts of the moving object in the field of view of each camera do not intersect, is yet to be solved. None of the previous strategies would work in this situation, as there is no correspondence between the features observed in the two cameras. In [31], Yan and Pollefeys suggest an algorithm that uses the correlation between the distributions of space-time interest points, representing special events in the videos, to synchronize them. This method does not explicitly require feature correspondences and static or jointly moving cameras, but their fields of view must intersect and its performance degrades for wide baselines.

Contributions. The main contribution of this work is a method that synchronizes two video sequences acquired by independently moving cameras that see (possibly) different parts of a common rigidly moving object. The scene recorded by the cameras must also include a second common object (typically a static background), whose motion must be independent of the one of the first object. From now on, this second object is referred to as the background. The fields of view of the two cameras may not intersect and no knowledge about the correspondence between the two video sequences, in terms of which trajectories belong to which objects, is required. This is a general synchronization problem that arises, for instance, when two people use handheld cameras to record a rigid object moving on a static background (e.g., a car moving on the street). The relative intercamera extrinsic parameters are unknown and the intrinsic parameters of the cameras are assumed to be known. These parameters can be self-calibrated (see [8]) or calibrated a priori using the typical approaches (see [33] and [3]). The proposed synchronization methods require tracking two sets of features in each video

sequence: one on the moving object and the other on the background. The feature tracks are assumed to be long enough and are used to retrieve the motion of the two objects with respect to each camera using state-of-the-art structure and motion methods; see [13] and [25]. These results can be used to obtain information about the motion of one object with respect to the other, which is used as clue for the synchronization process. When the correspondences between the features and the two trajectories are not known, subspace clustering algorithms, such as the ones presented in [7] and [14], can be used to segment the two motions.

In addition to the previous contribution, a new set of methods that synchronize two videos acquired by static or jointly moving cameras that see (possibly) different parts of a common rigidly moving object is also presented. These methods are closely related to the one mentioned in the previous paragraph and are introduced first in the document as they serve as a starting point for the general case of independently moving cameras. The assumptions about the intrinsic and extrinsic parameters, and fields of view of the cameras, are the same as before. In this case, the motion of the object with respect to each camera can be used directly as a clue for the synchronization, due to the constraints imposed on the motion of the cameras. A study of the uniqueness of the solutions obtained with these methods is also presented.

The frame rate of the two videos is assumed to be the same, hence a single temporal offset between them is considered. This is without loss of generality since accurate information about the frame rate is usually available on the metadata of the videos. Thus, the multirate problem can be tackled by interpolating the measurements of the features in the video acquired with the lowest frame rate (note that typical object motions are smooth). The strategies proposed in this work can be used after this resampling.

Finally, it should also be stressed that the proposed methods require some interaction with the user, as they rely on him to select features on one or two objects (depending on the application at hand) and to ensure that they are tracked properly during the whole video sequences used for the synchronization. This is an important step as inaccurate tracks may compromise the performance of the methods used to recover the motion of the objects. The user should also guarantee that the features tracked on the two videos stem from the same objects. The synchronization algorithms take as input the 2D trajectories of the features in the two video sequences and output the temporal offset between them.

This paper is an extended version of the work presented in [9]. The main difference is the inclusion of the following:

- the proof of the lemma that studies the uniqueness of the solution for static or jointly moving cameras;
- a new set of methods to synchronize static or jointly moving cameras, when the moving object describes certain types of trajectories;
- a detailed description of the strategy used to retrieve the 3D motion of the objects from the time evolution of the projections of their features into the images;
- simulation results that complement the experimental ones by assessing the performance of the proposed methods for different scenarios and noise conditions.

Notation and paper organization. In this document, the identity matrix with dimensions $k \times k$ is denoted $\mathbf{I}_k$, and $\mathbf{0}_{k\times n}$ is used to represent a matrix of zeros with k lines and n columns. The notation $||\mathbf{v}||$ denotes the Euclidean norm of the vector $\mathbf{v}$ and $[\mathbf{v}]_\times$ is used to represent

the skew-symmetric matrix obtained from a given vector $\mathbf{v} \in \mathbb{R}^3$. This matrix is such that $[\mathbf{v}]_\times \mathbf{s} = \mathbf{v} \times \mathbf{s}$, for any vector $\mathbf{s} \in \mathbb{R}^3$, where $\times$ represents the cross-product. The trace and determinant of a square matrix $\mathbf{A}$ are denoted $\mathrm{tr}(\mathbf{A})$ and $\det(\mathbf{A})$, respectively, and the set of rotation matrices is given by the special orthogonal group $SO(3) = \{\mathbf{R} \in \mathbb{R}^{3\times 3} : \mathbf{R}^T\mathbf{R} = \mathbf{R}\mathbf{R}^T = \mathbf{I}_3,\ \det(\mathbf{R}) = 1\}$. For a generic rotation matrix $\mathbf{R} \in SO(3)$, the corresponding unit quaternion is given by $\mathbf{q} = [\sin(\theta/2)\mathbf{v}^T\ \cos(\theta/2)]^T$, where θ and $\mathbf{v}$ denote, respectively, the associated nonnegative angle of rotation and the unit Euler axis. These quantities are such that $\mathbf{R} = e^{[\mathbf{v}]_\times \theta}$. The natural logarithm of a scalar x is given by $\log(x)$.

This paper is organized as follows. In section 2, a strategy to recover the 3D motion of an object from the time evolution of the images of its features is described. The new algorithms that synchronize video sequences acquired by static or jointly moving cameras are presented in section 3, as well as a study of the object trajectories that lead to a unique identification of the correct temporal offset. One of these algorithms is generalized for independently moving cameras in section 4. Simulation and experimental results illustrating the performance of the proposed synchronization algorithms are presented in section 5. Section 6 provides some concluding remarks.

2. Object motion recovery.

In this section, the strategy used to retrieve the 3D motion of an object (i.e., to obtain the time evolution of its rotation matrix and translation vector) from a video sequence is described. Our approach consists of two parts: finding a first estimate for the motion of the object and refining such estimate.

There are several strategies that can be used to retrieve both structure and motion from a sequence of images. A good overview of several methods that solve this problem, using either affine or perspective camera models, can be found in [13]. Approaches based on affine camera models are typically simpler but lead to poor results when the object thickness is significant when compared to its distance to the camera. In this work, a strategy based on the perspective camera model is used, so that such limitations are avoided.

Consider a set of 3D features that move rigidly, being their motion at time instant k, with respect to the initial time instant, described by a rotation $\mathbf{R}_f(k) \in SO(3)$ and a translation $\mathbf{t}_f(k) \in \mathbb{R}^3$, for all $k \in [k_0, k_0 + F]$, where k_0 and F are such that k_0 and $k_0 + F$ correspond to the times of acquisition of the first and final video frames. Given the evolution over time of the projection of such features into the images acquired by a camera, it is possible to retrieve such rotations and the direction of such translations, both expressed in the reference frame of the camera, using standard strategies based on epipolar geometry; see [11]. In this work, the intrinsic parameters of the cameras are known, thus essential matrices can be used, rather than fundamental matrices. These matrices can be obtained using different strategies, depending on the number of available features; see examples in [11] and [24]. A comparison between several methods that require a minimum of five, six, seven, and eight points is presented in [24]. Once the essential matrices are obtained, the aforementioned rotations and normalized translations $\mathbf{t}_{f_n}(k)$, which verify $\mathbf{t}_f(k) = \mathbf{t}_{f_n}(k)||\mathbf{t}_f(k)||$, can be computed using the algorithms described in [11]. These rigid body transformations do not enforce a globally consistent geometry, as only the directions $\mathbf{t}_{f_n}(k)$ are retrieved, rather than the full 3D translation vectors $\mathbf{t}_f(k)$. A strategy that enforces such global consistency and, in addition, imposes some smoothness on the trajectory of the object is described next.

Let $\mathbf{y}_n(k)$, for $n = 1, 2, \ldots, N$, and $k \in [k_0, k_0 + F]$, where N is the number of tracked features, denote the normalized homogeneous coordinates of feature n, at instant k, expressed on the image plane. These coordinates are obtained from the measured homogeneous coordinates by correcting for the lens distortion and multiplying by the inverse of the intrinsic parameters matrix. Using this notation, we have that

$$\lambda_n(k)\mathbf{y}_n(k) = \mathbf{R}_f(k)\lambda_n(k_0)\mathbf{y}_n(k_0) + \mathbf{t}_{f_n}(k)\psi(k) \tag{2.1}$$

for all $k \in [k_0, k_0 + F]$, where $\lambda_n(k)$ is the unknown projective depth of feature n at instant k, and $\psi(k)$ is a scaling factor that compensates for the unknown magnitude of $\mathbf{t}_f(k)$ at the same instant. If the $\mathbf{R}_f(k)$ and $\mathbf{t}_{f_n}(k)$ obtained from the computed essential matrices are used, and if all the features and time instants are considered, the previous expression leads to the linear system $\mathbf{Hx} = \mathbf{0}$, where $\mathbf{x} \in \mathbb{R}^{F+N(F+1)}$ has the form

$$\mathbf{x} = [\psi(k_0 + 1) \ldots \psi(k_0 + F), \lambda_1(k_0) \ldots \lambda_1(k_0 + F), \ldots, \lambda_N(k_0) \ldots \lambda_N(k_0 + F)]^T.$$

The term $\psi(k_0)$ is not included in $\mathbf{x}$ as $\mathbf{R}_f(k_0) = \mathbf{I}_3$ and $\mathbf{t}_{f_n}(k_0) = \mathbf{0}_{3\times 1}$. The structure of $\mathbf{H}$ can be easily obtained by rearranging the terms in (2.1).

The projective depths that compose $\mathbf{x}$ can be found from the $\hat{\mathbf{x}}$ that solves the optimization problem

$$\hat{\mathbf{x}} = \arg\min_{\mathbf{x}} ||\mathbf{Hx}||^2 + \mu_\psi \sum_{k=k_0+2}^{k_0+F} ||\mathbf{t}_{f_n}(k)\psi(k) - \mathbf{t}_{f_n}(k-1)\psi(k-1)||^2 + \mu_\lambda \sum_{n=1}^{N} \sum_{k=k_0+1}^{k_0+F} ||\lambda_n(k) - \lambda_n(k-1)||^2$$

subject to

$$\sum_{k=k_0+1}^{k_0+F} \psi(k) = \gamma,$$

$$\lambda_n(k) > 0 \text{ for all } k \in [k_0, k_0+F] \text{ and } n = 1, 2, \ldots, N.$$

In these expressions, μ_ψ and μ_λ are positive weighting coefficients, and γ is a positive scalar that excludes the obvious solution $\mathbf{x} = \mathbf{0}$ and defines the magnitude of the translational component of the motion of the object. The two terms associated with the weighting coefficients are used to impose some smoothness on the trajectory of the object, and the inequality constraints guarantee that all the projective depths are positive. This problem is a quadratic program and can be solved using the approaches detailed in [4].

By combining the normalized homogeneous coordinates $\mathbf{y}_n(k)$ with the projective depths obtained with the previous procedure, for all $k \in [k_0, k_0 + F]$ and $n = 1, 2, \ldots, N$, a set of 3D point clouds, one per time instant, that evolve smoothly in time results. These clouds can be aligned with respect to the cloud associated with a chosen instant, typically the first, using the algorithm in [1], which finds the optimal rotation and translation between two point

clouds in the least-squares sense. This strategy leads to estimates for $\mathbf{R}_f(k)$ and $\mathbf{t}_f(k)$, for all $k \in [k_0, k_0+F]$, that are smooth and geometrically consistent. There is an overall ambiguity in the scale of the translation vectors that cannot be removed since there is no metric information about the scene.

Even though the 3D motion estimates obtained with the described method could be used as final estimates, a second step, consisting of a refinement procedure, is performed. This refinement, known in the literature as bundle adjustment (see [28] and [15]), is typically used as the last step of feature-based structure and motion estimation but has an important disadvantage, as the nonlinear least-squares problem that results has high computational and memory storage costs, due to the large number of parameters involved. To overcome this problem, the sparse bundle adjustment algorithm presented in [15] is used here. This approach exploits the lack of interaction between certain subgroups of parameters, which leads to a sparse Jacobian, to achieve significant computational savings. The motion estimates used to attain the results presented in this work were obtained using the two-step approach described above. The C/C++ code made available by the authors [15] was used to implement the sparse bundle adjustment algorithm.

3. Static and jointly moving cameras. This section addresses the synchronization of static or jointly moving cameras when no correspondences between the features tracked in the two videos exist; see Figure 2. Instead of explicitly using the features tracked on a moving object to get the synchronization, the rigid body transformations that explain the trajectories of such features when expressed in the reference frame of each one of the cameras are used.

Synchronizing two static cameras that see a common moving object is equivalent to synchronizing two moving cameras, as long as they move jointly and their motion is such that the motion perceived from the images of the features on their fields of view (with respect to the cameras) is the same as the motion perceived for the moving object (with respect to the static cameras). Since no matching between the features observed by the two cameras is required, the two topologies presented in Figure 3 are equivalent to the one presented in Figure 2. Note that, throughout this document, the expression *jointly moving cameras* is used to convey the idea that the cameras move in such a way that there exists a time-invariant

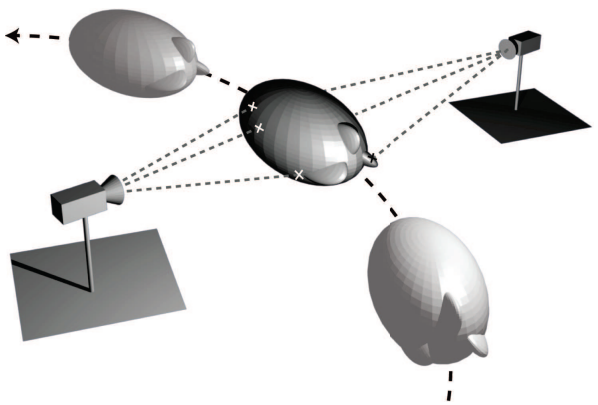
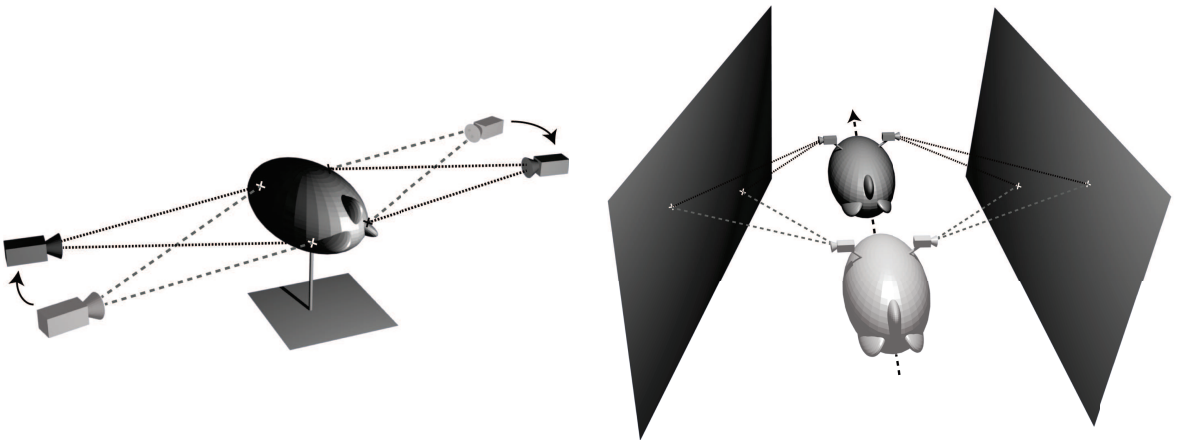


Figure 2. *Example of setup for the synchronization problem when the cameras are static. The blimp represents the moving object.*



(a) The fields of view of the two cameras may intersect, but no matching between the two sets of features exists.

(b) The fields of view of the two cameras do not intersect.

Figure 3. *Equivalent problems for static scenes and moving cameras.*

rigid body transformation between them over time, i.e., the relative position and orientation of one camera with respect to the other remain constant.

Let $\mathbf{x}_i(k) \in \mathbb{R}^3$, with k used to denote an integer in the range $[k_0, k_0 + F]$, where k_0 and F are such that k_0 and $k_0 + F$ correspond to the times of acquisition of the first and final frames of synchronized video sequences, denote the 3D coordinates of an object feature expressed in the reference frame of camera i, $i = 1, 2$, at the time of acquisition of the $(k - k_0 + 1)$th frame. These coordinates can be obtained from the coordinates of the same feature at the time instant associated with k_0 as $\mathbf{x}_i(k) = \mathbf{R}_i(k)\mathbf{x}_i(k_0) + \mathbf{t}_i(k)$, $i = 1, 2$, for all $k \in [k_0, k_0 + F]$, where $\mathbf{R}_i(k) \in SO(3)$ and $\mathbf{t}_i(k) \in \mathbb{R}^3$ denote, respectively, a rotation matrix and a translation vector that describe the evolution in time of the coordinates of object features expressed in the reference frame of camera i.

Since the cameras are static or jointly moving, there exist a constant rotation matrix and a constant translation vector that transform coordinates expressed in the reference frame of camera 1 into the one of camera 2, similarly to what happens in the hand-eye calibration problem; see [12]. If these rotation and translation are denoted $\mathbf{R} \in SO(3)$ and $\mathbf{t} \in \mathbb{R}^3$, respectively, then it is possible to show that $\mathbf{x}_2(k) = \mathbf{R}\mathbf{R}_1(k)\mathbf{R}^T\mathbf{x}_2(k_0) + \mathbf{R}\mathbf{t}_1(k) + (\mathbf{I}_3 - \mathbf{R}_2(k))\mathbf{t}$, and consequently

$$\mathbf{R}_2(k) = \mathbf{R}\mathbf{R}_1(k)\mathbf{R}^T \quad \text{and} \quad \mathbf{t}_2(k) = \mathbf{R}\mathbf{t}_1(k) + (\mathbf{I}_3 - \mathbf{R}_2(k))\mathbf{t}, \tag{3.1}$$

for all $k \in [k_0, k_0 + F]$, when the two videos are synchronized. These expressions are not valid for unsynchronized videos, except in the cases detailed in section 3.2.

3.1. Video synchronization—general case. In this section, a synchronization strategy for static cameras that uses information about both the rotational and translational components of the motion of the moving object is presented. Unlike the strategies presented in section 3.3,

this method copes with all types of trajectories of the object, as long as such trajectories have enough information for the synchronization to succeed.

There are several methods to track features on an object moving on a video, being one of the most used the KLT feature tracker [22]. By combining such strategies with structure and motion algorithms (see [13] and [25]), it is possible to retrieve the motion of the object, apart from a nonnegative scaling factor in the magnitude of its translational component. More details about this procedure are provided in section 2. By applying this strategy to the two videos, the quantities $\mathbf{R}_1(k)$, $\alpha_1\mathbf{t}_1(k)$, $\mathbf{R}_2(k)$, and $\alpha_2\mathbf{t}_2(k)$ are obtained for all $k \in [k_0, k_0+F]$. The constants α_1 and α_2 are nonnegative scalars that account for the scaling ambiguity in the magnitude of the translation of the moving object.

According to the discussion above, for unsynchronized videos the expressions in (3.1) have the form

$$\mathbf{R}_2(k') = \mathbf{R}\mathbf{R}_1(k)\mathbf{R}^T, \tag{3.2}$$

$$\alpha_2\mathbf{t}_2(k') = \mathbf{R}\mathbf{t}_1(k) + (\mathbf{I}_3 - \mathbf{R}_2(k'))\mathbf{t} \tag{3.3}$$

for all $k \in [k_0, k_0 + F]$, with $k' = k + \delta$, where δ denotes the temporal offset between the two sequences. This offset is considered to belong to a given interval, $\delta \in [-\Delta, \Delta]$, with $\Delta \leq F$ positive and known. Even though the two videos are unsynchronized, they are assumed to have at least $F + 1$ frames acquired at the same time instants; see Figure 4. In the expressions, α_1 is considered to be the unit. This is without loss of generality, as there is an overall ambiguity in the magnitude of the two members of (3.3).

If quaternions are used to parameterize attitude, (3.2) takes the form

$$\mathbf{q}_2(k').\mathbf{q} = \mathbf{q}.\mathbf{q}_1(k), \tag{3.4}$$

where $\mathbf{q}_2(k')$, $\mathbf{q}_1(k)$, and $\mathbf{q}$, are the unit quaternions (quaternions with unit norm) associated with $\mathbf{R}_2(k')$, $\mathbf{R}_1(k)$, and $\mathbf{R}$, respectively, and "." denotes the quaternion multiplication operator. See [19] and [6] for details about the use of quaternions to represent rotations and the end of section 1 for details about the notation used to represent quaternions. Under this framework, (3.4) can be written as

$$\mathbf{M}(\mathbf{q}_1(k), \mathbf{q}_2(k'))\mathbf{q} = \mathbf{0}_{4\times 1} \quad \text{with} \tag{3.5}$$

$$\mathbf{M}(\mathbf{q}_1(k), \mathbf{q}_2(k')) = \left[\mathbf{\Psi}(\mathbf{q}_2(k')) - \mathbf{\Xi}(\mathbf{q}_1(k)) \quad \mathbf{q}_2(k') - \mathbf{q}1(k)\right].$$

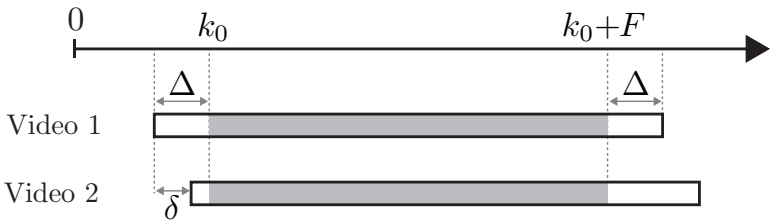


Figure 4. *Example of timeline for videos with $F+1+2\Delta$ frames, in which the regions in gray represent the $F+1$ frames used in the synchronization, and the remaining 2Δ frames guarantee that the algorithms proposed in this work can test all the delays in the interval $[-\Delta, \Delta]$.*

For a given quaternion $\mathbf{g} = [\mathbf{w}^T \; g_4]^T$, the matrices $\mathbf{\Xi}(\mathbf{g})$ and $\mathbf{\Psi}(\mathbf{g})$ have the form

$$\mathbf{\Xi}(\mathbf{g}) = \begin{bmatrix} g_4\mathbf{I}_3 + [\mathbf{w}]_\times \\ -\mathbf{w}^T \end{bmatrix} \quad \text{and} \quad \mathbf{\Psi}(\mathbf{g}) = \begin{bmatrix} g_4\mathbf{I}_3 - [\mathbf{w}]_\times \\ -\mathbf{w}^T \end{bmatrix}, \tag{3.6}$$

where $\mathbf{w} \in \mathbb{R}^3$ is a vector and $g_4 \in \mathbb{R}$ a scalar; see [6].

If (3.3) is also written as a function of the unit quaternion $\mathbf{q}$, we have that

$$\alpha_2 \mathbf{t}_2(k') = \mathbf{\Xi}^T(\mathbf{q})\mathbf{\Psi}(\mathbf{q})\mathbf{t}_1(k) + (\mathbf{I}_3 - \mathbf{R}_2(k'))\mathbf{t} \tag{3.7}$$

as $\mathbf{R} = \mathbf{\Xi}^T(\mathbf{q})\mathbf{\Psi}(\mathbf{q})$ [6]. The use of quaternions is to avoid singularities in the representation of rotations; see [19].

By combining (3.5) and (3.7), the synchronization problem for static or jointly moving cameras can be cast into the form of the minimization problem

$$\hat{\delta} = \arg\min_{\delta} \; E_s(\delta), \tag{3.8}$$

where $\hat{\delta}$ denotes the estimated temporal offset and $E_s(\delta)$ is the error function

$$E_s(\delta) = \min_{(\mathbf{q},\mathbf{t},\beta_2)} \mu_R E_R(\delta, \mathbf{q}) + \mu_T E_T(\delta, \mathbf{q}, \mathbf{t}, \beta_2) + \mu_q(\mathbf{q}^T\mathbf{q} - 1)^2 \tag{3.9}$$

with μ_R, μ_T, and μ_q, positive weighting coefficients. The last term in the expression forces $||\mathbf{q}||$ to be the unit and the other two are obtained from

$$E_R(\delta, \mathbf{q}) = \sum_{k=k_0}^{k_0+F} ||M(\mathbf{q}_1(k), \mathbf{q}_2(k+\delta))\mathbf{q}||^2,$$

$$E_T(\delta, \mathbf{q}, \mathbf{t}, \beta_2) = \sum_{k=k_0}^{k_0+F} ||\beta_2^2\mathbf{t}_2(k+\delta) - \mathbf{\Xi}^T(\mathbf{q})\mathbf{\Psi}(\mathbf{q})\mathbf{t}_1(k) - (\mathbf{I}_3 - \mathbf{R}_2(k+\delta))\mathbf{t}||^2.$$

The scalar β_2 is used to guarantee that α_2, with $\alpha_2 = \beta_2^2$, is not negative.

The optimization problem in (3.9) is a nonlinear least-squares problem due to the nonlinear dependence of $E_T(\delta, \mathbf{q}, \mathbf{t}, \beta_2)$ on $\mathbf{q}$ and β_2, and thus it can be solved using any nonlinear least-squares solver, such as the Levenberg–Marquardt method [17]. This problem transforms into a linear least-squares problem if α_2 is used and if $\mathbf{R}$, in (3.3), is considered to be a generic constant matrix $\mathbf{P} \in \mathbb{R}^{3\times3}$. Linear constraints on the trace of $\mathbf{P}$ and on the l_1 and l_∞ norms of its line and column vectors are imposed to guarantee that $\mathbf{P}$ is close to a rotation matrix; see [10] for details about these norms. If this relaxation is used and if the estimate found for $\mathbf{P}$, by solving the linear version of the problem, is approximated by a rotation matrix, an initial guess for $\mathbf{q}$, $\mathbf{t}$, and β_2 is easily found. This approximation can be obtained using the algorithm proposed in [21], which can be used to approximate a given matrix by the closest rotation matrix in the least-squares sense.

The temporal offset between the two videos is the one that solves (3.8) and is found by evaluating the error function in (3.9) for all the offsets in a given range.

3.2. Uniqueness of solution. There are situations in which the motion of the object does not have enough information for the synchronization to succeed (imagine, for instance, that the object is stopped or moves with constant velocity). In these cases, the solution of the minimization problem introduced in the previous section is not unique, i.e., there are several temporal offsets that minimize the error function in (3.9). This section presents the object trajectories that lead to such situation.

According to (3.2) and (3.3), it is possible to conclude that the solution of the optimization problem in (3.8) is unique in terms of the temporal offset δ (meaning that $E_s(\delta)$ is null only for the correct offset) if and only if there are not a nonnegative constant scaling factor, a constant rotation matrix, and a constant translation vector that verify such equations for some temporal offset, different from the real δ. The object trajectories that violate this condition are summarized in Lemma 3.1, where $\theta_i(k) \in \mathbb{R}$ and $\mathbf{v}_i(k) \in \mathbb{R}^3$ are used to denote, respectively, the nonnegative rotation angle and the corresponding Euler axis associated with $\mathbf{R}_i(k)$ for all $k \in [k_0 - \Delta, k_0 + F + \Delta]$. The conditions presented on the lemma depend on $\mathbf{t}_i$ and $\mathbf{R}_i$, but they do not need to be tested for both $i = 1$ and $i = 2$. It is enough to choose one of the cameras, for instance, camera 1, and test if $\mathbf{t}_1$ and $\mathbf{R}_1$ verify any of such conditions.

Lemma 3.1. *The solution of the optimization problem presented in (3.8) is unique if and only if none of the following three conditions are met for some nonnull $\overline{\delta}$ verifying $|\overline{\delta}| \leq \Delta$ with $\overline{\delta}_1 = \min[\overline{\delta}, 0]$ and $\overline{\delta}_2 = \max[0, \overline{\delta}]$:*

1. *$\theta_i(k) = 0$ for all $k \in [k_0 + \overline{\delta}_1,\ k_0 + F + \overline{\delta}_2]$ and there exist a nonnegative constant scalar $\overline{\alpha}$ and a constant rotation matrix $\overline{\mathbf{R}}$ such that $\overline{\alpha}\,\mathbf{t}_i(k + \overline{\delta}) = \overline{\mathbf{R}}\,\mathbf{t}_i(k)$ for all $k \in [k_0,\ k_0 + F]$;*
2. *$\theta_i(k)$ is periodic with period $|\overline{\delta}|$ for $k \in [k_0 + \overline{\delta}_1,\ k_0 + F + \overline{\delta}_2]$, the direction of $\theta_i(k)\mathbf{v}_i(k)$ is constant in the same interval, and there exist a nonnegative constant scalar $\overline{\alpha}$ and a constant rotation matrix $\overline{\mathbf{R}}$ such that $\theta_i(k + \overline{\delta})\mathbf{v}_i(k + \overline{\delta}) = \overline{\mathbf{R}}\,\theta_i(k)\mathbf{v}_i(k)$ for all $k \in [k_0,\ k_0 + F]$ and $\overline{\alpha}\,[\mathbf{t}_i(k + 2\overline{\delta}) - \mathbf{t}_i(k + \overline{\delta})] = \overline{\mathbf{R}}\,[\mathbf{t}_i(k + \overline{\delta}) - \mathbf{t}_i(k)]$ for all $k \in [k_0 - \overline{\delta}_1,\ k_0 + F - \overline{\delta}_2]$;*
3. *$\theta_i(k)$ is periodic with period $|\overline{\delta}|$ for $k \in [k_0 + \overline{\delta}_1,\ k_0 + F + \overline{\delta}_2]$, the direction of $\theta_i(k)\mathbf{v}_i(k)$ is not constant in the same interval, and there exist a nonnegative constant scalar $\overline{\alpha}$, a constant vector $\overline{\mathbf{t}}$, and a constant rotation matrix $\overline{\mathbf{R}}$ such that*

$$\begin{aligned} \theta_i(k + \overline{\delta})\mathbf{v}_i(k + \overline{\delta}) &= \overline{\mathbf{R}}\,\theta_i(k)\mathbf{v}_i(k), \\ \overline{\alpha}\,\mathbf{t}_i(k + \overline{\delta}) &= \overline{\mathbf{R}}\,\mathbf{t}_i(k) + (\mathbf{I}_3 - \mathbf{R}_i(k + \overline{\delta}))\overline{\mathbf{t}} \end{aligned}$$

for all $k \in [k_0,\ k_0 + F]$.

Proof. See Appendix A. ∎

Note that the conditions presented on the lemma can be easily tested for a given trajectory of the moving object.

3.3. Video synchronization—special cases. In this section, three new synchronization strategies for static cameras are presented. These strategies are particularly appropriate for situations where the motion of the object falls within the three categories that are addressed.

The method proposed in section 3.1 is general and can be used for all types of trajectories of the object, as long as its motion has enough information for the synchronization to succeed; see section 3.2. However, the underlying optimization problem requires the estimation of

the relative rigid body transformation between the two cameras, which contributes to the nonlinearity of the problem. This is not the case of the methods addressed in this section. The formulations that result do not require the estimation of the rigid body transformation between the two cameras, which significantly simplifies the optimization process. The special cases addressed here result from simplifications and/or transformations of (3.2) and (3.3) and occur when the target

1. rotates with nonconstant angle of rotation (this type of motion is here referred to as rotational motion);
2. translates and rotates, with respect to the reference frame of both cameras, and the Euler axis associated with this rotation is not orthogonal to the direction of translation during the whole video (this type of motion is here called nonconstrained motion);
3. translates but does not rotate (this type of motion is here referred to as purely translational motion).

Synchronization strategies that address these three situations are proposed next. Their simplicity is obtained at the cost of losing generality in their domain of applicability and using only part of the information that is available.

It is possible to test if the motion of the object falls within any of these three situations by retrieving such motion with respect to the reference frame of the cameras, using the procedure described in section 2, and checking if the angles of rotation, Euler axes, and translational vectors verify any of the aforementioned conditions.

Rotational motion. When the rotation matrix that defines the rotational component of the motion of the target varies over time, the use of (3.2) may be enough to synchronize the videos. According to the properties of the trace of a product of matrices (see [10]) and to the properties of rotation matrices, from (3.2) we have that $\text{tr}[\mathbf{R}_2(k')] = \text{tr}[\mathbf{R}_1(k)]$ for all $k \in [k_0, k_0 + F]$, which can be written in the form $\theta_2(k') = \theta_1(k)$ for all $k \in [k_0, k_0 + F]$, where $\theta_i(k)$, $i = 1, 2$, is the nonnegative angle of rotation associated with $\mathbf{R}_i(k)$, since $\text{tr}[\mathbf{R}_i(k)] = 1 + 2\cos(\theta_i(k))$, for $i = 1, 2$; see [19].

According to the previous reasoning, the angles of rotation associated with the rotation of the moving object with respect to each one of the cameras can be used for synchronization purposes, as long as they do not remain constant during the whole video sequences and are not periodic; see Lemma 3.1. In this case, the synchronization can be obtained by finding the temporal offset between the two videos that solves the optimization problem

$$\hat{\delta} = \arg\min_{\delta} \underbrace{\sum_{k=k_0}^{k_0+F} \left(\text{tr}[\mathbf{R}_2(k+\delta)] - \text{tr}[\mathbf{R}_1(k)]\right)^2}_{E_{tr}(\delta)}, \tag{3.10}$$

which is equivalent to the problem in (3.8), but with the error function $E_s(\delta)$ replaced by the function $E_{tr}(\delta)$.

Nonconstrained motion. Let $\theta_i(k) \in \mathbb{R}$ and $\mathbf{v}_i(k) \in \mathbb{R}^3$, $i = 1, 2$, denote, respectively, the nonnegative rotation angle and the corresponding unit Euler axis associated with $\mathbf{R}_i(k)$ for all

$k \in [k_0, k_0 + F]$. In other words, $\mathbf{v}_i(k)$ is the eigenvector of $\mathbf{R}_i(k)$ associated with eigenvalue 1, i.e.,

$$\mathbf{v}_i(k) = \mathbf{R}_i(k)\mathbf{v}_i(k) \quad \text{for all } k \in [k_0, k_0 + F]; \tag{3.11}$$

see [19]. Then, from (3.2), it is possible to conclude that

$$\theta_2(k')\mathbf{v}_2(k') = \mathbf{R}\theta_1(k)\mathbf{v}_1(k) \quad \text{for all } k \in [k_0, k_0 + F]. \tag{3.12}$$

If the inner product between $\theta_2(k')\mathbf{v}_2(k')$ and both members of (3.3) is computed, we have that

$$\alpha_2\, \theta_2(k')\, \mathbf{v}_2^T(k')\, \mathbf{t}_2(k') = \theta_2(k')\, \mathbf{v}_2^T(k')\, \mathbf{R}\, \mathbf{t}_1(k) + \theta_2(k')\, \mathbf{v}_2^T(k')\, (\mathbf{I}_3 - \mathbf{R}_2(k'))\, \mathbf{t}.$$

From this equation, and from (3.11) and (3.12), it results that

$$\alpha_2\, \theta_2(k')\, \mathbf{v}_2^T(k')\, \mathbf{t}_2(k') = \theta_1(k)\, \mathbf{v}_1^T(k)\, \mathbf{t}_1(k) \tag{3.13}$$

for all $k \in [k_0, k_0 + F]$.

This is an elegant expression that merges information from both (3.2) and (3.3), but it is not useful when

- the object does not rotate, as in this case both $\theta_1(k)$ and $\theta_2(k')$ are null for all $k \in [k_0, k_0 + F]$;
- the object does not translate with respect to the reference frames of the cameras, as in this case both $\mathbf{t}_1(k)$ and $\mathbf{t}_2(k')$ are null for all $k \in [k_0, k_0 + F]$;
- the object rotates and translates with respect to the reference frames of the cameras, but $\mathbf{v}_1^T(k)\, \mathbf{t}_1(k)$ and $\mathbf{v}_2^T(k')\, \mathbf{t}_2(k')$ are null for all $k \in [k_0, k_0 + F]$, i.e., the Euler axes are orthogonal to the directions of translation during the whole video sequences. This situation occurs, for instance, when the motion of the object is restricted to a plane (e.g., a car moving on the street).

In these cases, both members of (3.13) are null.

If the motion of the object is not in these categories, the cameras can be synchronized by solving the problem

$$\hat{\delta} = \arg\min_{\delta} \quad E_{vt}(\delta),$$

where the error function $E_{vt}(\delta)$ is given by

$$\min_{\alpha_2} \sum_{k=k_0}^{k_0+F} \big(\alpha_2 \theta_2(k+\delta)\mathbf{v}_2^T(k+\delta)\mathbf{t}_2(k+\delta) - \theta_1(k)\mathbf{v}_1^T(k)\mathbf{t}_1(k)\big)^2 \tag{3.14}$$

$$\text{subject to } \alpha_2 \geq 0,$$

which is equivalent to the problem in (3.8), but with $E_s(\delta)$ replaced by $E_{vt}(\delta)$. The constrained optimization problem can be solved using the methods described in [4].

In addition to the previous strategy, there is another neat approach, based on the use of essential matrices, that can be used for the nonconstrained motion. Instead of having static

cameras and a moving object, consider the dual situation in which the object is static and the cameras are moving (see the discussion in the beginning of section 3). In this case, the essential matrix associated with the pose of the ith camera in each time instant is obtained from $\mathbf{R}_i(k)$ and $\mathbf{t}_i(k)$ as $\mathbf{E}_i(k) = [\mathbf{t}_i(k)]_\times \mathbf{R}_i(k)$ with $i = 1, 2$; see [11]. From (3.2) and (3.3), it is possible to conclude that the essential matrix $\mathbf{E}_2(k')$, associated with camera 2, can be written as a function of the essential matrix $\mathbf{E}_1(k)$, associated with camera 1, and that this relation has the form

$$\alpha_2 \mathbf{E}_2(k') = \mathbf{R}\mathbf{E}_1(k)\mathbf{R}^T + [\mathbf{t}]_\times \mathbf{R}_2(k') - \mathbf{R}_2(k')[\mathbf{t}]_\times$$

for all $k \in [k_0, k_0+F]$. By computing the trace of each member of this equation, the expression

$$\alpha_2 \mathrm{tr}[\mathbf{E}_2(k')] = \mathrm{tr}[\mathbf{E}_1(k)] \quad \text{for all } k \in [k_0, k_0 + F] \tag{3.15}$$

results since, according to the properties of the trace of a product of matrices, $\mathrm{tr}([\mathbf{t}]_\times \mathbf{R}_2(k')) = \mathrm{tr}(\mathbf{R}_2(k')[\mathbf{t}]_\times)$ and $\mathrm{tr}(\mathbf{R}\mathbf{E}_1(k)\mathbf{R}^T) = \mathrm{tr}(\mathbf{E}_1(k))$; see [10].

The previous equation can also be used to synchronize two cameras but fails this task in the three situations mentioned above. In the first case, i.e., when the object does not rotate, it reduces to $\alpha_2 \mathrm{tr}\big([\mathbf{t}_2(k')]_\times\big) = \mathrm{tr}\big([\mathbf{t}_1(k)]_\times\big)$, which is null for all $k \in [k_0, k_0 + F]$, regardless of the type of translation of the target, due to the structure of skew-symmetric matrices. In the second case, i.e., when the object does not translate with respect to the reference frames of the cameras, the expression is also null, as $\mathbf{t}_1(k)$ and $\mathbf{t}_2(k')$ are both null for all $k \in [k_0, k_0+F]$. In the last case, in which the object rotates and translates with respect to the reference frames of the cameras, but $\mathbf{v}_1^T(k)\,\mathbf{t}_1(k)$ and $\mathbf{v}_2^T(k')\,\mathbf{t}_2(k')$ are null for all $k \in [k_0, k_0 + F]$, the two members of the expression in (3.15) are also null. In order to confirm this statement, let $\mathbf{R}_i(k)$, $i = 1, 2$, be written in the form

$$\mathbf{R}_i(k) = \mathbf{I}_3 + [\mathbf{v}_i(k)]_\times \sin(\theta_i(k)) + [\mathbf{v}_i(k)]_\times^2 \big(1 - \cos(\theta_i(k))\big),$$

where the Euler axis $\mathbf{v}_i(k)$ has unit norm; see Rodrigues' formula in [19]. According to this expression, we have that

$$\mathrm{tr}[\mathbf{E}_i(k)] = \mathrm{tr}\Big[[\mathbf{t}_i(k)]_\times + [\mathbf{t}_i(k)]_\times [\mathbf{v}_i(k)]_\times \sin(\theta_i(k)) + [\mathbf{t}_i(k)]_\times [\mathbf{v}_i(k)]_\times^2 \big(1 - \cos(\theta_i(k))\big)\Big].$$

Since the trace of a skew-symmetric matrix is null, due to its structure, and $\mathrm{tr}\big[[\mathbf{t}_i(k)]_\times [\mathbf{v}_i(k)]_\times\big] = -2\,\mathbf{t}_i^T(k)\mathbf{v}_i(k)$ (see [2] and references therein), then

$$\mathrm{tr}[\mathbf{E}_i(k)] = -2\,\mathbf{t}_i^T(k)\mathbf{v}_i(k)\sin(\theta_i(k)) + \mathrm{tr}\Big[[\mathbf{t}_i(k)]_\times [\mathbf{v}_i(k)]_\times^2 \big(1 - \cos(\theta_i(k))\big)\Big].$$

In this expression, $[\mathbf{v}_i(k)]_\times^2$ is symmetric, as it can be written in the form $[\mathbf{v}_i(k)]_\times^2 = \mathbf{v}_i(k)\mathbf{v}_i^T(k) - ||\mathbf{v}_i(k)||^2\mathbf{I}_3$; see [19]. Thus, according to Fact 3.7.23 in [2], which states that the trace of the product of a skew-symmetric and a symmetric matrix is null, the trace of $[\mathbf{t}_i(k)]_\times [\mathbf{v}_i(k)]_\times^2$ is zero. Since in the case of the trajectories under study $\mathbf{t}_i^T(k)\mathbf{v}_i(k) = 0$, it is possible to conclude that, for such trajectories, $\mathrm{tr}[\mathbf{E}_1(k)] = \mathrm{tr}[\mathbf{E}_2(k')] = 0$ for all $k \in [k_0, k_0 + F]$. Consequently, the two members of the expression in (3.15) are null for the aforementioned trajectories.

When the motion of the target is not among the three aforementioned categories, a new synchronization strategy, based on (3.15), can be used. Such strategy consists in solving the optimization problem

$$\hat{\delta} = \arg\min_{\delta} \quad E_{ess}(\delta),$$

where

$$E_{ess}(\delta) \quad = \quad \min_{\alpha_2} \quad \sum_{k=k_0}^{k_0+F} \left(\alpha_2 \mathrm{tr}[\mathbf{E}_2(k+\delta)] - \mathrm{tr}[\mathbf{E}_1(k)]\right)^2 \tag{3.16}$$
$$\text{subject to } \alpha_2 \geq 0,$$

which is equivalent to the problem in (3.8), but with $E_s(\delta)$ replaced by the error function $E_{ess}(\delta)$. This problem can be solved using one of the methods described in [4].

Purely translational motion. When the object does not rotate, (3.2) does not have useful information for the synchronization and (3.3) reduces to

$$\alpha_2 \mathbf{t}_2(k') = \mathbf{R}\mathbf{t}_1(k) \quad \text{for all } k \in [k_0,\, k_0+F],$$

as $\mathbf{R}_2(k')$ is the identity during the whole sequence. In this case, the cameras can be synchronized by analyzing if, for each possible time offset, the relation between $\mathbf{t}_1(k)$ and $\mathbf{t}_2(k')$ is accurately explained by the previous expression, i.e., if there exists a scaling factor α_2 and a rotation $\mathbf{R}$ that accurately map $\mathbf{t}_1(k)$ to $\mathbf{t}_2(k')$ for all $k \in [k_0,\, k_0+F]$. Thus, the cameras can be synchronized by solving the problem

$$\hat{\delta} = \arg\min_{\delta} \quad E_{pt}(\delta),$$

where

$$E_{pt}(\delta) \quad = \quad \min_{(\alpha_2,\mathbf{R})} \quad \sum_{k=k_0}^{k_0+F} \left|\left|\alpha_2 \mathbf{t}_2(k+\delta) - \mathbf{R}\mathbf{t}_1(k)\right|\right|^2$$
$$\text{subject to } \begin{array}{l} \alpha_2 \geq 0 \\ \mathbf{R} \in SO(3). \end{array}$$

The solution of this problem can be obtained using a singular value decomposition, as reported by Markley in [16]. Even though the problem addressed in that article, known as the Wahba's problem, does not include the ambiguity in scale introduced by α_2, this is not problematic. The only difference is that α_2 introduces an ambiguity in the magnitude of the singular values of the matrix whose singular value decomposition is used to solve the problem, but they are not used at any point in the algorithm. Thus, instead of the previous formulation, the optimization problem

$$E_{pt}(\delta) \quad = \quad \min_{\mathbf{R}} \quad \sum_{k=k_0}^{k_0+F} \left|\left|\mathbf{t}_2(k+\delta) - \mathbf{R}\mathbf{t}_1(k)\right|\right|^2 \tag{3.17}$$
$$\text{subject to } \mathbf{R} \in SO(3)$$

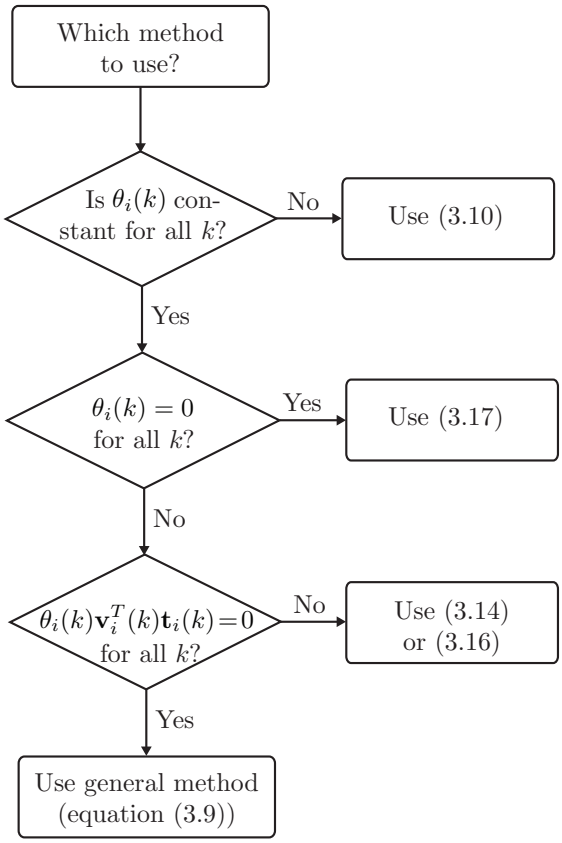


Figure 5. *Suggestion of procedure to decide which synchronization method to use, based on the trajectory of the moving object with respect to camera i, which is defined by the evolution of θ_i, $\mathbf{v}_i$, and $\mathbf{t}_i$, along time.*

can be solved to find the value of $E_{pt}(\delta)$. The optimal solution to this problem is provided in [16].

In all the situations presented in this section, the temporal offset between the videos is found according to the strategy described in the end of section 3.1, i.e., by evaluating the corresponding error functions, $E_{tr}(\delta)$, $E_{vt}(\delta)$, $E_{ess}(\delta)$, and $E_{pt}(\delta)$, for all the possible offsets in a given range.

Choosing the method that best suits a particular synchronization task is not easy, and usually there is not a single answer to this question. The flowchart in Figure 5 illustrates a possible procedure to help with this decision. In this figure, the general method associated with expression (3.9) appears as the option to use when all the other strategies fail, but recall that this method can be used in all the other situations.

4. Independently moving cameras. This section presents a new method that solves the video synchronization problem for situations where the cameras move independently and there are no correspondences between the features tracked in the two videos.

When videos are acquired with independently moving cameras, tracking features on a single rigidly moving object is not enough for the synchronization. This is because the projection of such features into acquired images results both from the motion of the object, which includes

information for the synchronization, and from the motion of the camera, which does not. In this situation, features on a second rigidly moving object, for instance, a static background, must be used. If the motion of this object is independent from the one of the first object, the relative motion between the two objects (i.e., the rotation and translation of one object with respect to the reference frame of the other object) has information for the synchronization.

Let ${}^{c_0}\mathbf{p}_i^j(k) \in \mathbb{R}^4$, for $k \in [k_0, k_0 + F]$, denote the homogeneous coordinates of a point of the jth object, $j = 1, 2$, at the time of acquisition of the kth frame. The superscript c_0 and the subscript i indicate that these coordinates are expressed in the reference frame $\{c_0\}$ of camera i, $i = 1, 2$, at the time of acquisition of the first frame k_0 of the video. The evolution in time of the coordinates of this point is given by ${}^{c_0}\mathbf{p}_i^j(k) = \mathbf{g}_i^j(k)\, {}^{c_0}\mathbf{p}_i^j(k_0)$, where $\mathbf{g}_i^j(k)$ denotes a homogeneous transformation, i.e., it is a 4×4 matrix that combines the rotation and translation information into a single matrix; see [19] for more details about homogeneous transformations. Moreover, let ${}^{c_k}_{c_0}\mathbf{g}_i(k)$ denote another homogeneous transformation, which converts coordinates of points expressed in $\{c_0\}$ into the coordinates of the same points expressed in $\{c_k\}$. Here, $\{c_k\}$ is used to identify the reference frame of camera i at the time of acquisition of frame k. This transformation represents the motion of camera i. If these two transformations are combined, a new transformation $\mathbf{g}_{T_i^j}(k)$ that includes both the motion of the jth object and the motion of the ith camera results:

$$ {}^{c_k}\mathbf{p}_i^j(k) = \underbrace{{}^{c_k}_{c_0}\mathbf{g}_i(k)\, \mathbf{g}_i^j(k)}_{\mathbf{g}_{T_i^j}(k)}\, {}^{c_0}\mathbf{p}_i^j(k_0). $$

This transformation relates ${}^{c_0}\mathbf{p}_i^j(k_0)$, the homogeneous coordinates in the initial instant of points of object j expressed in $\{c_0\}$, with ${}^{c_k}\mathbf{p}_i^j(k) \in \mathbb{R}^4$, their homogeneous coordinates at the time of acquisition of frame k expressed in $\{c_k\}$.

From the three aforementioned transformations, only $\mathbf{g}_{T_i^j}(k)$ can be obtained from the available features (apart from a nonnegative scaling factor) for all $k \in [k_0, k_0 + F]$. Thus, it is the only one that can be used to synchronize the cameras. Consider, for instance, the homogeneous transformation $\mathbf{g}_{T_i^1}(k) = {}^{c_k}_{c_0}\mathbf{g}_i(k)\, \mathbf{g}_i^1(k)$, associated with the motion of object 1 with respect to camera i, which can be written as $\mathbf{g}_{T_i^1}(k) = {}^{c_k}_{c_0}\mathbf{g}_i(k)\, \mathbf{g}_i^2(k)\, [\mathbf{g}_i^2(k)]^{-1}\, \mathbf{g}_i^1(k)$, since $\mathbf{g}_i^2(k)[\mathbf{g}_i^2(k)]^{-1} = \mathbf{I}_4$. If $[\mathbf{g}_i^2(k)]^{-1}\, \mathbf{g}_i^1(k)$, which does not depend on the motion of the cameras, is denoted by $\mathbf{g}_i(k)$, the previous expression can be rearranged in the form

$$ \mathbf{g}_i(k) = [\mathbf{g}_{T_i^2}(k)]^{-1}\, \mathbf{g}_{T_i^1}(k). \tag{4.1} $$

If the homogeneous transformation from the reference frame of camera 1, at the initial instant, to the reference frame of camera 2, at the same instant, is denoted $\mathbf{g}$, it is easy to show that $\mathbf{g}_2^j(k) = \mathbf{g}\, \mathbf{g}_1^j(k)\, \mathbf{g}^{-1}$, $j = 1, 2$, and consequently

$$ \mathbf{g}_2(k) = \mathbf{g}\, \mathbf{g}_1(k)\, \mathbf{g}^{-1}, \tag{4.2} $$

as $\mathbf{g}_i(k) = [\mathbf{g}_i^2(k)]^{-1}\, \mathbf{g}_i^1(k)$. When the two videos are synchronized, this expression is valid for all $k \in [k_0, k_0 + F]$.

If the rotations and translations associated with $\mathbf{g}$ and $\mathbf{g}_i(k)$, $i = 1, 2$, are denoted by $\mathbf{R} \in SO(3)$ and $\mathbf{t} \in \mathbb{R}^3$, and by $\mathbf{R}_i(k) \in SO(3)$ and $\mathbf{t}_i(k) \in \mathbb{R}^3$, respectively, then (4.2) can

be cast into the form of (3.1). The difference is that in section 3, $\mathbf{t}_i(k)$ was determined using structure and motion strategies, which is not possible in this case. Here, the translational components of $\mathbf{g}_{T_i^j}(k)$ can also be determined using structure and motion strategies, and thus they are known up to a scaling factor, but $\mathbf{t}_i(k)$ cannot. For independently moving cameras, $\mathbf{t}_i(k)$ is obtained using (4.1). This procedure induces some structure on $\mathbf{t}_i(k)$, which cannot be modeled with a single scaling factor. This is why the strategy proposed in section 3.1 cannot be used for independently moving cameras.

Video synchronization. Using the strategy described in the beginning of section 3.1, it is possible to retrieve the values of $\mathbf{R}_{T_i^j}(k)$ and $\alpha_i^j\,\mathbf{t}_{T_i^j}(k)$, with $i=1,2$, and $j=1,2$, for all $k\in[k_0,k_0+F]$. The rotation $\mathbf{R}_{T_i^j}(k)$ and translation $\mathbf{t}_{T_i^j}(k)$ are the ones associated with the homogeneous transformation $\mathbf{g}_{T_i^j}(k)$, and α_i^j is a nonnegative constant that accounts for the ambiguity in the magnitude of the translation of the jth object, when it is estimated using the features observed in camera i.

According to the discussion above and (4.1), we have that

$$\begin{aligned}\mathbf{R}_i(k) &= [\mathbf{R}_{T_i^2}(k)]^T\mathbf{R}_{T_i^1}(k),\\ \mathbf{t}_i(k) &= \alpha_i^1\underbrace{[\mathbf{R}_{T_i^2}(k)]^T\mathbf{t}_{T_i^1}(k)}_{\mathbf{h}_i^1(k)} - \alpha_i^2\underbrace{[\mathbf{R}_{T_i^2}(k)]^T\mathbf{t}_{T_i^2}(k)}_{\mathbf{h}_i^2(k)}\end{aligned}$$

for all $k\in[k_0,k_0+F]$ and for $i=1,2$. Note that the use of a single scaling factor is not enough to model the ambiguity in the determination of $\mathbf{t}_i(k)$. The vectors $\mathbf{h}_i^1(k)\in\mathbb{R}^3$ and $\mathbf{h}_i^2(k)\in\mathbb{R}^3$, introduced in the expression, are used in this section to make the notation easier to follow.

If the expression in (4.2) is separated into its rotational and translational parts, it takes the following form for unsynchronized video sequences:

$$\mathbf{R}_2(k')=\mathbf{R}\mathbf{R}_1(k)\mathbf{R}^T, \tag{4.3}$$

$$\alpha_2^1\mathbf{h}_2^1(k')-\alpha_2^2\mathbf{h}_2^2(k')=\mathbf{R}\left[\mathbf{h}_1^1(k)-\alpha_1^2\mathbf{h}_1^2(k)\right]+(\mathbf{I}_3-\mathbf{R}_2(k'))\mathbf{t} \tag{4.4}$$

for all $k\in[k_0,k_0+F]$, where $k'=k+\delta$ and δ is as defined in section 3.1. Note that α_1^1 was omitted in (4.4) as it is assumed to be the unit. This is without loss of generality since there is an ambiguity in the magnitude of the two members of (4.4).

If quaternions are used, (4.3) reduces to the form of (3.5) (see details in section 3.1), where $\mathbf{q}_1(k)$, $\mathbf{q}_2(k')$, and $\mathbf{q}$, are the unit quaternions associated, respectively, with the rotation matrices $\mathbf{R}_1(k)$, $\mathbf{R}_2(k')$, and $\mathbf{R}$, redefined in this section for the case of independently moving cameras.

The expression in (4.4) can also be written as a function of the quaternion $\mathbf{q}$, associated with the rotation $\mathbf{R}$ that relates the reference frames of the two cameras in the initial time instant. In this case, this expression takes the form

$$\alpha_2^1\mathbf{h}_2^1(k')-\alpha_2^2\mathbf{h}_2^2(k')=\mathbf{\Xi}^T(\mathbf{q})\mathbf{\Psi}(\mathbf{q})\left[\mathbf{h}_1^1(k)-\alpha_1^2\mathbf{h}_1^2(k)\right]+(\mathbf{I}_3-\mathbf{R}_2(k'))\mathbf{t},$$

where the matrices $\mathbf{\Xi}(\mathbf{q})$ and $\mathbf{\Psi}(\mathbf{q})$ are as defined in (3.6).

By combining the previous expression with the one relating the rotations perceived from both sequences, the synchronization problem for independently moving cameras can be cast

into the form of the minimization problem

$$\hat{\delta} = \arg\min_{\delta} \; E_m(\delta), \tag{4.5}$$

where $\hat{\delta}$ denotes the estimated temporal offset and $E_m(\delta)$ is the error function

$$E_m(\delta) = \min_{(\mathbf{q},\mathbf{t},\beta_2^1,\beta_2^2,\beta_1^2)} \mu_R E_R(\delta,\mathbf{q}) + \mu_T E_T(\delta,\mathbf{q},\mathbf{t},\beta_2^1,\beta_2^2,\beta_1^2) + \mu_q(\mathbf{q}^T\mathbf{q}-1)^2 \tag{4.6}$$

with μ_R, μ_T, and μ_q, positive weighting coefficients and

$$E_R(\delta,\mathbf{q}) = \sum_{k=k_0}^{k_0+F} ||M(\mathbf{q}_1(k),\mathbf{q}_2(k+\delta))\mathbf{q}||^2,$$

$$E_T(\delta,\mathbf{q},\mathbf{t},\beta_2^1,\beta_2^2,\beta_1^2) = \sum_{k=k_0}^{k_0+F} ||(\beta_2^1)^2\mathbf{h}_2^1(k+\delta) - (\beta_2^2)^2\mathbf{h}_2^2(k+\delta) - \boldsymbol{\Xi}^T(\mathbf{q})\boldsymbol{\Psi}(\mathbf{q})\left[\mathbf{h}_1^1(k) - (\beta_1^2)^2\mathbf{h}_1^2(k)\right] - (\mathbf{I}_3 - \mathbf{R}_2(k+\delta))\mathbf{t}||^2.$$

The scalars β_2^1, β_2^2, and β_1^2 are used in these expressions to guarantee that α_2^1, α_2^2, and α_1^2 (with $\alpha_2^1 = (\beta_2^1)^2$, $\alpha_2^2 = (\beta_2^2)^2$, and $\alpha_1^2 = (\beta_1^2)^2$) are not negative.

The optimization problem in (4.6) is a nonlinear least-squares problem due to the nonlinear dependence of $E_T(\delta,\mathbf{q},\mathbf{t},\beta_2^1,\beta_2^2,\beta_1^2)$ on $\mathbf{q}$, β_2^1, β_2^2, and β_1^2, and thus it can be solved using any nonlinear least-squares solver, such as the Levenberg–Marquardt method [17]. An initial guess for the unknowns $\mathbf{q}$, $\mathbf{t}$, β_2^1, β_2^2, and β_1^2 can be obtained by relaxing the problem, similarly to what was done in the end of section 3.1.

The temporal offset between the videos is the one that solves (4.5) and is found by evaluating the error function in (4.6) for all the possible offsets in a given range. Moreover, note that with the proposed strategy it is possible to estimate the relative scales between the two objects. This is possible only because two cameras are used. In the monocular multibody structure-from-motion problem, for instance, each reconstructed object has a different unknown scale, and thus objects are distorted with respect to each other; see [20].

In this work, the correspondence between the two videos, in terms of which trajectories belong to which objects, is assumed to be unknown. A set of features in one camera may correspond to either of the two sets in the other camera, and thus two combinations between the sets are possible (once an association is assumed, the other is implicitly defined). The correct combination can be found by solving the previous optimization problem for the two cases and choosing the one that leads to the minimum value for $E_m(\delta)$.

5. Results. In this section, proof of concept simulations and experiments with real data that illustrate the performance of the proposed synchronization methods are presented. The main goal of this section is to validate the theory that supports the methods that were presented, rather than providing an exhaustive evaluation of their performance, as the flavor of this work is more in the theoretical novelty than in the practical details and experimental assessment. The results obtained with the proposed methods are compared with the ones obtained with the strategy presented in [30]. This strategy was developed to synchronize videos

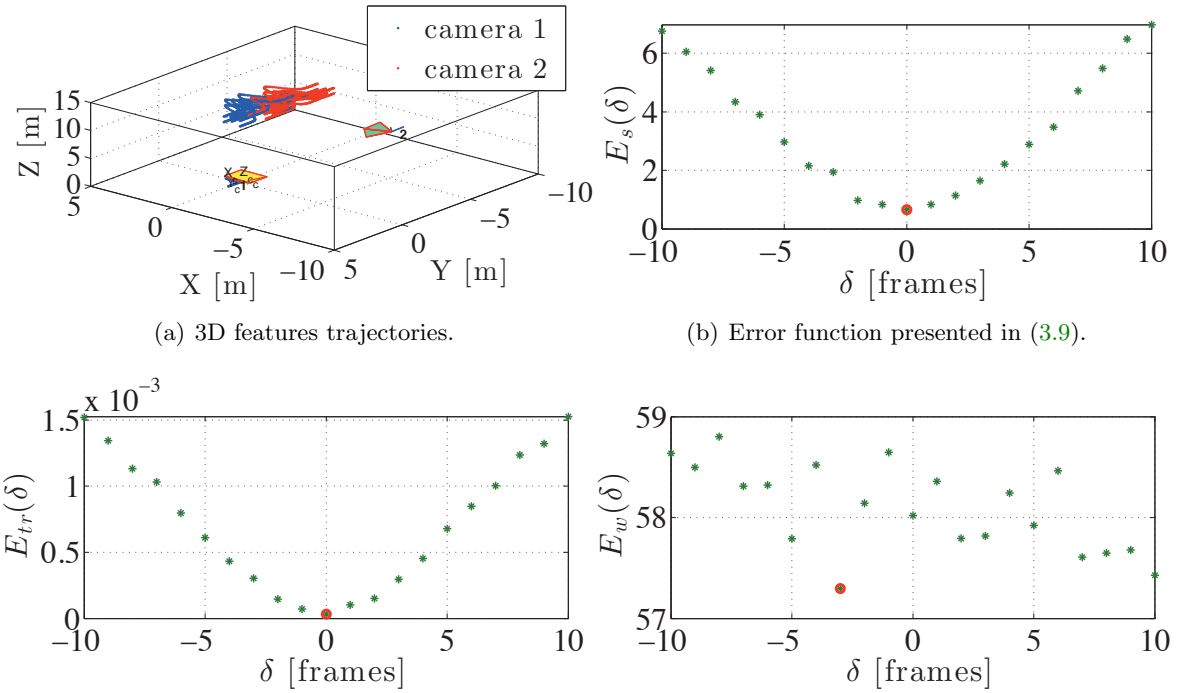
acquired with static or jointly moving cameras, when no correspondence between the features tracked in the two videos exists. It consists in using a heuristic to examine the effective rank of a matrix constructed from the measurements. The heuristic proposed in the paper and the suggested threshold were used in the implementation of this algorithm. The comparisons with [30] serve two purposes: (i) to understand how our algorithms for static or jointly moving cameras compare to a state-of-the-art approach, and (ii) to confirm that such approach cannot be used to synchronize videos acquired with cameras that move independently.

In the reported experiments, the temporal offset between the two videos is considered to be an integer in the range $[-10, 10]$ frames ($\Delta = 10$ frames). The estimates of this offset obtained according to a given strategy are found by minimizing the corresponding error function, which is done by evaluating it for all the possible offsets. For ease of presentation, the real offset between the two sequences is 0 frames in all situations, as the sequences are previously synchronized using information about the ground truth. This information is known in simulation and was obtained using a photo-flash to mark some of the frames, as suggested in [29], in the case of the experiments with real data.

5.1. Simulation results. The setup used for the simulations is the one described throughout this document, i.e., features on a moving object (in the case of static or jointly moving cameras) or features on a moving object and on a static background (in the case of independently moving cameras) are used to simulate the measurements provided to the algorithms. These measurements are generated using a pinhole model for the cameras, whose intrinsic parameters and spatial resolution were made equal to the ones of the cameras used to obtain the experimental results presented in section 5.2. The features tracked in the two cameras, either the ones on the moving object or the ones on the background, do not match. In the implementation of the proposed strategies, more emphasis was given to the cost function terms coming from the rotational motion of the objects, as these terms are usually more robust to noise; thus the parameters $\mu_T = 1$ and $\mu_R = 10$ were used. Moreover, μ_q was made equal to the number of frames of the video sequences, i.e., $\mu_q = F$, since it multiplies a single term, whereas both μ_T and μ_R are associated with a summation of several terms, each one corresponding to a different time instant. For the simulations presented in this section, sequences with $F = 120$ frames were considered and 10 object features were used. In the case of the simulations with independently moving cameras, 20 features belonging to the static background were tracked.

The trajectories used to assess the algorithms proposed for static or jointly moving cameras are presented in Figures 6(a), 7(a), and 8(a). The first experiment illustrates a situation where the object moves on a plane, the second shows the case of a nonconstrained motion, and in the third the motion of the object is purely translational. These trajectories illustrate the special cases described in section 3.3. Since the methods proposed in that section are particular for some types of trajectories, only the results obtained with the strategies that can be used (at least in theoretical terms) to synchronize the cameras in each case are presented.

The motion of the object in the experiment reported in Figure 6 includes rotational and translational components but is restricted to a plane. This situation occurs, for instance, when a car moving on the street is used to synchronize the cameras. In this case, both the general strategy proposed in section 3.1 and the one that only uses information coming from



(a) 3D features trajectories.

(b) Error function presented in (3.9).

(c) Error function presented in (3.10).

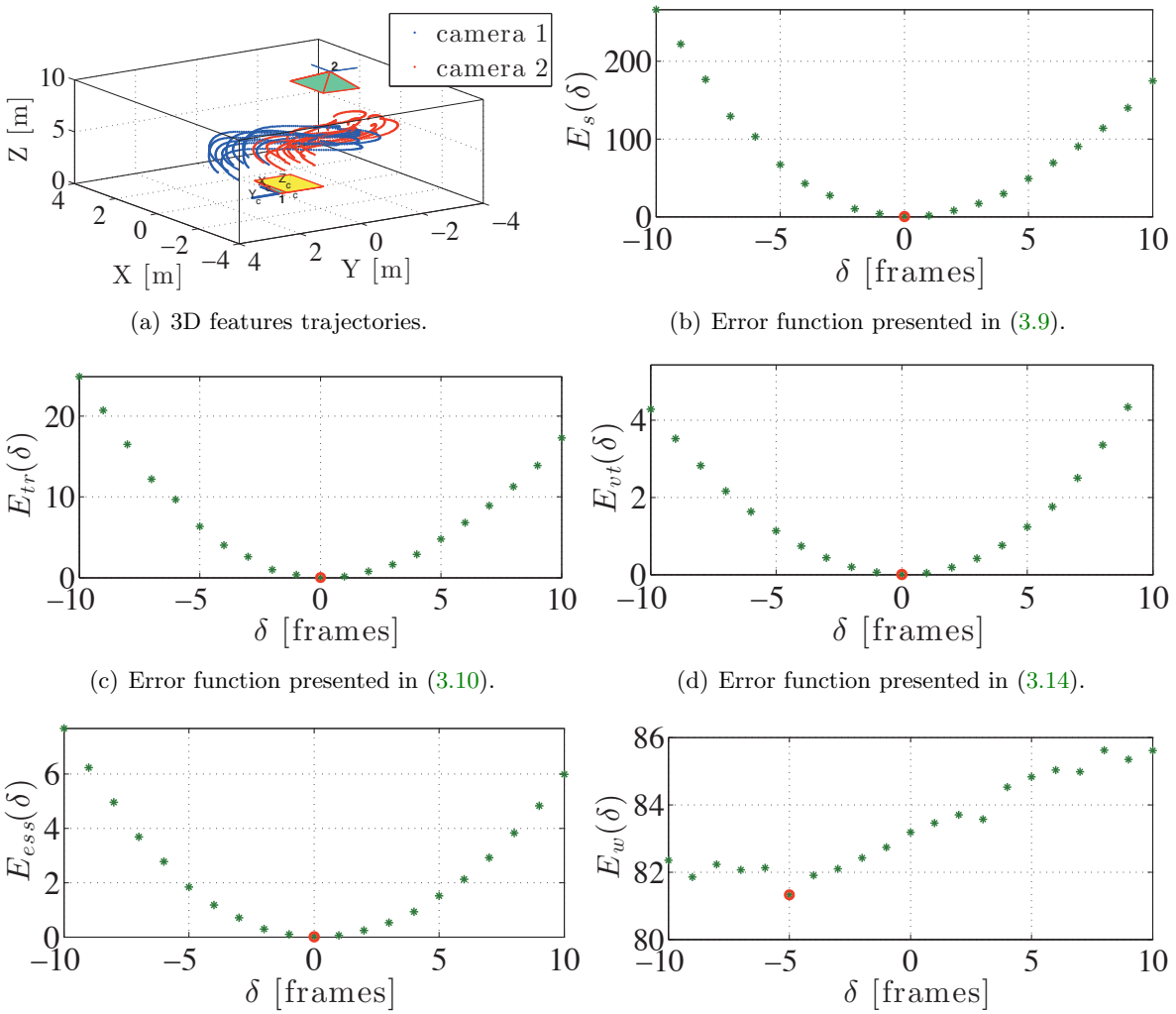
(d) Method presented in [30].

Figure 6. *Performance of the algorithms for static cameras in the case of an object that moves on a plane. In* (a)*, the 3D trajectories of the features tracked in camera* 1 *(blue dots) and camera* 2 *(red dots) are depicted. The two pyramids in the figure represent the position and attitude of the cameras. In* (b)*,* (c)*, and* (d)*, the values of the error functions* $E_s(\delta)$*,* $E_{tr}(\delta)$*, and* $E_w(\delta)$*, when the feature measurements are corrupted by zero-mean white Gaussian noise with standard deviation* $\sigma = 0.25$ *pixel, are presented. The circles in red identify the minima of the functions.*

the rotation matrices (see the error function in (3.10)) correctly identify the temporal offset $\delta = 0$. These results were obtained using feature measurements corrupted by zero-mean white Gaussian noise with standard deviation $\sigma = 0.25$ pixel. The method proposed by Wolf and Zomet in [30] fails to identify this offset. The main reason for this failure has to due with the use of an affine approximation for the cameras. Recall that all our strategies consider a projective model for the cameras. In Figure 6, and in the other figures cited in this section, $E_w(\delta)$ denotes the error function associated with the method proposed in [30].

The type of motion addressed in Figure 7 is very general, as the object rotates and translates freely throughout the 3D space (it is not restricted to a plane), with a motion that falls into the scope of the nonconstrained motion described in section 3.3. In this case, four of the five synchronization strategies proposed in this paper can be used. Only the one regarding objects that do not rotate, i.e., the one associated with the cost function in (3.17), is not appropriate. As can be seen, for the type of noise under consideration, the error functions $E_s(\delta)$, $E_{tr}(\delta)$, $E_{vt}(\delta)$, and $E_{ess}(\delta)$ reach their minimum at the correct temporal offset. This does not occur with the method proposed by Wolf and Zomet (see Figure 7(f)) due to the use of an affine model for the cameras.

The simulation presented in Figure 8 addresses the case of an object with a purely translational motion. There are two of the proposed synchronization strategies that are appropriate



(a) 3D features trajectories.

(b) Error function presented in (3.9).

(c) Error function presented in (3.10).

(d) Error function presented in (3.14).

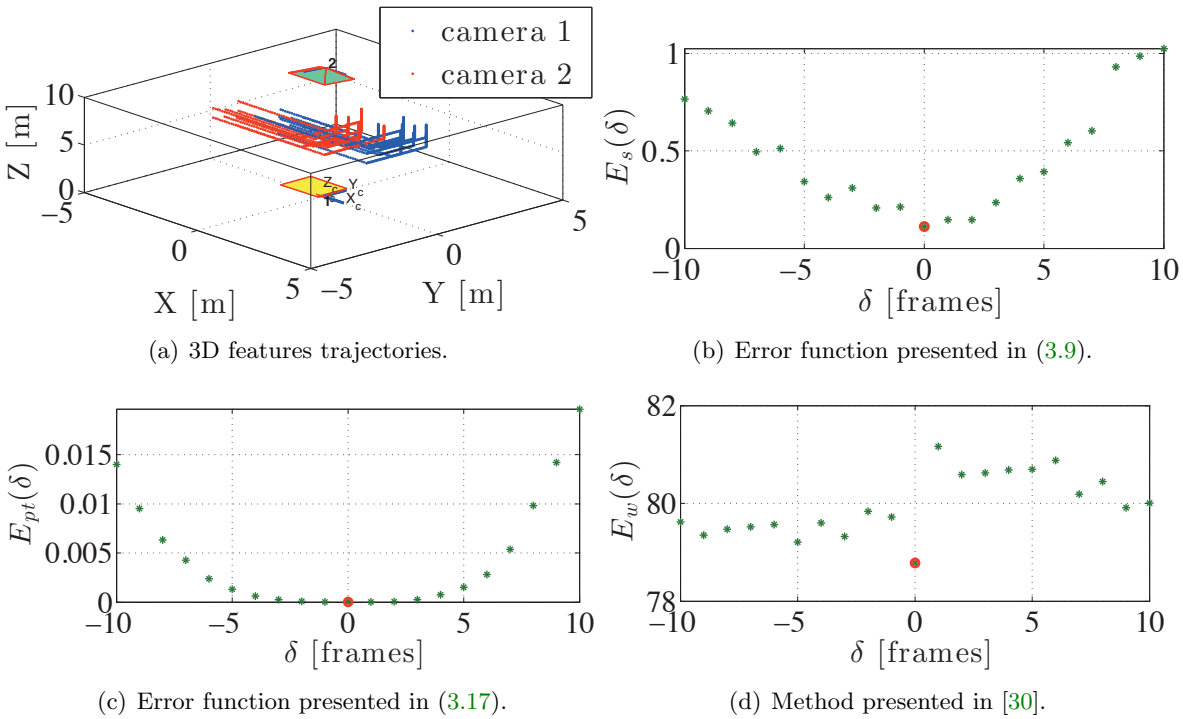
(e) Error function presented in (3.16).

(f) Method presented in [30].

Figure 7. *Performance of the algorithms for static cameras in the case of a nonconstrained motion. In* (a)*, the 3D trajectories of the features tracked in camera* 1 *(blue dots) and camera* 2 *(red dots) are depicted. The two pyramids in the figure represent the position and attitude of the cameras. In* (b)*,* (c)*,* (d)*,* (e)*, and* (f)*, the values of the error functions* $E_s(\delta)$*,* $E_{tr}(\delta)$*,* $E_{vt}(\delta)$*,* $E_{ess}(\delta)$*, and* $E_w(\delta)$*, when the feature measurements are corrupted by zero-mean white Gaussian noise with standard deviation* $\sigma = 0.25$ *pixel, are presented. The circles in red identify the minima of the functions.*

for this type of trajectory: the general approach proposed in section 3.1 and the refinement of such approach for the case of objects that do not rotate, i.e., the method associated with the error function in (3.17). Results obtained with the algorithm proposed in [30] are also presented. As can be seen from the figure, the three strategies successfully identify the temporal offset between the two sequences.

From the discussion above, it is possible to confirm that the method proposed in section 3.1 covers a wide range of situations and that it leads to a unique solution, in terms of the



(a) 3D features trajectories.

(b) Error function presented in (3.9).

(c) Error function presented in (3.17).

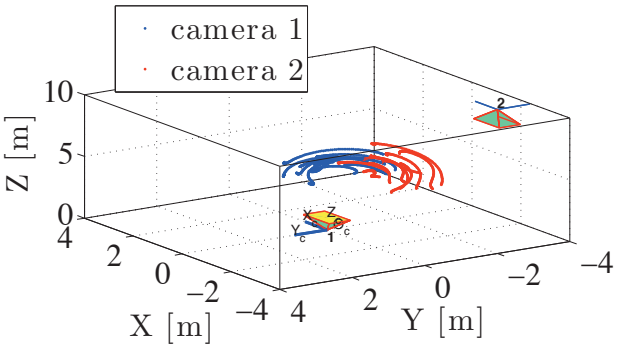
(d) Method presented in [30].

Figure 8. *Performance of the algorithms for static cameras in the case of an object that translates but does not rotate. In* (a)*, the 3D trajectories of the features tracked in camera* 1 *(blue dots) and camera* 2 *(red dots) are depicted. The two pyramids in the figure represent the position and attitude of the cameras. In* (b)*,* (c)*, and* (d)*, the values of the error functions* $E_s(\delta)$*,* $E_{pt}(\delta)$*, and* $E_w(\delta)$*, when the feature measurements are corrupted by zero-mean white Gaussian noise with standard deviation* $\sigma = 0.25$ *pixel, are presented. The circles in red identify the minima of the functions.*

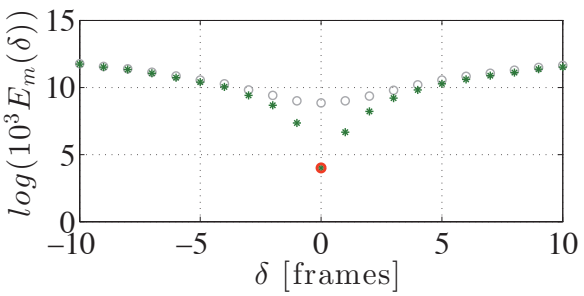
identification of the temporal offset between the two sequences, when the motion of the object includes enough information for the synchronization, as expected according to the conclusions drawn in section 3.2. Moreover, the results discussed above also confirm that the methods presented in section 3.3, for some particular types of trajectories, work properly when the type of motion of the object falls into the scope of their domain of applicability.

According to the figures, the error functions associated with the strategies proposed in this paper are typically smoother than the ones associated with the method proposed by Wolf and Zomet. This is probably a consequence of the fact that their method consists in examining the effective rank of a matrix, which is usually not a smooth function.

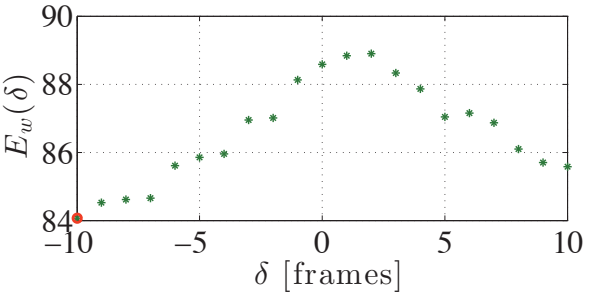
In order to make a first assessment of the algorithm proposed in section 4, two simulations with independently moving cameras are presented next: one in which the motion of the object with respect to the background includes both rotational and translational components, and another in which such motion is purely translational. In addition to the object features shown in Figures 9 and 10, features that simulate a static background are also used. They are not shown here to avoid that the figures become illegible. The motions of the simulated cameras were made similar to the ones obtained for the real cameras used in section 5.2, in order to mimic the natural movements of a person holding a camera. A logarithmic scale is used in the



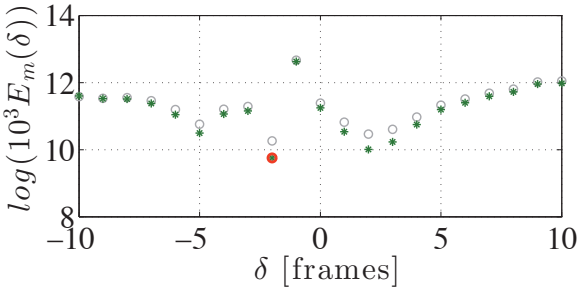
(a) 3D features trajectories.



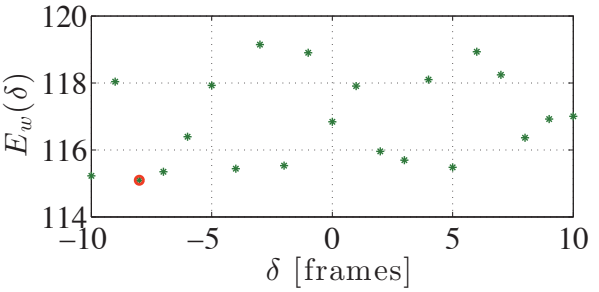
(b) Error function presented in (4.6): $\sigma = 0.25$ pixel.



(c) Method presented in [30]: $\sigma = 0.25$ pixel.



(d) Error function presented in (4.6): $\sigma = 1$ pixel.

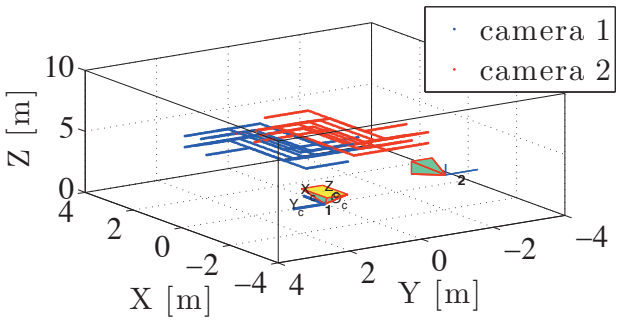


(e) Method presented in [30]: $\sigma = 1$ pixel.

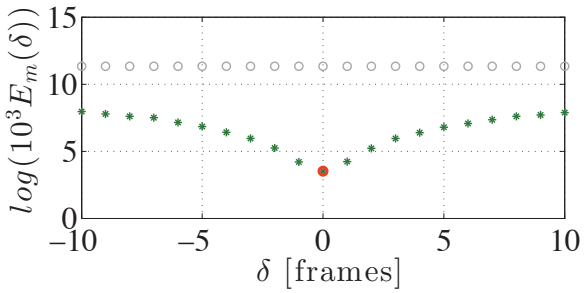
Figure 9. *Performance of the algorithms for independently moving cameras in the case of an object that rotates and translates with respect to the background. In* (a)*, the 3D trajectories of the features tracked in camera* 1 *(blue dots) and camera* 2 *(red dots) are depicted. The two pyramids in the figure represent the position and attitude of the cameras in the initial time instant. In* (b) *and* (d)*, the values of the error function* $E_m(\delta)$*, when the feature measurements are corrupted by zero-mean white Gaussian noise with the indicated standard deviations, are presented. The same is depicted in* (c) *and* (e) *for* $E_w(\delta)$*. In* (b) *and* (d)*, the green and gray curves result, respectively, from evaluating* $E_m(\delta)$ *when the correct and wrong combinations, between the sets of features associated with the moving object and with the static background, are considered. The circles in red identify the minima of the functions.*

figures that depict the values of $E_m(\delta)$, so that the two curves that result from considering the two possible associations between the two sets of features tracked in both sequences are legible.

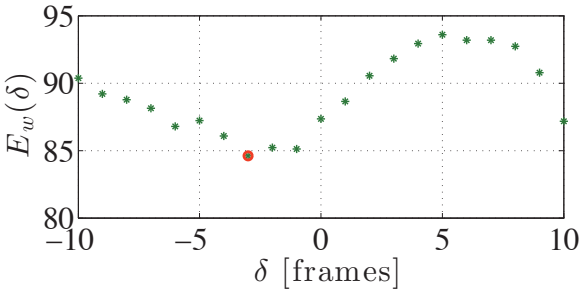
The purpose of Figures 9 and 10 is to illustrate a situation in which two people use handheld cameras to record different parts of a rigid object moving between them. When the



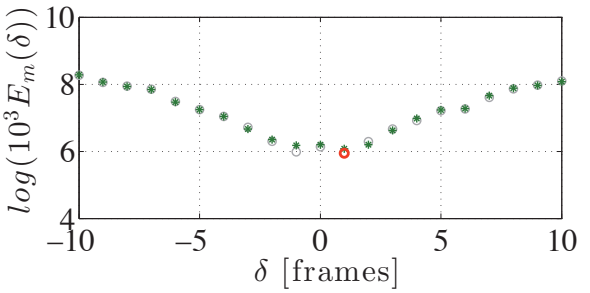
(a) 3D features trajectories.



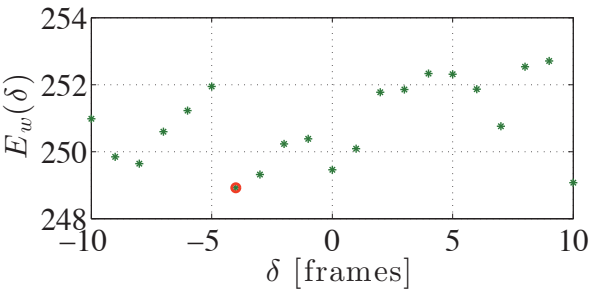
(b) Error function presented in (4.6): $\sigma = 0.25$ pixel.



(c) Method presented in [30]: $\sigma = 0.25$ pixel.



(d) Error function presented in (4.6): $\sigma = 1$ pixel.



(e) Method presented in [30]: $\sigma = 1$ pixel.

Figure 10. *Performance of the algorithms for independently moving cameras in the case of an object that translates but does not rotate with respect to the background. In* (a)*, the 3D trajectories of the features tracked in camera* 1 *(blue dots) and camera* 2 *(red dots) are depicted. The two pyramids in the figure represent the position and attitude of the cameras in the initial time instant. In* (b) *and* (d)*, the values of the error function* $E_m(\delta)$*, when the feature measurements are corrupted by zero-mean white Gaussian noise with the indicated standard deviations, are presented. The same is depicted in* (c) *and* (e) *for* $E_w(\delta)$*. In* (b) *and* (d)*, the green and gray curves result, respectively, from evaluating* $E_m(\delta)$ *when the correct and wrong combinations, between the sets of features associated with the moving object and with the static background, are considered. The circles in red identify the minima of the functions.*

standard deviation of the noise that corrupts the measurements is small ($\sigma = 0.25$ pixel), the strategy presented in section 4 succeeds in the identification of the correct temporal offset in both cases. For larger standard deviations, a degradation in the performance of the proposed method occurs. In particular, it fails to synchronize the sequences in both experiments for

$\sigma = 1$ pixel; see Figures 9(d) and 10(d).

The main reason for the degradation of the performance of the presented algorithm has to do with the accuracy of the estimates of the motions of the object and camera, as structure from motion strategies are very sensitive to noise. Another issue with an important influence on the accuracy of the synchronization is the type of motion of the object. Motions that are not rich enough (for instance, in terms of changes in the velocity or acceleration or in the direction of travel of the object) make the synchronization more difficult, as there is very little information that can be used.

The strategy presented in section 4 found the correct association, in terms of which set of features corresponds to a given set in the other sequence, in the two experiments, for small noise conditions ($\sigma = 0.25$ pixel), as the minimum of the two error functions is achieved for the correct association (green curve) in both experiments. For $\sigma = 1$ pixel, i.e., when the synchronization of the sequences is not successful, there is a situation in which the correct correspondence is found (see Figure 9(d)) and one in which it is not (see Figure 10(d)). Even though the correct association can be identified when the synchronization does not succeed, as happens in Figure 9(d), the failure in Figure 10(d) is understandable, since the synchronization process and the identification of the association between the two sets of features depend on each other.

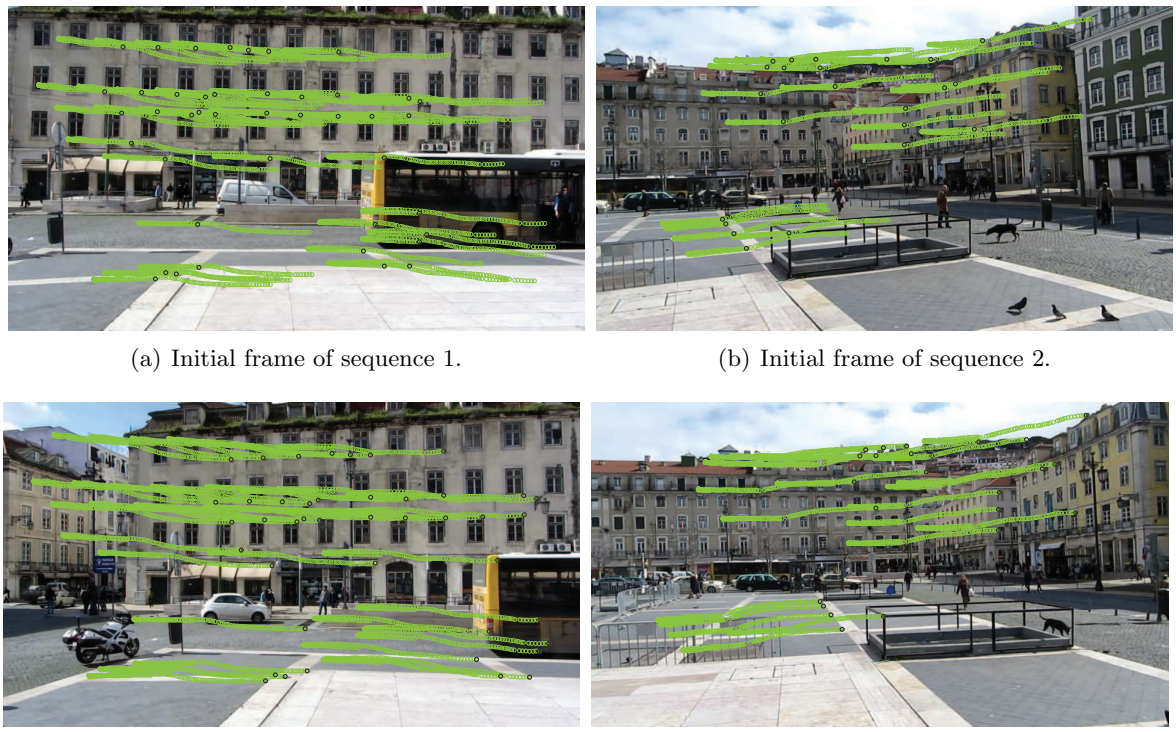
Finally, note that the method proposed in [30] fails to synchronize the sequences in both experiments, which was expected since this algorithm was developed for static or jointly moving cameras.

5.2. Experimental results. In order to show that the proposed methods can also be used to synchronize real videos, this section presents results obtained for videos acquired with cameras of regular mobile phones, working at 29 fps and providing images with a spatial resolution of 960×540 pixels. The intrinsic parameters of the cameras were calibrated a priori using the toolbox in [3]. When a priori calibration is not possible, self-calibration should be used; see [8] and references therein for details.

The features used for the synchronization were selected manually in the first frame of each sequence and tracked along the videos with the KLT feature tracker; see [22]. No strategy to deal with occlusions or outliers was implemented, as this is not the focus of this work; thus good features had to be selected to guarantee that the motion recovery algorithm works properly.

Two experiments are presented in this section. In the first, two cameras are mounted on the same rigidly moving platform, so that their fields of view do not intersect (they were facing opposite directions, similarly to the configuration in Figure 3(b)). The intercamera extrinsic parameters remain constant over time, and features on the static background are used. As explained in section 3, this problem is equivalent to having two static cameras recording an object moving between them. In the second experiment, a tram is recorded with two cameras held by two different people (i.e., with two independently moving cameras), and the static background is used as the second object.

The videos used in the first experiment have 121 frames and were obtained with two cameras moving jointly in the center of a public square. The first and final video frames are depicted in Figure 11, with the motion of the features (that results from the motion of



(a) Initial frame of sequence 1. (b) Initial frame of sequence 2.

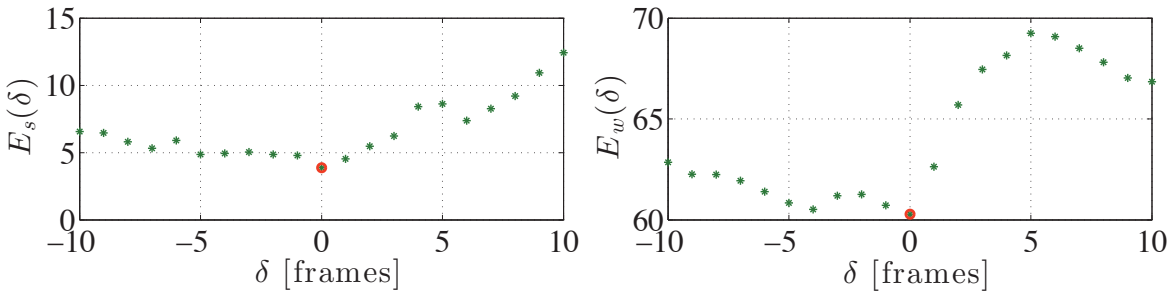
(c) Final frame of sequence 1. (d) Final frame of sequence 2.

Figure 11. *Initial and final frames of the two videos in the experiment with jointly moving cameras. Green dots represent the evolution over time of features on the background, and black dots identify their position at the time of acquisition of the presented frames.*

the platform where the cameras were installed) superimposed on them. No correspondence between the features tracked in the two sequences exists, as the fields of view of the cameras do not intersect.

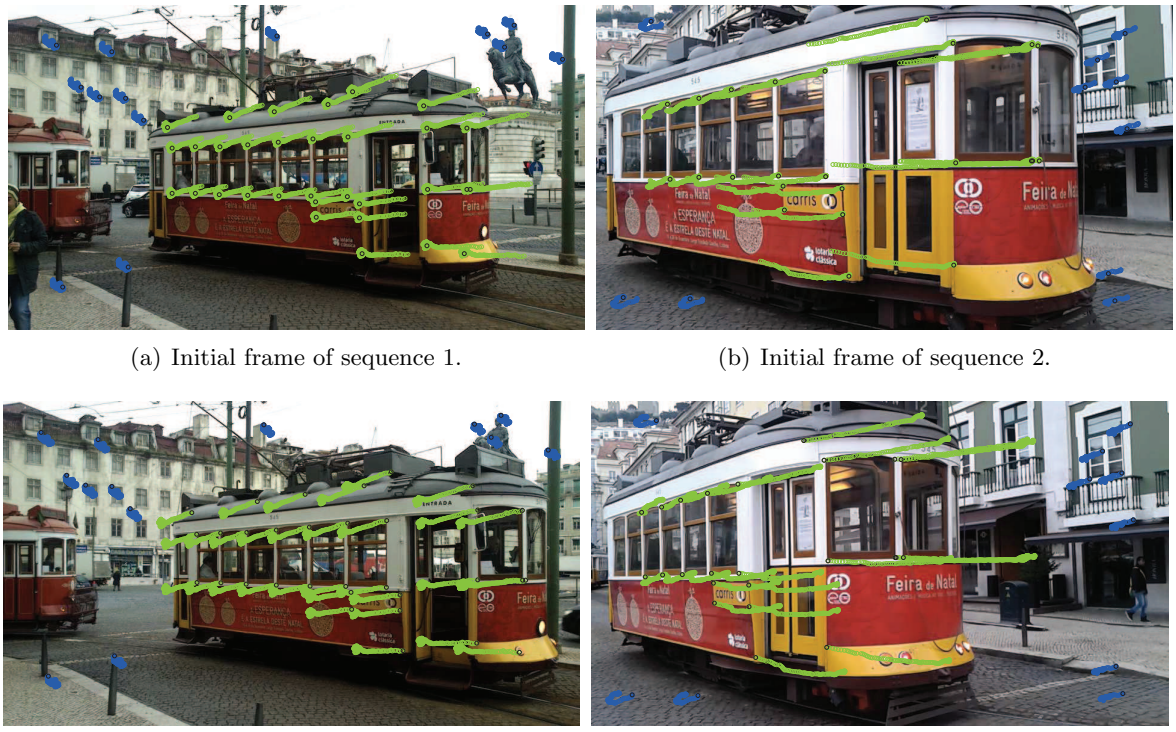
A comparison between the approaches presented in section 3.1 and in [30] can be found in Figure 12. In particular, the values of the error functions $E_s(\delta)$ and $E_w(\delta)$ associated with such strategies are presented for each one of the considered temporal offsets. Both methods succeed in the identification of the offset ($\delta = 0$) between the videos.

The sequences used in the experiment with independently moving cameras have 96 frames. As before, there is no time offset between them as they were previously synchronized using the ground truth. The first and final frames of the videos are depicted in Figure 13, with the time evolution of the features superimposed on them. The motion of the features in blue results from the motion of the users that were holding the cameras. No correspondence between the features tracked in the two sequences exists, as the cameras were in opposite sides of the tram (note the open/closed door or the differences on the background). The error functions $E_m(\delta)$ and $E_w(\delta)$, proposed in section 4 and in [30], are presented in Figure 14. Our method succeeds in the identification of the offset ($\delta = 0$) between the videos, whereas the one in [30] does not, which was expected since it was designed for static or jointly moving cameras.



(a) Our method (error function presented in (3.9)). (b) Method proposed in [30].

Figure 12. *Error functions for the experiment with jointly moving cameras (videos previously synchronized using the ground truth obtained by marking some of the frames with a photo-flash). The circles in red identify the minima of the functions.*

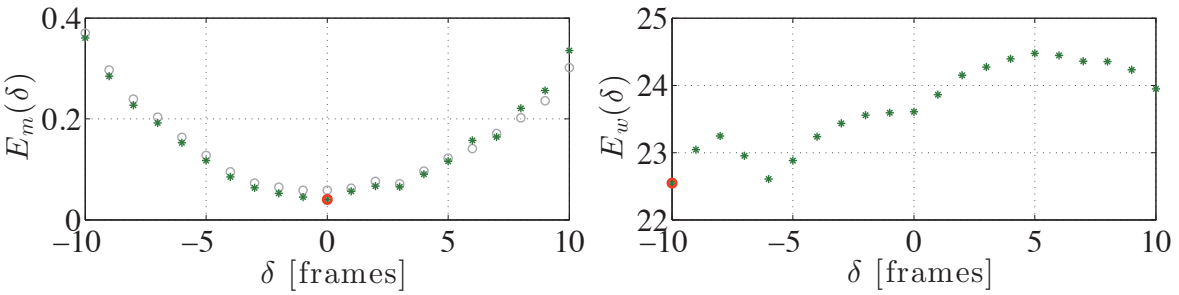


(a) Initial frame of sequence 1. (b) Initial frame of sequence 2.

(c) Final frame of sequence 1. (d) Final frame of sequence 2.

Figure 13. *Initial and final frames of the two sequences in the experiment with independently moving cameras. The evolution over time of features in the tram and features in the background are represented in green and blue, respectively. The black dots identify the position of the features at the time of acquisition of the presented frames.*

The two curves in Figure 14(a) correspond to the two possible associations between the sets of features identified in the two videos. The correct association (that corresponds to the green curve) is successfully identified, as it is the one that minimizes the minimum of the two error functions.



(a) Our method (error function presented in (4.6)).

(b) Method proposed in [30].

Figure 14. *Error functions for the experiment with independently moving cameras (videos previously synchronized using the ground truth obtained by marking some of the frames with a photo-flash). The green and gray curves in* (a) *result, respectively, from evaluating $E_m(\delta)$ when the correct and wrong combinations, between the sets of features associated with the moving object and with the static background, are considered. The circles in red identify the minima of the functions.*

6. Conclusions. In this work, the video synchronization problem that arises when there is no correspondence between the features that are visible in the different video sequences was addressed. In particular, two typical scenarios, one in which the cameras move independently and another in which they are static or move jointly, were studied. The method that was proposed for the case of independently moving cameras is particularly relevant, as it is the first to solve the synchronization problem while simultaneously allowing that the cameras move independently and that the scene does not include features that are visible in all the videos. The solution that we proposed consists in using the relative motion between two objects that move independently as a clue for the synchronization. A similar strategy was used for the case of static or jointly moving cameras, which led to a new set of synchronization methods for this type of setup. All the solutions that were presented were tested and validated with simulated and/or real data, and the most general strategy proposed for the synchronization of static or jointly moving cameras was shown to perform similarly or to outperform a state-of-the-art approach.

Appendix A. Proof of Lemma 3.1. According to (3.2) and (3.3), it is possible to conclude that the solution of (3.8) is unique in terms of the temporal offset (meaning that $E_s(\delta)$ is null only for the correct offset) if and only if there are not a nonnegative constant α_2', a constant unit quaternion $\mathbf{q}'$ (i.e., a constant rotation $\mathbf{R}' = \mathbf{\Xi}^T(\mathbf{q}')\mathbf{\Psi}(\mathbf{q}')$), and a constant translation $\mathbf{t}'$ that verify

$$\mathbf{R}_2(k'+\overline{\delta}) = \mathbf{R}'\mathbf{R}_1(k)\mathbf{R}'^T, \tag{A.1}$$

$$\alpha_2'\mathbf{t}_2(k'+\overline{\delta}) = \mathbf{R}'\mathbf{t}_1(k) + (\mathbf{I}_3 - \mathbf{R}_2(k'+\overline{\delta}))\mathbf{t}' \tag{A.2}$$

for some nonnull temporal offset $\overline{\delta}$ such that $|\overline{\delta}| \leq \Delta$, and for all $k \in [k_0,\ k_0+F]$. If $\mathbf{R}_2(k'+\overline{\delta})$ and $\mathbf{t}_2(k'+\overline{\delta})$ are written as a function of $\mathbf{R}_1(k+\overline{\delta})$ and $\mathbf{t}_1(k+\overline{\delta})$, the previous expressions

take the form

$$\mathbf{R}_1(k+\overline{\delta}) = \overline{\mathbf{R}}\mathbf{R}_1(k)\overline{\mathbf{R}}^T, \tag{A.3}$$

$$\overline{\alpha}_2\mathbf{t}_1(k+\overline{\delta}) = \overline{\mathbf{R}}\mathbf{t}_1(k) + (\mathbf{I}_3 - \mathbf{R}_1(k+\overline{\delta}))\overline{\mathbf{t}} \tag{A.4}$$

for all $k \in [k_0, k_0+F]$, where $\overline{\alpha}_2 = \alpha_2'/\alpha_2 \geq 0$, $\overline{\mathbf{R}} = \mathbf{R}^T\mathbf{R}'$, and $\overline{\mathbf{t}} = \mathbf{R}^T(\mathbf{t}' - \overline{\alpha}_2\mathbf{t})$. If the exponential map is used to represent $\mathbf{R}_1(k)$ and $\mathbf{R}_1(k+\overline{\delta})$, (A.3) can be written as

$$\theta_1(k+\overline{\delta})\mathbf{v}_1(k+\overline{\delta}) = \overline{\mathbf{R}}\theta_1(k)\mathbf{v}_1(k) \quad \text{for all } k \in [k_0, k_0+F], \tag{A.5}$$

where $\theta_1(k)$ and $\mathbf{v}_1(k)$ are, respectively, the nonnegative angle of rotation and the corresponding unit Euler axis associated with $\mathbf{R}_1(k)$; see [19]. Thus, checking if the solution of the optimization problem in (3.8) is unique is equivalent to checking if there exists a nonnull $\overline{\delta}$, verifying $|\overline{\delta}| \leq \Delta$, such that (A.4) and (A.5) are valid for some $\overline{\alpha}_2$, $\overline{\mathbf{R}}$, and $\overline{\mathbf{t}}$.

Since $||\mathbf{v}_1(k+\overline{\delta})|| = ||\overline{\mathbf{R}}\mathbf{v}_1(k)|| = 1$, recall that $\mathbf{v}_1(k+\overline{\delta})$ and $\mathbf{v}_1(k)$ are unit Euler axes; then, from (A.5), we have that

$$\theta_1(k+\overline{\delta}) = \theta_1(k) \tag{A.6}$$

must be verified for all $k \in [k_0, k_0+F]$. Thus, there is not a rotation matrix $\overline{\mathbf{R}}$ that solves (A.5) for some nonnull $\overline{\delta}$ such that $|\overline{\delta}| \leq \Delta$, unless $\theta_1(k)$ is periodic with period $|\overline{\delta}|$ in the interval $[k_0+\overline{\delta}_1, k_0+F+\overline{\delta}_2]$.

A particular case of periodic functions is the constant null function. If $\theta_1(k) = 0$, i.e., if $\mathbf{R}_1(k) = \mathbf{I}_3$, for all $k \in [k_0, k_0+F]$, there exists a nonnull $\overline{\delta}$, with $|\overline{\delta}| \leq \Delta$, such that (A.4) and (A.5) are verified for some $\overline{\alpha}_2 \geq 0$, $\overline{\mathbf{R}}$, and $\overline{\mathbf{t}}$, if and only if there exists a $\overline{\delta} \neq 0$, with $|\overline{\delta}| \leq \Delta$, such that $\theta_1(k) = 0$, for all $k \in [k_0+\overline{\delta}_1, k_0+F+\overline{\delta}_2]$, and such that

$$\overline{\alpha}_2\,\mathbf{t}_1(k+\overline{\delta}) = \overline{\mathbf{R}}\,\mathbf{t}_1(k) \quad \text{for all } k \in [k_0, k_0+F]$$

is verified for some constant scalar $\overline{\alpha}_2 \geq 0$ and for some constant rotation matrix $\overline{\mathbf{R}}$.

When $\theta_1(k)$ is periodic with period $|\overline{\delta}|$, but not null, in the interval $[k_0+\overline{\delta}_1, k_0+F+\overline{\delta}_2]$, two situations are possible: either the direction of $\theta_1(k)\mathbf{v}_1(k)$ is constant in that interval or it is not. In the first case, the direction of $\theta_1(k)\mathbf{v}_1(k)$ defines the (constant) Euler axis associated with $\mathbf{R}_1(k)$. Together with (A.6), this implies that

$$\mathbf{R}_1(k+\overline{\delta}) = \mathbf{R}_1(k)$$

for all $k \in [k_0, k_0+F]$. Using this equality and (A.4), it is straightforward to show that, in this case,

$$\overline{\alpha}_2\,[\mathbf{t}_1(k+2\overline{\delta}) - \mathbf{t}_1(k+\overline{\delta})] = \overline{\mathbf{R}}\,[\mathbf{t}_1(k+\overline{\delta}) - \mathbf{t}_1(k)] \tag{A.7}$$

for all $k \in [k_0-\overline{\delta}_1, k_0+F-\overline{\delta}_2]$. Thus, in this situation, there exists a $\overline{\delta} \neq 0$, with $|\overline{\delta}| \leq \Delta$, such that (A.4) and (A.5) are verified for some $\overline{\alpha}_2 \geq 0$, $\overline{\mathbf{R}}$, and $\overline{\mathbf{t}}$, if and only if there exists a constant scalar $\overline{\alpha}_2 \geq 0$ and a constant rotation $\overline{\mathbf{R}}$ such that (A.5) and (A.7) are verified, where $|\overline{\delta}|$ denotes the period of $\theta_1(k)$.

Consider now the case when $\theta_1(k)$ is periodic, but not null, in the interval $[k_0 + \overline{\delta}_1, k_0 + F + \overline{\delta}_2]$, and the direction of $\theta_1(k)\mathbf{v}_1(k)$ is not constant in the same interval. In this situation, if a rotation matrix $\overline{\mathbf{R}}$ that solves (A.4) and (A.5), for some $\overline{\delta}$, $\overline{\alpha}_2 \geq 0$, and $\overline{\mathbf{t}}$, exists, it is uniquely defined by the directions of the vector $\theta_1(k)\mathbf{v}_1(k)$ over time; see [21]. Thus, in this case, there exists a $\overline{\delta} \neq 0$, with $|\overline{\delta}| \leq \Delta$, such that (A.4) and (A.5) are verified for some $\overline{\alpha}_2 \geq 0$, $\overline{\mathbf{R}}$, and $\overline{\mathbf{t}}$ if and only if there exists a nonnegative constant scalar $\overline{\alpha}_2$ and a constant vector $\overline{\mathbf{t}}$ that verify (A.4), for all $k \in [k_0, k_0 + F]$, when the $\overline{\mathbf{R}}$ defined by the directions of $\theta_1(k)\mathbf{v}_1(k)$, according to (A.5), is considered. Here, $|\overline{\delta}|$ corresponds to the period of $\theta_1(k)$.

If $\mathbf{R}_1(k)$ and $\mathbf{t}_1(k)$, in (A.1) and (A.2), are written as a function of $\mathbf{R}_2(k')$ and $\mathbf{t}_2(k')$, a similar deduction can be done for the rotations and translations obtained for the second camera. Thus, the conditions derived above are valid for both cameras, i.e., for $i = 1, 2$, and can be summarized as in the statement of this lemma.

REFERENCES

[1] K. Arun, T. Huang, and S. Blostein, *Least-squares fitting of two 3-d point sets*, IEEE Trans. Pattern Anal. Machine Intelligence, 9 (1987), pp. 698–700.

[2] D. S. Bernstein, *Matrix Mathematics: Theory, Facts, and Formulas*, Princeton University Press, Princeton, NJ, 2009.

[3] J. Bouguet, *Camera Calibration Toolbox for Matlab*, www.vision.caltech.edu/bouguetj/calib{_}doc/.

[4] S. Boyd and L. Vandenberghe, *Convex Optimization*, Cambridge University Press, New York, 2004.

[5] Y. Caspi and M. Irani, *Spatio-temporal alignment of sequences*, IEEE Trans. Pattern Anal. Machine Intelligence, 24 (2002), pp. 1409–1424.

[6] J. Crassidis, F. Markley, and Y. Cheng, *Survey of nonlinear attitude estimation methods*, J. Guidance Control Dynamics, 30 (2007), pp. 12–28.

[7] E. Elhamifar and R. Vidal, *Sparse subspace clustering*, in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2009, pp. 2790–2797.

[8] O. Faugeras, Q.-T. Luong, and S. J. Maybank, *Camera self-calibration: Theory and experiments*, in European Conference on Computer Vision, Springer, 1992, pp. 321–334.

[9] T. Gaspar, P. Oliveira, and P. Favaro, *Synchronization of two independently moving cameras without feature correspondences*, in Proceedings of the European Conference on Computer Vision, Zurich, Switzerland, 2014, pp. 189–204.

[10] G. H. Golub and C. F. V. Loan, *Matrix Computations*, vol. 1, 3rd ed., Johns Hopkins University press, Baltimore, 1996.

[11] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*, 2nd ed., Cambridge University Press, New York, 2004.

[12] R. Horaud and F. Dornaika, *Hand-eye calibration*, Internat. J. Robotics Research, 14 (1995), pp. 195–210.

[13] T. Kanade and D. Morris, *Factorization methods for structure from motion*, Philos. Trans. R. Soc. Lond. Ser. A Math. Phys. Eng. Sci., 356 (1998), pp. 1153–1173.

[14] G. Liu, Z. Lin, S. Yan, J. Sun, Y. Yu, and Y. Ma, *Robust recovery of subspace structures by low-rank representation*, IEEE Trans. Pattern Anal. Machine Intelligence, 35 (2013), pp. 171–184.

[15] M. I. A. Lourakis and A. A. Argyros, *SBA: A software package for generic sparse bundle adjustment*, ACM Trans. Math. Software, 36 (2009), p. 2.

[16] F. L. Markley, *Attitude determination using vector observations and the singular value decomposition*, J. Astronaut. Sci., 36 (1988), pp. 245–258.

[17] D. Marquardt, *An algorithm for least-squares estimation of nonlinear parameters*, J. SIAM, 11 (1963), pp. 431–441.

[18] B. Meyer, T. Stich, M. Magnor, and M. Pollefeys, *Subframe temporal alignment of non-stationary cameras*, in Proceedings of the British Machine Vision Conference, 2008.

[19] R. M. Murray, Z. Li, and S. S. Sastry, *A Mathematical Introduction to Robotic Manipulation*, CRC Press, Boca Raton, FL, 1994.
[20] K. E. Ozden, K. Schindler, and L. V. Gool, *Multibody structure-from-motion in practice*, IEEE Trans. Pattern Anal. Machine Intelligence, 32 (2010), pp. 1134–1141.
[21] P. Schönemann, *A generalized solution of the orthogonal procrustes problem*, Psychometrika, 31 (1966), pp. 1–10.
[22] J. Shi and C. Tomasi, *Good features to track*, in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 1994, pp. 593–600.
[23] S. N. Sinha and M. Pollefeys, *Synchronization and calibration of camera networks from silhouettes*, in Proceedings of the IEEE International Conference on Pattern Recognition, vol. 1, 2004, pp. 116–119.
[24] H. Stewénius, C. Engels, and D. Nistér, *Recent developments on direct relative orientation*, J. Photogrammetry Remote Sensing, 60 (2006), pp. 284–294.
[25] P. Sturm and B. Triggs, *A factorization based algorithm for multi-image projective structure and motion*, in Proceedings of the European Conference on Computer Vision, 1996, pp. 709–720.
[26] C. Tomasi and T. Kanade, *Shape and motion from image streams under orthography: A factorization method*, Int. J. Comput. Vis., 9 (1992), pp. 137–154.
[27] P. Tresadern and I. Reid, *Synchronizing image sequences of non-rigid objects*, in Proceedings of the British Machine Vision Conference, 2003, pp. 629–638.
[28] B. Triggs, P. F. McLauchlan, R. I. Hartley, and A. W. Fitzgibbon, *Bundle adjustment—a modern synthesis*, in Vision Algorithms: Theory and Practice, Springer, New York, 2000, pp. 298–372.
[29] T. Tuytelaars and L. Van Gool, *Synchronizing video sequences*, in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, vol. 1, 2004, pp. 762–768.
[30] L. Wolf and A. Zomet, *Correspondence-free synchronization and reconstruction in a non-rigid scene*, in Proceedings of the Workshop on Vision and Modelling of Dynamic Scenes, 2002.
[31] J. Yan and M. Pollefeys, *Video synchronization via space-time interest point distribution*, in Proceedings of the Advanced Concepts for Intelligent Vision Systems, 2004.
[32] A. Yilma and M. Shah, *Recognizing human actions in videos acquired by uncalibrated moving cameras*, in Proceedings of the IEEE International Conference on Computer Vision, vol. 1, 2005, pp. 150–157.
[33] Z. Zhang, *A flexible new technique for camera calibration*, IEEE Trans. Pattern Anal. Machine Intelligence, 22 (2000), pp. 1330–1334.
[34] C. Zhou and H. Tao, *Dynamic depth recovery from unsynchronized video streams*, in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, vol. 2, 2003, pp. 351–358.