Line Feature based Multiple Video Synchronization

Cheng Lei and Yee-Hong Yang

Abstract— **In this paper, we present a novel method for synchronizing multiple (more than 2) un-calibrated video sequences recording the same event by free-moving full-perspective cameras. Unlike previous point feature based synchronization methods, our method takes advantage of line features for their better performance in measuring geometric alignment when video frames are synchronized. In particular, the tri-ocular geometric constraint of line features, which is evaluated by tri-focal transfer, is enforced when building the timeline maps for sequences to be synchronized. A hierarchical approach is used to reduce the computational complexity. To achieve sub-frame synchronization accuracy, a Levenberg-Marquardt method based optimization is performed to refine the synchronization. The experimental results on several synthetic and real video datasets demonstrate the effectiveness and robustness of our method over previous methods.**

Index Terms— **geometric alignment, trifocal tensor, sub-frame synchronization, video synchronization**

1. INTRODUCTION

As the cost of cameras is reduced, the use of multiple cameras has become popular. Unlike using a single camera, multiple views of a physical event can provide more information of the event. As a result, many novel applications have been developed using multiple view videos, e.g. blind-spot-free video surveillance [22], video based metrology/forensics [11], multiple video based rendering including the free-viewpoint video [2, 18, 20], video mosaic/panorama, image-based rendering [7] and so on. A fundamental assumption for these applications to extract accurate semantic, geometric or graphic information of the captured scene is that all the video sequences must be synchronized. Although using hardware to synchronize cameras may be feasible in some applications, it is not a practical solution for applications where the cameras are physically separated or are mobile. To synchronize videos manually is tedious and error prone, especially when there are more than two sequences. In this paper, a new line feature based method is proposed that can synchronize more than 2 video sequences with minimum manual efforts.

Cheng LEI is with the Department of Computing Science, University of Alberta, Edmonton, AB, T6G 2E8, CANADA (e-mail: clei@cs.ualberta.ca).

Yee-Hong YANG, is with the Department of Computing Science, University of Alberta, Edmonton, AB, T6G 2E8, CANADA (phone: 780-492-3059; fax: 780-492-1071; e-mail: yang@cs.ualberta.ca).

The organization of this paper is as follows. The next section gives an overview of previous works. In section 3, our new line feature based synchronization method is introduced. In particular, an overview of our methodology is given first and then some implementation details are elaborated. Next in section 4, we show the experimental results on various synthetic and real video sequences, and the comparison with two previous methods. Finally, we conclude the paper in section 5.

2. BACKGROUND

2.1 Problem Formulation

Suppose that we are given $m\,(m \geq 2)$ video sequences $\mathcal{S}^{(k)}(k \in [1,m])$ captured by multiple stationary or free-moving affine or full-perspective cameras. The sequence length (i.e. frame number) of the *k*-th sequence is denoted by $N^{(k)}(k \in [1,m])$ and the frame-rate by $Fps^{(k)}(k \in [1,m])$. Each sequence $\mathcal{S}^{(k)}$ that defines a local timeline $\mathcal{T}^{(k)}$ consists of discrete samples, each of which corresponds to a captured frame $\mathcal{I}_{t_k}^{(k)}(t_k \in \mathcal{T}^{(k)})$. W.l.o.g., taking $\mathcal{S}^{(1)}$ as the reference sequence, the synchronization problem can be formulated as: given a timeline sample $\mathcal{I}_{t_1}^{(1)}(t_1 \in \mathcal{T}^{(1)})$ in $\mathcal{S}^{(1)}$, find in the other sequence(s) $\mathcal{S}^{(k)}$ the corresponding timeline sample $\mathcal{I}_{t_k}^{(k)}(t_k \in \mathcal{T}^{(k)})$ that is captured at the exact same instant. Namely, a one-to-one timeline map $\mathcal{M}_{\mathcal{T}^{(1)} \to \mathcal{T}^{(k)}}$ from $\mathcal{S}^{(1)}$ to $\mathcal{S}^{(k)}$ should be established for synchronization.

Based on the specific context, various forms of the timeline map $\mathcal{M}$ can be used. The simplest one is the "offset-only" form given as $\mathcal{M}_{\mathcal{T}^{(1)} \to \mathcal{T}^{(k)}} : t_1 + \Delta_k = t_k$ when $Fps^{(1)} = Fps^{(k)}$. The more general "1D-Affine" form defined as:

$$\mathcal{M}_{\mathcal{T}^{(1)} \to \mathcal{T}^{(k)}} : \frac{Fps^{(k)}}{Fps^{(1)}} t_1 + \Delta_k = \alpha^{(k)} t_1 + \Delta_k = t_k \qquad (1)$$

can be used when $Fps^{(1)} \neq Fps^{(k)}$. However, if the frame rates are constant and known, then only Δ_k is required. In some special cases, the dynamic timeline map has to be defined, e.g. in [10].

Given a specific group of timeline maps $\{\mathcal{M}_k\}$, the "supporting voters," defined as, the *m*-frame tuples $\mathbf{\Omega}_{\{\mathcal{M}_k\}}$ that satisfy such mapping relations, must be in

synchrony if and only if the sequences are synchronized through $\{\mathcal{M}_k\}$.

2.2 Previous Works

Based on the inter-sequence temporal correlation constraint used, current software-based sequence synchronization methods can be roughly categorized as: feature (trajectory) based, intensity based and camera movement based.

Since the first paper [13] in investigating the sequence synchronization problem was published, most synchronization methods proposed up to now fall into the feature based category [1, 5, 9, 10, 13, 14, 15, 17, 19]. Feature based synchronization methods usually require tracking features in sequences and optionally matching features across sequences. In the literatures, the most used feature is the point. The tracked point features in each sequence can be treated locally as single points or globally as trajectory curves.

Such feature based methods are usually based on the fact that, for exactly synchronized frames, the corresponding dynamic 3D scene features can be regarded as a stationary rigid configuration at the corresponding time instant. Therefore, some form of multiple view geometric alignment constraint(s) must hold for their 2D projections in the synchronized frames. By finding the timeline maps $\{\mathcal{M}_k\}$ that best satisfy such constraint(s) for all supporting frame-tuples $\in \mathbf{\Omega}_{\{\mathcal{M}_k\}}$, the sequence can be synchronized. Commonly exploited geometric alignment constraints include the binocular epipolar geometry constraint [5, 9, 10], the plane-induced homography [5, 13], rankness properties arose from the special projection model [14, 15] or feature movements [19] and so on.

The intensity based synchronization methods try to minimize the sum of squared differences (SSD) between the sequences that can be spatially and temporally warped through a parametric model. As in the representative work [3], usually the homography based spatial transform and the 1D affine temporal transform are used. All the pixels can provide constraints to such a model, while not just the limited number of salient features so that feature tracking and matching can be avoided. Moreover, the temporal and spatial information in the sequences can be utilized in the unified framework for the simultaneous spatial and temporal alignments.

As an example of camera movement based synchronization methods, in [4], the non-overlapping sequences can be aligned spatially and temporally under the assumption that the cameras are fixed rigidly (sharing a common optical center so that they are related by a homography) and move together. The synchronization is done by using the homography induced constraint between the frame-to-frame transformations across the sequences.

2.3 Recent Progress

All of the above methods can not handle the general cases of free-moving cameras and multiple (>2) sequences, for which some effort has been made recently. In [17], a 5-point method is proposed for synchronizing 2 video sequences captured by affine moving cameras in 3D instead of in 2D by evaluating the line-to-line distance of the back-projection lines of the matching points as the geometric alignment measure. While in [1], with fixed inter-camera epipolar geometry recovered using stationary feature points, for each moving feature point detected in the reference sequence, if the corresponding epipolar line intersects with any tracked feature trajectory across consecutive frames in the other sequences, a tentative timeline map voter is formed. With enough voters collected, a RANSAC procedure is applied to synchronize more than 2 sequences with the robust feature matching done implicitly.

As discussed earlier, almost all current video synchronization methods have some limitations on the captured scene or on the cameras. No general framework has been developed for synchronizing sequences captured by multiple free-moving full-perspective cameras. Though it is more intuitive to regard synchronization as a pairwise temporal matching problem, novel methods to explore constraints unique to the context of synchronizing more than 2 sequences are also needed. Furthermore, in all of the previous feature-based synchronization methods, point features are explored extensively. Not much effort has been made to utilize other types of features such as line features, which are common in man-made environment and allow more precise and robust tracking than points. Therefore, the main goal of this paper is to present a new general multiple sequence synchronization framework using line features, by which uncalibrated video sequences captured by multiple free-moving full-perspective cameras can be synchronized.

3. LINE FEATURE BASED MULTIPLE VIDEO SYNCHRONIZATION

3.1 Overview

Suppose we want to synchronize three or more sequences $\mathcal{S}^{(k)}(k \in [1,m], m > 2)$, whose frame-rates $Fps^{(k)}(k \in [1,m])$ might be different but should be constant and known in advance. Therefore only the synchronization offsets $\Delta_k(k \in (1,m])$ in timeline map (1) need to be recovered for synchronization.

Our new video synchronization method belongs to the feature based category. Therefore we also reformulate the video synchronization problem as a geometric alignment problem for the set of matching features among multiple video sequences. Namely, for each specific frame-tuple, how well the frames are synchronized is evaluated as a geometric alignment measure of the matched features.

Instead of deducing such a measure from the binocular epipolar constraint of point feature as in many previous methods, in this paper, we propose to use the geometric incidence constraint of line features. Specifically, as shown in Figure 1, for the matched 2D line features $(\mathbf{l},\mathbf{l}',\mathbf{l}''\ldots)$ on exactly synchronized frames $(\mathcal{I},\mathcal{I}',\mathcal{I}''\ldots)$, their corresponding back-projection planes $(\boldsymbol{\pi},\boldsymbol{\pi}',\boldsymbol{\pi}''\ldots)$ must intersect in a single 3D line, i.e., forming a sheaf (or pencil) of

the back-projection planes. Otherwise, no such plane sheaf can be formed in general. Such a constraint works for only 3 or more views since two back-projection planes always intersect, even if they correspond to different 3D lines at the same time instant or to the same 3D line at different time instants.

Our new method works as follows. First, the line and/or point features are matched across sequences manually or automatically and then tracked automatically within each sequence. We assume that the features can be tracked throughout the whole sequence for simplicity, i.e., no missing data. Then with the features matched and tracked, all the possible integral synchronization offset combinations are evaluated to find the best one that maximizes the line feature geometric alignment measure w.r.t all available supporting *m*-frame tuples. Specifically, w.r.t. the reference sequence $\mathcal{S}^{(1)}$, the verifiable integral synchronization offset Δ_k corresponding to sequence $\mathcal{S}^{(k)}$ is in the range of $\mathcal{R}^{(k)}=[-N^{(1)}+1, N^{(k)}-1]$. That is, two extreme cases of potential synchronization between two sequence $\mathcal{S}^{(1)}$ and $\mathcal{S}^{(k)}$ are when the first frame of $\mathcal{S}^{(1)}$ is synchronized with the last frame of $\mathcal{S}^{(k)}$ or the last frame of $\mathcal{S}^{(1)}$ is synchronized with the first frame of $\mathcal{S}^{(k)}$. Then for each offset combination candidate $(\Delta_2,\cdots,\Delta_k,\cdots\Delta_m)$ with integral offset $\Delta_k \in \mathcal{R}^{(k)}(k\in(1,m])$, all the available supporting m-frame tuples $\in \mathbf{\Omega}_{(\Delta_2,\cdots,\Delta_k,\cdots\Delta_m)}$, which satisfy the corresponding timeline maps defined as in (1), are checked with the corresponding line feature geometric alignment measure evaluated and stored in an m-dimensional evaluation matrix $\mathbf{E}$ as $\mathbf{E}(i,i+\Delta_2,\cdots,i+\Delta_m)$ with $i\in\mathcal{R}_{\mathbf{\Omega}_{(\Delta_2,\cdots,\Delta_k,\cdots\Delta_m)}}=\{i\,|\,0\le i<N^{(1)}, 0\le i+\Delta_k<N^{(k)}, k\in(1,m]\}$. Hence, for an offset combination candidate $(\Delta_2,\cdots,\Delta_k,\cdots\Delta_m)$, a corresponding number $\left|\mathcal{R}_{\mathbf{\Omega}_{(\Delta_2,\cdots,\Delta_k,\cdots\Delta_m)}}\right|$ (size of the set $\mathcal{R}_{\mathbf{\Omega}_{(\Delta_2,\cdots,\Delta_k,\cdots\Delta_m)}}$) of geometric alignment evaluations $\mathbf{E}(i,i+\Delta_2,\cdots,i+\Delta_m)$ can be collected, whose median $\underset{i\in\mathcal{R}_{\mathbf{\Omega}_{(\Delta_2,\cdots,\Delta_k,\cdots\Delta_m)}}}{Median}\left(\mathbf{E}(i,i+\Delta_2,\cdots,i+\Delta_m)\right)$ in turn is used as the overall synchronization fitness evaluation of $(\Delta_2,\cdots,\Delta_k,\cdots\Delta_m)$. Here the median value is used instead of the mean value for better robustness. Also please note that the initial value of each entry in the evaluation matrix $\mathbf{E}$ is initialized to zero.

Therefore, after checking all possible integral synchronization offset combinations, the best integral synchronization offsets $(\tilde{\Delta}_2,\cdots,\tilde{\Delta}_m)$ can be recovered such that the synchronization fitness evaluation is maximized, that is, $(\tilde{\Delta}_2,\cdots,\tilde{\Delta}_m)=\underset{(\Delta_2,\cdots,\Delta_m)}{\arg\max}\left(\underset{i\in\mathcal{R}_{\mathbf{\Omega}_{(\Delta_2,\cdots,\Delta_k,\cdots\Delta_m)}}}{Median}\left(\mathbf{E}(i,i+\Delta_2,\cdots,i+\Delta_m)\right)\right)$. Then by using the recovered integral synchronization offsets as initial values, a post-optimization using the Levenberg-Marquardt (LM) method could be performed to further achieve sub-frame synchronization accuracy.

In the following, the implementation details of our method are presented.

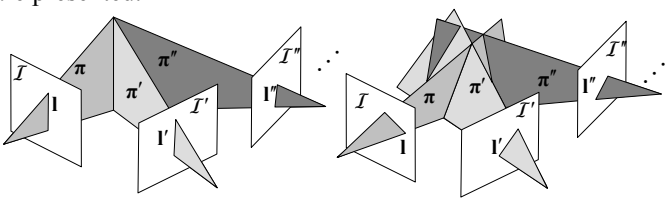

Figure 1. Illustration of the difference between synchronized and unsynchronized cases when the lines are back projected.

3.2 Implementation Details

3.2.1 Issue of computational Complexity

Synchronization using an exhaustive search strategy has poor scalability due to the computation complexity when the sequences are very long or when many sequences are to be synchronized. To address this problem, the following steps are used in our implementation.

First, instead of processing all the sequences at once, we process them in groups of three. A sequence may be present in more than one group. Then the synchronization process is performed for each group independently. The results are integrated together through the common sequences in different groups so that a global synchronization can be established.

Even by sub-grouping, the intrinsically cubic computation complexity may still be too high to be practical. To further speed up the synchronization process, a hierarchical coarse-to-fine approach is taken. In particular, each sequence in a group is first temporally down-sampled appropriately so that the total number of possible synchronization offset combinations decreases dramatically. After synchronizing the down-sampled sequences, the result is transformed properly so that the sequences at the original frame rates are coarsely synchronized. Then three sub-sequences around the coarsely synchronized frames are extracted and further synchronized without the temporal down-sampling. For long sequences, multiple hierarchies could be used as well. Another advantage of processing sequences hierarchically is that, the larger inter-frame motions make it easier to distinguish off-synchronization between frames. Moreover, if prior knowledge on the possible synchronization range between different sequences were available, the synchronization process could also be accelerated. As shown in the following experimental results section, by sub-grouping and hierarchical processing, the efficiency can be improved dramatically.

3.2.2 Geometric alignment of line features

As mentioned before, each potential synchronization offset combination is evaluated on how well the induced timeline maps can synchronize the sequences in question by checking all of its supporting frame tuples, each of which in turn is evaluated individually by measuring the geometric alignment of the matched 2D line features. W.l.o.g., suppose

that we are evaluating a 3-frame tuple $\Gamma(\mathcal{I},\mathcal{I}',\mathcal{I}'')$ and the matching point and line features (not all stationary or move rigidly) are $\{\mathbf{x}_k^i \mid i=1\cdots n_P, k=1\cdots 3\}$ and $\{\mathbf{l}_k^i \mid i=1\cdots n_L, k=1\cdots 3\}$, respectively. Then our evaluation approach is fairly straightforward. First, the tri-focal tensor $\mathbf{T}_i^{jk}$ of 3 views $(\mathcal{I},\mathcal{I}',\mathcal{I}'')$ is recovered using the point features[1]. In this paper, the perspective factorization method [16] is used. Other methods such as RANSAC based 6-point algorithm [8] can be used as well. Then for each matched line feature triplet $(\mathbf{l},\mathbf{l}',\mathbf{l}'')$, the tensor-based line transferring $((\mathbf{l}',\mathbf{l}'')\rightarrow\mathbf{l}$, $(\mathbf{l},\mathbf{l}'')\rightarrow\mathbf{l}'$ and $(\mathbf{l},\mathbf{l}')\rightarrow\mathbf{l}'')$ distance errors are evaluated, from which the corresponding geometric alignment measure is deduced for evaluating how well frames $(\mathcal{I},\mathcal{I}',\mathcal{I}'')$ are synchronized.

Specifically, given the tri-focal tensor $\mathbf{T}_i^{jk}$ of 3 views $(\mathcal{I},\mathcal{I}',\mathcal{I}'')$ and the matched lines in the 2nd $(\mathbf{l}'=(l_1',l_2',l_3')^T)$ and 3rd view $(\mathbf{l}''=(l_1'',l_2'',l_3'')^T)$, then the corresponding line in the first view $\mathbf{l}=(l_1,l_2,l_3)^T$ can be obtained by transferring from $\mathbf{l}'$ and $\mathbf{l}''$ through the tensor operation of $l_i=l_j'l_k''\mathbf{T}_i^{jk}$. Similarly, the above property also holds for the line transfer process of $(\mathbf{l},\mathbf{l}'')\rightarrow\mathbf{l}'$ and $(\mathbf{l},\mathbf{l}')\rightarrow\mathbf{l}''$ w.r.t. the different transferred destination frames (with the trifocal tensors calculated accordingly).

When $(\mathcal{I},\mathcal{I}',\mathcal{I}'')$ are not synchronized, the estimated trifocal tensor in general will be invalid and hence, the alignment constraint of the matched line features will be violated, which is reflected in the large distance between the transferred and the observed line features. Therefore, the line transfer distance errors could be used to measure the geometric alignment for evaluating the synchronization fitness of the three frames. Specifically, for a given 3-frame tuple $\Gamma(\mathcal{I},\mathcal{I}',\mathcal{I}'')$, the corresponding line transfer distance error between the *i*-th line triplet $(\mathbf{l},\mathbf{l}',\mathbf{l}'')$ is calculated as

$$\varepsilon_\Gamma^{(i)}(\mathbf{l},\mathbf{l}',\mathbf{l}'')=d_\perp^2(\mathbf{x}_A,\tilde{\mathbf{l}})+d_\perp^2(\mathbf{x}_B,\tilde{\mathbf{l}})+d_\perp^2(\mathbf{x}_A',\tilde{\mathbf{l}}')+d_\perp^2(\mathbf{x}_B',\tilde{\mathbf{l}}')+d_\perp^2(\mathbf{x}_A'',\tilde{\mathbf{l}}'')+d_\perp^2(\mathbf{x}_B'',\tilde{\mathbf{l}}'') \quad (2)$$

where $\tilde{\mathbf{l}}(\tilde{\mathbf{l}}',\tilde{\mathbf{l}}'')$ are the transferred lines, $\mathbf{x}_{A(B)}$ ($\mathbf{x}'_{A(B)}$, $\mathbf{x}''_{A(B)}$) are two arbitrary (or end) points on line $\mathbf{l}(\mathbf{l}',\mathbf{l}'')$, and $d_\perp(\mathbf{x},\mathbf{l})$ denotes the 2D Euclidean distance from point $\mathbf{x}$ to line $\mathbf{l}$. Then based on the transfer distance errors of all matched line triplets in frames $(\mathcal{I},\mathcal{I}',\mathcal{I}'')$ of Γ, we define the line alignment measure for tuple Γ as $C(\Gamma)=3\cdot n_L/(\sum_{i=1}^{n_L}\varepsilon_\Gamma^{(i)}+\mu_0)$, where μ_0 is a small positive value for avoiding the divide-by-zero problem. In this paper, $\mu_0=10^{-6}$. Therefore, the smaller the sum of the line transfer distance errors, the larger the line alignment measure, and in turn the better the frames are synchronized. To minimize the transfer distance error, it is equivalent to maximize the alignment measure.

[1] It can be extended to using line features.

One important remark must be made regarding the possible degenerate cases of the trifocal tensor based line transfer when two back-projection planes are coplanar. Numerically, such degeneracy can be detected by inspecting the parameters of the transferred line since all of them will become very close to zero in the degenerate cases [8], all of which will be excluded in the alignment measure.

3.2.3. Post-optimization for sub-frame synchronization

With the integral frame synchronization offsets $\{\Delta_2\cdots\Delta_m\}$ recovered using the above hierarchical approach, we may further perform a post-optimization to obtain sub-frame synchronization accuracy.

Specifically, the following cost function $f(\Delta_2,\cdots,\Delta_k,\cdots\Delta_m)=\frac{1}{2}\sum_{\Gamma\in\mathbf{\Omega}(\Delta_2,\cdots,\Delta_k,\cdots\Delta_m)}\varepsilon(\Gamma)^2$ is minimized w.r.t. $\{\Delta_2\cdots\Delta_m\}$ using the LM method. For the case of synchronizing three sequences, $\varepsilon(\Gamma)^2$ is defined as $\sum_{i=1}^{n_L}\varepsilon_\Gamma^{(i)}$, that is, the sum of the transfer distance errors of matched lines of frame-tuple Γ. For the general case of synchronizing more than three sequences, $\varepsilon(\Gamma)^2$ can be defined as the sum of squares of the re-projection error of all the matching line features of frame-tuple Γ.

Using the integral frame synchronization result as initial values, $\{\Delta_2\cdots\Delta_m\}$ are updated iteratively by the LM method to minimize $f(\Delta_2,\cdots,\Delta_k,\cdots\Delta_m)$. In particular, in each iteration, $\{\Delta_2\cdots\Delta_m\}$ are updated based on the cost function value and its partial derivative with respect to each offset being optimized. With offsets updated, the corresponding supporting frame-tuple set $\mathbf{\Omega}$ is updated as well with the linear frame interpolation done as needed. It is noteworthy that we don't need to interpolate the frame, but only the point/line features. On the other hand, since the current state-of-the-art line trackers can not guarantee that the endpoints of the tracked lines are always matched between consecutive frames, the line features are interpolated by interpolating the normalized line equations instead of the line end points.

To optimize the synchronization offsets $\Delta_k(k=2,\cdots,m)$ for sub-frame accuracy using the LM method, the partial derivatives $\frac{\partial f}{\partial\Delta_k}(k=2,\cdots,m)$ of the cost function to be minimized with respect to each offset being optimized have to be calculated. Since it is very difficult, if not impossible, to establish the exact analytic relation between the cost function $f(\Delta_2,\cdots,\Delta_k,\cdots\Delta_m)$ and the synchronization offsets $\{\Delta_2\cdots\Delta_m\}$, an approximate method for calculating the required partial derivatives is used. In particular, for $\frac{\partial f}{\partial\Delta_k}(k=2,\cdots,m)$, we evaluate $f_0=f(\Delta_2,\cdots,\Delta_k,\cdots\Delta_m)$ and $f_{\bar{\Delta}}=f(\Delta_2,\cdots,\Delta_k+\bar{\Delta},\cdots\Delta_m)$ respectively with $\bar{\Delta}$ being a small positive value ($\bar{\Delta}=0.002$ in this paper), and then

$\frac{\partial f}{\partial \Delta_k}$ is approximated as $(f_{\bar{\Delta}} - f_0)/\bar{\Delta}$. As can be seen from the experiments, such approximation is quite accurate.

3.3 Work flow review

The workflow of our line feature based synchronization method can be summarized as follows:

Step 1: Match and then track point and/or line features across and along sequences

Step 2: Hierarchically recover the best integral synchronization offsets by evaluating corresponding supporting frame-tuples as done in section 3.2.2.

Step 3: Optimize the recovered integral synchronized offsets using the LM method for sub-frame accuracy as shown in section 3.2.3.

4. EXPERIMENTS

To test the effectiveness of our proposed method, extensive experiments using synthetic and real video sequences have been done. Additionally, two other pairwise synchronization methods, i.e. the 5-point method [17] and the affine factorization measurement matrix rank based method [14] are also implemented and compared. In the comparison, the 5-point method is repeated for different 5-point configurations for five times, with the best result selected.

For simplicity, the following terminology is used to describe the experimental details. In particular, "$[u:v]$" denotes the sub-sequence frame range starting from frame u to frame v and "$\%n$" denotes the corresponding sequence temporally down-sampled with a down-sampling rate n. As an example, $\mathcal{S}^{(k)}_{[u:v]\%n}$ represents a sequence obtained by down-sampling with a rate n to the $[u:v]$ sub-sequence of $\mathcal{S}^{(k)}$. For a specific group of sequences $\left\{\mathcal{S}^{(1)}_{[u_1:v_1]\%n_1}, \mathcal{S}^{(2)}_{[u_2:v_2]\%n_2}, \cdots, \mathcal{S}^{(m)}_{[u_m:v_m]\%n_m}\right\}$, as the corresponding synchronization result recovered at the j-th hierarchical level, $(\Delta_2, \Delta_3, \cdots, \Delta_m)_{L_j}$ means that the corresponding timeline map between sequence $\mathcal{S}^{(1)}$ and $\mathcal{S}^{(k)}$ is $\mathcal{M}_{\mathcal{T}^{(1)}\to\mathcal{T}^{(k)}}: \alpha^{(k)}(t_1 - u_1) + \Delta_k \cdot n_k = t_k - u_k$. Or alternatively speaking, all the frame-tuples $\Gamma(\mathcal{I}_{t_1}, \mathcal{I}_{\alpha^{(2)}(t_1-u_1)+(\Delta_2\cdot n_2+u_2)}, \cdots, \mathcal{I}_{\alpha^{(m)}(t_1-u_1)+(\Delta_m\cdot n_m+u_m)})$ in the original sequences are in exact synchrony. Moreover, the 3D evaluation graph in the following illustrations shows the line geometric alignment measurements surface $z(x,y)$ for 3 sequences $\left\{\mathcal{S}^{(1)}_{[u_1:v_1]\%n_1}, \mathcal{S}^{(2)}_{[u_2:v_2]\%n_2}, \mathcal{S}^{(3)}_{[u_3:v_3]\%n_3}\right\}$ being synchronized, where $z(x,y)$ is the median of the line alignment measurements of all the frame-tuples supporting the timeline maps based on specific synchronization offset relations $(\Delta_2 = x, \Delta_3 = y)$ as described above. Therefore, the offsets $(\tilde{x}, \tilde{y})$ corresponding to the highest peak in the 3D evaluation graph is just the offsets best synchronize the sequences $\left\{\mathcal{S}^{(1)}_{[u_1:v_1]\%n_1}, \mathcal{S}^{(2)}_{[u_2:v_2]\%n_2}, \mathcal{S}^{(3)}_{[u_3:v_3]\%n_3}\right\}$ in question.

In the following experiments, the features are matched manually across sequences in the first frames and then tracked automatically. For point features, the KLT [12] point tracker is used. While for line features, a line tracker based on the work of [6] is used. Here some remarks should be made on feature selection. First the selected features should be as spatially distributed as possible in the frames. Second it is better to use features whose relative position changes between frames are large since otherwise the difference of the recovered geometry due to little off-synchronization between frames may not be significant enough to incur large geometric alignment error. So to some degree, camera movements can sometimes even help in the synchronization. Finally, all the features are normalized [8] for better robustness in computing perspective factorization and tri-focal tensor transferring.

4.1 Synthetic videos

The synthetic video experiments are done to quantitatively evaluate the performance of our new method with known ground truth. In particular, the synthesized dynamic scenes are constructed and rendered using POV-Ray [21] with the virtual cameras moving appropriately. The frame rates are constant but not necessarily the same. As an example shown in Figure 2, 5 sequences are synthesized, each of which starts at a different global time instant and spans for a different duration (with $Fps^{(1,2,3)} = 25$ and $Fps^{(4,5)} = 50$). Figure 2 depicts the global timeline relations between all the five sequences.

For our line feature based method, the 5 sequences are divided into two 3-sequence groups $\left\{\mathcal{S}^{(1)}, \mathcal{S}^{(2)}, \mathcal{S}^{(3)}\right\}$ and $\left\{\mathcal{S}^{(1)}, \mathcal{S}^{(4)}, \mathcal{S}^{(5)}\right\}$, each of which is synchronized first and then the results are integrated together. As for the other two methods, the synchronization is done pairwise w.r.t. the reference sequence $\mathcal{S}^{(1)}$.

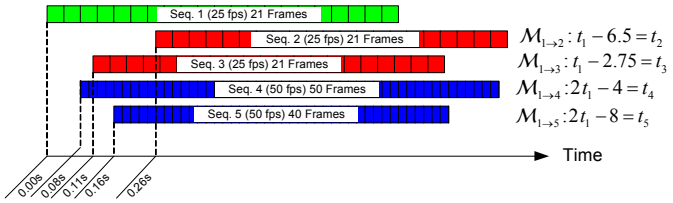

Figure 2. Global timeline relations between 5 synthetic videos.

In Figure 3, one can see that the line based method recovers the timeline maps correctly. The corresponding peak of the 3D line alignment evaluation is rather dominant and the sub-frame synchronization result is quite accurate $(\Delta_2 = -6.503, \Delta_3 = -2.712, \Delta_4 = -3.977, \Delta_5 = -8.030)$ as compared with the ground truth: $(\Delta_2 = -6.5, \Delta_3 = -2.75, \Delta_4 = -4.0, \Delta_5 = -8.0)$. In contrast, the other two methods incurred fairly large errors. For the 5-point method, the best recovered synchronization offsets are $(\Delta_2 = -8, \Delta_3 = -9, \Delta_4 = 1, \Delta_5 = -5)$ and for the rank based method, they

are $(\Delta_2 = -9, \Delta_3 = -12, \Delta_4 = 1, \quad \Delta_5 = -6)$. Therefore it can be seen that these two affine geometric property based methods are not good for synchronizing sequences with large perspective distortions.

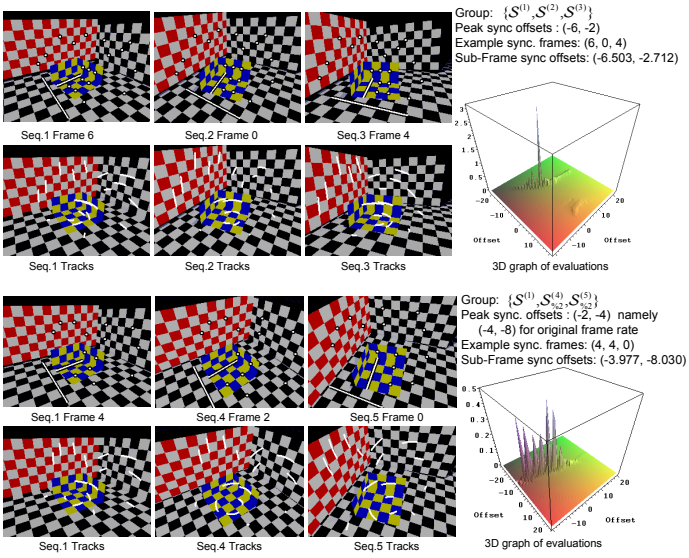

Figure 3. Synchronization result of 5 synthetic videos.

4.2 Real videos

Experiments are also conducted using real video sequences captured in the lab and outdoor.

4.2.1 Lab scenes

In the lab scene, a client-server based multiple camera capturing system is used. The CCD firewire cameras and DV camcorders are controlled to start streaming video by a server PC. In the simultaneous capturing mode, the synchronization offsets between the captured sequences are mainly caused by the network communication delays and the intrinsic response of the controlled cameras, which are usually in a small range of a couple of frames. However, to test the performance of our method to handle large offsets, in the lab scene experiment, we intentionally configure the server to start each camcorder asynchronously at relatively large offsets.

As shown in Figure 4, a lab scene with the electronic stopwatch shown on the wall is captured by two stationary and one moving DV camcorders. For synchronization, one moving line on the pattern board and one stationary line on the wall are tracked. Since the sequences are a little long, they are synchronized hierarchically for higher efficiency. Specifically, the temporally down-sampled sequences $\{\mathcal{S}^{(1)}_{\%4}, \mathcal{S}^{(2)}_{\%4}, \mathcal{S}^{(3)}_{\%4}\}$ are first synchronized at level L_1 with the result of $(\Delta_2 = 5, \Delta_3 = -5)_{L_1}$, namely $(\Delta_2 = 20, \Delta_3 = -20)$ after transforming to the original frame-rate context. Then guided by this initial result, 3 sub-sequences $\{\mathcal{S}^{(1)}_{[20,45]}, \mathcal{S}^{(2)}_{[40,65]}, \mathcal{S}^{(3)}_{[0,25]}\}$ are extracted and further synchronized at level L_0. The resulting synchronization offsets are $(\Delta_2 = 3, \Delta_3 = 4)_{L_0}$ which means that all frame tuples $\Gamma(\mathcal{I}^{(1)}_{t_1}, \mathcal{I}^{(2)}_{t_1+23}, \mathcal{I}^{(3)}_{t_1-16})$ are synchronized. From the timing information transitions of the synchronized frames shown in the middle column of Figure 4 and by further considering the sub-frame optimization result $(\Delta_2 = 3.226, \Delta_3 = 4.127)_{sub}$ (namely, the sub-frame synchronization offsets for the original sequence $\{\mathcal{S}^{(1)}, \mathcal{S}^{(2)}, \mathcal{S}^{(3)}\}$ is $(\Delta_2 = 23.226, \Delta_3 = -15.873)_{sub}$), it can be seen that our result is very accurate. In contrast, for the other two compared methods, the 5-point based one gives the result of $(\Delta_2 = 26, \Delta_3 = -8)$ while the rank-based one gives the result of $(\Delta_2 = 26, \Delta_3 = -7)$, both of which incur quite large errors.

4.2.2 Outdoor scenes

For the outdoor scenes, only the hand-held camcorders are used. The capturing process is triggered manually and independently so that the synchronization offsets might be fairly large. The sequences are not directly streamed to the hard drive but to the tape first and then transferred to the computer using video editing software. In the following, we present the skating sequence and the ping-pong sequence experiments as representative examples.

As the first example, 3 skating sequences captured in a shopping mall shown in Figure 5 are synchronized. This experiment is quite challenging in that the 3 DV camcorders are all panning (or swinging) and one of them is zooming. Furthermore, the features available are very close to the co-planar degenerate case, which makes accurate tri-focal tensor recovery more difficult. However, our synchronization method still works pretty well. In particular, 10 point features and 2 line features are used and a 2-level sequence pyramid is built for the hierarchical synchronization. At the top level L_1, the original sequences are temporally down-sampled by 4, resulting with the sequences $\{\mathcal{S}^{(1)}_{\%4}, \mathcal{S}^{(2)}_{\%4}, \mathcal{S}^{(3)}_{\%4}\}$, for which the synchronization result is $(\Delta_2 = -5, \Delta_3 = 0)_{L_1}$, namely $(\Delta_2 = -20, \Delta_3 = 0)$ when considered in the original frame-rate context. Then going to the lower level L_0 (of original frame rate), the sub-sequences $\{\mathcal{S}^{(1)}_{[20:50]}, \mathcal{S}^{(2)}_{[0:30]}, \mathcal{S}^{(3)}_{[20:50]}\}$ are extracted based on the result from the upper level L_1 and then further synchronized with the result of $(\Delta_2 = 0, \Delta_3 = 0)_{L_0}$ and the corresponding sub-frame optimization result of $(\Delta_2 = 0.122, \Delta_3 = -0.037)_{sub}$. That is, for the original sequences $\{\mathcal{S}^{(1)}, \mathcal{S}^{(2)}, \mathcal{S}^{(3)}\}$, the sub-frame synchronization offsets are $(\Delta_2 = -19.878, \Delta_3 = -0.037)_{sub}$. By inspecting the hand-grasping procedure of two skating girls shown in the middle column of Figure 6, such a synchronization result is very accurate and coincides with the manually identified ground truth perfectly. While for the other two methods, both get the result of $(\Delta_2 = -16, \Delta_3 = 2)$, incurring synchronization error of $3 \sim 4$ frames.

Shown in Figure 6 is another experiment in which three ping-pong sequences captured in a gym using 3 stationary DV camcorders are synchronized. In total 10 point features and 2

line features are matched and tracked. Since all the sequences are quite short, they are directly synchronized without building the hierarchical sequence pyramid. In particular, for the sequences $\{\mathcal{S}^{(1)},\mathcal{S}^{(2)},\mathcal{S}^{(3)}\}$, the integral synchronization offsets are recovered as $(\Delta_2 = 10, \Delta_3 = 8)$. Although such result is quite close to the ground truth identified manually by checking the trajectory of the moving ping-pong ball bounced by the person walking from the right to left, the off-synchronization error is still fairly large. Then by applying sub-frame synchronization optimization, more accurate sub-frame synchronization offsets are obtained, that is, $(\Delta_2 = 10.498, \Delta_3 = 9.830)_{sub}$. By carefully checking the timing instant (frame) at when the ping-pong ball hits the pad and its corresponding height from the pad in the next instant (frame), we can see that the sub-frame synchronization offsets are quite accurate. Specifically, the height of the ball from the pad in frame $\mathcal{I}_{15}^{(1)}$ of sequence $\mathcal{S}^{(1)}$ is obviously higher than that in frame $\mathcal{I}_{25}^{(2)}$ and lower than that in frame $\mathcal{I}_{26}^{(2)}$ of sequence $\mathcal{S}^{(2)}$. Therefore, the synchronization offset Δ_2 must be in the range of $(25-15, 26-15)$, that is, $(10,11)$. By further considering the kinetics characteristics of the ball movement, we estimate that the ground truth of Δ_2 should be very close to 10.5 which coincides with our result very well. While for sequence $\mathcal{S}^{(3)}$, we can also see that the inaccurate integral result is successfully corrected by the sub-frame optimization. While in the comparison, the 5-point and the rank-based methods also get quite accurate results of $(\Delta_2 = 10, \Delta_3 = 8)_{5-point}$ and $(\Delta_2 = 10, \Delta_3 = 9)_{rank\ based}$ respectively, although no corresponding sub-frame optimization process can make further refinement.

Many other similar real video experiments have been performed with similar performance and accuracy.

5. Conclusions

Our extensive experimental results suggest that our proposed method is robust and effective. It provides a general framework for synchronizing multiple sequences captured by free-moving full-perspective cameras. Unlike previous feature based methods, the line features are used for checking the synchrony between frames, resulting in improved robustness and accuracy in synchronization. Furthermore, our proposed method is more adaptive to general scenes with large perspective distortions than previous ones. As well, with accurate input feature data, very accurate sub-frame synchronization can be obtained through a LM based optimization.

As to possible extensions, our future research will be focused on the following topics. First, we want to enable automatic synchronization by automating the process of point/line feature matching across the wide-base-line sequences. Second, we plan to improve the efficiency of our method by segmenting a long sequence into shorter sub-sequences and by coarse-to-fine checking instead of exhaustive checking all possible synchronization offset combinations. Furthermore, the issues on how to synchronize variable frame-rate sequences or sequences with more complex timeline maps are also interesting topics to explore.

REFERENCES

[1] R. L. Carceroni, F. L. C. Pádua, G. A. M. R. Santos, "Linear sequence-to-sequence alignment," in *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, Washington DC, USA, 2004, Vol. 1, pp. 746- 753.

[2] J. Carranza, C. Theobalt, M. A. Magnor, and H. P. Seidel. "Free-viewpoint video of human actors," in *Proc. of ACM SIGGRAPH*, San Diego, USA, July 2003, pp. 569-577.

[3] Y. Caspi and M. Irani, "A step towards sequence-to-sequence alignment," in *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, Hilton Head Island, South Carolina, June 13-15, 2000, Vol. 2, pp. 682–689.

[4] Y. Caspi and M. Irani, "Alignment of non-overlapping sequences," in *Proc. of the 8th International Conference. on Computer Vision*, Vancouver, B.C., July 2001, Vol. 2, pages 76–83.

[5] Y. Caspi, D. Simakov, and M. Irani, "Feature-based sequence-to-sequence matching," in *Proc. Workshop on Vision and Modelling of Dynamic Scenes, Copenhagen*, Denmark, May 2002.

[6] N. Chiba and T. Kanade, "A Tracker for broken and closely spaced lines," in *Proc. of the International Society for Photogrammetry and Remote Sensing Conference (ISPRS '98)*, Stuttgart, Germany, 1998, Vol. XXXII, No. 5, pp. 676 - 683.

[7] M. Gong and Y.H. Yang, "Camera field rendering of static and dynamic scenes," *Graphical Models*, Vol. 67, 2005, pp. 73-99.

[8] R. I. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*, Cambridge University Press, 2000.

[9] D. W. Pooley, M. J. Brooks, A. van den Hengel, and W. Chojnacki, "A voting scheme for estimating the synchrony of moving camera videos," *in Proc. of International Conference on Image Processing* (ICIP 2003), Barcelona, Sept. 2003, Vol. 1 pp. 413-416.

[10] C. Rao, A. Gritai, M. Shah, and T. Syeda-Mahmood, "View-invariant alignment and matching of video sequences," in *Proc. IEEE International Conference on Computer Vision*, Nice, France, Oct. 13-16, 2003, pp. 939-045.

[11] I. Reid and A. Zisserman，"Goal-directed Video Metrology，" in *Proc. of the 4th European Conference on Computer Vision*, vol. 2 LNCS 1065, Cambridge , April 1996, pp. 647-658.

[12] J. Shi and C. Tomasi, "Good features to track," in *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition*, Seattle, Washington, June 1994, pp. 593-600.

[13] G. P. Stein, "Tracking from multiple view points: Self-calibration of space and time," in *DARPA IU Workshop*, 1998, pp. 521-527.

[14] P. Tresadern and I. Reid, "Synchronizing image sequences of non-rigid objects," in *Proc. British Machine Vision Conference*, Norwich, Sept. 9-11 2003, Vol. 2, pp. 629-638.

[15] P. Tresadern and I. Reid, "Uncalibrated and unsynchronized human motion capture: a stereo factorization approach," in *Proc. IEEE Conf on Computer Vision and Pattern Recognition*, Washington, D. C., June 27 - July 2, 2004, pp.128-134.

[16] B. Triggs, "Factorization methods for projective structure and motion," in *Proc. IEEE Conference on Computer Vision and Pattern* Recognition, San Francisco, CA, USA, 1996, pp 845–851.

[17] T. Tuytelaars and L. Van Gool, "Synchronizing video sequences," in *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, Washington DC, USA, 2004, Vol. 1, pp. 762- 768.

[18] S. Vedula, S. Baker, and T. Kanade, "Spatio-Temporal View Interpolation," in *Proc. of the 13th ACM Eurographics Workshop on Rendering*, Pisa, Italy, June, 2002, pp 65-76.

[19] L. Wolf and A. Zomet, "Correspondence-free synchronization and reconstruction in a non-rigid scene," in *Proc. Workshop on Vision and Modeling of Dynamic Scenes, Copenhagen*, Denmark, May 2002.

[20] C. L. Zitnick, S. B. Kang, M. Uyttendaele, S. A. J. Winder, and R. Szeliski , "High-quality Video View Interpolation Using a Layered Representation," in *Proc. of ACM SIGGRAPH,* Los Angeles, USA, Aug. 2004, pp 600-608.

[21] The Persistence of Vision Raytracer, http://www.povray.org/

[22] Video insight PC video surveillance systems, http://www.securityideas.com/viinpcvisu.html

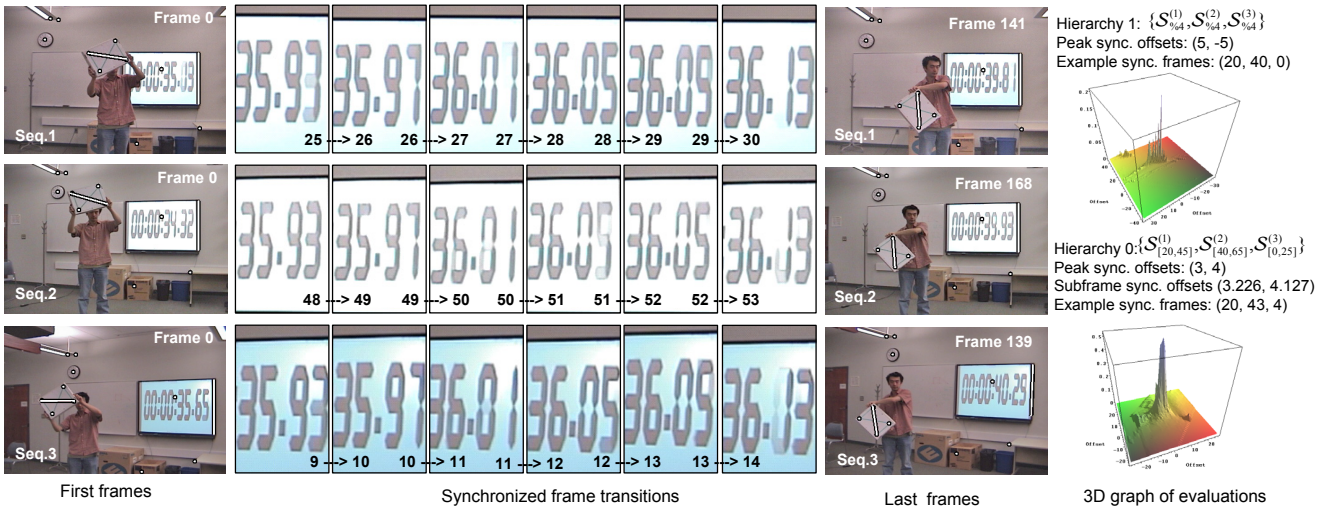

Figure 4. Synchronization result of stop-watch sequence experiment.

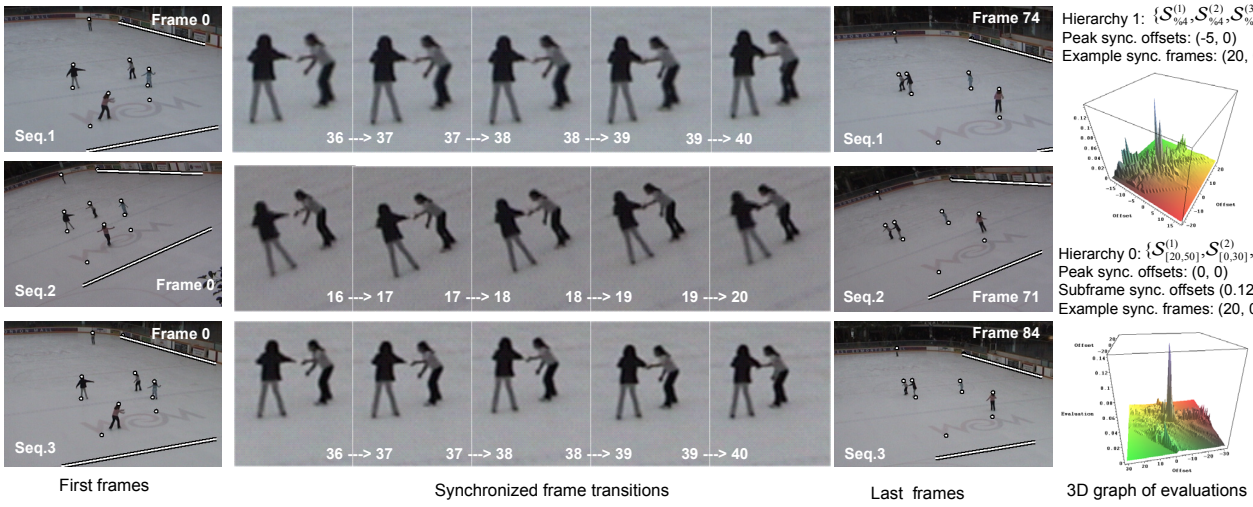

Figure 5. Synchronization result of the skating sequence experiment.

Figure 6. Synchronization result of the ping-pong sequence experiment.