Spatio-Temporal Alignment of Non-Overlapping Sequences from Independently Panning Cameras

S. Morteza Safdarnejad
Michigan State University
East Lansing, MI, USA
safdarne@egr.msu.edu

Xiaoming Liu
Michigan State University
East Lansing, MI, USA
liuxm@cse.msu.edu

Abstract
This paper addresses the problem of spatio-temporal alignment of multiple video sequences. We identify and tackle a novel scenario of this problem referred to as Non-overlapping Sequences (NOS). NOS are captured by multiple freely panning handheld cameras whose field of views (FOV) might have no direct spatial overlap. With the popularity of mobile sensors, NOS rise when multiple cooperative users capture a public event to create a panoramic video, or when consolidating multiple footages of an incident into a single video. To tackle this novel scenario, we first spatially align the sequences by reconstructing the background of each sequence and registering these backgrounds, even if the backgrounds are not overlapping. Given the spatial alignment, we temporally synchronize the sequences, such that the trajectories of moving objects (e.g., cars or pedestrians) are consistent across sequences. Experimental results demonstrate the performance of our algorithm in this novel and challenging scenario, quantitatively and qualitatively.

1. Introduction
Spatio-temporal alignment of multiple videos [7–9, 11, 14, 21, 24, 28, 31] is a well-studied vision problem with a wide range of applications, e.g., human action recognition [25, 29], video editing [31], markerless motion capture [14], video mosaicing, change detection [8], and abandoned object detection [16]. Previous works study different aspects and scenarios of the spatio-temporal alignment. Some works target sequences from the same scene but different viewpoints [14, 21]. Some can handle sequences recorded at different times by independent moving cameras that follow a similar trajectory [9,11,31]. The seminal work of Caspi and Irani [7] studies spatially non-overlapping sequences when two fixed cameras move jointly in space.

Our work covers a novel unexplored aspect of spatio-temporal alignment of sequences, for non-overlapping sequences (NOS). Targeted NOS are captured by freely and independently panning cameras, from nearby viewpoints, with limited translation, especially in the optical axis direction. In NOS, sequences might not have any pair of frames that have spatial overlap and belong to the same world time instant. More interestingly, sequences might even not cover some common regions of the same scene over the progression of time. In other words, if we reconstruct the observed background by these sequences, the backgrounds may be non-overlapping, i.e., in Fig. 1 (a), the overall spatial overlap does not exist.

Given the ubiquitousness of smartphones and wearcams, NOS are increasingly common. When amateur users unsynchronizedly shoot videos of an event, aligning these videos leads to a single comprehensive video, with greater spatial and temporal spans (Fig. 1 (b)). This resultant video is essentially a panoramic video, shot by smartphones, without the need to fix the cameras to each other or use tripods. Further, when many witnesses capture videos during crime actions or violations, each sequence may cover part of the story. Aligning these videos into a unified large-scale 3D volume provides a better grasp of the full picture.

The existing spatio-temporal alignment algorithms fail in the case of NOS, since even if there is some overall
spatial overlap, spatial alignment of apparently overlapping frames, e.g., frames \(a\) and \(b\) in Fig. 1 (a), obviously violates the temporal alignment. However, by decomposing the task to spatial alignment first and then temporal alignment based on scene dynamics, the problem can be solved. In general our proposed algorithm assumes NOS satisfy the following two assumptions. 1) Although the sequences have no corresponding frames that share a common scene at the same world time stamp, and no overall overlap as in Fig. 1 (a), they cover nearby parts of a scene from similar view angles. 2) There are moving objects in the scene which move from the field of view (FOV) of one camera to FOV of other cameras. Note that in panoramic imaging the best results are obtained if the camera has nodal camera motion, otherwise parallax-tolerant methods should be used to hide artifacts from parallax [15, 32]. Similarly, our proposed algorithm creates the best result if the camera baseline is small, although due to the non-overlapping videos, larger baselines are handled with minor visual degradation.

Our algorithm utilizes global motion compensation to map each frame to a camera-motion-removed video and reconstruct the background for each sequence, independently. With the two assumptions, these potentially non-overlapping backgrounds are aligned via appearance cues and also the prediction that \textit{where} a moving object leaving FOV of a camera will appear in FOV of another camera. Collection of the former mappings and the latter background alignment, can spatially align each frame of each sequence with respect to frames from other sequences. Given the spatial alignment and the assumption 2, we predict \textit{when} a moving object leaving FOV of one camera will appear in FOV of another. We mathematically formulate this prediction and estimate the temporal alignment.

In summary, this paper makes these contributions:

\begin{itemize}
  \item A new scenario in spatio-temporal alignment of sequences is identified and studied.
  \item A spatial alignment algorithm for NOS via alignment of non-overlapping reconstructed backgrounds and consistency of objects movement is proposed.
  \item The trajectory of moving objects with smooth path is used as a clue for temporal alignment of NOS.
\end{itemize}

2. Previous Work

The prior works in spatio-temporal alignment of sequences mostly differ in their assumptions and scenarios, e.g., the camera movement (static, jointly moving, or moving), camera view-point (similar or distinct), extent of overlap in sequences, and extent of similarity of camera motion paths. The work of [12] presents an excellent taxonomy of these assumptions, one of which is that, to align sequences from the same event captured by freely moving cameras, coherent scene appearance is assumed. We lift this assumption by handling non-overlapping sequences, although we do assume negligible camera movement in the optical axis direction. We now review key scenarios in prior work.

**Jointly moving cameras** Caspi and Irani align spatially non-overlapping sequences when two closely \textit{attached} cameras move \textit{jointly} in space (Fig. 2 (a)) [7]. Assuming cameras share the \textit{same} projection center, their relationship is modeled as a fixed homography \(H\). Esquivel et al. [10] relax the projection center assumption and calibrate a multi-camera rig from non-overlapping views, assuming synchronized sequences. Recently, some works generate panoramic videos from synchronized sequences of a multi-camera rig [15, 22]. Given that the cameras are fixed relative to each other and have overlapping FOV, these works concentrate on removing the parallax artifacts. In contrast, we focus on creating video panoramas of \textit{unsynchronized} sequences from \textit{independently panning} cameras, removing the requirement of joint cameras and overlapping sequences.

**Cameras following similar trajectories** The authors of [9, 11, 12, 31] align sequences recorded at \textit{different} times by independent moving cameras that follow a \textit{similar} trajectory (Fig. 2 (b)). Assuming one sequence is entirely contained (temporally) within the other, in [9], the alignment is formulated as an energy minimization alternately solved for temporal and spatial alignment parameters and is evaluated on four sets of real videos. In [31] an interactive method for nonlinear temporal video alignments is proposed for video editing. All these methods require coherent scene appearance and cannot handle sequences from moving cameras with no overlap in FOV — the targeted scenario of NOS.

**Stationary cameras at different views** Padua et al. [21] target sequences from the \textit{same} scene but \textit{different} viewpoints (Fig. 2 (c)). The \textit{stationary} cameras allow the estimated camera’s epipolar geometry remain fixed. Motion trajectories are used as cues for both spatial and temporal alignment. Experimental results are provided for five sequences. For each sequence, the optimal tracker is chosen based on the application in hand.

**Time synchronization** Assuming the known 3D object location and calibrated stationary cameras, [6] synchronizes non-overlapping sequences of these cameras. Gaspar et al. [13] synchronize sequences from independently moving cameras, assuming known intrinsic parameters and visibility of two rigid moving objects in both sequences.
3. Proposed Method

We discussed the assumptions for the proposed spatio-temporal alignment of NOS in Sec. 1. The intrinsic and extrinsic camera parameters are not required. Also, the cameras might start capturing at different times, i.e. unsynchronized, with possibly distinct frame rates, and are panned freely and independently. However, best results are achieved by a small camera baseline and limited translation of cameras, especially in the optical axis direction, as we rely on global motion compensation and large variations in scale, degrade the resultant video in its rendering phase.

The proposed algorithm has two stages, (1) spatial alignment (Fig. 3(b)), which relies on the reconstructed backgrounds’ appearance and consistency of movement of objects across the sequences, (2) temporal alignment (Fig. 3(d)), which uses the continuity of objects’ trajectories to synchronize the videos.

Notations As shown in Fig. 3, frame coordinate refers to the pixel coordinate in the input video, sequence coordinate to the global coordinate of the reconstructed background of one video, and world coordinate to the global coordinate of all input videos where the final aligned video is rendered. We denote the coordinates and time stamps in the frame coordinate with plain letters, in the sequence coordinate with \( \sim \) over the notation, e.g., \( \bar{x} \), and in the world coordinate with double \( \sim \), e.g., \( \bar{\bar{x}} \). Accordingly, a transformation from the frame to sequence coordinate has \( \sim \) over the notation, and a transformation from the sequence to world coordinate has double \( \sim \). We use superscript for the sequence number and subscript for, either the frame number or trajectory number. E.g., \( H^s_i \) is the transformation of frame \( i \) in sequence \( s \) from the frame coordinate to sequence coordinate.

3.1. Spatial alignment

We break down the spatial alignment to two phases. First, for each sequence, we map all the frames to the sequence coordinate, via global motion compensation (GMC), which also produces a reconstructed background mosaic (Fig. 3(a)). A crucial assumption for successful GMC is the camera having small motion in the optical axis direction. Second, image alignment is conducted on the reconstructed backgrounds and maps them to the world coordinate (Fig. 3(b)). However, if the backgrounds are non-overlapping, common image alignment cannot be used. Thus, a new alignment scheme is proposed in Sec. 3.1.2.

3.1.1 Global motion compensation

GMC removes intentional or unwanted camera motion in a sequence, creating a video with static background [26, 27]. Essentially, GMC estimates a per frame transformation to the sequence coordinate. We utilize the TRGMC algorithm [26] which handles dynamic scenes and estimates transformations by jointly aligning input frames. TRGMC detects SURF [5] keypoints in each frame, and matches keypoints to densely interconnect all frames, regardless of
their temporal offset. These connections are referred as links. Then, appropriate transformation is applied to each frame and its links, such that the spatial coordinates of the end-points of each link are as similar as possible. Using TRGMC independently for each sequence \( s \), we estimate the mapping \( \tilde{h}^s_i \) to map frame \( i \) to the sequence coordinate.

TRGMC defines the problem as the concealing of densely connected keypoints in a stack of frames. For the convenience of readers, we briefly introduce this algorithm. Given a stack of frames with indices \( i \in \mathbb{K} = \{k_1, \ldots, k_N\} \), TRGMC is formulated as an optimization problem,

\[
\min_{\{h^s_i\}} \epsilon^s = \sum_{i \in \mathbb{K}} [e_i(h^s_i)]^T \Omega^s_i [e_i(h^s_i)],
\]

where \( h^s_i \) is an 8-dim homography transformation parameter from frame \( i \) of sequence \( s \) to the sequence coordinate, \( e_i(h^s_i) \) collects pair-wise alignment errors of frame \( i \) relative to all other frames, and \( \Omega^s_i \) is a weight matrix. Since TRGMC uses homography transformation, it works best with nodal camera motion. In the case of camera translation, TRGMC still works by matching the dominant background, although the result may downgrade with parallax.

The alignment error of frame \( i \) relative to all other frames is the sum of squared differences (SSD) between the start and end coordinates of the links connected to frame \( i \), denoted for the \( k \)th link as \( (x_{i,k}, y_{i,k}) \) and \( (u_{i,k}, v_{i,k}) \), respectively. Thus, the error \( e_i(h^s_i) \) is,

\[
e_i(h^s_i) = [\Delta x_i(h^s_i)^T, \Delta y_i(h^s_i)^T]^T,
\]

where \( \Delta x_i(h^s_i) = \tilde{w}_i^{(x)} - u_i \) and \( \Delta y_i(h^s_i) = \tilde{w}_i^{(y)} - v_i \) denote the errors in \( x \) and \( y \)-axes. The vectors \( \tilde{w}_i^{(x)} \) and \( \tilde{w}_i^{(y)} \) are the \( x \) and \( y \)–coordinates of \( (x_{i,k}, y_{i,k}) \) warped by the parameter \( h^s_i \), respectively.

Equation 1 is solved by taking the Taylor expansion around \( h^s_i \) and finding the increment \( \Delta h^s_i \) that minimizes,

\[
\arg\min_{\Delta h^s_i} [e_i(h^s_i) + \frac{\partial e_i(h^s_i)}{\partial h^s_i} \Delta h^s_i] \Omega^s_i [e_i(h^s_i) + \frac{\partial e_i(h^s_i)}{\partial h^s_i} \Delta h^s_i] + \gamma \Delta h^s_i \mathcal{I} \Delta h^s_i,
\]

where \( \mathcal{I} \) is a diagonal matrix specifying which elements of \( \Delta h^s_i \) need a constraint. By setting the first-order derivative of Eqn. 3 to zero, a closed-form solution for \( \Delta h^s_i \) is obtained. \( h^s_i \) is estimated after enough iterations.

**Spatial alignment of overlapping backgrounds** Given the \( h^s_i \) for all the input videos, we follow [26] to reconstruct the backgrounds \( B^s \) for them. If there exists enough overlap between the backgrounds, common image alignment algorithms may be used. Specifically, we estimate the transformation \( \tilde{h}^s \) that maps the background of sequence \( s \) to the world coordinate, by matching SURF keypoints on background images via the vector field consensus algorithm [20]. In summary, a point with the homogeneous coordinate \( (x, y, 1) \) in frame \( i \) of sequence \( s \) is mapped to the sequence coordinate of sequence \( s \), denoted as \( (\tilde{x}, \tilde{y}, 1) \), and the world coordinate of all sequences, denoted as \( (\hat{x}, \hat{y}, 1) \),

\[
[\hat{x}, \hat{y}, 1]^T = \tilde{h}^s [\tilde{x}, \tilde{y}, 1]^T = \tilde{h}^s h^s_i [x, y, 1]^T.
\]

Thus, the transformation \( \tilde{h}^s h^s_i \) conducts spatial alignment for frame \( i \) in sequence \( s \). Given the homography transformation of \( \tilde{h}^s \), as the cameras’ baseline increases, the dominant background plane is aligned, and the foreground may be affected by parallax in the final composite video.

**3.1.2 Spatial alignment of non-overlapping sequences** With freely panning cameras, it is likely that the backgrounds of sequences have no overlap, or the overall overlap is too small to reliably estimate the spatial alignment. One potential solution is to extrapolate the background images, and align the extrapolated images [23]. However, our experiments reveal that this is not reliable. First, extrapolation introduces many artifacts [3], or blurred areas [1, 23], leading to poor keypoint matching. Second, extrapolation in the horizontal direction, helps with alignment in the vertical direction, but leaves lots of ambiguity in horizontal alignment. Third, a rigid Euclidean transformation, as in [23], does not suffice for a proper background alignment.

On the other hand, how objects move across the sequences in the spatial world coordinate, irrespective of the temporal synchronization, provides hints for spatial alignment (Fig. 4). There is ambiguity in the exact spatial alignment, however, as more objects move across the sequences and in more diverse directions, the ambiguity is decreased.

To this end, we propose a spatial alignment algorithm for NOS that combines both aforementioned ideas. We first extrapolate the background images of all sequences. Then, we perform motion tracking to obtain trajectories of all keypoints in each sequence. By transforming the trajectories to the sequence coordinate using \( h^s_i \) and filtering out static trajectories, we collect moving object trajectories. We create motion tracks by matching moving object trajectories across sequences. Finally, we incrementally update the transformation applied to the background images to increase the motion track smoothness in the world coordinate,

![Figure 4. Spatial alignment of non-overlapping sequences using background extrapolation and smoothness of object trajectories.](image)
while maintaining the appearance consistency of extrapolated backgrounds in the overlap region. In essence, instead of relying only on extrapolated appearance, which is blurry and unreliable, we base our method on the extrapolation that movement of objects provides across the FOV of sequences.

**Motion tracking** We perform tracking in consecutive frames to form the trajectories. We prefer keypoint-based tracking for two reasons. 1) Object-based tracking requires detecting generic objects on each frame, which could be error-prone and inefficient. 2) Our experiments and also the analysis in [17] reveal that optical flow-based tracking such as dense trajectories [30] leads to spurious motion trajectories around the motion boundaries. We use SURF [5] keypoint detection and description. To detect newly emerging objects, we start tracking all the keypoints on frame $i$ that have no corresponding matches from frame $i-1$.

Denote the $j$th trajectory in sequence $s$ as $P_j^s = [x_j^s, y_j^s, t_j^s]$, where $x_j^s$ and $y_j^s$ are the frame coordinates, and $t_j^s$ is the time stamp. To handle sequences at different frame rates, $t_j^s$ should be the absolute time unit such as milliseconds not frame number. We then compute the trajectory $P_j^s$ in the sequence coordinate via $h_j^s$. In this coordinate, trajectories of moving and stationary keypoints are easily distinguishable, as sequence coordinates of static objects remain constant over time (Fig. 3(d), bold vs. dashed lines). Denoting the trajectory length as $t_j^s$ and width and height of the sequence as $w^s$ and $h^s$, we omit stationary trajectories if,

$$\frac{1}{t_j^s} \sum_{k=1}^{t_j^s} \left( \frac{|\tilde{x}_{j,k}^s - x_{j,k+1}^s|}{w^s} + \frac{|\tilde{y}_{j,k}^s - y_{j,k+1}^s|}{h^s} \right) < \tau_1,$$  

where $\tau_1$ is a threshold for the total displacement of the tracked object, and $\tilde{x}_{j,k}^s$ and $\tilde{y}_{j,k}^s$ denote the $k$th element in the vectors $x_j^s$ and $y_j^s$, respectively.

**Creating motion tracks** We describe each moving object trajectory $j$ of sequence $s$ with two SURF descriptors, one for the keypoint starting the trajectory ($S_j^s$) and one for the keypoint ending it ($E_j^s$). To match two trajectories $j$ and $k$ from sequences $s_1$ and $s_2$, a classical keypoint matching algorithm [18] is used to match all 4 combinations of keypoints, i.e., $(S_j^{s_1}, S_k^{s_2})$, $(E_j^{s_1}, E_k^{s_2})$, $(S_j^{s_1}, E_k^{s_2})$ and $(E_j^{s_1}, S_k^{s_2})$, and the minimum distance decides a match. This can achieve more robustness against view point variation, as the nearby keypoints of the trajectories (in the world coordinate) will be the deciding factor in trajectory matching. We call each set of matched trajectories a track, denoted by $\Pi_k$. For simplicity of notation, we assume that the trajectories within a track have been re-indexed such that $\Pi_k = \{ \tilde{P}_k^s : s \in [1, S] \}$. For a certain sequence $s, \tilde{P}_k^s$ might be empty, i.e., no trajectories from this sequence is part of the track $\Pi_k$. Note that not all the moving object trajectories should be matched to form tracks, due to noisy trajectories or objects with non-smooth motion path. Sec. 3.2.2 presents a method to remove non-smooth trajectories.

**Spatial alignment formulation** For simplicity, we discuss the alignment of 2 sequences, as more sequences may be aligned in the same manner, sequentially. We set $\tilde{H}^1 = I_{3 \times 3}$ and use $p$ for $\tilde{H}^2$ to avoid cluttered equations. Given $N$ tracks indexed by $i$, and extrapolated backgrounds $B^1$ and $B^2$, the goal is to find a transformation $p$ which maps $B^2$ to $B^1$, such that the extrapolated background are similar in the overlap region $O(p)$ and trajectories of sequence 2 reside on the extension of trajectories in sequence 1, in $\tilde{x} - \tilde{y}$ coordinate. For image extrapolation, we use PatchMatch algorithm [3]. Then, we formulate the optimization problem (Fig. 4).

$$\min_p \sum_{x \in O(p)} \left[ B^2(\mathcal{W}(\tilde{x}; p)) - B^1(\tilde{x}) \right]^2 + \beta \sum_i e_i(p)^T e_i(p),$$

where $\mathcal{W}(\tilde{x}; p)$ warps $\tilde{x}$ to the world coordinate by transformation $p$, and $e_i(p)$ represents how far trajectory $i$ of sequence 2 is from spatial extension of matching trajectory in sequence 1. The first term in Eqn. 6 is similar to Lucas-Kanade algorithm [2], operated only in the overlapping area. To define $e_i(p)$, we fit a line, which works better than fitting polynomials in our experiments, to the $i$th trajectory in sequence 1 (in the sequence coordinate), denoted by $f_i(x)$. The vector $e_i(p)$ collects the $y$-distance between each point on the $i$th trajectory in sequence 2, after warped by $p$, and the fitted line,

$$e_i(p) = \tilde{w}_i \text{(-)} f_i(\tilde{w}_i) \text{,(7)}$$

where $\tilde{w}_i = [W_i(x_{i,1}, y_{i,2}; p)]$ and $\tilde{w}_i \text{(-)} = [W_i(\tilde{x}_{i,1}, \tilde{y}_{i,2}; p)]$ are the warped $\tilde{x}$ and $\tilde{y}$-coordinates of the $i$th trajectory in sequence 2 to the world coordinate.

The optimization problem is solved by taking the Taylor expansion around $p$ and finding the increment $\Delta p$ by,

$$\text{argmin}_{\Delta p} \sum_{x \in O(p)} \left[ B^2(\mathcal{W}(\tilde{x}; p)) - B^1(\tilde{x}) \right]^2 + \beta \sum_i [e_i(p) + J_e \Delta p]^T e_i(p) + J_e \Delta p + \alpha \Delta p^T \mathcal{I} \Delta p,$$

where $S_B = \nabla B^2 \frac{\partial \mathcal{W}}{\partial p}$ is the steepest decent image, $J_e = \frac{\partial e_i(p)}{\partial p}$, and $\Delta p^T \mathcal{I} \Delta p$ is a regularization term penalizing some special changes on $\Delta p$ controlled by $\mathcal{I}$ and a positive constant $\alpha$. By setting $\mathcal{I} = diag([0, 0, 0, 0, 0, 1, 0, 0, 0, 0])$, we penalize large changes on translation elements of $\Delta p$, so that frames are first aligned by warping them rather than translating them. Based on our experiments, this leads to more stable results. The solution to Eqn. 8 is,

$$\Delta p = H^{-1} \left( \sum_{x \in O(p)} S_B^T \left[ B^1(\tilde{x}) - B^2(\mathcal{W}(\tilde{x}; p)) \right] - \beta \sum_i J_e e_i(p) \right),$$

(9)
in which \( H = \sum_{x \in \mathcal{O}(p)} S^T B S + \beta \sum_{1} J^T J + \alpha I \).

We initialize the algorithm by setting the sequences side by side (spatially) with the two possible layouts, and use the alignment result of the layout with lower final cost.

### 3.2. Temporal alignment

NOS are assumed to have moving objects, without which the temporal alignment is neither necessary nor possible. Given moving objects and spatial alignment results, the temporal alignment of NOS amounts to estimating when an object will appear in FOV of another camera, after it moves out of the current FOV. If both cameras observe the object’s motion at the same time, the problem is easier. For this purpose, we create motion tracks as discussed in Sec. 3.1. Then, we estimate the temporal offset between sequences such that trajectories from the identical object follow a continuous path in \( \tilde{x} - t \) and \( \tilde{y} - \tilde{t} \) coordinates, i.e., the motion tracks are smooth. Since not all trajectories are due to moving objects, we filter motion trajectories with non-smooth paths, before matching trajectories.

#### 3.2.1 Estimation of temporal offset

Given the collection of tracks, the objective is to make each track a smooth curve, by shifting the temporal coordinates of the contributing trajectories appropriately (Fig. 5). For \( S \) sequences, \( \tilde{x} - \tilde{t} \) coordinate of trajectories forming the \( k \)th track is the vector \( \tilde{x}_k^1 \tilde{x}_k^2 \cdots \tilde{x}_k^N \). \( \tilde{y} - \tilde{t} \) coordinate of each track is defined similarly. We assume that by temporally shifting each sequence \( s \) for \( \Delta t_s \), the sequences are temporally aligned. To estimate \( \Delta t_s \), we fit a polynomial curve of degree \( m \) to time stamps versus \( \tilde{x} \) and \( \tilde{y} \)–coordinates of each track independently and estimate the time shifts, in order to achieve the lowest curve fitting error. Here, we discuss only the \( \tilde{x} - \tilde{t} \) curve, and the \( \tilde{y} - \tilde{t} \) curve is similar.

We denote the trajectory coordinates of sequence \( s \) and all the power terms of the polynomial time-space curve as

\[
\tilde{x}_k^{s(m)} = [1_k \tilde{x}_k^1 \tilde{x}_k^2 \cdots \tilde{x}_k^m],
\]

where \( 1_k \) is a \( l_k \)-dim vector of all ones, and \([\cdot ]^m \) denotes an element-wise power operation. For the track \( k \), all the required terms of the polynomial space-time curve are collected in a matrix \( \tilde{X}_k \) of size \( \sum_k l_k \times (m + 1) \), and all the time stamps in a vector \( \tilde{T}_k(\Delta t) \) of length \( \sum_k l_k \).

We denote the coefficients of the \( k \)th polynomial curve fitting to the \( k \)th track as \( c_k = [c_q] q \in \{0, ..., m\} \). We can estimate the coefficients by solving a linear system, \( \text{arg} \min_{c_k} ||\tilde{T}_k(\Delta t) - \tilde{X}_k c_k|| \). Since all tracks share the same \( \Delta t \), we can efficiently solve for all tracks jointly,

\[
c^*, \Delta t^* = \text{argmin}_{c, \Delta t} ||\tilde{T}(\Delta t) - \tilde{X} c||.
\]

in which

\[
\tilde{X} = \begin{bmatrix} \tilde{x}_1 & 0 & \cdots & 0 \\ 0 & \tilde{x}_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \tilde{x}_K \end{bmatrix}, \tilde{T}(\Delta t) = \begin{bmatrix} \tilde{T}_1(\Delta t) \\ \tilde{T}_2(\Delta t) \\ \vdots \\ \tilde{T}_K(\Delta t) \end{bmatrix}, c = \begin{bmatrix} c_1 \\ c_2 \\ \vdots \\ c_K \end{bmatrix}.
\]

Here, \( \tilde{X} \) is a \( N_K \times K(m + 1) \) matrix where \( N_K = \sum_k \sum_s l_k^2 \) is the count of keypoints in all \( K \) tracks. We alternatively estimate \( c \) and \( \Delta t \), until the change in \( \Delta t \) is negligible. We first estimate \( c \), with fixed \( \Delta t \). Since \( N_K \gg K(m + 1) \), this linear system is over-constrained for \( c \). We solve \( c \) by Orthogonal-triangular decomposition, which is numerically more accurate than the pseudo inverse of \( \tilde{X} \). Then, for a given \( c^* \), we set \( \Delta t_s \) as the average of residuals from the keypoints in trajectories belonging to sequence \( s \),

\[
\Delta t_s = -\frac{1}{N_s} (\tilde{T} - \tilde{X} c^*)^T I_s,
\]

where \( I_s \) is a binary indicator vector with an element equal to 1 if the corresponding row in \( \tilde{T} \) comes from a trajectory in sequence \( s \), and \( N_s = \| I_s \| \) is the count of such rows.

#### 3.2.2 Motion trajectory filtering

As mentioned before, not all trajectories are resulted from object motion with a smooth path. In other words, some trajectories might be due to noise in keypoint locations while the camera moves. So, before matching trajectories across sequences and collecting them to a track, we filter out the trajectories that cannot be well approximated with a smooth path, by fitting the order-\( m \) polynomial to the trajectory,

\[
c_k^{**} = \text{argmin}_{c_k} \| \tilde{t}_k - \tilde{x}_k^{s(m)} c_k^{**} \|_2,
\]

and thresholding the total fitting residual to remove non-smooth trajectories, i.e., \( \frac{1}{l_k} \| \tilde{t}_k - \tilde{x}_k^{s(m)} c_k^{**} \|_1 < \tau_2 \).

### 4. Experimental results

In this section, we present the experimental setup and both quantitative and qualitative results. Note that since
NOS is a novel scenario for spatio-temporal alignment of sequences, there is no prior work for comparison. We set $\beta = 100$, $\alpha = 10^3$, $m = 3$ for the temporal curve fitting step, and $\tau_1 = 0.03$ and $\tau_2 = 0.15$ for trajectory filtering. **Dataset** Given that there is no public dataset in this new scenario, we collect a NOS dataset including ten real-world sequence sets, and five synthetic sequence sets. Real sets are captured by two or three people using handheld smartphones with the distance between the cameras, i.e. baseline, as shown in Table 1. Synthetic sets provide sequences for which the ground truth result are exactly known, and are created by taking a sequence and cropping out two spatio-temporal tubes from the 3D sequence volume. This emulates the case of independently panning cameras with almost identical optical centers. To simulate a freely panning camera and hand shake, the spatial region used for each tube at each frame has a fixed size of $640 \times 360$ pixels, but the region location has an additive zero-mean Gaussian noise. Also, if the original video is stationary, the regions shift in $x$-direction to create a pan-like effect. The dataset is available at http://cvlab.cse.msu.edu/project-sequence-alignment.html.

**Qualitative results** Figure 6 presents the reconstructed backgrounds along with image extrapolation results. Further, it is shown how the backgrounds are transformed so that moving object trajectories have smooth path. Figure 7 shows the alignment results for five sets of real sequences. The first two sets include sequences with some overall spatial overlap while the rest have no/minimal spatial overlap. For each set, two or three sample frames with moving objects are shown, at the time shift estimated by the proposed algorithm. Also, keypoint trajectories from both sequences in the world coordinate after spatio-temporal alignment are shown. Trajectories of moving objects have considerable extent in the $x-$direction, whereas trajectories of stationary objects are roughly parallel to $t-$axis. Finally, the two input frames are warped to the world coordinate to make a composite image. Although the input frames may not have direct overlap, perceived continuity of the scene and also relative location of the moving objects, demonstrate capabilities of the proposed algorithm and the application scenarios. Note that in all test sequences cameras move freely and independently, as shown by the range of trajectories in the world coordinate. For the case of “R7” in Fig. 7, the sequences are non-overlapping, but only a person is tracked moving across the FOVs. Thus, as shown in this figure, spatial alignment has some error, which consequently affects the accuracy of the temporal alignment.

Figure 8 represents a synthetic set where two sequences are created from a video of a car accident. The two cropped frames after spatio-temporal alignment are shown in a composite image and for comparison, the corresponding frame from the original video is also shown, demonstrating the accuracy of spatio-temporal alignment.

**Quantitative results** To quantitatively evaluate the proposed algorithm, we compare the alignment errors with the ground truth. For the case of synthetic sets, the original video from which the synthetic sequences are cropped, provides the ground truth location of the center points of the cropped frames. We measure the spatial location error of each aligned frame w.r.t. the ground truth location and report sum of absolute errors in $x$ and $y$-direction, averaged over the sum of the length of the sequences, as the spatial alignment error. Also, since we create the synthetic sequences, the ground truth time shift is known. For real sets, when the input frames do not have overlap, quantifying the spatial error is not feasible. For quantification of temporal alignment, we manually align the sequences by relying on visual cues such as body pose, moving object location relative to background landmarks, and consistency of appearance of moving objects in the composite image. Table 1 provides the quantified temporal and spatial errors. As may be observed, temporal alignment works well even when the camera baseline distance increases, although the final consolidated result may suffer from parallax.

<table>
<thead>
<tr>
<th>Sequence ID</th>
<th>R1</th>
<th>R2</th>
<th>R3</th>
<th>R4</th>
<th>R5</th>
<th>R6</th>
<th>R7</th>
<th>R8</th>
<th>R9</th>
<th>R10</th>
<th>S1</th>
<th>S2</th>
<th>S3</th>
<th>S4</th>
<th>S5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Camera baseline (meter)</td>
<td>1</td>
<td>3</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>5</td>
<td>10</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Temporal error (sec.)</td>
<td>0.13</td>
<td>0.07</td>
<td>0.07</td>
<td>0.13</td>
<td>0.07</td>
<td>0.07</td>
<td>0.10</td>
<td>0.03</td>
<td>0.07</td>
<td>0.07</td>
<td>0.07</td>
<td>0.03</td>
<td>0.07</td>
<td>0.03</td>
<td>0.07</td>
</tr>
<tr>
<td>Spatial error (pixel)</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>2</td>
<td>3</td>
<td>7</td>
<td>2</td>
<td>2</td>
</tr>
</tbody>
</table>

Table 1. Temporal and spatial alignment error in seconds and pixels, respectively, for real (R) and synthetic (S) sequences.

![Figure 6](http://cvlab.cse.msu.edu/project-sequence-alignment.html)
Figure 7. Each row shows spatio-temporal alignment results on a set of real NOS. For each sequence, input frames at the estimated time shift and trajectories of moving objects in the world coordinate are shown. The input frames are transformed to the world coordinate to make a composite image.

Computational cost The main computational cost of the proposed algorithm comes from TRGMC. On average, for a video of 15-second long, we spend 450 seconds on TRGMC and background reconstruction, using a PC with an Intel i5-3470@3.2GHz CPU, and 8GB RAM. Spatial alignment is independent of sequence length and takes \( \sim 162 \) seconds on average for NOS. Finally, temporal alignment takes about 13 seconds on average over the dataset.

Limitations The proposed algorithm is a first step for generating panoramic videos in the challenging scenario of NOS. While this work relaxes many common assumptions of prior works, violation of some assumptions, especially existence of moving objects with a trajectory that spans FOVs of multiple cameras, results in alignment failures. Furthermore, when relying on non-rigid or articulated moving objects for alignment, many keypoints are not tracked long enough due to change of appearance, making alignment difficult. Also, in this case, matching trajectories among different sequences is less reliable. Since our algorithm is independent of the type of tracker, other tracking algorithms can be investigated in the future. Finally, alignment of non-overlapping background images suffers from ambiguity and is error prone, although the proposed algorithms makes use of available cues to conduct this task.

5. Conclusions

We proposed an algorithm for spatio-temporal alignment of sequences, referred to as non-overlapping sequences (NOS), from freely panning cameras whose FOVs might not even observe a common region over progression of time. This new scenario of video alignment is useful in reconstructing events, incidents, or crime scenes from multiple amateur-captured sequences, or creation of panoramic videos from cooperative users via handheld cameras. The spatial alignment of our algorithm relies on reconstructing background for each sequence and aligning the backgrounds. When backgrounds are non-overlapping, the spatial alignment uses clues from smoothness of moving objects’ paths and coherent appearance of background after image extrapolation. Smoothness of trajectory of moving objects is also utilized as a clue for temporal alignment. Our experiments demonstrate capabilities of the proposed method, despite the challenging scenario of NOS.

Acknowledgement This work was partially supported by TechSmith Corporation.
References


