Efficient Non-consecutive Feature Tracking for Structure-from-Motion

Guofeng Zhang[1], Zilong Dong[1], Jiaya Jia[2], Tien-Tsin Wong[2], and Hujun Bao[1]

[1] State Key Lab of CAD&CG, Zhejiang University
{zhangguofeng,zldong,bao}@cad.zju.edu.cn
[2] The Chinese University of Hong Kong
{leojia,ttwong}@cse.cuhk.edu.hk

Abstract. Structure-from-motion (SfM) is an important computer vision problem and largely relies on the quality of feature tracking. In image sequences, if disjointed tracks caused by objects moving in and out of the view, occasional occlusion, or image noise, are not handled well, the corresponding SfM could be significantly affected. In this paper, we address the non-consecutive feature point tracking problem and propose an effective method to match interrupted tracks. Our framework consists of steps of solving the feature 'dropout' problem when indistinctive structures, noise or even large image distortion exist, and of rapidly recognizing and joining common features located in different subsequences. Experimental results on several challenging and large-scale video sets show that our method notably improves SfM.

1 Introduction

Large-scale 3D reconstruction [1,2,3,4] is a very active research topic and finds many practical applications in, for example, Google Earth and Microsoft Virtual Earth. Recent work essentially relies on the SfM algorithms [5,6,7,4] to automatically estimate 3D features given the input of image or video collections.

Compared to images, videos usually contain denser geometrical and structural information, and are the main source of SfM in the movie and commercial industry. A common strategy for video SfM estimation is by employing feature point tracking [8,9,10,11], which takes care of the temporal relationship among frames. It is also a basic tool for solving a variety of computer vision problems, such as automatic camera tracking, video matching, and object recognition.

In this paper, we discuss two critical and non-trivial problems of feature point tracking, which could seriously handicap SfM especially for large-scale scene modeling, and propose novel methods to address them. One problem is the high vulnerability of feature tracking to object occlusions, illumination change, noise, and large motion, which easily causes occasional feature dropout and distraction. This problem makes developing a robust feature tracking system with the input of long sequences very challenging.

The other problem is the inability of sequential feature tracking to cope with feature matching over non-consecutive subsequences. To our best knowledge, this

K. Daniilidis, P. Maragos, N. Paragios (Eds.): ECCV 2010, Part V, LNCS 6315, pp. 422–435, 2010.

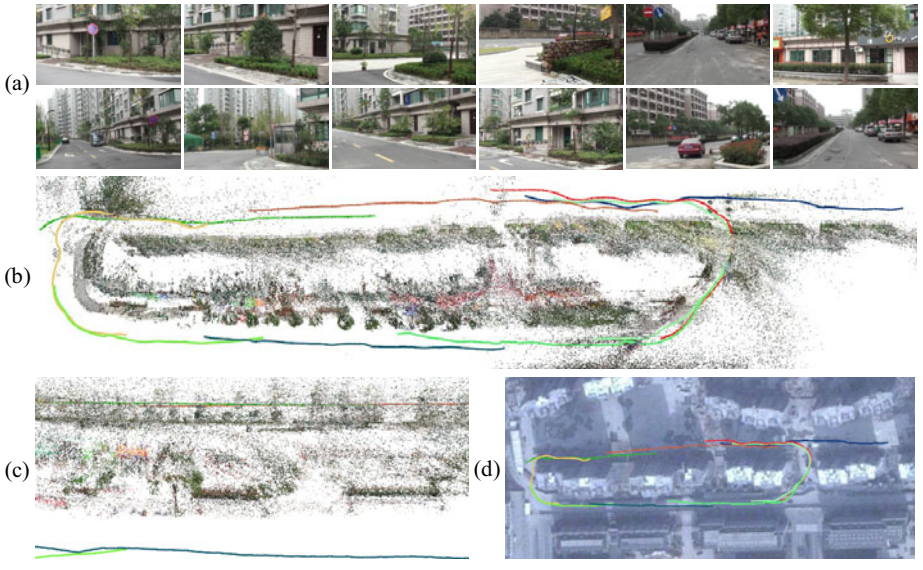


Fig. 1. "Street" example. (a) The snapshots of the input videos containing around 23,000 frames. (b) With the matched feature tracks, we register many 3D points and camera trajectories in a large 3D system. The camera trajectories are differently color-coded. (c) Close-up of the recovered trajectories and 3D points. (d) Superimposing the recovered camera trajectories onto a satellite image from Google Earth.

impact has not yet been thoroughly studied in existing literatures. A typical scenario is that the tracked object moves out and then re-enters the field-of-view of the camera. This yields two discontinuous subsequences containing the same object. Although there are common features in the two subsequences, they cannot be matched and included in a single track using conventional tracking methods. Addressing this issue can alleviate the drift problem of SfM, which in turn benefits high-quality 3D reconstruction as demonstrated in our experimental results. A naïve solution to this problem is to exhaustively search all features. But this consumes much unnecessary computation as many temporally far away frames simply share no content.

Our new feature tracking framework efficiently addresses the above problems in two phases, namely *consecutive point tracking* and *non-consecutive track matching*. We demonstrate their significance for SfM estimation using a few challenging videos. *Consecutive point tracking* detects and matches invariant features distributed over consecutive frames. A new two-pass matching strategy is proposed to greatly increase the matching rate of the detected invariant features and extend the lifetime of the tracks. Then in the *non-consecutive track matching* phase, by rapidly computing a matching matrix, a set of disjointed subsequences with overlapping content can be detected. The common feature tracks scattered over these subsequences can also be reliably matched.

Our method can naturally handle feature tracking in multiple videos and register sequences in a large-scale 3D system. Fig. 1 shows a challenging example, which contains 9 videos (about 23,000 frames in total) in a large-scale scene (500 meters long). With our method, a set of long and accurate feature tracks are efficiently obtained. The computation time is only 1.3 seconds per frame with our software implementation (single working thread). Our system also greatly improves SfM by registering videos in a 3D system, as shown in Fig. 1(b). The accuracy of SfM is verified by superimposing the recovered camera trajectories onto a satellite image from Google Earth, as shown in Fig. 1(d). Please refer to our supplementary video [1] for the complete results.

2 Related Work

For video tracking, sequential matchers are used for establishing correspondences between consecutive frames. Kanade-Lucas-Tomasi (KLT) tracker [8,9,12] is widely used for small baseline matching. Other advanced methods [11,13,14,15] detect image local features and match them with descriptors.

Both the KLT tracker and invariant feature algorithms depend on modeling feature appearance, and can be distracted by occlusion, similar structures, noise, and image distortion. Generally, sequential matchers cannot match non-consecutive frames under large image transformation. Scale-invariant feature detection and matching algorithms [11,16,2] are effective in recognizing panoramas and in matching wide-baseline images. But they are not easy to be used in consecutive point tracking due primarily to the global indistinctiveness and feature dropout problems in matching, which yield many short tracks.

In addition, invariant features are sensitive to large image distortion. Although variations, such as ASIFT [17], can improve the feature matching performance under substantial viewpoint change, computational overhead significantly increases owing to exhaustive simulation. In this paper, we propose a novel two-pass matching method to solve this problem.

There is work using invariant features for object and location recognition in images/videos [18,19,20,21,22]. These methods typically use the bag-of-words technique to perform global localization and loop-closure detection in an image classification scheme. To reduce the matching ambiguity, they generally suppress indistinctive features. This operation is not suitable for producing long and accurate point tracks.

Engels et al. [23] propose integrating wide-baseline local features with the tracked features to improve SfM. The method creates small and independent submaps over short periods of time and links them together via feature recognition. This approach generally cannot produce many long and accurate point tracks. Only short tracks are found insufficient for drift-free SfM estimation in our experiments. In comparison, our method is effective in high-quality point

[1] The supplementary video can be downloaded from http://www.cad.zju.edu.cn/home/gfzhang/

track estimation. We also address the ubiquitous nondistinctive feature matching problem in dense frames, and utilize track descriptors, instead of the feature descriptors, to reduce computation redundancy.

3 Our Approach

Given a video sequence $\hat{I}$ with n frames, $\hat{I} = \{I_t | t = 1, ..., n\}$, the objective of our feature tracking method is to extract and match features in all frames in order to form a set of *feature tracks*. A feature track $\mathcal{X}$ is defined as a series of feature points in images: $\mathcal{X} = \{\mathbf{x}_t | t \in f(\mathcal{X})\}$, where $f(\mathcal{X})$ denotes the frame set spanned by track $\mathcal{X}$. Each invariant feature $\mathbf{x}_t$ in frame t is associated with an appearance descriptor $\mathbf{p}(\mathbf{x}_t)$ [11] and we denote all description vectors in a feature track as $\mathcal{P}_\mathcal{X} = \{\mathbf{p}(\mathbf{x}_t) | t \in f(\mathcal{X})\}$.

Table 1. Overview of Our Method

1. Detect invariant features over the entire sequence.
2. **Consecutive point tracking** (Section 4):
2.1 Match features between consecutive frames with descriptor comparison.
2.2 Perform the second-pass matching to extend track lifetime.
3. **Non-consecutive track matching** (Section 5):
3.1 Use hierachical k-means to cluster the constructed invariant tracks.
3.2 Estimate the matching matrix with the grouped tracks.
3.3 Detect overlapping subsequences and join the matched tracks.

Our method has two main steps, i.e., consecutive point tracking and non-consecutive track matching. The algorithm overview is given in Table 1.

Step 2 in Table 1 suppresses the influence of image noise and distortion in feature tracking, which usually cause spurious feature appearance variation and feature dropout in matching. We locate missing features (as well as the unmatched ones) by a constrained spatial search with planar motion segmentation as described in Section 4.2.

Step 3 is a non-consecutive track matching process. It first uses a hierarchical K-means method to cluster the obtained track descriptors. Based on it, overlapping confidence among non-consecutive frames is measured using a matching matrix, which helps robustly join common features in subsequences. This step is described in Section 5.

4 Two-Pass Matching for Consecutive Tracking

In the first place, we use the SIFT algorithm [11] to detect and describe image features. We extract SIFT features from all frames in the input sequence and match them among temporally adjacent frames. The matched features constitute sequential feature tracks. Note that previous KLT methods typically detect

features in the first frame and then track them down consecutively without the invariant feature descriptor constraint. Our method, contrarily, obtains not only a set of feature tracks, but also descriptors to represent tracks, which avail further non-consecutive track matching.

We propose a *two-pass matching* strategy to efficiently reduce false matches caused by structure similarity and feature dropout due to image noise and distortion. The *first-pass matching* is used to obtain high-confidence matches. In the second pass, tracks are extended with planar motion segmentation and constrained spatial search.

4.1 First-Pass Matching by Descriptor Comparison

In this section, we discuss tracking a feature $\mathcal{X}$ from I_t to I_{t+1}. It can be generalized to tracks spanning multiple frames. An invariant feature in I_t is denoted as $\mathbf{x}_t$ with descriptor $\mathbf{p}(\mathbf{x}_t)$. To determine if there is a corresponding feature $\mathbf{x}_{t+1}$ with descriptor $\mathbf{p}(\mathbf{x}_{t+1})$ in I_{t+1}, we employ the 2NN heuristic proposed by Lowe [11].

Specifically, we search for the two nearest neighboring features of $\mathbf{x}_t$ in I_{t+1} with respect to the Euclidean distance of the descriptor vectors and denote them as $\mathcal{N}_1^{t+1}(\mathbf{x}_t)$ and $\mathcal{N}_2^{t+1}(\mathbf{x}_t)$. Their corresponding descriptor vectors are denoted as $\mathbf{p}(\mathcal{N}_1^{t+1}(\mathbf{x}_t))$ and $\mathbf{p}(\mathcal{N}_2^{t+1}(\mathbf{x}_t))$ respectively. The matching confidence between $\mathbf{x}_t$ and $\mathcal{N}_1^{t+1}(\mathbf{x}_t)$ is defined as

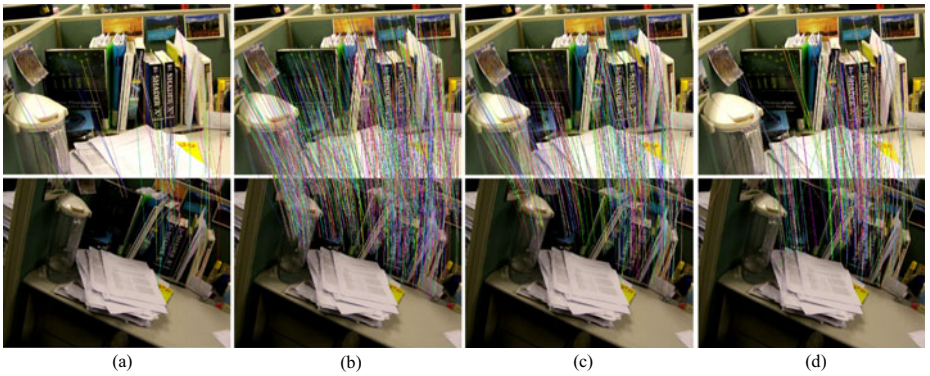
$$c = \frac{||\mathbf{p}(\mathcal{N}_1^{t+1}(\mathbf{x}_t)) - \mathbf{p}(\mathbf{x}_t)||}{||\mathbf{p}(\mathcal{N}_2^{t+1}(\mathbf{x}_t)) - \mathbf{p}(\mathbf{x}_t)||}, \tag{1}$$

where c measures the global distinctiveness of one feature $\mathbf{x}_t$ with respect to the ratio of the smallest feature distance and the second smallest one. If $c < \varepsilon$, we assign $\mathbf{x}_{t+1} = \mathcal{N}_1^{t+1}(\mathbf{x}_t)$ and mark these detected features as *globally distinctive*. In our experiments, ε is set to 0.7.

However, this metric is easily interfered by image noise, repeated structures, and image distortion, which make it difficult to find matches for some features even in the adjacent frames. This common problem usually results in breaking a long track into several short ones. One example is shown in Fig. 2. Given an image pair, we detect $1,246$ features. Only 50 features can be matched by descriptor comparison, as shown in Fig. 2(a). In the next step, we propose a spatial search method to help identify more matches.

4.2 Second-Pass Matching by Planar Motion Segmentation

With a few high-confidence matches in neighboring frames (I_t, I_{t+1}) computed in the first step, we use the RANSAC algorithm [24] to estimate the fundamental matrix $F_{t,t+1}$ and remove outliers. For those unmatched features, it is possible to search for their correspondences along the conjugate epipolar line $l_{t,t+1}(\mathbf{x}_t) = F_{t,t+1}\mathbf{x}_t$. However, if significant image distortion exists, naive



(a) (b) (c) (d)

Fig. 2. Feature matching comparison. (a) First-pass matching by SIFT descriptor comparison. There are 1,246 features detected; but only 50 matches are found. (b) Second-pass matching by planar motion segmentation. 717 new matches are included; but quite a number of them are outliers. (c) The final result with our outlier rejection. A total of 343 matches are retained. (d) The matching result by ASIFT [17]. 220 matches are found.

Algorithm 1. Second-Pass Matching

1. Use the inlier matches to estimate a set of homographies $\{H^k_{t,t+1}|k = 1, ..., N\}$ by Algorithm 2, and then use them to obtain a set of rectified images $\{\hat{I}^k_t|k = 1, ..., N\}$.
2. **for** each unmatched feature $\mathbf{x}_t$ in I_t **do**
   for $k = 1, ..., N$ **do**
   Find the best match $\mathbf{x}^k_{t+1}$ by minimizing (2) with $H^k_{t,t+1}$.
   end for
   Find the best match $\mathbf{x}^i_{t+1}$ among $\{\mathbf{x}^k_{t+1}|k = 1, ..., N\}$ which minimizes $S^k_{t,t+1}(\mathbf{x}_t)$. Further refine $\mathbf{x}^i_{t+1}$ to $\mathbf{x}^*_{t+1}$ with the KLT tracking. If $||\mathbf{x}^*_{t+1} - \mathbf{x}^i_{t+1}||$ is large, reject this match.
   end for

window-based matching becomes unreliable. Also, an exhaustive search is time-consuming and ambiguous with many potential correspondences. To address these problems, we propose a segmentation-based method (sketched in Algorithm 1) to robustly identify missing matches.

We base our method on the observation that many feature points undergo similar motion. This allows computing inlier matches to estimate a set of homographies $\{H^k_{t,t+1}|k = 1, ..., N\}$, which represent possible local image transformation, as described in Algorithm 2. We then rectify images with their homographies. This scheme is similar to that of [25] where a set of dominant scene planes are extracted to generate a piecewise planar depth map. For an unmatched feature in image I_t, if its transformation towards I_{t+1} is coincident with any of these homographies after rectification, a match in I_{t+1} can possible be found. Incorrect

Algorithm 2. Planar Motion Segmentation

1. Put all matches into a set Ω.
2. For $k = 1, ..., N_{\max}$, $\quad \%N_{\max}$ is the maximum number of the homographies.
   2.1 Use RANSAC to estimate homography $H^k_{t,t+1}$ that has the maximum inliers.
   2.2 Remove the inliers from Ω. If the size of Ω is small enough, stop; otherwise, continue.

homographies are unlikely to yield high-confidence matches. To handle illumination change, we estimate the global illumination variation $L_{t,t+1}$ between images I_t and I_{t+1} by computing the average intensity ratio between the matched features.

With the image transformation $H^k_{t,t+1}$, we rectify I_t to $\hat{I}^k_t$ such that $\hat{I}^k_t = H^k_{t,t+1}(L_{t,t+1} \cdot I_t)$. Correspondingly, $\mathbf{x}_t$ in image I_t is rectified to $\hat{\mathbf{x}}^k_t$ where $\hat{\mathbf{x}}^k_t \sim H^k_{t,t+1}\mathbf{x}_t$ in $\hat{I}^k_t$. If $\hat{\mathbf{x}}^k_t$ largely deviates from the epipolar line (i.e., $d(\hat{\mathbf{x}}^k_t, l_{t,t+1}(\mathbf{x}_t)) > 5.0$), we reject $H^k_{t,t+1}$ since it does not describe the motion of $\mathbf{x}_t$ well. Otherwise, we search for the match along the epipolar line by minimizing the matching cost

$$S^k_{t,t+1}(\mathbf{x}_t) = \min_{\mathbf{x}' \in l_{t,t+1}(\mathbf{x}_t)} \sum_{\mathbf{y} \in W} ||\hat{I}^k_t(\hat{\mathbf{x}}^k_t + \mathbf{y}) - \hat{I}_{t+1}(\mathbf{x}' + \mathbf{y})||^2, \tag{2}$$

where W is a 11×11 matching window, and $\mathbf{x}'$ is in the local searching area where $||\hat{\mathbf{x}}^k_t - \mathbf{x}'|| < r$ (usually $r = 15$ in our experiments). The best match is denoted as $\mathbf{x}^k_{t+1}$. With the set of homographies $\{H^k_{t,t+1}|k = 1, ..., N\}$, we can find several matches $\{\mathbf{x}^k_{t+1}|k = 1, ..., N\}$. Only the best one $i = \min_k S^k_{t,t+1}(\mathbf{x}_t)$ is kept.

In case the feature motion cannot be described by all of the homographies or the feature correspondence is indeed missing in the other image, the computed match is actually an outlier. Simply applying threshold $S^i_{t,t+1}(\mathbf{x}_t) < \tau$ cannot perform satisfactorily, as shown in Fig. 2(b). In addition, the best match may not strictly lie on the epipolar line due to estimation error. We adopt the following procedure to detect outliers.

Our strategy is to relax the epipolar geometry constraint and use the KLT method instead to locally search the best match $\mathbf{x}^*_{t+1}$. The intuition is that true correspondence produces the minimum matching cost locally; so searching with and without the epipolar constraint should return the same result. We thus calculate the distance between $\mathbf{x}^*_{t+1}$ and $\mathbf{x}^i_{t+1}$. If $||\mathbf{x}^*_{t+1} - \mathbf{x}^i_{t+1}||$ is large (over 3.0 in our experiments), $\mathbf{x}^i_{t+1}$ is considered as an outlier; or else, $\mathbf{x}^*_{t+1}$ is the correspondence of $\mathbf{x}_t$ and its descriptor is set to that of $\mathbf{x}_t$, i.e. $\mathbf{p}(\mathbf{x}^*_{t+1}) = \mathbf{p}(\mathbf{x}_t)$.

Applying this criterion effectively rejects most outliers, as shown in Fig. 2(c). Compared to ASIFT [17], our method adaptively estimates a set of dominant homographies, without exhaustively simulating all views. So the computation is much less. Besides, it is hard to apply ASIFT to consecutive point tracking because features, after the simulation of viewpoint change, are no longer the original SIFT ones. Our method has no such problem.

Most of the above steps can be performed hierarchically to yield high efficiency. For a pair of images (resolution 640×480) with $1,196$ features and 4 estimated homographies, the second-pass matching only requires 0.4 second with our software implementation.

The two-pass matching can also produce many long tracks as shown in our supplementary video. Each track has a group of description vectors, denoted as $\mathcal{P}_{\mathcal{X}} = \{\mathbf{p}(\mathbf{x}_t)|t \in f(\mathcal{X})\}$. These descriptors must be similar to each other in the same track due to the matching criteria. We compute an average of them and denote it as *track descriptor* $\mathbf{p}(\mathcal{X})$. It is used in the following non-consecutive track matching.

5 Non-consecutive Track Matching

Given the invariant feature information encoded in tracks, we detect and match features scattered over non-consecutive frames. The following process consists of two main phases, namely *matching matrix estimation* and *non-consecutive track matching*.

5.1 Fast Matching Matrix Estimation

To allow non-consecutive track matching, we first estimate a matching matrix for the whole sequence to describe the frame overlapping confidence. Obviously, exhaustive all-to-all frame matching is computationally expensive especially for long sequences. We propose fast estimation of the matching confidence among different frames with regard to the track descriptors.

In [26], extracted image descriptors are used to construct a vocabulary tree for fast image indexing. Note that our consecutive point tracking has already clustered matchable features in sequential frames. Instead of locating similar features, we propose constructing a vocabulary tree based on track descriptors for finding similar tracks. This approach can not only significantly reduce the size of the tree, but improve the matching accuracy among non-consecutive frames as well.

We use a hierarchical K-means approach to cluster the track descriptors. The root cluster contains all the descriptors. It is partitioned into b subgroups by the K-means method. Each sub-cluster consists of the descriptor vectors closest to the center. The same procedure is recursively applied to all subgroups and terminates when the variance of all descriptors in a final (leaf) cluster is less than a threshold. The leaf clusters provide a detailed partition of all tracks. We measure the *overlapping confidence* between any two frames based on the descriptor similarity (depicted in Algorithm 3). The scores are stored in the matching matrix M, which is with size $n \times n$. n is the number of all frames. The confidence value between images I_i and I_j is saved in $M(i, j)$.

All elements in M are first initialized to zeros. In each iteration of Algorithm 3, $M(i, j)$ is increased by 1 if two features respectively in frames i and j are in the same leaf node of the tree. With the objective of non-consecutive frame matching,

Algorithm 3. Matching Matrix Estimation

1. Initialize M as a zero matrix.
2. For each track cluster G_k $(k = 1, ..., K)$, % K is the number of the final clusters.
   For each track pair $(\mathcal{X}_u, \mathcal{X}_v)$ in G_k, if $f(\mathcal{X}_u) \cap f(\mathcal{X}_v) = \emptyset$,
   For any $i \in f(\mathcal{X}_u)$ and $j \in f(\mathcal{X}_v)$,
   $M(i, j)$ += 1,
   $M(j, i)$ += 1.

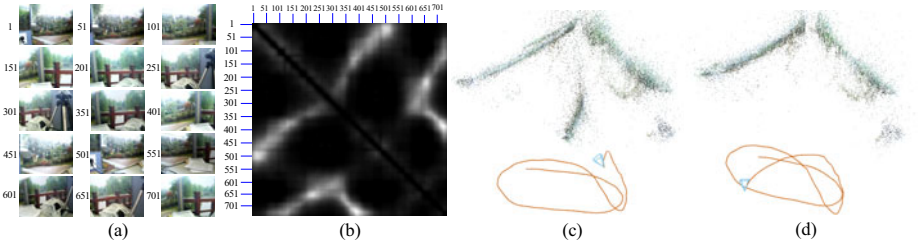


Fig. 3. Matching matrix estimation and non-consecutive track matching evaluation. (a) Selected frames from the "wallpaper" sequence. (b) Computed matching matrix that is linearly scaled for visualization. (c) Reconstruction result without non-consecutive track matching. (d) With non-consecutive track matching, the 3D points and camera motion are better estimated. The drift problem is also eliminated, as shown in our supplementary video.

we exclude the cases that two tracks in the same group span common frames (i.e., $f(\mathcal{X}_u) \cap f(\mathcal{X}_v) \neq \emptyset$).

For acceleration, we only select long tracks in the confidence estimation. In our experiments, for a sequence with 735 frames, the matching matrix estimation only requires 6 seconds, with a total of 22,573 selected feature tracks. Fig. 3(b) visualizes the computed matching matrix from a video, beside which a few selected frames are shown. Bright pixels indicate high confidence. It can be observed that these bright pixels are clustered in different regions in the matching matrix, reflecting the content similarity among subsequences in the input video. The diagonal band has no value because we exclude track self-matching.

5.2 Non-consecutive Track Matching

We identify overlapped subsequences by detecting rectangular regions containing the brightest pixels in the matching matrix. Suppose a rectangular region spans $[u_b, u_e]$ horizontally and $[v_b, v_e]$ vertically, video subsequences with frame sets $\phi_1 = \{u_b, ..., u_e\}$ and $\phi_2 = \{v_b, ..., v_e\}$ are correlated.

Since the matching matrix is symmetric, we only consider either the upper or lower triangle. We use the following method to estimate $[u_b, u_e]$ and $[v_b, v_e]$. In the beginning, we search for the element M_{ij} with the largest similarity value. Then we search for the maximum range $[u_b, u_e]$ such that $u_b < i < u_e$ and for any $t \in [u_b, u_e]$, $M_{tj}/M_{ij} > \delta$, where δ is a threshold. $[v_b, v_e]$ is computed similarly.

Algorithm 4. Track Joining

1. Track Matching:
   For $i = u_b, ..., u_e$, // one subsequence ϕ_1
   For $j = v_b, ..., v_e$, // another subsequence ϕ_2
   Match features in I_i and I_j and join their corresponding tracks.
2. Outlier Rejection:
   For joined tracks $\mathcal{X}_s$ (in ϕ_1) and $\mathcal{X}_t$ (in ϕ_2), if any of their feature pair $\{(\mathbf{x}_i^s, \mathbf{x}_j^t) | i \in f(\mathcal{X}_s) \cap \phi_1, j \in f(\mathcal{X}_t) \cap \phi_2\}$ do not satisfy the epipolar geometry constraint, the match is rejected.

Finally, we set $\phi_1 = \{u_b, ..., u_e\}$ and $\phi_2 = \{v_b, ..., v_e\}$ and set the corresponding elements in the matrix M to zeros. So in the next round, we again select a new M_{ij} from the updated matrix M to detect another subsequence pair for track matching. This process repeats until no high overlapping-confidence frames can be found.

Given the estimated subsequence pair (ϕ_1, ϕ_2), we reliably join tracks scattered over these frame sets (described in Algorithm 4). For each two frames, if their two distinctive features $\mathbf{x}_i^s$ and $\mathbf{x}_j^t$, belonging to $\mathcal{X}_s$ and $\mathcal{X}_t$ respectively, are matched using the method described in Section 4.1, we join tracks $\mathcal{X}_s$ and $\mathcal{X}_t$ as well. To reject outliers, we apply the geometric constraint to check whether all features in $\mathcal{X}_s$ and $\mathcal{X}_t$ satisfy the epipolar geometry constraint, i.e., $(\mathbf{x}_i^s, \mathbf{x}_j^t)$ consistent with a fundamental matrix F_{ij} estimated with the potential matches between frame pair (I_i, I_j) by the RANSAC algorithm [24]. If the two tracks qualify, they can be safely joined.

The example shown in Fig. 3 demonstrates the effectiveness of our non-consecutive track matching. We perform feature tracking and use the SfM method of [2] to recover camera poses together with sparse 3D points. In the first part of the experiment, we only use sequential tracks to estimate SfM. It is shown in Fig. 3(c) that this scheme produces erroneous camera pose estimate. Then we perform non-consecutive track matching to automatically join common tracks. It improves SfM, as shown in Fig. 3(d). The reconstruction quality can be assessed by inserting a virtual object into the scene, as demonstrated in our supplementary video. When skipping the non-consecutive track matching, the drift problem of the virtual object caused by inaccurate camera pose estimation is severe. In comparison, no such problem is observable after non-consecutive track matching.

5.3 Tracks in Multiple Videos

To describe a large-scale scene, multiple videos can generally be obtained from internet or be captured in different geographic regions but generally with overlaps. How to efficiently match multiple videos and register them in a common 3D system was seldom discussed in previous work. In our feature tracking system, this can be naturally accomplished. We first track feature points for each video independently and then detect overlap between each pair of the videos. The

Table 2. Running time of a few examples

Datasets	Resolution	Frames	Feature Tracking Time	
			Consecutive	Non-Consecutive
Wallpaper	640×480	735	6 minutes	2 minutes
Circle	960×540	1,991	30 minutes	12 minutes
Yard	960×540	3,201	50 minutes	20 minutes
Street	960×540	$\sim 23,000$	6 hours	2 hours

algorithm described in Section 5.1 is used to rapidly estimate the matching matrix such that related subsequences in different videos can be found. Afterwards, we match the common tracks distributed in various subsequences using Algorithm 4. This method quickly yields a matching graph for the collected videos, which finally leads to a robust global 3D registration, as shown in Fig. 1(b).

6 Results

We have evaluated our method on several challenging sequences. All results are generated using a PC with an Intel Core2Duo CPU 2.0GHz and 2GB memory. Running time for feature tracking on the tested data is listed in Table 2.

As our consecutive point tracking can handle wide-baseline images, frame-by-frame tracking is generally not necessary. In our experiments, the system extracts one frame for every $5 \sim 10$ frames to apply feature tracking. The tracked features are then propagated to other frames by simple KLT sequential tracking. This trick saves a lot of running time and results in feature tracking in a video sequence (1000 features per image and image resolution 640×480) only taking about 0.5 second per frame with our software implementation (single working thread). The running time of KLT[2] is about 0.4 second per frame. Note that the camera pose estimates from KLT could drift while our method avoids this problem because the computed matches are with very high quality and large quantity.

We compare our method to the brute-force SIFT matching in the Bundler software [27]. The brute-force SIFT matching method does not make use of image ordering. It extracts the SIFT features in all frames and exhaustively compares them. Although a K-d tree is used for matching speedup, the complexity is still quadratic to the number of the processed frames. In contrast, the complexity of our method is almost linear to the frame number.

For the "circle" example with 1991 frames. Performing the brute-force SIFT matching in the whole sequence will take days using our desktop computer. To save time, we pick out one frame for every 5, to compose a new sequence containing only 399 frames. The brute-force SIFT matching spends 187 minutes (6 minutes for SIFT feature extraction) on it, while our method only requires 25 minutes in total. When excluding the SIFT feature extraction time,

[2] We use the CPU implementation downloaded from http://www.ces.clemson.edu/~stb/klt/.

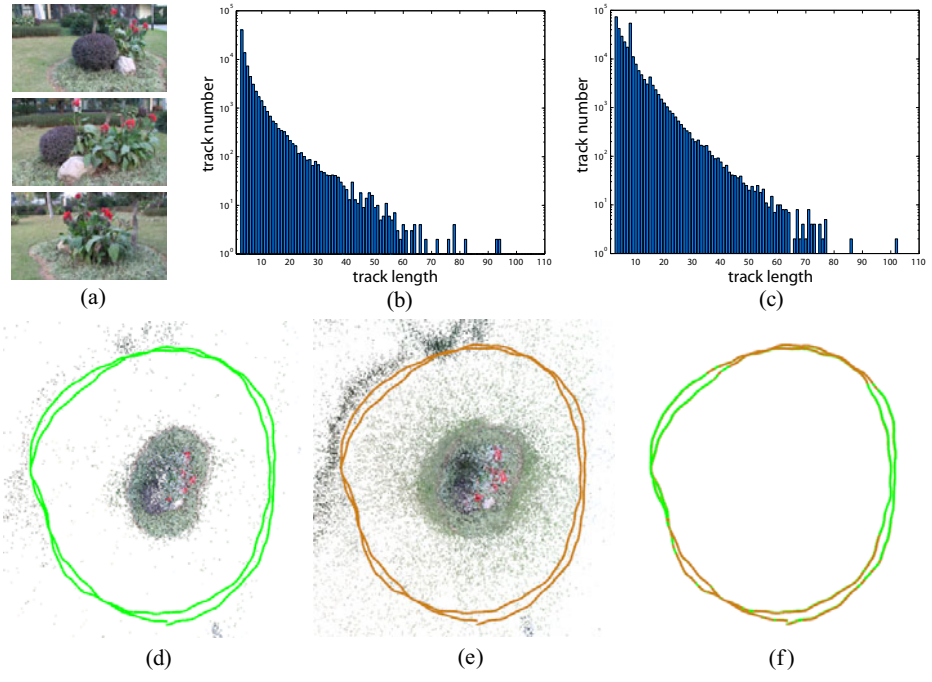


Fig. 4. Comparison with the brute-force SIFT matching. (a) Three selected frames from the "circle" sequence. (b-c) The track length histograms of the brute-force SIFT matching and our non-consecutive feature tracking, respectively. (d) The SfM result using the feature tracks computed by brute-force SIFT matching. (e) The SfM result using the feature tracks computed by our method. (f) Superimposing the camera trajectory in (d) to (e).

our method is about one order of magnitude faster. Figs. 4(a) and 4(b) show the track length histograms to compare the tracking quality. Our method yields many long feature tracks thanks to the effective two-pass matching and subsequence joining. The SfM results are shown in Figs. 4(d)-(f). The aligned two camera trajectories (shown in Fig. 4(f)) are with average camera position difference 0.000425 (normalized w.r.t. the total length of the camera trajectory).

We tested our method on a challenging large-scale "street" example containing a total of 9 videos, each of which has around $2000 \sim 3000$ frames. This example has been shown in Fig. 1. The camera moved along a street and captured several buildings. We first track feature points for each video independently, and then use our non-consecutive track matching algorithm to detect and match common feature tracks across different videos. We perform SfM estimation for each video independently. By aligning the computed 3D points, we register these videos in a 3D system. There are as many as $558,392$ estimated 3D points in this example. Superimposing the recovered camera trajectories onto a satellite image shows the high accuracy of the results as all trajectories are on streets and are not drifted.

7 Conclusion and Discussion

We have presented a robust and efficient non-consecutive feature tracking system for SfM, which consists of two main steps, i.e., consecutive point tracking and non-consecutive track matching. Different from the typical sequential matcher (e.g. KLT tracker), we use the invariant features and propose a two-pass matching strategy to significantly extend the track lifetime and reduce the feature sensitivity to noise and image distortion. The obtained tracks contain not only a set of 2D image positions, but also descriptors. They avail estimating a matching matrix to detect a set of disjointed subsequences with overlapping views. Our method can also handle tracking and registering multiple videos. Experimental results demonstrate the significance for SfM in middle- and large-scale scenes.

Our method is designed for SfM, and thus consider feature tracking only on rigid (non-deforming) objects in this paper. Part of our future work is to handle deforming or dynamic objects. Besides, although the proposed method is based on the SIFT features, there is no limitation to use other representations, especially in the general two-pass matching process. Further investigation will be conducted.

Acknowledgements

The work described in this paper was supported by the 973 program of China (No. 2009CB320804), NSF of China (Nos. 60633070 and 60903135), and the Research Grants Council of the Hong Kong Special Administrative Region (Project Nos. 413110 and 417107).

References

1. Pollefeys, M., Nistér, D., Frahm, J.M., Akbarzadeh, A., Mordohai, P., Clipp, B., Engels, C., Gallup, D., Kim, S.J., Merrell, P., Salmi, C., Sinha, S.N., Talton, B., Wang, L., Yang, Q., Stewénius, H., Yang, R., Welch, G., Towles, H.: Detailed real-time urban 3d reconstruction from video. International Journal of Computer Vision 78, 143–167 (2008)
2. Snavely, N., Seitz, S.M., Szeliski, R.: Photo tourism: exploring photo collections in 3d. ACM Trans. Graph. 25, 835–846 (2006)
3. Li, X., Wu, C., Zach, C., Lazebnik, S., Frahm, J.M.: Modeling and recognition of landmark image collections using iconic scene graphs. In: Forsyth, D., Torr, P., Zisserman, A. (eds.) ECCV 2008, Part I. LNCS, vol. 5302, pp. 427–440. Springer, Heidelberg (2008)
4. Agarwal, S., Snavely, N., Simon, I., Seitz, S.M., Szeliski, R.: Building rome in a day. In: ICCV, pp. 72–79 (2009)
5. Hartley, R.I., Zisserman, A.: Multiple View Geometry in Computer Vision, 2nd edn. Cambridge University Press, Cambridge (2004) ISBN: 0521540518
6. Fitzgibbon, A., Zisserman, A.: Automatic camera tracking. In: Video Registration, pp. 18–35 (2003)

7. Zhang, G., Qin, X., Hua, W., Wong, T.T., Heng, P.A., Bao, H.: Robust metric reconstruction from challenging video sequences. In: CVPR (2007)
8. Lucas, B.D., Kanade, T.: An iterative image registration technique with an application to stereo vision. In: IJCAI, pp. 674–679 (1981)
9. Shi, J., Tomasi, C.: Good features to track. In: CVPR, pp. 593–600 (1994)
10. Georgescu, B., Meer, P.: Point matching under large image deformations and illumination changes. IEEE Trans. Pattern Anal. Mach. Intell. 26, 674–688 (2004)
11. Lowe, D.G.: Distinctive image features from scale-invariant keypoints. International Journal of Computer Vision 60, 91–110 (2004)
12. Zach, C., Gallup, D., Frahm, J.M.: Fast gain-adaptive klt tracking on the gpu. In: CVPR Workshop on Visual Computer Vision on GPU's (CVGPU) (2008)
13. Lazebnik, S., Schmid, C., Ponce, J.: A sparse texture representation using local affine regions. IEEE Trans. Pattern Anal. Mach. Intell. 27, 1265–1278 (2005)
14. Mikolajczyk, K., Schmid, C.: A performance evaluation of local descriptors. IEEE Trans. Pattern Anal. Mach. Intell. 27, 1615–1630 (2005)
15. Matas, J., Chum, O., Urban, M., Pajdla, T.: Robust wide-baseline stereo from maximally stable extremal regions. Image Vision Comput. 22, 761–767 (2004)
16. Brown, M., Lowe, D.G.: Recognising panoramas. In: ICCV, pp. 1218–1227 (2003)
17. Morel, J.M., Yu, G.: ASIFT: A new framework for fully affine invariant image comparison. SIAM J. Img. Sci. 2, 438–469 (2009)
18. Sivic, J., Zisserman, A.: Video google: A text retrieval approach to object matching in videos. In: ICCV, pp. 1470–1477 (2003)
19. Schaffalitzky, F., Zisserman, A.: Automated location matching in movies. Computer Vision and Image Understanding 92, 236–264 (2003)
20. Ho, K.L., Newman, P.M.: Detecting loop closure with scene sequences. International Journal of Computer Vision 74, 261–286 (2007)
21. Schindler, G., Brown, M., Szeliski, R.: City-scale location recognition. In: CVPR (2007)
22. Irschara, A., Zach, C., Frahm, J.M., Bischof, H.: From structure-from-motion point clouds to fast location recognition. In: CVPR (2009)
23. Engels, C., Fraundorfer, F., Nistér, D.: Integration of tracked and recognized features for locally and globally robust structure from motion. In: VISAPP (Workshop on Robot Perception), pp. 13–22 (2008)
24. Fischler, M.A., Bolles, R.C.: Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. Commun. ACM 24, 381–395 (1981)
25. Sinha, S.N., Steedly, D., Szeliski, R.: Piecewise planar stereo for image-based rendering. In: ICCV, pp. 1881–1888 (2009)
26. Nister, D., Stewenius, H.: Scalable recognition with a vocabulary tree. In: CVPR, Washington, DC, USA, pp. 2161–2168. IEEE Computer Society, Los Alamitos (2006)
27. Snavely, N.: Bundler: Structure from motion for unordered image collections, `http://phototour.cs.washington.edu/bundler/`