# Trajectory Matching from Unsynchronized Videos

Han Hu and Jie Zhou
State Key Laboratory on Intelligent Technology and Systems
Tsinghua National Laboratory for Information Science and Technology (TNList)
Department of Automation, Tsinghua University, Beijing, China, 100084
huh04@mails.thu.edu.cn, jzhou@tsinghua.edu.cn

## Abstract

*This paper studies the problem of spatio-temporal matching between trajectories from two videos of the same scene. In real applications, trajectories are usually extracted independently in different videos. So possibly a lot of trajectories stay "alone" (have no corresponding trajectory in the other video). In this paper, we propose a novel matching algorithm which can not only find the existing correspondences between trajectories, but also recover the corresponding trajectories of "alone" ones. First, we cast trajectory matching problem as an element recovering problem from a matrix constructed by matched trajectories of the two videos, which is naturally incomplete. Then, under affine camera assumption, we recover the matrix by sparse representation and $\ell_1$ regularization techniques. Finally, the results are refined to the case of perspective projection by a local depths estimation procedure. Our algorithm can handle noisy, incomplete or outlying data. Experiments on both synthetic data and real videos show that the proposed method has good performance.*

## 1. Introduction

An essential problem of computer vision is image registration, and its goal is to determine the spatial correspondences between two images of the same scene. There has been a large amount of studies on this topic (see [23] for a summary), and feature based methods are one of the most popular kinds being used. They come out by first extracting features, e.g. Harris Corners [7], SIFT [12], SURF [2], FAST [20, 21], etc., from the images, and then matching the features across different images.

According to the organization form of resource images, there are mainly two matching strategies. One occasion is matching features of static images taken from different viewpoints. The images may have much different zoom factors or viewpoints, which is also referred as wide-baseline stereo. A common way to tackle this problem is using robust features and finding their corresponding ones by searching the most similar descriptors on the other image. However, usually only a small fraction of the originally extracted features remains to be matched [14]. The other occasion is matching features of adjacent images in a video sequence. Since inter-frame motions are usually very small, the features can be efficiently matched by local searching or optical flow, e.g. SURFTrac [24] and Lucas-kanade-Tracker [22].

There are many potential applications for matching of image features, i.e. 3D reconstruction. However, the reconstruction from either static images or a video sequence becomes difficult or even impossible when with nonrigid scene. Using static images, apart from insufficient matched features, the photos should be taken strictly in the same time. While for the case of a single video sequence, the unknown 3D motion models largely increase the difficulty.

The above limitations have led to more and more interests on studies of multiple videos. Similarly, a fundamental problem is registration. It is a more challenging work than that of images for both spatial and temporal relationships should be recovered. During the last few years, there have been quantities of methods on feature-based video registration. They can be roughly grouped into two categories: temporal first and spatial first.

Temporal first methods usually obtain the temporal shift by minimizing an "energy" function based on the relationships of trajectories in different videos. Since there are no exact spatial correspondences, we need strong assumptions to define the "energy" in most cases. For example, [27] used rank constraints of the trajectory matrix as the "energy", which needs a large amount of frames and can be only applied for the case of jointly moving cameras.

Due to the limitations of temporal first methods, the spatial first methods become the most popular trend for video registration [26, 17, 16, 19]. Given spatial correspondences between trajectories, the temporal relationship can be stably fitted by RANSAC [19] or Hough transform [16, 26]. Our

978-1-4244-6985-7/10/$26.00 ©2010 IEEE 

algorithm also follows this mode. However, most of these methods obtained the spatial correspondences of trajectories by manually selecting or empirically assigning, which makes them far away from real applications. To the best of our knowledge, the work in [29] is the only attempt to find the exact spatial correspondences between trajectories without much empirical information. Given a few matched trajectories, it found a new match by searching the pair leading to minimal rank rise when added to the initial trajectory matrix. The limitations of this algorithm involve: it needs the trajectories in different videos are projected from the same set of 3D points and the trajectories should be all complete.

In this paper, we propose a novel approach to trajectory matching problem. We use local rigidity assumption to describe general nonrigid scene, which means for each 3D point, there exist sufficient other 3D points lying on the same rigid structure with it. This is a very popular and also a weak assumption for nonrigid scenes [4, 29]. Under local rigidity assumption, the trajectories satisfy certain subspace constraints, which can be used for trajectory matching. The main contributions of this paper include:

1. Our work is the first to cast trajectory matching problem as an element recovering problem from a subspace-constrained matrix. In this way, we can not only find the already existed matches, but also recover the corresponding trajectories of "alone" ones, which enables the applications that need a large quantity of matched trajectories.

2. We propose a *local* depths estimation algorithm, so that the matching results under the affine projection assumption, can be refined to perspective cases. Traditionally, depths estimation is usually carried out in the global domain, i.e. first segmenting the trajectories into several rigid groups and then estimating depths on each one [10]. However, motion segmentation usually needs the number of rigid objects is small and should be known as a priori. Another shortcoming is revealed by that when there are segmenting errors for a rigid group, all trajectories on the group are affected. Instead of using the global segments for depths estimation, our algorithm finds the local rigid structure for each trajectory, which is obtained by directly selecting the trajectories contributing nonzero reconstruction coefficients in sparse representation of the very trajectory. Since the rigid parts are calculated locally and independently for trajectories, the proposed algorithm is expected to be immune to the two problems which trouble global methods.

The paper is organized as follows: we present the subspace constraints of a local-rigid scene in Section 2; Section 3 proposes a novel matching algorithm based on subspace constraints and sparse representation under affine camera model; after that, we refine the results to the perspective case in Section 4; then, Section 5 shows the experimental results on both synthetic and real data; and finally we conclude with a brief summary in Section 6 .

## 2. Subspace Constraints of a Local-rigid Scene using Affine Cameras

Let $\{x_{fi} = (u_{fi}, v_{fi}, 1)^T \in \mathbb{R}^3\}_{f=1,\dots,F}^{i=1,\dots,P}$ be the 2D projections in $F$ frames of $P$ 3D points $\{X_i \in \mathbb{R}^4\}_{i=1}^{P}$ from a rigid structure. Under the affine camera model, the trajectories and their 3D points satisfy the following equations [9],

$$x_{fi} = A_f X_i, \text{ and} \tag{1}$$

$$W = \overbrace{\begin{bmatrix} x_{11} & \dots & x_{1P} \\ \vdots & \ddots & \vdots \\ x_{F1} & \dots & x_{FP} \end{bmatrix}}^{3F\times P} = \overbrace{\begin{bmatrix} A_1 \\ \vdots \\ A_F \end{bmatrix}}^{3F\times 4} \overbrace{\begin{bmatrix} X_1^T \\ \vdots \\ X_P^T \end{bmatrix}^T}^{4\times P} = MS, \tag{2}$$

where $A_f = K_f \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} R_f & \mathbf{t}_f \\ \mathbf{0}^T & 1 \end{bmatrix} \in \mathbb{R}^{3\times 4}$ is an affine matrix at frame $f$, which depends on the camera intrinsic parameters $K_f$ and the object pose relative to the camera $(R_f, \mathbf{t}_f)$.

Eq.(2) indicates that,

$$rank(W) \leq \min(rank(M), rank(S)) \leq 4. \tag{3}$$

Eq.(3) assumes that trajectories from a rigid structure lie on a linear subspace of $\mathbb{R}^{3F}$ with dimension no more than 4.

For an arbitrary trajectory of a local-rigid scene, there exist sufficient trajectories which are on the same rigid structure with it, so they together span a single linear subspace with dimension no more than 4.

## 3. The Proposed Algorithm under Affine Projection Model

Let $W_1 \in \mathbb{R}^{3F_1\times P_1}$ and $W_2 \in \mathbb{R}^{3F_2\times P_2}$ be two trajectory matrices from two video sequences, respectively. If the scene only includes a single rigid motion, then according to Eq.(2), we have $W_1 = M_1 S_1$ and $W_2 = M_2 S_2$. If $P_1 = P_2 = P$, and $S_1$, $S_2$ share the same set of 3D points, such that there exists a permutation matrix $\Gamma^* \in \mathbb{R}^{P\times P}$ satisfying $S_1 = S_2\Gamma^*$, we have,

$$W^* = \begin{bmatrix} W_1 \\ W_2\Gamma^* \end{bmatrix} = \begin{bmatrix} M_1 & 0 \\ 0 & M_2 \end{bmatrix} \begin{bmatrix} S_1 \\ S_2\Gamma^* \end{bmatrix} = \begin{bmatrix} M_1 \\ M_2 \end{bmatrix} S_1. \tag{4}$$

Given an arbitrary permutation matrix $\Gamma$, we obtain the following inequality,

$$\begin{aligned} rank(W^*) &= rank\left(\begin{bmatrix} M_1 \\ M_2 \end{bmatrix} S_1\right) \\ &\leqslant rank\left(\begin{bmatrix} M_1 & 0 \\ 0 & M_2 \end{bmatrix} \begin{bmatrix} S_1 \\ S_2\Gamma \end{bmatrix}\right) = rank(\begin{bmatrix} W_1 \\ W_2\Gamma \end{bmatrix}). \end{aligned} \tag{5}$$

Eq.(5) indicates that the correct permutation yields a compound matrix $W$ with the lowest rank. Thus the spatial matching problem can be cast as the following optimization problem,

$$\min_{\Gamma} \ rank(\begin{bmatrix} W_1 \\ W_2\Gamma \end{bmatrix}). \tag{6}$$

When $W_1$ and $W_2$ share the same set of 3D points, Eq.(6) can be used for determining the matches. However, in practice, the trajectories of different videos are usually extracted independently, and therefore, a vast amount of trajectories stay "alone", for which the above algorithm is useless. Here we exploit another way to address the matching problem. Suppose we have obtained several initial matches across different videos (we will discuss the details in Section 3.4). Denote the initial matched trajectories on two videos by $W_{1m} \in \mathbb{R}^{3F_1 \times P_m}$ and $W_{2m} \in \mathbb{R}^{3F_2 \times P_m}$. Denote the remaining unmatched trajectories on the first video and their unknown corresponding trajectories on the other video by $W_{11} \in \mathbb{R}^{3F_1 \times P_1}$ and $W_{21} \in \mathbb{R}^{3F_2 \times P_1}$. The remaining unmatched trajectories on the second video and their unknown corresponding trajectories on the first video are $W_{22} \in \mathbb{R}^{3F_2 \times P_2}$ and $W_{12} \in \mathbb{R}^{3F_1 \times P_2}$. Thus the matching and recovering task is equal to fill in the missing elements of the matrix below,

$$W = \begin{bmatrix} W_{11} & W_{1m} & (W_{12}) \\ (W_{21}) & W_{2m} & W_{22} \end{bmatrix}, \tag{7}$$

where $(\cdot)$ indicates that the entries are missing. If two compound trajectories from $[W_{11}^T \ W_{21}^T]^T$ and $[W_{12}^T \ W_{22}^T]^T$ are close enough, we judge they are the same and the recovered entries are replaced by the corresponding existing entries; otherwise, we use the recovered entries as their corresponding trajectories.

In the following subsections, we first show how the missing entries can be recovered by using subspace constraints and sparse representation. Then we deal with noisy, incomplete or outlying data. Third, a temporal matching algorithm is presented. Finally, we discuss details of obtaining the trajectories as well as the initial matches.

### 3.1. Spatial Matching and Recovery via Sparse Representation

Sparse representation has proven to be a very powerful tool for representing and compressing high-dimensional signals. In the last few years, this technique also has seen significant impact in computer vision (see [28] for some instances). Now we show how sparse representation technique is used for matching and recovering of trajectories from two video sequences of the same scene.

We have shown in Section 2 that the trajectories from a video sequence of a local-rigid scene satisfy certain subspace constraints, which also acts on the compound trajectory matrix of Eq.(7). Therefore, the compound trajectories of a local rigid structure lie on a linear subspace with dimension no more than 4. Denote its dimension by $D$. We know that any trajectory $\mathbf{y} \in \mathbb{R}^{3(F_1+F_2)}$ can be represented in a basis of $3(F_1+F_2)$ vectors $\{\psi_i \in \mathbb{R}^{3(F_1+F_2)}\}_{i=1}^{3(F_1+F_2)}$, such that,

$$\mathbf{y} = \sum_{i=1}^{3(F_1+F_2)} \alpha_i \psi_i = \Psi \mathbf{a}, \tag{8}$$

where $\Psi = [\psi_1, \psi_2, \ldots, \psi_{3(F_1+F_2)}]$, and $\mathbf{a} = [\alpha_1, \alpha_2, \ldots, \alpha_{3(F_1+F_2)}]^T$. As $3(F_1+F_2)$ is usually much larger than D, $\mathbf{a}$ can be very sparse with a properly chosen basis $\Psi$. In principle, given such proper basis, the sparse representation problem can be solved by the $\ell_0$ optimization problem:

$$\min \|\mathbf{a}\|_0 \quad \text{subject to} \quad \mathbf{y} = \Psi \mathbf{a}, \tag{9}$$

where $\|\cdot\|_0$ denotes the $\ell_0$-norm, i.e. the number of nonzero entries in a vector. However, such an optimization problem is NP-hard and is even difficult to approximately solved [1]. Fortunately, [5] claims that if the solution is sparse enough, the solution of Eq.(9) is equal to the solution of the following $\ell_1$ optimization problem:

$$\min \|\mathbf{a}\|_1 \quad \text{subject to} \quad \mathbf{y} = \Psi \mathbf{a}. \tag{10}$$

In trajectory matching problem, the bases can be chosen as the full-entry compound trajectories. To be specific, if we assume that for each trajectory, there are sufficient full-entry compound trajectories lying on the same rigid structure with it, and the trajectories locate in general positions that they together span the whole local rigid subspace, then the trajectory can be represented as a linear combination of at most 4 full-entry ones. So using the compound trajectories $W_m = [W_{1m}^T \ W_{2m}^T]^T$ as the bases, any trajectory $y_i \in [W_{11}^T \ W_{21}^T]^T$ can have a sparse representation with at most 4 nonzero coefficients. We can then determine the sparse reconstruction coefficients $\mathbf{a}_{1i}^* \in \mathbb{R}^{m \times 1}$ by the $\ell_1$ optimization,

$$\min \|\mathbf{a}_{1i}\|_1 \quad \text{subject to} \quad y_{1i} = W_{1m}\mathbf{a}_{1i}, \tag{11}$$

and the invisible trajectory $y_{2i}$ is recovered by $W_{2m}\mathbf{a}_{1i}^*$.

Similarly, we can recover the corresponding ones of the trajectories in the second video.

### 3.2. Dealing with Noises, Incompletion or Outliers

In reality, trajectories are usually noisy, incomplete, or corrupted by outliers, which may come from occlusions, perspective projection, low resolution, or limitations of feature tracker. In this section, we handle these "dirty" trajectories with Lasso and sparse representation techniques, which were also exploited in the preprocessing step for motion segmentation [18, 6]. In the following descriptions, we

only take the trajectories from the first video as examples, and the processing of the other video is similar.

When data is contaminated with noises, the constraint condition of Eq.(11) may not hold any more. Denote the noise bound by $\varepsilon$. The constraint condition should be rewritten as a soft one, $\|y_{1i} - W_{1m}\mathbf{a_{1i}}\|_2 \leq \varepsilon$. We further convert this problem into a nonrestraint optimization problem which can be efficiently solved by many Lasso algorithms, e.g. [25],

$$\min \|\mathbf{a_{1i}}\|_1 + \gamma\|y_{1i} - W_{1m}\mathbf{a_{1i}}\|_2. \tag{12}$$

Now consider the case of incompletion. Denote the indices of missing entries in the to-be-matched trajectory $y_{1i} \in \mathbb{R}^{3F_1}$ by $I_i \subset \{1, ..., 3F_1\}$. Then we obtain a complete vector $\tilde{y}_{1i}$ and the corresponding data matrix $\tilde{W}_{1m}$ by eliminating the $I_i$ rows of $y_{1i}$ and $W_{1m}$, respectively. Then we find a sparse representation coefficients, $\mathbf{a}^*_{1i}$, by optimizing the following problem,

$$\min \|\mathbf{a}_{1i}\|_1 + \gamma\|\tilde{y}_{1i} - \tilde{W}_{1m}\mathbf{a}_{1i}\|_2. \tag{13}$$

The missing entries of $y_{1i}$ are recovered by $W_{1m}\mathbf{a}^*_{1i}$.

And next we attempt to handle outliers. This can be easily achieved by examine the cost of Eq.(12) or Eq.(13). And the trajectories with high costs are discarded.

### 3.3. Temporal Matching

Since quantities of matched trajectories have been obtained, many trajectory-based synchronizing algorithms [26, 17, 16, 19, 3] can be used to find the temporal correspondences between two video sequences. Here, similar with [3], we exploit two-view epipolar constraints to determine the synchronizing time:

$$\min_{F,\Delta t} \sum_{i=1}^{P} \sum_{t} \|x_{1i}(t + \Delta t)^T F_t x_{2i}(t)\|^2, \tag{14}$$

where $P = P_1 + P_2 + P_m$, and $x_{1i}(t)$ and $x_{2i}(t)$ are the homogeneous coordinates of the $i$th trajectory at frame t from two videos, respectively. $F_t$ is the fundamental matrix between frame $t + \Delta t$ of the first video and frame $t$ of the second video.

We further refine the time shift to subframe precision by the following scheme: first we find the two adjacent time shifts $t_1$ and $t_2$ ($t_2 = t_1 + 1$) by minimizing (14); then we obtain subframe time shift $\Delta t = t_1 + \beta$ by alternatively optimizing $F$ and $\beta$ from the following optimization problem:

$$\begin{aligned}\min_{F \text{ or } \beta} \quad & \textstyle\sum_{i=1}^{P} \sum_t \|((1-\beta)\cdot x_{1i}(t + t_1) \\ & +\beta \cdot x_{1i}(t + t_1 + 1))^T F_t x_{2i}(t)\|^2.\end{aligned} \tag{15}$$

### 3.4. Feature Tracking and Initial Matching

For spatial-matching of trajectories, there are two tasks. The first one is to match features between consecutive frames of a video, which is also referred as feature tracking. The other is to match the features of frames across videos with different viewpoints or scales. Generally speaking, optical flow methods, e.g. Lucas-Kanade tracker [13], are preferential ones for feature tracking, for they exploit the coherent information between adjacent frames. However, only corner-like features which indicate gray discontinuities, such as FAST corners [20, 21] or Harris corners [7], are good ones for tracking [22]. While, for the second task, robust features such as SIFT [12] and SURF [2] are usually adopted since they are invariant to scale and rotation. Our goal is to achieve both matching tasks with the same features. Unfortunately, the two kinds of features mentioned above are unlikely to coincide with each other, for the one is corner-like and the other is blob-like.

To tackle the above difficulty, there have been several attempts to track robust features exploiting coherence information. For example, [11] proposed a SIFT Flow approach which utilizes a dense SIFT descriptors. SURFTrac [24] found the matches by searching the highest Normalized-Cross-Correlation (NCC) in Hessian domain over the local area. These algorithms gain merits in several properties, however, they are still not good enough for many applications: SIFT Flow is much time consuming, while SURF-Trac can only reach pixel-precision and often ends with high error rate.

In this paper, we adopt a hybrid algorithm to achieve the two tasks. Since initially we only need a small amount of matched features across videos, we use Harris corners, which are good to track, as extracted features. The Lucas-Kanade tracker is adopted to track features of the same video. And several SIFT descriptors on multiple scales are computed for each corner to robustly match features across different videos. There may be some wrongly tracked or matched features. So we execute an outlier discarding procedure as a post-processing step which has been presented in Section 3.2.

## 4. Local Perspective Refining

In the case of perspective cameras, the projection model (1) for a rigid structure becomes

$$\lambda_{fi} x_{fi} = A_f X_i, \tag{16}$$

where $\lambda_{fi}$ is the projective depth of $x_{fi} \in \mathbb{R}^3$. Then the subspace constraint of a rigid structure is rewritten as,

$$W(\lambda) = \begin{bmatrix} \lambda_{11}x_{11} & \dots & \lambda_{1P}x_{1P} \\ \vdots & \ddots & \vdots \\ \lambda_{F1}x_{F1} & \dots & \lambda_{FP}x_{FP} \end{bmatrix} = MS. \tag{17}$$

To enable the modified subspace constraint of Eq.(17) for trajectory matching, we need to first recover the projective depths. Most of existing algorithms need the scene to be rigid [9]. However, for a local-rigid scene, the rigid motion patterns as well as the trajectories belonging to each are both unknown. [10] proposed a framework to settle this problem by alternating between single rigid depths estimation and motion segmentation. We refer it as *global* method for it obtains the global segments of trajectories. The shortcomings of this method involve: 1) motion segmentation algorithms usually need the number of rigid structures is small and known as a priori; 2) when there exist wrongly segmented trajectories for a rigid structure, the errors can affect the recovering procedure for all trajectories in the structure. Different with *global* method, we propose to calculate a local rigid structure for each trajectory and complete the trajectory by a local depths estimation procedure on the structure. We refer our method as *local* one. Because the local rigid structures are obtained locally and independently for trajectories, the above mentioned limitations are eased.

The local rigid structure is directly selected as the set of basis trajectories with nonzero reconstruction coefficients. There may be some outlying trajectories in the set for a local rigid structure. Nevertheless, because the coefficients of these outlying trajectories are usually very small, we can ease the problem by discarding trajectories with small coefficients or defining the reconstruction coefficients as weights to the basis trajectories.

### 4.1. Depths Estimation and Trajectory Recovering

For a to-be-reconstructed compound trajectory, after obtaining its local rigid structure, we can use rigid-scene based depths estimation methods to recover the relative depths. As in [9][15], we achieve the goal by alternating between the estimation of structure-and-motion and the estimation of depths using the subspace constraints. The main differences from the previous methods are: 1) we use weighted trajectories for matrix factorization and non-weighted trajectories for depths estimation; 2) the to-be-reconstructed incomplete compound trajectory is also introduced into the iteration step.

For a compound trajectory $y$, denote its existing entries indexed by I. The basis trajectories with nonzero reconstruction coefficients $\mathbf{a}^*$ compose a data matrix $\Phi$. The depths estimation algorithm is as follows:

1. construct weighted data matrix as, $\Phi^* = [\alpha_1^*\Phi_1 \dots \alpha_n^*\Phi_n]$, where $\mathbf{a}^* = [\alpha_1^* \dots \alpha_n^*]$ and $\Phi = [\Phi_1 \dots \Phi_n]$;

2. calculate the rank 4 approximation of $\Phi^*$ to get $\hat{\Phi}^*$ as, $\hat{\Phi}^* = \hat{U}\hat{\Sigma}\hat{V}^T = U(:, 1:4)\Sigma(1:4, 1:4)V(:, 1:4)^T$, where $U$, $\Sigma$ and $V$ are the SVD factorization of $\Phi^*$;

3. obtain the non-weighted rank 4 data matrix as $\hat{\Phi} = [1/\alpha_1^* \cdot \hat{\Phi}_1^* \dots 1/\alpha_n^* \cdot \hat{\Phi}_n^*]$;

4. approximately represent $y_I$ by $\hat{y}_I$ which is a linear combination of the orthonormal vectors, in least square sense,

$$\hat{y}_I = \hat{U}_I(\hat{U}_I^T\hat{U}_I)^{-1}\hat{U}_I^T y_I; \tag{18}$$

5. estimate the depths of the points on $y_I$ and $\Phi$ by minimizing the Euclidian distance between $y_I$ and $\hat{y}_I$, as well as that between each column of $\Phi$ and $\hat{\Phi}$ by,

$$\lambda_f = \hat{x}_f^T x_f / x_f^T x_f, \tag{19}$$

where $\hat{x}_f \in \mathbb{R}^3$ and $x_f \in \mathbb{R}^3$ are the corresponding entries in $\hat{y}_I$, $\hat{\Phi}$ and $y_I$, $\Phi$;

6. iteratively normalize existing entries in the columns and rows of the depths matrix to be 1 as suggested by [9];

7. iterate steps 1-6 until the distance between $y_I$, $\Phi$ and $\hat{y}_I$, $\hat{\Phi}$ is below a threshold.

After obtaining the depths, the missing entries of $y$ are recovered by $y_{\bar{I}} = \hat{U}_{\bar{I}}(\hat{U}_I^T\hat{U}_I)^{-1}\hat{U}_I^T y_I$.

## 5. Experiments

The proposed algorithm uses homogeneous coordinates to form the trajectory matrix as well as to calculate the time shift. As suggested in [8, 9], we normalize the coordinates as a preprocessing step. In the following of this section we demonstrate the performance of the proposed algorithm on both synthetic data and real video sequences.

### 5.1. Quantitative Evaluation on Synthetic Data

We first evaluate our algorithm on synthetic data of a local-rigid scene. The synthetic scene has three rigid bodies. Initial 3D points are uniformly generated in a sphere with center on the origin and radius 200. Then the 3 rigid bodies are randomly rotated and translated at each frame. Two perspective cameras, which initially locate in [-40,000, -400] on the z-axis and x-axis, are also freely rotated and translated with random values. We obtain the views by projecting the points onto images with $400 \times 400$ pixels. And zero-mean Gaussian noises with standard deviation $\sigma \in [0, 1]$ are added to the obtained 2-D points. We generate totally 1200 3D points, and 240 ones are in both two views while the others appear in only one view. A half of the "non-alone" ones are initially matched.

The performance on full non-outlying data is measured by *matching correct rates* (MCR) and *recovering accuracies* (RA). We compare our algorithm with the spatial matching method presented in [29]. Since the method in [29] is designed under affine camera model and can only find the existing matches for complete data, we only compare the method with our without-refining algorithm using MCR curves. In our algorithm, for each recovered compound trajectory visible in one video, we search the nearest

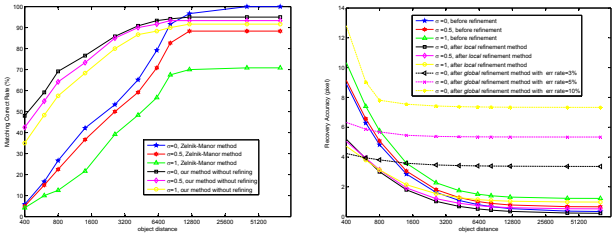
one visible in the other video and label the two trajectories as a match candidate. In [29], given initial matches, it searches the pair which adds a minimal residual to the matrix constructed by initial matches as a match candidate. For both methods, we select a set of 120 candidates with minimal Euclidian distances or residual as the ultimate matches, and MCR is the proportion of the correct matches in the set.

We also compare our *local* refining method with the *global* refining method. The RA, which represents the average coordinates error, is used to measure their performances. The *global* refining method is executed under different motion segmentation error rates between [3%,10%], which is generated by randomly selecting the wrongly labeled trajectories as well as their wrong labels.

For trajectories with missing entries, we use RA curves for evaluation. And *outlier detection rate* (ODR), which is measured by the proportion of correct ones in the detected outliers, is adopted for outliers detection experiments. We generate the outlying trajectories by choosing a random initial point in the first frame, and then performing a random walk through the following frames. Each increment is generated by taking the difference between the coordinates of a randomly chosen point in two randomly chosen consecutive frames. In this way the outlying trajectories will qualitatively have the same statistical properties as the other trajectories, but will not obey the subspace constraints.

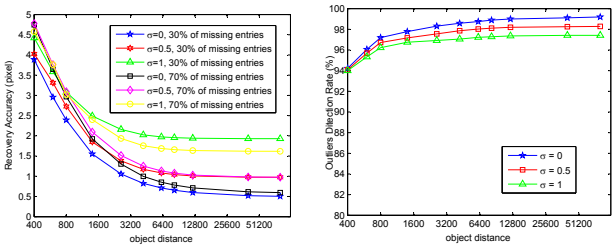
For each noise level, we run 5 trials with different randomly generated noises; for each incomplete rate, 3 trials are carried out with different randomly selected missing entry indexes; for each outlier detection experiment, we try 10 times; and for each segmentation error rate, it is 20. Figure 1 shows the evaluation curves of our algorithm compared with other methods. In Figure 1(a), MCRs of the method in [29] and our algorithm are presented. We can see that when the degree of perspective effects is very high, or when noises exist, our method (without refining) performs much better than the method in [29]. Figure 1(b) shows the RAs before and after perspective refining procedure versus object distance (which indicates the degree of perspective effect). The refining procedure is carried out either by our *local* method or the *global* one. We can see, even with a very low segmentation error rate, i.e. 3%, the *global* refining method hardly takes effect, while the *local* method stably reduces RA in almost all the cases.

Figure 2 shows the performance of the proposed method on data with missing entries or outliers. Figure 2(a) presents the RAs versus object distance under 30% and 70% missing entries. We can see that our algorithm recovers the missing entries with a small coordinate error and more missing entries does not necessarily lead to performance reduction. This is possibly because the 30% existing entries can provide enough information to determine the relationships among the trajectories. Figure 2(b) are the ODR curves with 240 outliers. The figure illustrates that our method can reach the detection rate with about 95% even in the worst case.




(a) (b)

Figure 1. The evaluation curves of our method compared with other methods. (a): the MCR curves of our without-refining method and the method in [29]; (b) the RA curves of our *local* refining method and the *global* one with different segmentation error rates.




(a) (b)

Figure 2. The evaluation curves of our method with incomplete or outlying data. (a): the RA curves for incomplete data; (b): the ODR curves for data with outliers.

## 5.2. Qualitative Evaluation on Real Video Sequences

We test the performance of our algorithm on two real scenes called *rotatingbooks* and *deformablepaper*. The *rotatingbooks* scene contains three rigid motions, two caused by two persons rotating books independently and the rest by the static background. The *deformablepaper* scene approximately obeys local rigidity assumption. The neighborhood area of each trajectory can be approximately regarded as a rigid structure. The two scenes are both recorded by two independent hand-held cameras from different viewpoints.

For each pair of videos, we get the trajectories and initial matches by the method presented in Section 3.4. In each video, we only maintain features tracked successfully through more than 20 frames. Figure 3 illustrates the "incompleteness" of compound trajectories for *rotatingbooks*, of which the black area indicates the missing entries.

The initial matches are then used to recover correspondences between all trajectories from two videos. Table 1 gives the details about videos and the recovered temporal

Figure 3. Existing entries of the compound matrix obtained from the *rotatingbooks* scene.

| Sequence | *rotatingbooks.* | *deformablepaper* |
|---|---|---|
| Num of points on left video | 869 | 851 |
| Num of points on right video | 869 | 874 |
| Num of frames on left video | 90 | 95 |
| Num of frames on right video | 103 | 101 |
| Initial matches | 93 | 213 |
| Temporal shift (frame) | -29.02 | -4.22 |

Table 1. Details about videos and the recovered temporal shifts.

shifts. Figure 4 shows the spatial matching results of the two scenes by our method. Notice that in order to show the matches more clearly, we only draw 100 randomly selected recovered correspondences in Figure 4(c)(f).

The results illustrate that the proposed algorithm can find the existing matches as well as recover quantities of matches which are originally nonexistent from different videos of the same scene. To qualitatively compare the *local* refining method with *global* one, we show an instance in Figure 5. We use the *deformablepaper* scene for the purpose, of which each trajectory and its neighbors approximately lie on a rigid structure. *Global* method tends to find exact multiple segments. See Figure 5(a) for an example. It groups the trajectories into 4 rigid segments using the method in [6]. Trajectories near the centers of segments benefit from the segmentation result since the segment can be an approximation to their optimal local rigid structures. While for trajectories near the boundaries, it becomes a disaster. See the red-rectangle point in Figure 5(a). It is near the boundary of black and green groups. However, neither of the two groups can well represent its local structure. And after executing a depths estimation procedure on the black area, the corresponding point is wrongly recovered as the blue-rectangle points on the right image.

The *local* refining method can avoid this problem. By using *local* refining method, almost all of the trajectories locate near the center of the estimated local rigid structure, which are illustrated as Figure 5(b). Green circles represent the local rigid structure for the red-rectangle point, and the larger the circle, the more important the trajectory is for spanning the structure. Using this local rigid structure, its corresponding trajectory is correctly recovered (see the blue-rectangle on the right image).

## 6. Conclusions

We have presented a novel algorithm to match trajectories from unsynchronized videos of the same nonrigid scene based on subspace constraints and sparse representation. We showed that, under affine projection assumption,

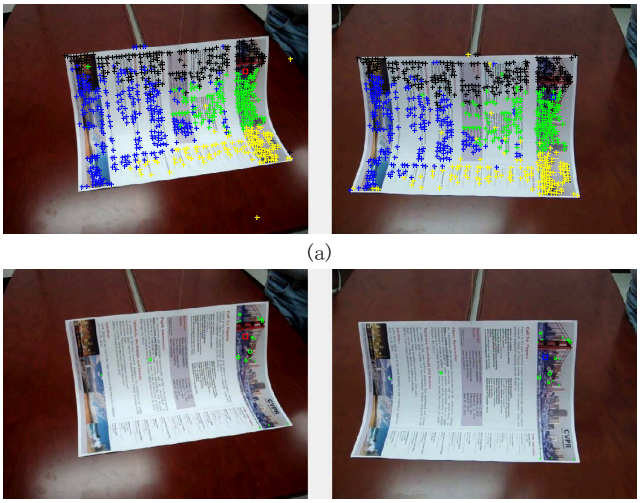


Figure 5. An instance for *global* refining method and our *local* refining method on *deformablepaper* scene. (a): the matching recovering result (blue-rectangle point) for the red-rectangle point using *global* refining method. "+" markers with different colors indicate different segments. (b): the matching recovering result of the red-rectangle point using our *local* refining method. The size of green "o" markers indicates the weight of the points contributing to the local rigid structure.

the compound trajectories jointed by the matches from a local rigid structure lie on a low dimensional linear subspace. We tackled the matching problem by first casting it into an element recovering problem and then using sparse representation and $\ell_1$ regularization techniques for the recovering procedure. We showed that the coefficients of sparse representation can be exploited to determine the local structure of each trajectory, which are further used for perspective refining. We analyzed our *local* refining method as well as the traditional *global* one and concluded that the *local* refining strategy can be used in more general cases as well as can ease the coupling problem which troubles *global* method.

## Acknowledgement

This work was supported in part by the Natural Science Foundation of China under Grants 60721003,60673106, 60875017, in part by the National 863 Hi-Tech Development Program of China under Grant 2008AA01Z123, and in part by the National Key Project of China under Grant 2009BAH40B03.

## References

[1] E. Amaldi and V. Kann. On the approximability of minimizing nonzero variables or unsatisfied relations in linear systems. *Theoretical Computer Science*, 209:237–260, 1998.

[2] H. Bay, T. Tuytelaars, and L. V. Gool. Surf: Speeded up robust features. *European Conference on Computer Vision*, 1:404–417, 2006.

[3] Y. Caspi, D. Simakov, and M. Irani. Feature-based sequence-to-sequence matching. *International Journal of Computer Vision*, 68(1):53–64, 2006.

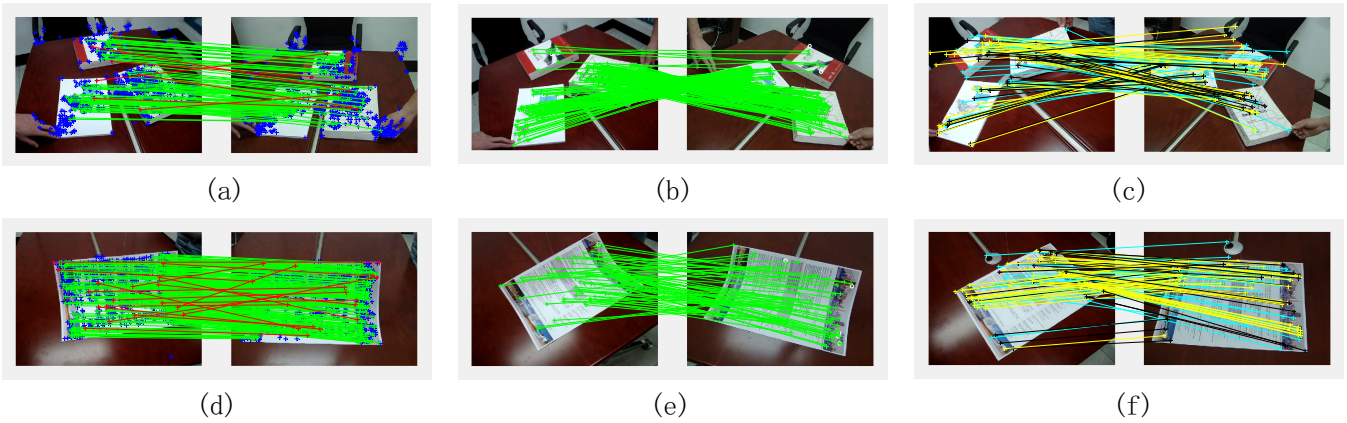



Figure 4. Spatial Matching results of the two scenes. (a)(b)(c) are the matching results of the *rotatingbooks* scene. The frames are at times $t = 1, 31, 90$ for the left video and $t = 1, 60, 103$ for the right video. (d)(e)(f) are the results of the *deformablepaper* scene. We show the frames at times $t = 1, 46, 95$ and $t = 1, 50, 101$ for two videos respectively. (a)(d) the extracted features and initial matches. The feature points are marked by blue "+" ; the red lines indicate the discarded outlying matches; the green lines represent correct initial matches obtained by using multi-scale SIFT descriptors after a discarding procedure. (b)(e) the obtained correspondences between "non-alone" trajectories (green lines) and the detected outliers (white "o" markers). (c)(f) a part of recovered matches for "alone" trajectories. Cyan lines indicate that the corresponding points on the right frame are invisible; yellow lines indicate that the corresponding points on the left frame are invisible; and black lines indicate that the corresponding points on both frames are invisible.


[4] J. Costeira and T. Kanade. A multibody factorization method for independently moving objects. *International Journal of Computer Vision*, 29(3):108–121, 1998.

[5] D. Donoho. For most large underdetermined systems of linear equations the minimal l1-norm solution is also the sparsest solution. *Communications on Pure and Applied Mathematics*, 59(6):797–829, 2006.

[6] E. Elhamifar and R. Vidal. Sparse subspace clustering. *IEEE Conference on Computer Vision and Pattern Recognition*, pages 2790–2797, 2009.

[7] C. Harris and M. Stephens. A combined corner and edge detection. *Proc. of the Fourth Alvey Vision Conference*, pages 147–151, 1988.

[8] R. Hartley. In defense of the eight-point algorithm. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(6):580–593, 1997.

[9] R. I. Hartley and A. Zisserman. Multiple view geometry in computer vision. *Cambridge University Press, second editionn*, 2004.

[10] T. Li, V. Kallem, D. Singaraju, and R. Vidal. Projective factorization of multiple rigid-body motions. *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–6, 2007.

[11] C. Liu, J. Yuen, A. Torralba, J. Sivic, and W. T. Freeman. Sift flow: Dense correspondence across different scenes. *European Conference on Computer Vision*, 2008.

[12] D. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(1):91–110, 2004.

[13] B. D. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. *IJCAI*, pages 674–679, 1981.

[14] K. Mikolajczyk, T. Tuytelaars, C. Schmid, A. Zisserman, J. Matas, F. Schaffalitzky, T. Kadir, and L. V. Gool. A comparison of affine region detectors. *International Journal of Computer Vision*, 65(1):43–72, 2005.

[15] J. Oliensis and R. Hartley. Iterative extensions of the sturm/triggs algorithm: Convergence and nonconvergence. *European Conference on Computer Vision*, pages 214–227, 2006.

[16] D. Pooley, M. Brooks, A. Hengel, and W. Chojnacki. A voting scheme for estimating the synchrony of moving-camera videos. *IEEE Conference on Image Processing*, pages 413–416, 2003.

[17] C. Rao, A.Gritai, and M. Shah. View-invariant alignment and matching of video sequences. *IEEE Conference on Computer Vision*, pages 939–945, 2003.

[18] S. Rao, R. Tron, R. Vidal, and Y. Ma. Motion segmentation via robust subspace separation in the presence of outlying, incomplete, or corrupted trajectories. *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8, 2008.

[19] R.L.Carceroni, F.L.C.Padua, G.A.M.R.Santos, and K.N.Kutulakos. Linear sequence-to-sequence alignment. *IEEE Conference on Computer Vision and Pattern Recognition*, pages 746–753, 2004.

[20] E. Rosten and T. Drummond. Fusing points and lines for high performance tracking. *IEEE International Conference on Computer Vision*, 2, 2005.

[21] E. Rosten and T. Drummond. Machine learning for high-speed corner detection. *European Conference on Computer Vision*, 1:430–443, 2006.

[22] J. Shi and C. Tomasi. Good features to track. *IEEE Conference on Computer Vision and Pattern Recognition*, pages 939–945, 1994.

[23] R. Szeliski. Image alignment and stitching: A tutorial. *Fundamental Trends in Computer Graphics and Vision*, 2:1–104, 2006.

[24] D. Ta, W. Chen, N. Gelfand, and K. Pulli. Surftrac: Efficient tracking and continuous object recognition using local feature descriptors. *IEEE Conference on Computer Vision and Pattern Recognition*, pages 2937 – 2944, 2009.

[25] R. Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society B*, 58(1):267–288, 1996.

[26] P. A. Tresadern and I. D. Reid. Video synchronization from human motion using rank constraints. *Computer Vision and Image Understanding*, 113:891–906, 2009.

[27] L. Wolf and A. Zomet. Wide baseline matching betweeen unsynchronized video seqeuences. *International Journal of Computer Vision*, 68(1):43–52, 2006.

[28] J. Wright, Y. Ma, J. Mairal, G. Sapiro, T. Huang, and S. Yan. Sparse representation for computer vision and pattern recognition. *To appear in Proceedings of the IEEE*, 2009.

[29] L. Zelnik-Manor and M. Irani. On single-sequence and multi-sequence factorizations. *International Journal of Computer Vision*, 67(3):313–326, 2006.