Multi-Image Matching via Fast Alternating Minimization

Xiaowei Zhou, Menglong Zhu, Kostas Daniilidis
GRASP Laboratory, University of Pennsylvania
{xiaowz,menglong,kostas}@seas.upenn.edu

Abstract

In this paper we propose a global optimization-based approach to jointly matching a set of images. The estimated correspondences simultaneously maximize pairwise feature affinities and cycle consistency across multiple images. Unlike previous convex methods relying on semidefinite programming, we formulate the problem as a low-rank matrix recovery problem and show that the desired semidefiniteness of a solution can be spontaneously fulfilled. The low-rank formulation enables us to derive a fast alternating minimization algorithm in order to handle practical problems with thousands of features. Both simulation and real experiments demonstrate that the proposed algorithm can achieve a competitive performance with an order of magnitude speedup compared to the state-of-the-art algorithm. In the end, we demonstrate the applicability of the proposed method to match the images of different object instances and as a result the potential to reconstruct category-specific object models from those images.

1. Introduction

Finding feature correspondences between two images is a fundamental problem in computer vision with various applications such as structure from motion, image registration, shape analysis, to name a few. While previous efforts were mostly focused on matching a pair of images, many tasks require to find correspondences across multiple images. A typical example is nonrigid structure from motion [3, 12], where one can hardly reconstruct a nonrigid shape from two frames. Furthermore, recent work has shown that leveraging multi-way information can dramatically improve matching results compared to pairwise matching [29, 16].

The most important constraint for joint matching is the cycle consistency, i.e., the composition of matches along a loop of images should be identity, as illustrated in Figure 1. Given pairwise matches, one can possibly identify true or false matches by checking all cycles in the image collection. But there are many difficulties for this approach [10]. For example, the input pairwise matches are often very noisy with many false matches and missing matches, and the features detected from different images may only have a partial overlap even if the same feature detector is applied [27]. Therefore, it is likely that very few consistent cycles can be found. Moreover, how to sample cycles is not straightforward due to the huge number of possibilities [16]. Recent work on joint matching has shown that, if all feature correspondences within multiple images are denoted by a large binary matrix, the cycle consistency can be translated into the fact that such a matrix should be positive semidefinite and low-rank [18, 29, 16]. Based on this observation, convex optimization-based algorithms were proposed, which achieved the state-of-the-art performances with theoretical guarantees [16, 10]. But these algorithms rely on semidefinite programming (SDP), which is not computationally efficient to handle image matching problems in practice.

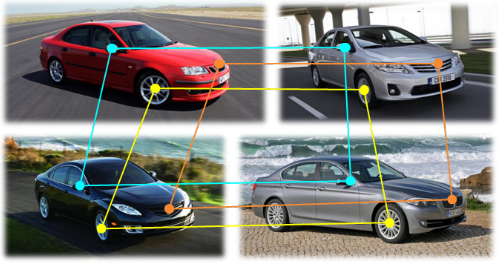

Figure 1. An illustration of consistent multi-image matching.

In this paper, we propose a novel algorithm for multi-image matching. The inputs to our algorithm are original similarities between feature descriptors such as SIFT descriptors [25] and deep features [35], or optimized affinities provided by existing graph matching solvers [22]. The outputs are feature correspondences between all pairs of images. Unlike many previous methods starting from quantized pairwise matches [29, 10], we postpone the decision until we optimize for both pairwise affinities and multi-image consistency. Instead of using SDP relaxation, we formulate the problem as a low-rank matrix recovery problem and employ the nuclear-norm relaxation for rank minimization (Section 4.1). We show that the positive semidefiniteness of a desired solution can be spontaneously fulfilled (Section 4.2). Moreover, we derive a fast alternating minimization algorithm to globally solve the problem in the low-dimensional variable space (Section 5). Besides validating our method on both simulated and real benchmark datasets, we also demonstrate the applicability of the proposed method combined with deep learning and graph matching to match images with different objects and reconstruct category-specific object models (Section 6).

2. Related work

The early work on joint matching aimed to select cycle-consistent matches and identify incorrect matches from bad cycles [41, 28]. The assumption for this family of methods is that correct matches are dominant in the raw input. Otherwise, it is difficult to find a sufficient number of closed cycles [16]. Some works proposed to use the cycle consistency as an explicit constraint for sparse feature matching [38, 37, 39, 36] or pixel-wise flow computation [42], but the resulting optimization problems are nonconvex and can hardly be solved globally. Recent results in [18, 16, 29] showed that the consistent matches could be extracted from the spectrum (top eigenvectors) of the matrix composed of all pairwise matches. The rationale behind this spectral technique is that the problem can be formulated as a quadratic integer program and relaxed into a generalized Rayleigh problem. But the relaxation assumes full feature correspondences (bijection) between images [29]. Recently, Huang and Guibas [16] proposed an elegant solution based on convex relaxation and derived the theoretical conditions for exact recovery. The result is further improved in [10] by assuming that the underlying rank of the variable matrix can be reliably estimated. In these works, the problem is formulated as SDP, which has a limited computational efficiency in real applications.

Regarding methodology, our work is inspired by the recent advances on low-rank matrix recovery which make use of convex relaxation [8, 7] and explore the underlying low-rank structure to accelerate computation [5, 15]. Our work is also related to some other problems that aim to find global estimates from pairwise estimates such as rotation averaging [14, 34] and model fusion [40].

3. Preliminaries and notation

Suppose we have n images and p_i features from each image i. The objective is to find feature correspondences between all pairs of images. Before introducing the proposed method, we first give a brief introduction to pairwise matching techniques and the definition of cycle consistency.

3.1. Pairwise matching

To match an image pair (i,j), one can compute similarities for all pairs of feature points from two images and store them in a matrix $\boldsymbol{S}_{ij} \in \mathbb{R}^{p_i \times p_j}$.

We represent the feature correspondences for image pair (i,j) by a partial permutation matrix $\boldsymbol{X}_{ij} \in \{0,1\}^{p_i \times p_j}$, which satisfies the doubly stochastic constraints:

$$\mathbf{0} \leq \boldsymbol{X}_{ij}\mathbf{1} \leq \mathbf{1}, \quad , \mathbf{0} \leq \boldsymbol{X}_{ij}^T\mathbf{1} \leq \mathbf{1}. \tag{1}$$

To find $\boldsymbol{X}_{ij}$, we can maximize the inner product between $\boldsymbol{X}_{ij}$ and $\boldsymbol{S}_{ij}$ subject to the constraints in (1) resulting in a linear assignment problem, which has been well studied and can be efficiently solved by the Hungarian algorithm.

In image matching, spatial rigidity is usually preferred, i.e., the relative location between two features in an image should be similar to that between their correspondences in the other image. This problem is well known as graph matching and formulated as a quadratic assignment problem (QAP). While QAP is NP-hard, many efficient algorithms have been proposed to solve it approximately, e.g., [22, 1, 11]. Those solvers basically relax the binary constraint on the permutation matrix, solve the optimization, and output the confidence of a candidate match being correct. We refer readers to the related literature for details. Here we aim to emphasize that the outputs of graph matching solvers are basically optimized affinity scores of candidate matches, which consider both feature similarity and spatial rigidity. We will use these scores (saved in $\boldsymbol{S}_{ij}$) as our input in some cases.

3.2. Cycle consistency

Some recent work proposed to use the cycle consistency as a constraint to match a bunch of images [29, 37, 10]. The cycle consistency can be described by

$$\boldsymbol{X}_{ij} = \boldsymbol{X}_{iz}\boldsymbol{X}_{zj}, \tag{2}$$

for any three images (i,j,z) and can be extended to the case with more images.

The recent results in [16, 29] show that the cycle consistency can be described more concisely by introducing a virtual "universe" that is defined as the set of unique features that appear in the image collection. Each point in the universe may be observed by several images and the corresponding image points should be matched. In this way, consistent matching should satisfy $\boldsymbol{X}_{ij} = \boldsymbol{A}_i\boldsymbol{A}_j^T$, where $\boldsymbol{A}_i \in \{0,1\}^{p_i \times k}$ denotes the map from Image i to the universe, k is the number of points in the universe, and $k \geq p_i$ for all i.

Suppose the correspondences for all $m = \sum_{i=1}^n p_i$ features in the image collection is denoted by $\boldsymbol{X} \in \{0,1\}^{m \times m}$:

$$\boldsymbol{X} = \begin{pmatrix} \boldsymbol{X}_{11} & \boldsymbol{X}_{12} & \cdots & \boldsymbol{X}_{1n} \\ \boldsymbol{X}_{21} & \boldsymbol{X}_{22} & \cdots & \boldsymbol{X}_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ \boldsymbol{X}_{n1} & \cdots & \cdots & \boldsymbol{X}_{nn} \end{pmatrix}, \tag{3}$$

and all $\boldsymbol{A}_i$s are concatenated as rows in a matrix $\boldsymbol{A} \in \{0,1\}^{m\times k}$. Then, one can write $\boldsymbol{X}$ as

$$\boldsymbol{X} = \boldsymbol{A}\boldsymbol{A}^T, \tag{4}$$

From (4), it is clear to see that a desired $\boldsymbol{X}$ should be both positive semidefinite and low-rank:

$$\boldsymbol{X} \succeq 0, \quad \text{rank}\,(\boldsymbol{X}) \leq k. \tag{5}$$

Using (5) the cycle consistency can be effectively imposed without checking all cycles of pairwise matches. Moreover, partial matching is allowed, while bijection needs to be assumed in (2).

4. Joint matching via rank minimization

Given affinity scores $\{\boldsymbol{S}_{ij} \mid 1 \leq i,j \leq n\}$, we aim to find globally consistent matches $\boldsymbol{X}$. Note that $\boldsymbol{S}_{ij}$ can be all-zero if matching is not performed for a pair (i,j). Moreover, affinity scores can be computed from either feature similarities or graph matching solvers according to specific scenarios, as described in Section 3.1.

4.1. Formulation

We formulate the problem as a low-rank matrix recovery problem. We maximize the inner product between $\boldsymbol{X}_{ij}$ and $\boldsymbol{S}_{ij}$ for all i and j as multiple linear assignment problems. At the same time, we minimize the rank of $\boldsymbol{X}$ to enforce the cycle consistency. We ignore the positive semidefinite constraint on $\boldsymbol{X}$ and will explain the reasons later.

To make the optimization tractable, we make the following relaxations: (1) $\boldsymbol{X}$ is treated as a real matrix $\boldsymbol{X} \in [0,1]^{m\times m}$ instead of a binary matrix, which is a general practice in solving matching problems. Experimentally, we found that the solution values were very close to 0 or 1 and could be stably quantized by a threshold of 0.5. This might be attributed to the existence of a linear term in the cost function [26]. (2) Rank of $\boldsymbol{X}$ is replaced by the nuclear norm $\|\boldsymbol{X}\|_*$ (sum of singular values), which is a tight convex relaxation proven to be very effective in various low-rank problems such as matrix completion [8] and robust principal component analysis [7].

The estimated $\boldsymbol{X}$ should be sparse since at most one value in each row of $\boldsymbol{X}_{ij}$ can be nonzero. To induce sparsity, we minimize the sum of values in $\boldsymbol{X}$. Combining all three terms, we obtain the following cost function:

$$\begin{aligned} f(\boldsymbol{X}) &= -\sum_{i=1}^{n}\sum_{j=1}^{n}\langle \boldsymbol{S}_{ij}, \boldsymbol{X}_{ij}\rangle + \alpha\langle \mathbf{1}, \boldsymbol{X}\rangle + \lambda\|\boldsymbol{X}\|_*, \\ &= -\langle \boldsymbol{S} - \alpha\mathbf{1}, \boldsymbol{X}\rangle + \lambda\|\boldsymbol{X}\|_*, \end{aligned} \tag{6}$$

where $\langle\cdot,\cdot\rangle$ denotes the inner product and $\boldsymbol{S} \in \mathbb{R}^{m\times m}$ is the matrix collecting all $\boldsymbol{S}_{ij}$s. α is the weight of sparsity, which can be interpreted as a threshold to remove small scores in $\boldsymbol{S}_{ij}$s. In our implementation, we normalize the scores to let them lie between 0 and 1 and empirically set $\alpha = 0.1$. λ controls the weight of the nuclear norm. We will discuss λ in Section 4.2 and Section 6.1.2.

Besides the doubly stochastic constraints in (1), additional constraints shall be imposed on $\boldsymbol{X}$ after relaxation:

$$\boldsymbol{X}_{ii} = \boldsymbol{I}_{p_i}, \quad 1 \leq i \leq n, \tag{7}$$

$$\boldsymbol{X}_{ij} = \boldsymbol{X}_{ji}^T, \quad 1 \leq i,j \leq n, i \neq j, \tag{8}$$

$$\mathbf{0} \leq \boldsymbol{X} \leq \mathbf{1}, \tag{9}$$

where (7) constrains self-matching to be identity, (8) constrains $\boldsymbol{X}$ to be symmetric, and (9) constrains the values in $\boldsymbol{X}$ to lie in $[0,1]$.

Finally, we obtain the following optimization problem:

$$\begin{aligned} &\min_{\boldsymbol{X}} \ \langle \boldsymbol{W}, \boldsymbol{X}\rangle + \lambda\|\boldsymbol{X}\|_*, \\ &\text{s.t.} \quad \boldsymbol{X} \in \mathcal{C}, \end{aligned} \tag{10}$$

where $\boldsymbol{W} = \alpha\mathbf{1} - \boldsymbol{S}$ and $\mathcal{C}$ denotes the set of matrices satisfying the constraints given in (1), (7), (8) and (9).

Upon our experimental observation, the result doesn't degrade noticeably when removing the doubly stochastic constraints in (1). This might be attributed to the existence of the sparsity regularizer. Therefore, we remove (1) in implementation to accelerate the computation.

4.2. Positive semidefiniteness

We ignore the positive semidefinite constraint for two reasons: (1) solving SDP is generally unscalable; (2) with the constraints in (7) and (8), the solution to (10) turns out to be nearly positive semidefinite if λ is sufficiently large.[1]

Suppose $\sigma_1, \cdots, \sigma_m$ are eigenvalues of $\boldsymbol{X}$. From (7), we have $X_{ii} = 1$ for all i, and $\sum_{i=1}^m \sigma_i = \text{trace}\,(\boldsymbol{X}) = m$, which implies that the sum of σ_is is fixed. From (8), we have $\boldsymbol{X}$ is symmetric, and σ_is are all real numbers. When we choose a large λ, $\|\boldsymbol{X}\|_* = \sum_{i=1}^m |\sigma_i|$ dominates the cost function, and a solution with all nonnegative σ_is will give the lowest cost, because $\sum_{i=1}^m |\sigma_i| \geq \sum_{i=1}^m \sigma_i = m$ and the equality holds iff. $\sigma_i \geq 0$ for all i.

The boundness $\|\boldsymbol{X}\|_* \geq m$ also implies that the solution to (10) will be insensitive to λ when λ is sufficiently large, and then minimizing the nuclear norm is equivalent to adding a positive semidefinite constraint. The effect of λ is experimentally illustrated in Section 6.1.2.

[1] We use the term "nearly positive semidefinite" to refer to the property that the negative eigenvalues of a matrix, if there exist, are negligible compared to the norm of the matrix.

$[n, p]$	$m = np$	MatchLift	Partial SVD	MatchALS
$[5, 20]$	1.0×10^2	0.005	0.016	0.001
$[10, 20]$	2.0×10^2	0.009	0.016	0.003
$[20, 20]$	4.0×10^2	0.034	0.033	0.009
$[20, 100]$	2.0×10^3	1.472	2.023	0.283
$[20, 500]$	1.0×10^4	166.8	219.3	9.804

Table 1. The CPU time (seconds) for one iteration of MatchALS, MatchLift [10] and partial SVD [21]. n, p, and m denote the number of images, the number of points per image, and the dimension of $\boldsymbol{X}$, respectively. We set $k = 2p$ for MatchALS.

5. Fast alternating minimization

5.1. Optimization in the low-rank space

The nuclear norm minimization in (10) is convex and the state-of-the-art methods to solve this family of problems are the proximal method [30] or ADMM [2] based on iterative singular value thresholding [6]. However, singular value decomposition (SVD) needs to be performed in each iteration, which is extremely expensive even for a medium-sized problem. For instance, if there are 20 images with 500 features per image to match, we have to optimize for an $10,000 \times 10,000$ matrix. A single SVD for such a matrix takes hundreds of seconds on a typical PC even if a partial SVD solver [21] is used. See Table 1 and Section 5.3.

Fortunately, recent results on low-rank optimization have shown that one can solve the problem more efficiently via a change of variables $\boldsymbol{X} = \boldsymbol{A}\boldsymbol{B}^T$ [5, 15], where $\boldsymbol{A}, \boldsymbol{B} \in \mathbb{R}^{m \times k}$ are new variables with a smaller dimension $k < m$. More importantly, the change of variables will not introduce additional local minima if k is larger than the rank of the original solution. This result was originally derived for low-rank SDP [4, 20] but also applies here since a nuclear-norm minimization problem can be rewritten as SDP [31].

Inspired by these works, we propose the following low-rank factorization-based formulation in order to leverage the underlying low dimensionality of our problem:

$$\begin{aligned} &\min_{\boldsymbol{A},\boldsymbol{B}} \ \langle \boldsymbol{W}, \boldsymbol{A}\boldsymbol{B}^T \rangle + \lambda \|\boldsymbol{A}\boldsymbol{B}^T\|_*, \\ &\text{s.t.} \quad \boldsymbol{A}\boldsymbol{B}^T \in \mathcal{C}. \end{aligned} \tag{11}$$

Moreover, with the following equation [31],

$$\|\boldsymbol{X}\|_* = \min_{\boldsymbol{A},\boldsymbol{B}: \boldsymbol{A}\boldsymbol{B}^T = \boldsymbol{X}} \frac{1}{2}\left(\|\boldsymbol{A}\|_F^2 + \|\boldsymbol{B}\|_F^2\right), \tag{12}$$

we finally obtain the following formulation:

$$\begin{aligned} &\min_{\boldsymbol{A},\boldsymbol{B}} \ \langle \boldsymbol{W}, \boldsymbol{A}\boldsymbol{B}^T \rangle + \frac{\lambda}{2}\|\boldsymbol{A}\|_F^2 + \frac{\lambda}{2}\|\boldsymbol{B}\|_F^2, \\ &\text{s.t.} \quad \boldsymbol{A}\boldsymbol{B}^T \in \mathcal{C}. \end{aligned} \tag{13}$$

The selection of matrix dimension k is critical to the success of change of variables, while it directly affects the computational complexity. We will first provide the algorithm, analyze its complexity and then discuss the selection of k.

5.2. Algorithms

The problem in (13) is not straightforward to solve due to the constraint on the product of variables. Instead, we rewrite the problem as

$$\begin{aligned} &\min_{\boldsymbol{X},\boldsymbol{A},\boldsymbol{B}} \ \langle \boldsymbol{W}, \boldsymbol{X} \rangle + \frac{\lambda}{2}\|\boldsymbol{A}\|_F^2 + \frac{\lambda}{2}\|\boldsymbol{B}\|_F^2, \\ &\text{s.t.} \quad \boldsymbol{X} = \boldsymbol{A}\boldsymbol{B}^T, \ \ \boldsymbol{X} \in \mathcal{C}, \end{aligned} \tag{14}$$

and apply the ADMM [2] to solve (14).

The augmented Lagrangian of (14) reads:

$$\begin{aligned} \mathcal{L}_\mu\left(\boldsymbol{X}, \boldsymbol{A}, \boldsymbol{B}, \boldsymbol{Y}\right) = &\langle \boldsymbol{W}, \boldsymbol{X} \rangle + \frac{\lambda}{2}\|\boldsymbol{A}\|_F^2 + \frac{\lambda}{2}\|\boldsymbol{B}\|_F^2 \\ &+ \langle \boldsymbol{Y}, \boldsymbol{X} - \boldsymbol{A}\boldsymbol{B}^T \rangle + \frac{\mu}{2}\|\boldsymbol{X} - \boldsymbol{A}\boldsymbol{B}^T\|_F^2 \end{aligned} \tag{15}$$

where $\boldsymbol{Y}$ is the dual variable and μ is a parameter controlling the step size in optimization. We keep the constraint $\boldsymbol{X} \in \mathcal{C}$ since it can be easily handled as we will show later. Then, the ADMM alternately updates each primal variable by minimizing $\mathcal{L}_\mu$ and updates the dual variable via gradient ascent while fixing all other variables. The overall algorithm is summarized in Algorithm 1.

Algorithm 1: Multi-Image Matching via Alternating Least Squares (MatchALS)

```
  Input: Pairwise affinity scores $\boldsymbol{S}$
  Output: Globally consistent matches $\boldsymbol{X}$
1 randomly initialize $\boldsymbol{A}$ and $\boldsymbol{B}$, $\boldsymbol{Y} = \boldsymbol{0}$ ;
2 $\boldsymbol{W} = \alpha\boldsymbol{1} - \boldsymbol{S}$ ;
3 while not converged do
4     $\boldsymbol{A} \leftarrow \left(\boldsymbol{X} + \frac{1}{\mu}\boldsymbol{Y}\right)\boldsymbol{B}\left(\boldsymbol{B}^T\boldsymbol{B} + \frac{\lambda}{\mu}\boldsymbol{I}\right)^\dagger$ ;
5     $\boldsymbol{B} \leftarrow \left(\boldsymbol{X} + \frac{1}{\mu}\boldsymbol{Y}\right)\boldsymbol{A}\left(\boldsymbol{A}^T\boldsymbol{A} + \frac{\lambda}{\mu}\boldsymbol{I}\right)^\dagger$ ;
6     $\boldsymbol{X} \leftarrow \mathcal{P}_\mathcal{C}\left(\boldsymbol{A}\boldsymbol{B}^T - \frac{1}{\mu}(\boldsymbol{W} + \boldsymbol{Y})\right)$ ;
7     $\boldsymbol{Y} \leftarrow \boldsymbol{Y}^k + \mu\left(\boldsymbol{X} - \boldsymbol{A}\boldsymbol{B}^T\right)$ ;
8 end
9 quantize $\boldsymbol{X}$ with a threshold equal to 0.5.
```

Minimizing $\mathcal{L}_\mu$ over $\boldsymbol{A}$ turns out to be a regularized least squares problem with a closed-form solution given in Step 4 in Algorithm 1. The update of $\boldsymbol{B}$ can be solved similarly. The update of $\boldsymbol{X}$ requires to solve:

$$\min_{\boldsymbol{X} \in \mathcal{C}} \|\boldsymbol{X} - \boldsymbol{A}\boldsymbol{B}^T + \frac{1}{\mu}\left(\boldsymbol{W} + \boldsymbol{Y}\right)\|_F^2, \tag{16}$$

and the solution turns out to be a projection to $\mathcal{C}$. Since the constraints in $\mathcal{C}$ are all linear, the projection can be solved conveniently. We denote the solution by $\mathcal{P}_{\mathcal{C}}(\cdot)$ and leave the details to the supplementary material.

5.3. Computational complexity

The time complexity of an iteration in Algorithm 1 is dominated by matrix multiplication that requires $O(m^2k)$ flops[2]. We compare it to the state-of-the-art algorithm MatchLift [10], which is based on SDP. The time complexity of an iteration in MatchLift is dominated by the eigenvalue decomposition that requires $O(m^3)$ flops. As m is much larger than k, MatchALS has a lower complexity compared to MatchLift. Moreover, matrix multiplication is parallelizable and has been inherently multithreaded in Matlab, while the parallelization of eigenvalue decomposition is an open problem. Both MatchALS and MatchLift are based on ADMM and require similar numbers of iterations to converge upon our observation.

The CPU time for some problem sizes is shown in Table 1. The algorithms are implemented in Matlab and tested on a PC with an Intel i7 3.4GHz CPU and 8G RAM. We also compare the time cost of partial SVD using PROPACK [21], a toolbox widely used to solve large-scale matrix completion problems [24]. In partial SVD, only r leading singular vectors are computed, which is much faster than full SVD when r/m is extremely small. But it is not efficient for a relatively large r. In our problem, r should be larger than the true rank and we test partial SVD with $r = p$ in Table 1.

5.4. Selection of k and rank reduction

From the previous subsections we see that k determines the complexity of MatchALS and k should be larger than the rank of true solution, i.e. the size of universe. While some spectral techniques have been proposed in previous work for rank estimation [10], we found that the estimation was inaccurate when the input was noisy and incomplete. Fortunately, our solution doesn't depend on k if k is larger than the underlying true rank (demonstrated later in Figure 3). A heuristic choice is to set $k = 2\hat{r}$, where $\hat{r}$ is a rough estimate of the size of universe.

In real applications, there are likely to be many isolated features in each image which don't have any correspondence in other images. However, the constraint in (7) implies that every image feature must be matched to a point in the universe. To see this, recall that we hope $\boldsymbol{X} = \boldsymbol{A}\boldsymbol{A}^T$ in (4). If diagonal values of $\boldsymbol{X}$ are all ones, every row of $\boldsymbol{A}$ has a unit norm, which indicates a match to the universe. Therefore, the size of universe is dramatically increased by those isolated features, and consequently a very large k needs to be selected, which severely increases the computation. To address this issue, we loose the constraint in (7) to be

$$\begin{aligned} &\text{trace}\left(\boldsymbol{X}\right) = m', \\ &\text{off-diagnal values}\{\boldsymbol{X}_{ii}\} = \boldsymbol{0},\ 1 \le i \le n, \end{aligned} \qquad (17)$$

where $m' \le m$ is a predefined constant. When $m' = m$, (17) is reduced to (7). When $m' < m$, we allow some rows and columns in $\boldsymbol{X}$ to be null, which is most likely to happen for the rows and columns corresponding to the isolated features, since "switching" them off will not lose many affinity scores but be able to reduce the nuclear norm immediately. By using such a "rank reduction" strategy, the algorithm can automatically prune the isolated features and reduce the size of universe, which enables us to select a smaller k for better computational efficiency. We set $m' = m$ in simulation since there is no isolated feature and $m' = 0.7m$ in real experiments.

6. Experiments

6.1. Simulation

We evaluate the performance of the proposed method using synthesized data. Given a permutation matrix $\boldsymbol{X}$ and the ground truth $\boldsymbol{X}^*$, we measure the error rate by intersection over union:

$$1 - \frac{|\tau(\boldsymbol{X}) \cap \tau(\boldsymbol{X}^*)|}{|\tau(\boldsymbol{X}) \cup \tau(\boldsymbol{X}^*)|}, \qquad (18)$$

where τ denotes the matches defined by a permutation matrix and $|\cdot|$ means the size of a set.

6.1.1 Matching errors

We follow the settings in [10] to evaluate the performance of MatchALS and compare it to alternative methods. The size of universe is fixed as 20 points and in each image a random sample of the points are observed with a probability denoted by ρ_o. The number of images is denoted by n. Then, the ground-truth pairwise matches are established, and random corruptions are simulated by removing some true matches and adding some false matches to achieve an error rate of ρ_e. Finally, the corrupted permutation matrix is fed into Algorithm 1 as the input affinity scores.

We evaluate the performance of MatchALS under various ρ_o, ρ_e and n. We compare MatchALS to two related methods: MatchLift [10] and the spectral method [29]. Both of the alternative methods require to know the size of universe and we provide the true value $r^* = 20$. For MatchALS parameters, we set $k = 2r^*$ and $\lambda = 50$.

The output error rates under various settings are shown in Figure 2. When the number of images is sufficiently large, all methods can achieve nearly exact recovery even

[2] The detail is given in the supplementary material

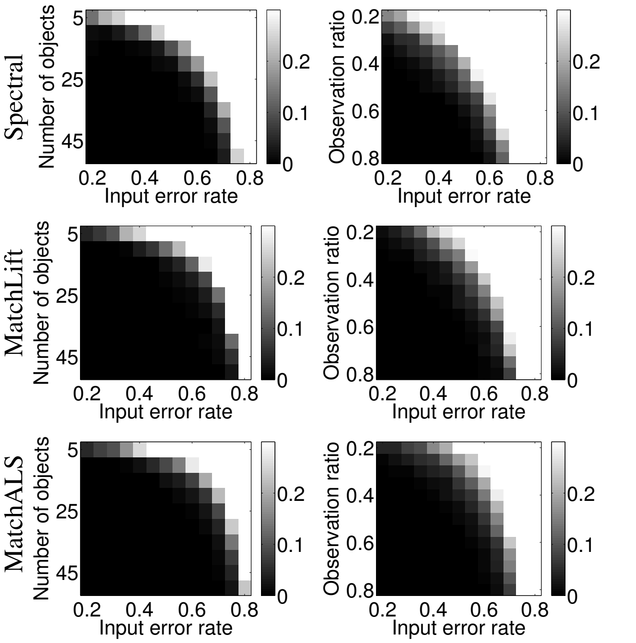

Figure 2. The 2D plot of matching errors under various problem settings for the spectral method [29], MatchLift [10] and the proposed MatchALS. In the left column, the number of images n and the input error rate ρ_e are varying, while the observation ratio $\rho_o = 0.6$. In the right column, ρ_o and ρ_e are varying, while $n = 20$. Lower intensity indicates smaller error and overall a larger dark region indicates a better performance.

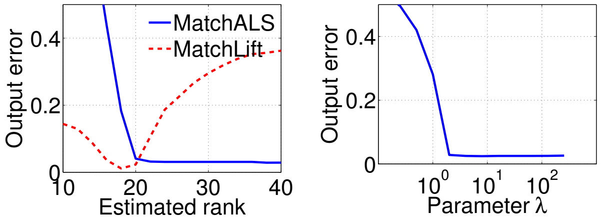

Figure 3. The estimation error versus the input rank $\hat{r}$ and the weight of nuclear norm λ. The true rank $r^* = 20$. Here we set $k = \hat{r}$ for MatchALS.

if the input error rate is larger than 50%, which demonstrates the power of joint matching. MatchALS and MatchLift achieve very similar performances and outperform the spectral method especially when the observation ratio is small. Compared to MatchLift, the proposed method obtains a competitive performance without exactly knowing the true rank and requires much less computation time.

6.1.2 Sensitivity to parameters

The sensitivity of MatchALS to the parameters in (13) is illustrated in Figure 3. The figure shows that MatchALS is insensitive to the predefined dimension of factor matrices k when k is larger than the true rank r^*, as we explained in Section 5.1. When $k < r^*$, the problem in (13) is no longer equivalent to the original convex problem in (10), and consequently the alternating minimization fails. In practice, we choose $k = 2\hat{r}$ as a compromise between safety and efficiency. The right panel in Figure 3 illustrates that the algorithm is insensitive to λ when λ is sufficiently large as we explained in Section 4.2. In all our experiments, we set $\lambda = 50$.

6.2. Real experiments

6.2.1 Graffiti datasets

We evaluate the performance of our algorithm on six benchmark datasets from the Graffiti datasets[3]. In each dataset, there are six images of a scene with various image transformations such as viewpoint change, blurring, illumination variation, etc.

We detect 1000 affine covariant features [27] with SIFT [25] descriptors from each image using the VLFeat library [32]. For each image pair (i, j), we compute the inner products between feature descriptors as affinity scores and only keep the scores larger than 0.7 and collect them in S_{ij}. If the ratio between the first and the second largest scores in a row/column is smaller than 1.1, we set all scores in this row/column to be zero in order to remove indistinctive features. After computing all $\boldsymbol{S}_{ij}$, we remove the features that have candidate matches in less than two images since they have no contribution to joint matching. Finally, we input the affinity scores to Algorithm 1 to obtain the optimized joint matches.

For evaluation, we adopt the metric used in [10]: for a testing point in an image, we calculate the distance between its estimated correspondence and the true correspondence in another image. If the distance is smaller than a threshold, we regard that a correct match is found for this testing point. Then, we plot the percentages of testing points with correct matches versus the threshold values and obtain a curve analogous to a precision-recall curve. If a testing point is not aligned with any detected point, its estimated correspondence is obtained by interpolation. In this experiment, we use all detected feature points in the first image as testing points and evaluate the matches from the first image to the other five images. True correspondences are computed from the homography matrices provided in the datasets.

The performance curves on three datasets are shown in Figure 4. A curve closer to the upper-left corner indicates a better performance. The area under curve and computation time for all datasets are summarized in Table 2. All of the joint matching methods achieve obvious improvements compared to the original pairwise matching. MatchALS and MatchLift perform similarly and outperforms the spectral method, which coincides with the observation in simulation. Regarding computation time, MatchALS achieves

[3] http://www.robots.ox.ac.uk/ vgg/data/data-aff.html

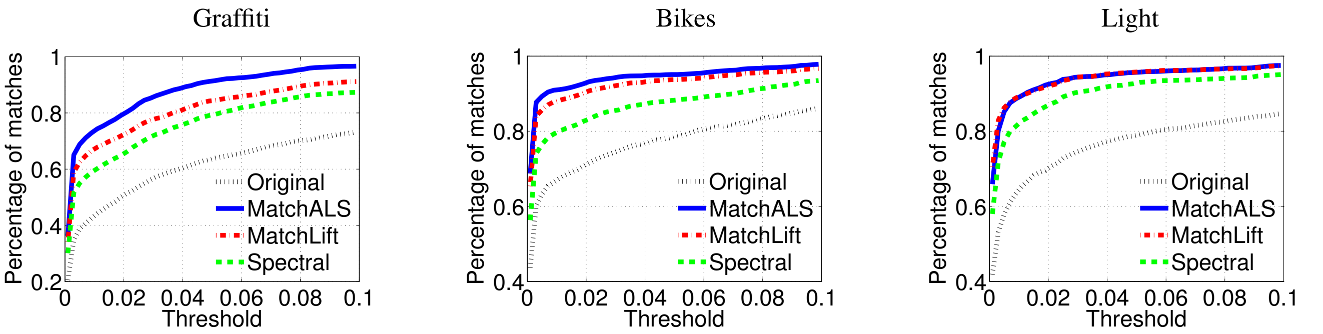

Figure 4. The performance curves on the Graff, Bikes and Light datasets. The y-axis shows the percentages of correct matches. The x-axis shows the distance threshold over the image width. Please see Section 6.2.1 for details. Four methods are compared: MatchALS, MatchLift [10], the spectral method [29], and the original pairwise matching. The areas under curves for all six datasets are given in Table 2.

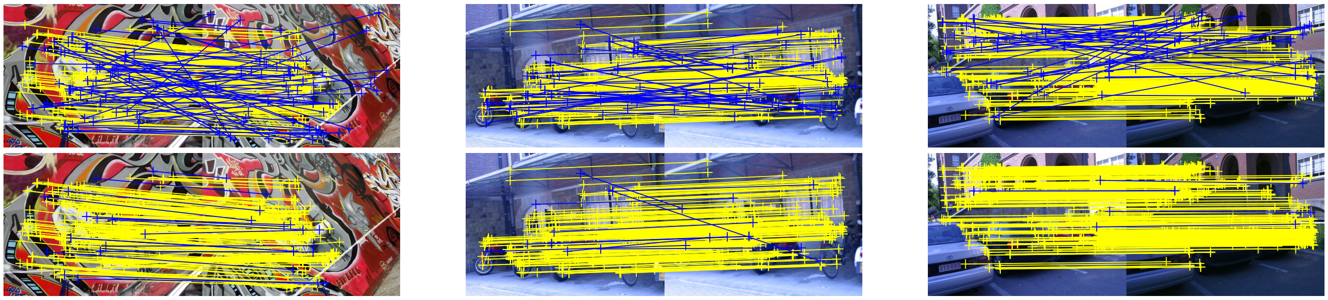

Figure 5. The matches between the 1st and the 4th images on the Graff, Bikes and Light datasets. Best viewed in color. The true matches and false matches are shown in yellow and blue, respectively. The top and bottom rows correspond to the results of pairwise matching and joint matching by MatchALS, respectively.

	Original	MatchALS	Spectral	MatchLift
Graffiti	60.2%	**87.3%**	75.6%	80.6%
Bikes	76.8%	**94.3%**	86.7%	92.5%
Boat	86.2%	**93.9%**	87.7%	91.7%
Light	76.0%	93.9%	90.0%	**94.0%**
Bark	71.7%	**92.2%**	91.2%	90.0%
UBC	88.0%	**97.0%**	92.9%	96.8%
Time	-	85.8	86.4	2518.4

Table 2. The matching scores and the average computation time (seconds) on the Graffiti datasets. The score is calculated as the area under the curve shown in Figure 4.

a remarkable speedup (∼30 times on average) compared to MatchLift.

We select three image pairs to visually demonstrate the effect of joint matching in Figure 5. A match with a deviation less than five pixels from the ground truth is declared as true. Clearly, the joint matching can prune the false matches (fewer blue lines), complete some missing matches (denser yellow lines), and achieve almost correct matching for these image pairs with large disparities in viewpoints, blurring and illumination changes.

6.2.2 Matching different objects

Recent years have witnessed growing interest in reconstructing category-specific object models from single images, which is still an open problem [33, 9]. Among a series of challenges, feature matching for different object instances is the foremost and previous work usually assumed that correspondences of some keypoints were given [33, 9]. In this section, we demonstrate the applicability of joint matching to solve this problem.

We use the FG3DCar datasets [23] and try to match the images of different car models in the same category (e.g., sedan or SUV). Following the general practice in object reconstruction [33, 9], we assume segmentation is provided such that background can be ignored, and we only match images with similar views. We select nine sedans and eight SUVs and match two sets of images separately. Note that the car models are all different from each other. See Figure 6 for examples.

To exact descriptive features we first detect image edges by the structured forests [13] and sample a number of points on the edges with constant spacing. On average, we obtain ∼600 feature points for each image. Since the object appearance is changed from image to image and the features are automatically extracted, the matching is extremely difficult. Inspired by recent works [35, 9], we adopt

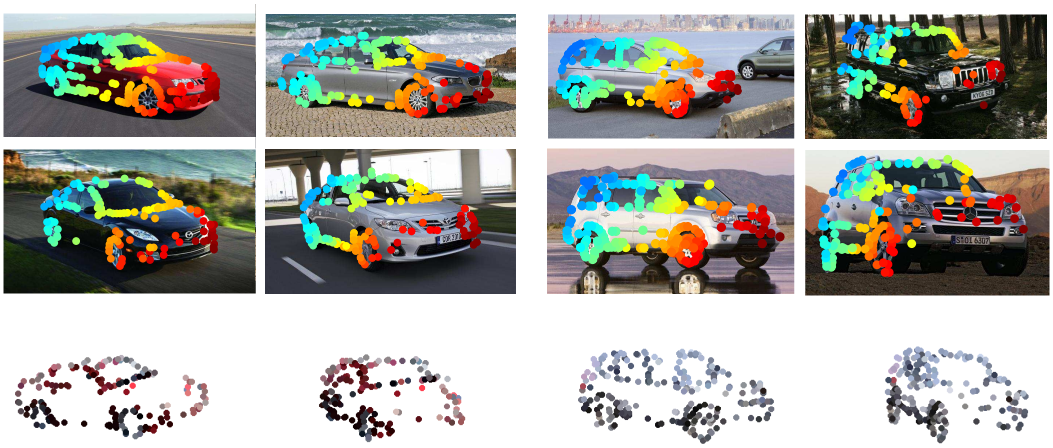

Figure 6. Matching different cars. Best viewed in color. Left: the correspondences of sedan images and the reconstruction. Right: the correspondences of SUV images and the reconstruction. Only four selected images are shown for each image set. Note that the cars in images are all different and the feature points are automatically detected. The markers with the same color indicate the matched feature points. The 3D reconstruction is rendered with the colors in the first image and visualized in two viewpoints.

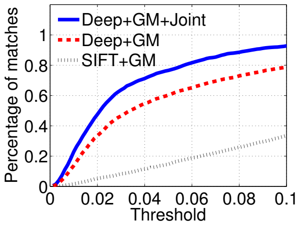

Figure 7. The performance curve of car image matching. "Deep" represents deep features. "GM" denotes graph matching. "Joint" means joint matching using the proposed method.

deep features, i.e., middle-layer responses of convolutional neural nets (CNN), as descriptors for feature matching. More specifically, we use the publicly available deep learning toolbox Caffe [17] and the pre-trained CNN Alexnet [19]. We feed a 192 × 192 patch around each feature point forward through the Alexnet. The center columns of conv4 and conv5 layers are concatenated and normalized to form a 640 dimensional feature vector. To leverage the prior on object rigidity, we use pairwise graph matching solved by the Reweighted Random Walk algorithm [11] and collect the output scores of candidate matches as affinity scores. Then, we delete the points with candidate matches in less than two images and run MatchALS.

We adopt the same metric introduced in Section 6.2.1 for quantitative evaluation and use the manually-annotated landmarks provided in the datasets as ground truth. The result on the sedan images is shown in Figure 7. Matching with SIFT features fails since local image patterns are different for two cars. Graph matching with deep features obtains a much better performance, which is further improved by the proposed joint matching algorithm. We obtain a very similar result on the SUV images, which is not plotted.

The results are visualized in Figure 6. The corresponding parts of cars are basically matched in spite of the large differences in appearances and viewpoints. Note that the features are automatically detected and therefore not fully overlapped for two images. For a simple demonstration, we run rigid reconstruction from the estimated feature correspondences by triangulation with an orthographic camera model and the viewpoints provided in the dataset. Despite some noises and missing points, we can clearly see the 3D structures of a sedan and a SUV. We believe that more appealing reconstructions can be obtained by using sophisticated reconstruction techniques and more information such as object silhouette and surface smoothness, while they are out of the scope of this paper.

7. Conclusion

In this paper, we proposed a practical solution to multi-image matching. We use pairwise feature similarities or graph matching scores as input and obtain accurate matches by an efficient algorithm that globally optimizes for both feature affinities and cycle consistency of matches. The experiments not only validate the effectiveness of the proposed method but also demonstrate that joint matching is a promising approach to matching images with different object instances as the first step towards reconstructing object models from crowd-sourced image collections. As future work, we would like to explore more applications and incremental algorithms for joint matching.

Acknowledgments: Grateful for support through the following grants: NSF-DGE-0966142, NSF-IIS-1317788, NSF-IIP-1439681, NSF-IIS-1426840, ARL RCTA W911NF-10-2-0016, and ONR N000141310778

References

[1] A. C. Berg, T. L. Berg, and J. Malik. Shape matching and object recognition using low distortion correspondences. In *CVPR*, 2005. 2

[2] S. Boyd. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends in Machine Learning*, 3(1):1–122, 2010. 4

[3] C. Bregler, A. Hertzmann, and H. Biermann. Recovering non-rigid 3d shape from image streams. In *CVPR*, 2000. 1

[4] S. Burer and R. D. Monteiro. Local minima and convergence in low-rank semidefinite programming. *Mathematical Programming*, 103(3):427–444, 2005. 4

[5] R. Cabral, F. De la Torre, J. P. Costeira, and A. Bernardino. Unifying nuclear norm and bilinear factorization approaches for low-rank matrix decomposition. In *ICCV*, 2013. 2, 4

[6] J. Cai, E. Candès, and Z. Shen. A singular value thresholding algorithm for matrix completion. *SIAM Journal on Optimization*, 20:1956, 2010. 4

[7] E. Candès, X. Li, Y. Ma, and J. Wright. Robust principal component analysis? *Journal of ACM*, 58(3):11, 2011. 2, 3

[8] E. Candès and B. Recht. Exact matrix completion via convex optimization. *Foundations of Computational Mathematics*, 9(6):717–772, 2009. 2, 3

[9] J. Carreira, A. Kar, S. Tulsiani, and J. Malik. Virtual view networks for object reconstruction. *arXiv preprint arXiv:1411.6091*, 2014. 7

[10] Y. Chen, L. Guibas, and Q. Huang. Near-optimal joint object matching via convex relaxation. In *International Conference on Machine Learning*, 2014. 1, 2, 4, 5, 6, 7

[11] M. Cho, J. Lee, and K. M. Lee. Reweighted random walks for graph matching. In *ECCV*, 2010. 2, 8

[12] Y. Dai, H. Li, and M. He. A simple prior-free method for non-rigid structure-from-motion factorization. In *CVPR*, 2012. 1

[13] P. Dollár and C. L. Zitnick. Structured forests for fast edge detection. In *ICCV*, 2013. 7

[14] R. Hartley, J. Trumpf, Y. Dai, and H. Li. Rotation averaging. *IJCV*, 103(3):267–305, 2013. 2

[15] T. Hastie, R. Mazumder, J. Lee, and R. Zadeh. Matrix completion and low-rank svd via fast alternating least squares. *arXiv preprint arXiv:1410.2596*, 2014. 2, 4

[16] Q.-X. Huang and L. Guibas. Consistent shape maps via semidefinite programming. *Computer Graphics Forum*, 32(5):177–186, 2013. 1, 2

[17] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell. Caffe: Convolutional architecture for fast feature embedding. *arXiv preprint arXiv:1408.5093*, 2014. 8

[18] V. G. Kim, W. Li, N. J. Mitra, S. DiVerdi, and T. A. Funkhouser. Exploring collections of 3d models using fuzzy correspondences. *ACM Transactions on Graphics*, 31(4):54, 2012. 1, 2

[19] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, 2012. 8

[20] B. Kulis, A. C. Surendran, and J. C. Platt. Fast low-rank semidefinite programming for embedding and clustering. In *International Conference on Artificial Intelligence and Statistics*, 2007. 4

[21] R. Larsen. Propack-software for large and sparse svd calculations., 2004. Available at http://soi.stanford.edu/~rmunk/PROPACK/. 4, 5

[22] M. Leordeanu and M. Hebert. A spectral technique for correspondence problems using pairwise constraints. In *ICCV*, 2005. 1, 2

[23] Y.-L. Lin, V. I. Morariu, W. Hsu, and L. S. Davis. Jointly optimizing 3d model fitting and fine-grained classification. In *ECCV*, 2014. 7

[24] Z. Lin, M. Chen, and Y. Ma. The augmented Lagrange multiplier method for exact recovery of corrupted low-rank matrices. *arXiv preprint arXiv:1009.5055*, 2010. 5

[25] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *IJCV*, 60(2):91–110, 2004. 1, 6

[26] J. Maciel and J. P. Costeira. A global solution to sparse correspondence problems. *T-PAMI*, 25(2):187–199, 2003. 3

[27] K. Mikolajczyk, T. Tuytelaars, C. Schmid, A. Zisserman, J. Matas, F. Schaffalitzky, T. Kadir, and L. Van Gool. A comparison of affine region detectors. *IJCV*, 65(1-2):43–72, 2005. 1, 6

[28] A. Nguyen, M. Ben-Chen, K. Welnicka, Y. Ye, and L. Guibas. An optimization approach to improving collections of shape maps. *Computer Graphics Forum*, 30(5):1481–1491, 2011. 2

[29] D. Pachauri, R. Kondor, and V. Singh. Solving the multi-way matching problem by permutation synchronization. In *NIPS*, 2013. 1, 2, 5, 6, 7

[30] N. Parikh and S. Boyd. Proximal algorithms. *Foundations and Trends in Optimization*, 1(3):123–231, 2013. 4

[31] B. Recht, M. Fazel, and P. Parrilo. Guaranteed minimum-rank solutions of linear matrix equations via nuclear norm minimization. *SIAM Review*, 52(3):471–501, 2010. 4

[32] A. Vedaldi and B. Fulkerson. VLFeat: An open and portable library of computer vision algorithms. http://www.vlfeat.org/, 2008. 6

[33] S. Vicente, J. Carreira, L. Agapito, and J. Batista. Reconstructing pascal voc. In *CVPR*, 2014. 7

[34] L. Wang and A. Singer. Exact and stable recovery of rotations for robust synchronization. *Information and Inference*, 2(2):145–193, 2013. 2

[35] P. Weinzaepfel, J. Revaud, Z. Harchaoui, and C. Schmid. Deepflow: Large displacement optical flow with deep matching. In *ICCV*, 2013. 1, 7

[36] J. Yan, M. Cho, H. Zha, X. Yang, and S. Chu. Multi-graph matching via affinity optimization with graduated consistency regularization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2015. 2

[37] J. Yan, Y. Li, W. Liu, H. Zha, X. Yang, and S. M. Chu. Graduated consistency-regularized optimization for multi-graph matching. In *ECCV*, 2014. 2

[38] J. Yan, Y. Tian, H. Zha, X. Yang, Y. Zhang, and S. M. Chu. Joint optimization for consistent multiple graph matching. In *ICCV*, 2013. 2

[39] J. Yan, J. Wang, H. Zha, X. Yang, and S. Chu. Consistency-driven alternating optimization for multigraph matching: A unified approach. *IEEE Transactions on Image Processing*, 24(3):994–1009, 2015. 2

[40] G. Ye, D. Liu, I. Jhuo, and S. Chang. Robust late fusion with rank minimization. In *CVPR*, 2012. 2

[41] C. Zach, M. Klopschitz, and M. Pollefeys. Disambiguating visual relations using loop constraints. In *CVPR*, 2010. 2

[42] T. Zhou, Y. Jae Lee, S. X. Yu, and A. A. Efros. Flowweb: Joint image set alignment by weaving consistent, pixel-wise correspondences. In *CVPR*, 2015. 2