Tensor Power Iteration for Multi-Graph Matching

Xinchu Shi[1], Haibin Ling[2*], Weiming Hu[1], Junliang Xing[1], Yanning Zhang[3]
[1]CAS Center for Excellence in Brain Science and Intelligence Technology, National Laboratory of Pattern Recognition, Institute of Automation, CAS, Beijing, China
[2]Department of Computer and Information Sciences, Temple University, Philadelphia, USA
[3]School of Computer Science, Northwestern Polytechnical University, Xian, China
{xcshi, wmhu, jlxing}nlpr.ia.ac.cn, hbling@temple.edu, ynzhang@nwpu.edu.cn

Abstract

Due to its wide range of applications, matching between two graphs has been extensively studied and remains an active topic. By contrast, it is still under-exploited on how to jointly match multiple graphs, partly due to its intrinsic combinatorial intractability. In this work, we address this challenging problem in a principled way under the rank-1 tensor approximation framework. In particular, we formulate multi-graph matching as a combinational optimization problem with two main ingredients: unary matching over graph vertices and structure matching over graph edges, both of which across multiple graphs. Then we propose an efficient power iteration solution for the resulting NP-hard optimization problem. The proposed algorithm has several advantages: 1) the intrinsic matching consistency across multiple graphs based on the high-order tensor optimization; 2) the free employment of powerful high-order node affinity; 3) the flexible integration between various types of node affinities and edge/hyper-edge affinities. Experiments on diverse and challenging datasets validate the effectiveness of the proposed approach in comparison with state-of-the-arts.

1. Introduction

Finding the correspondences across multiple sets of visual features is an essential problem in computer vision, and has a wide spectrum of applications such as object categorization [14, 23], shape analysis [4], feature tracking [24], and action recognition [5, 36]. A large amount of graph matching algorithms have been proposed in past decades, most of them are designed for the matching between two graphs, while *multiple graph matching* (MGM) is left under-exploited. Many scenarios in computer vision involve sets of multiple similar/same targets, thus MGM has wide perspectives in multi-view geometry and 3D reconstruction [15, 2], 3D shape matching [18, 17], cross temporal object analysis [28, 35], pattern recognition [6, 30, 26], and so on.

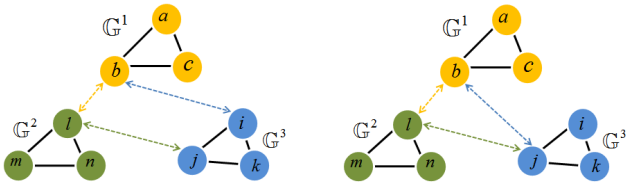

Figure 1. Matching consistency illustration (with 3-graph matching as an example). Left: inconsistency arises with the contradictory node matches such as $b \leftrightarrow l$, $l \leftrightarrow j$ and $i \leftrightarrow b$. Right: the consistent matching triple $b \leftrightarrow l$, $l \leftrightarrow j$ and $j \leftrightarrow b$. Note that the implicit assumption of the one-to-one mapping constraint is used.

A simple strategy for multi-graph matching is to first perform pairwise graph matching for graph pairs independently and then merge the results. However, such strategy meets the known *matching inconsistency* problem which is illustrated in Figure 1. Most existing multi-graph matching approaches use pairwise matching as the key building block. These methods inherit advantages of off-the-shelf pairwise graph matching algorithms, but meanwhile suffer from the matching inconsistency problem.

Some recent studies aim to address issues in both pairwise matching optimization and matching consistency. For example, Pachauri et al. [26] apply spectral analysis for smoothing the matching inconsistency and Yan et al. [33] design an elegant iterative solution to balance the matching score and consistency. More examples are provided in the Sec. 2. While two-graph matching leverages pairwise between-object affinities, MGM deals with affinities of order higher than two. In many cases, such high order affinity can not be decomposed into, or even approximated by, pairwise ones. Previous MGM algorithms, however, often

*Correspondence author.

restrict themselves on pairwise affinities and thus ignore the important high-order information across multiple objects.

To address the above issue, we formulate MGM as a combinatorial optimization dealing with two types of high-order information: the vertex affinity over a group of vertices and the structure affinity over a set of hyper-edges. By treating these affinities as high-order tensors, we formulate MGM in the low rank tensor approximation framework and design a tensor power iteration algorithm in this work.

The proposed solution enjoys serveral advantages from various components. First, working on high-order affinity tensors naturally avoids the matching inconsistency problem. Second, it is expediently to explore various types of powerful high-order affinities. Moreover, the framework is very flexible to integrate the vertex affinities and edge/hyper-edge affinities. To evaluate the proposed solution, we test it along with state-of-the-arts on several popular benchmarks, and it generates very promising results on all the experiments.

To summarize, the main contributions in this work include: (1) a new MGM optimization formulation based on high-order node and edge/hyper-edge affinities; (2) a novel tensor power iteration algorithm for solving the high-order MGM optimization; and (3) a sequence of convincing experimental validations.

2. Related work

The problem of matching two graphs has been extensively studied in the literature [10]. Graph matching is traditionally formulated as an optimization problem, and there are various algorithms such as the Graduated Assignment algorithm (GAGM) [16], the Integer Projected Fixed Point (IPFP) method [22], the Spectral Matching methods [21, 11], the path-following algorithms [37, 25, 39], and so on. Both the pairwise edge affinity and the hyper-edge affinity can be exploited in graph matching, the pairwise edge affinity is generally sensitive to the scaling and rotation, while hyper-edge affinity explores high-order structure information and can be robust to certain geometric transformations [38, 13, 20].

While matching two graphs has been intensively studied, multi-graph matching (MGM), which we focus on in this paper, receives relatively less attention. The state-of-the-arts MGM algorithms can be roughly divided into two categories, affinity-driven and the consistency-driven approaches. The affinity-driven approaches [30, 31, 33] formulate MGM as an optimization problem, in which the objective is usually the summation of the overall pairwise matching affinity scores [31], sometimes supplemented by a matching consistency regularizer [33]. For example, the GAGM method [16] is applied repeatedly across graph pairs to achieve cross graph matching [31]. In the work [33], the pairwise matching score is regularized with the inconsistency measure, and followed by the alternative optimization iterations. The consistency-driven approaches [26, 34] put more attention on the matching consistency. An iterative optimization solution is proposed in [34] with the rigid matching consistency constraint. The approach [26] pools all pairwise matching solutions into a single matrix, then the globally consistent array of matchings are estimated with the spectral smoothing algorithm.

The proposed MGM algorithm is very different from previous work. The key novelty lies in the new tensor approximation based optimization framework, the proposed approach works directly on the global matching space. Such novelty allows us to deal with high-order matching information, which can effectively boost the matching performance as demonstrated in the experiments.

On the algorithm side, our study is inspired by recent work on using tensor-based high-order matching, in particular [13, 7, 28, 27]. The algorithm in [13] uses high-order tensor for hyper-graph matching between two graphs, and the framework in [28] uses tensor approximation for multi-target tracking, which can be viewed as a degenerated case of MGM (no edges involved). Both algorithms exploit tensor power iteration for solving their models. By contrast, our work targets different problems. Moreover, our work deals with high-order information not only across multiple graphs, but also across hyper-edges.

3. Problem Formulation

We first introduce some notations used in this paper, then provide a short retrospective on pairwise graph matching, and finally formulate MGM used in our study.

3.1. Notation

A graph is generally represented as $\mathbb{G} = (\mathbb{V}, \mathbb{E}, \mathbb{A})$, where $\mathbb{V}$ denotes the node set, $\mathbb{E}$ the edge set and $\mathbb{A}$ the attribute set. The attribute set $\mathbb{A}$ includes the node features such as the position, appearance information, as well as the edge properties such as the distance and orientation.

To reduce the complexity of formulations and derivations, we use various notations for conciseness. Through this paper, by default we use font such as $\mathbb{X}$ for a set, $\mathcal{X}$ a tensor, $\mathbf{X}$ a matrix, $\mathbf{x}$ a vector and x a scalar number.

The element of a datum is consistently indexed by the subscripts. For example, the i-th entry of a vector $\mathbf{a}$ is denoted by a_i, the (i, j)-th entry of a matrix $\mathbf{A}$ by a_{ij} and the $(j_1, j_2, \dots, j_K)$-th entry of a K-th order tensor $\mathcal{A}$ by $a_{j_1:j_K}$. By contrast, a datum term itself is indexed by its superscript, e.g., the k-th graph is represented as $\mathbb{G}^k = \left(\mathbb{V}^k, \mathbb{E}^k, \mathbb{A}^k\right)$.

Finally, when lower-case italic letters $(i, j, \dots)$ are used for indices in summation, they by default run from 1 to their corresponding upper-case ones $(I, J, \dots)$. For example, $\sum_i$ denotes for $\sum_{i=1}^{I}$.

3.2. Pairwise Graph Matching

Given two graphs $\mathbb{G}^1 = (\mathbb{V}^1, \mathbb{E}^1, \mathbb{A}^1)$ and $\mathbb{G}^2 = (\mathbb{V}^2, \mathbb{E}^2, \mathbb{A}^2)$. The numbers of vertices in $\mathbb{V}^1$ and $\mathbb{V}^2$ are I_1 and I_2 respectively. A solution of matching is defined as a subset of possible correspondences between $\mathbb{V}^1$ and $\mathbb{V}^2$, that can be represented by an assignment matrix $\mathbf{X} \in \{0,1\}^{I_1 \times I_2}$. If vertex $v^1_{i_1} \in \mathbb{V}^1$ matches vertex $v^2_{i_2} \in \mathbb{V}^2$, then $x_{i_1 i_2} = 1$, and $x_{i_1 i_2} = 0$ otherwise. We further denote $\mathbf{x} \in \{0,1\}^{I_1 I_2}$ as the vectorized replica of $\mathbf{X}$.

The objective function $f(\mathbf{x})$ measures the overall similarity between graphs $\mathbb{G}^1$ and $\mathbb{G}^2$. It has two components, the unary similarity $s_V(v^1_{i_1}, v^2_{i_2})$ for a vertex pair $(v^1_{i_1} \in \mathbb{V}^1, v^2_{i_2} \in \mathbb{V}^2)$, and the compatibility/similarity $s_E(e^1_{i_1 i'_1}, e^2_{i_2 i'_2})$ over an edge pair $(e^1_{i_1 i'_1} \in \mathbb{E}^1, e^2_{i_2 i'_2} \in \mathbb{E}^2)$. The two similarities are popularly formulated into a symmetric matrix $\mathbf{B} \in \mathbb{R}^{I_1 I_2 \times I_1 I_2}$ whose diagonal terms are taken from s_V and non-diagonal terms from s_E. This way, the objective function has the form

$$\begin{aligned} f_2(\mathbf{x}) &= \sum_{i_1, i_2} x_{i_1 i_2} b_{i_1 i_2, i_1 i_2} + \sum_{i_1, i_2, i'_1, i'_2} x_{i_1 i_2} b_{i_1 i_2, i'_1 i'_2} x_{i'_1 i'_2} \\ &= \mathbf{x}^\top \mathbf{B} \mathbf{x}. \end{aligned} \tag{1}$$

For hyper-graph matching, the pairwise edge similarity in (1) takes a more complicated form. Finally, the graph matching problem can be expressed as to find an assignment solution $\mathbf{x}^*$ that maximizes $f_2(\mathbf{x})$ with the following matching constraints

$$\begin{cases} \sum_{i_1} x_{i_1 i_2} \le 1, & \forall i_2 = 1, 2, \dots, I_2 \\ \sum_{i_2} x_{i_1 i_2} \le 1, & \forall i_1 = 1, 2, \dots, I_1 \\ x_{i_1 i_2} \in \{0,1\}, & \forall i_1 = 1, \dots, I_1; i_2 = 1, \dots, I_2 \end{cases} \tag{2}$$

3.3. Multi-graph Matching

MGM extends pairwise graph matching to multiple graphs. Given $K+1$ graphs $\{\mathbb{G}^k = (\mathbb{V}^k, \mathbb{E}^k, \mathbb{A}^k) : k = 0, 1, \dots, K\}$, the problem of multi-graph matching (MGM) is to find a subset of possible correspondences across all $K+1$ graphs, and the solution can be denoted by a $K+1$-th order assignment tensor $\mathcal{X} \in \mathbb{R}^{I_0 \times \cdots \times I_K}$, where $I_k (k = 0, \dots, K)$ is the cardinality of node set $\mathbb{V}^k$. The element $x_{i_0 : i_K}$ in $\mathcal{X}$ denotes the group-wise matching across the vertices $\{v^0_{i_0}, \dots, v^K_{i_K} : v^k_{i_k} \in \mathbb{V}^k\}$. Specifically, $x_{i_0 : i_K} = 1$ means that the correspondence is selected in the solution, and 0 otherwise. The vertex affinity of the group-wise matching is denoted by $a_{i_0 : i_K}$, and the structural affinity over the edge set $\{e^0_{i_0 i'_0}, \dots, e^K_{i_K i'_K}\}$ by $s_{\mathbb{E}}(e^0_{i_0 i'_0}, \dots, e^K_{i_K i'_K})$. With these definitions, the objective function $f_{\text{mgm}}(\mathcal{X})$ for MGM can be formulated as

$$\begin{aligned} f_{\text{mgm}}(\mathcal{X}) = &\sum_{\mathbb{I}} x_{i_0 : i_K} a_{i_0 : i_K} + \\ &\lambda \sum_{\mathbb{I}} \sum_{\mathbb{I}'} x_{i_0 : i_K} s_{\mathbb{E}} \left(e^0_{i_0 i'_0}, \dots, e^K_{i_K i'_K} \right) x_{i'_0 : i'_K}, \end{aligned} \tag{3}$$

where $\mathbb{I} \doteq \{i_0, \dots, i_K\}$ and $\mathbb{I}' \doteq \{i'_0, \dots, i'_K\}$ are the index sets, and λ is the weighting parameter. The MGM problem is to find a solution that maximizes $f_{\text{mgm}}(\mathcal{X})$ subject to the following group-wise matching constraints

$$\begin{cases} \sum\limits_{\mathbb{I} \setminus \{i_k\}} x_{i_0 : i_K} \le 1, & \forall k = 0, \dots, K; i_k = 1, \dots, I_k \\ x_{i_0 : i_K} \in \{0,1\}, & \forall i_k = 1, \dots, I_k \end{cases} \tag{4}$$

where '$\setminus$' denotes set difference.

The objective function $f_{\text{mgm}}(\mathcal{X})$ is the high-dimensional extension of the objective function $f_2(\mathbf{x})$ in pairwise graph matching. The first component in $f_{\text{mgm}}(\mathcal{X})$ is the multi-dimensional assignment score extended from the linear assignment; while the second part measures the affinity score depending on edges from all graphs.

4. Proposed framework and algorithm

The maximization of the objective (3) subject to (4) is an NP hard problem in general, and the exact solution is extremely difficult to be obtained. In this section, we first bring in some relaxations for this optimization, and then propose a tensor based approximate solution.

4.1. Relaxed Optimization

We use three main relaxations, high-order matching decomposition, edge affinity relaxation and real-valued relaxation, to make the MGM objective in (3) tractable.

High-order matching decomposition: The high-order group-wise matching $x_{i_0 : i_K}$ results in an extremely challenging optimization in the very high-dimensional solution space. To reduce the computation complexity, we decompose the high-order matching $x_{i_0 : i_K}$ into pairwise ones as

$$x_{i_0 : i_K} = x^1_{i_0 i_1} x^2_{i_1 i_2} \dots x^K_{i_{K-1} i_K} = x^1_{j_1} x^2_{j_2} \dots x^K_{j_K}, \tag{5}$$

where $x^k_{i_{k-1} i_k}$ is an element in the assignment matrix $\mathbf{X}^k$ for matching graphs $\mathbb{G}^{k-1}$ and $\mathbb{G}^k$, $x^k_{j_k} \in \mathbf{x}^k$ is the vectorized replica of $x^k_{i_{k-1} i_k}$ with relation between j_k and (i_{k-1}, i_k) defined by

$$\begin{cases} j_k \doteq (i_{k-1} - 1) \times I_k + i_k, \\ \underline{j_k} \doteq i_k, \ \overline{j_k} \doteq i_{k-1}. \end{cases} \tag{6}$$

Note that, for notation simplicity, the elements of both the vector and matrix assignments use the same scalar symbol (e.g., x) but with different subscripts (e.g., x_j for an element of $\mathbf{x}$ and $x_{i_1 i_2}$ an element in $\mathbf{X}$).

The decomposition (5) is valid since the group-wise matching $x_{i_0 : i_K}$ is true if and only if all pairwise items $\{x^k_{i_{k-1} i_k}, k = 1, \cdots, K\}$ are true.

Edge affinity relaxation: We approximate the group-wise edge affinity $s_{\mathbb{E}}\left(e^0_{i_0 i'_0}, \dots, e^K_{i_K i'_K}\right)$ with pairwise edge

affinities as

$$s_{\mathbb{E}}\left(e^0_{i_0 i'_0},\dots,e^K_{i_K i'_K}\right)\approx\sum_{k=1}^{K}s^k_{i_{k-1}i_k,i'_{k-1}i'_k}, \tag{7}$$

where $s^k_{i_{k-1}i_k,i'_{k-1}i'_k}$ denotes the pairwise similarity between edges $e^{k-1}_{i_{k-1}i'_{k-1}}$ and $e^k_{i_k i'_k}$.

Note that such approximation does not hold for group-wise vertex affinity, i.e., $a_{i_0:i_K}$, which is critical to encode high-order group-wise information across multiple graphs.

Real-valued relaxation: The constraints (4) make MGM a difficult integer programming. We relax them to real-valued domain and get the new constraints as

$$x_{i_0:i_K}\in[0,1]\,. \tag{8}$$

The pairwise matching constraints $x^k_{i_{k-1}i_k}\in[0,1]$ are relaxed in the same way.

With the above relaxations, we obtain a new MGM formulation as:

$$\begin{aligned}\arg\max_{\mathbb{X}} f_{\text{mgm}}(\mathbb{X})=&\sum_{\mathbb{I}}a_{i_0:i_K}\prod_{k=1}^{K}x^k_{i_{k-1}i_k}+\\&\lambda\sum_k\sum_{\mathbb{P}^k}x^k_{i_{k-1}i_k}s^k_{i_{k-1}i_k,i'_{k-1}i'_k}x^k_{i'_{k-1}i'_k},\end{aligned} \tag{9}$$

$$\text{s.t.}\begin{cases}\sum_{i_{k-1}}x^k_{i_{k-1}i_k}\le 1, & \forall i_k=1,\dots,I_k\\ \sum_{i_k}x^k_{i_{k-1}i_k}\le 1, & \forall i_{k-1}=1,\dots,I_{k-1}\\ x^k_{i_{k-1}i_k}\in[0,1]\,, & \forall i_{k-1},i_k\end{cases} \tag{10}$$

where $\mathbb{P}^k=\{i_{k-1},i_k,i'_{k-1},i'_k\}$ is the index set, and $\mathbb{X}=\{\mathbf{X}^1,\dots,\mathbf{X}^K\}$ is the set of pairwise matching.

4.2. Tensor based Optimization

Similar to matrix-vector product form in the pairwise graph matching objective (1), we now reformulate the MGM objective (9) as the tensor-vector product formulation and exploit the tensor tools for the final optimization.

Firstly, we take advantage of the vectorized matching item $x^k_{j_k}$ other than its matrix replica $x^k_{i_{k-1}i_k}$. Secondly, the high-order affinity $a_{i_0:i_K}$ is rearranged by the vectorized indices. A new high-order affinity $c_{j_1:j_K}$ is represented as

$$c_{j_1:j_K}=\begin{cases}a_{\overline{j_1:j_K}\underline{j_K}}, & \text{if } \underline{j_k}=\overline{j_{k+1}},\forall k=1,\dots,K-1\\ 0, & \text{otherwise.}\end{cases} \tag{11}$$

In this way, each $a_{i_0:i_K}$ has a corresponding element $c_{j_1:j_K}$, while the elements $c_{j_1:j_K}$ with no counterpart in $a_{i_0:i_K}$ are set to 0. The same strategy has been used in [28] where details can be found. Similarly, the vectorized index-based pairwise edge affinity can be re-written as

$$s^k_{j_k j'_k}=s^k_{i_{k-1}i_k,i'_{k-1}i'_k}. \tag{12}$$

Using the above two reformulations, we denote the vertex affinity as a K-th order tensor $\mathcal{C}=(c_{j_1:j_K})$, and the edge affinity matrix as $\mathbf{S}^k=(s^k_{j_k j'_k})$. Now we can reformulate the objective in (9) in the form of tensor-vector product as

$$\begin{aligned}f_{\text{mgm}}(\mathbb{X})=&\sum_{\mathbb{J}}c_{j_1:j_K}\prod_{k=1}^{K}x^k_{j_k}+\lambda\sum_{k=1}^{K}\sum_{j_k,j'_k}x^k_{j_k}s^k_{j_k j'_k}x^k_{j'_k}\\=&\,\mathcal{C}\times_1\mathbf{x}^1\dots\times_K\mathbf{x}^K+\lambda\sum_{k=1}^{K}{\mathbf{x}^k}^\top\mathbf{S}^k\mathbf{x}^k,\end{aligned} \tag{13}$$

where $\mathbb{J}=\{j_1,\dots,j_K\}$ is the index set, and '$\times_k$' denotes the *k-mode* product of the tensor with a vector [12].

In this work, we make a step further to formulate the problem as the *multi-hyper-graph matching*[1] by replacing the pairwise edge affinity with a hyper-edge one. Suppose that $x^k_{j_k}$, $x^k_{l_k}$ and $x^k_{h_k}$ represent the matches on node pairs $v^{k-1}_{i_{k-1}}\leftrightarrow v^k_{i_k}$, $v^{k-1}_{p_{k-1}}\leftrightarrow v^k_{p_k}$ and $v^{k-1}_{q_{k-1}}\leftrightarrow v^k_{q_k}$ respectively. Hyper-edges $e^{k-1}_{i_{k-1}p_{k-1}q_{k-1}}$ and $e^k_{i_k p_k q_k}$ are based on the node triples $\{v^{k-1}_{i_{k-1}},v^{k-1}_{p_{k-1}},v^{k-1}_{q_{k-1}}\}$ and $\{v^k_{i_k},v^k_{p_k},v^k_{q_k}\}$ respectively, and the hyper-edge affinity between $e^{k-1}_{i_{k-1}p_{k-1}q_{k-1}}$ and $e^k_{i_k p_k q_k}$ is defined as $s^k_{j_k l_k h_k}\in\mathcal{S}^k$. The objective of multi-hyper-graph matching is extended from (13) and computed as

$$\begin{aligned}f_{\text{mgm}}(\mathbb{X})=&\sum_{\mathbb{J}}c_{j_1:j_K}\prod_{k=1}^{K}x^k_{j_k}+\lambda\sum_{k=1}^{K}\sum_{\mathbb{L}^k}s^k_{j_k l_k h_k}x^k_{j_k}x^k_{l_k}x^k_{h_k}\\=&\,\mathcal{C}\times_1\mathbf{x}^1\dots\times_K\mathbf{x}^K+\lambda\sum_{k=1}^{K}\mathcal{S}^k\times_1\mathbf{x}^k\times_2\mathbf{x}^k\times_3\mathbf{x}^k,\end{aligned} \tag{14}$$

where $\mathbb{L}^k=\{j_k,l_k,h_k\}$ is the index set for hyper-edges.

Now our MGM problem is to find the solution that maximizes the objective (14) subject to constraints in (10).

4.3. Power Iteration Solution

The objective (14) has two tensor-vector products, and each product is exactly a *rank-1 tensor approximation* (R1TA) objective [28]. Inspired by the work [28], we use tensor power iteration to solve the above MGM optimization. While there exist power iteration solutions for R1TA problems such as in [12, 28, 13], our task is different in two aspects. First, the objective in our optimization is a combination of two R1TA components. Second, the proposed optimization has the row/colum ℓ_1 unit norm constraints (10), while classical constraints in R1TA are the ℓ_2 unit norm ones.

[1] We emphasize that the proposed algorithm is general for either graph or hyper-graph matching, i.e., using either edge or hyper-edge affinity.

Algorithm 1: Tensor power iteration based multi-graph matching

Input: the node affinity $\mathcal{C}: c_{j_1:j_K}$.
the hyper-edge affinity $\mathcal{S}^k: s^k_{j_k l_k h_k}, \forall k = 1, \ldots, K$.
Output: pairwise matching
$\mathbb{X} = \{\mathbf{X}(k_1, k_2): k_1, k_2 = 0, \ldots, K\}$.

1 Initialize $\mathbf{x}^1, \ldots, \mathbf{x}^K$;
2 **for** *iter* $= 1$ *to max_iter* **do**
3 **for** $k = 1$ *to* K **do**
4 /* power iteration */
5 **for** $i_{k-1} = 1$ *to* I_{k-1} **do**
6 **for** $i_k = 1$ *to* I_k **do**
7 $j_k = (i_{k-1} - 1) \times I_k + i_k$.
8 $\phi^k_{i_{k-1} i_k} \doteq x^k_{j_k} \sum_{\mathbb{J} \setminus \{j_k\}} c_{j_1:j_K} x^1_{j_1} \ldots x^K_{j_K}$.
9 $\psi^k_{i_{k-1} i_k} \doteq x^k_{j_k} \sum_{l_k, h_k} s^k_{j_k l_k h_k} x^k_{l_k} x^k_{h_k}$.
10 **end**
11 $C = \sum_{i_k} \left(\phi^k_{i_{k-1} i_k} + \lambda \psi^k_{i_{k-1} i_k} \right)$
12 $x^k_{i_{k-1} i_k} = \left(\phi^k_{j_k} + \lambda \psi^k_{j_k} \right) / C, \ i_k = 1, \cdots, I_k$
13 **end**
14 /* alternative row/column normalization[2] */
15 **while** X^k *does not converge* **do**
16 **for** $i_{k-1} = 1$ *to* I_{k-1} **do**
17 $Q = \sum_{i_k} x^k_{i_{k-1} i_k}$
18 $x^k_{i_{k-1} i_k} = x^k_{i_{k-1} i_k} / Q, \ i_k = 1, \cdots, I_k$
19 **end**
20 **for** $i_k = 1$ *to* I_k **do**
21 $Q = \sum_{i_{k-1}} x^k_{i_{k-1} i_k}$
22 $x^k_{i_{k-1} i_k} = x^k_{i_{k-1} i_k} / Q, \ i_{k-1} = 1, \cdots, I_{k-1}$
23 **end**
24 **end**
25 **end**
26 **end**
27 Discretize $\mathbf{x}^k (1 \le k \le K)$ through the Hungarian algorithm.
28 Get the pairwise matching $\mathbf{X}(k_1, k_2)$ for graphs $\mathbb{G}^{k_1}$ and $\mathbb{G}^{k_2}, \forall k_1, k_2 = 0, \ldots, K$
29 **return** $\mathbb{X} = \{\mathbf{X}(k_1, k_2): k_1, k_2 = 0, \ldots, K\}$.

Inspired by the algorithm proposed in the work [28] for multi-target tracking, we design an efficient power iteration solution for the optimization of (14) as listed in Algorithm 1. The algorithm adapts the 'block-update' strategy, in which each time a section of the solution evolves itself with the rest fixed. In this way, all vectors $\{\mathbf{x}^1, \ldots, \mathbf{x}^K\}$ are updated in turn in the iteration.

The proposed algorithm has three key ingredients: 1) updating the node affinity score (i.e., $\phi^k_{i_{k-1} i_k}$) and hyper-edge affinity score (i.e., $\psi^k_{i_{k-1} i_k}$), listed in line $8 - 9$; 2) the ℓ_1 norm power iteration listed in line $11 - 12$; and 3) the alternative row/column normalizations in line $15 - 24$. Alternative row/column normalizations make the matching matrix double-stochastic according to the Sinkhorn's theorem [29], as has been used in previous work on graph matching [11, 13].

With the proposed power iteration solution, there are a sequence of pairwise matching outputs $\mathbf{x}^k (k \in \{1, \ldots, K\})$. However, this matching is real-valued and must be discretized. We first reshape $\mathbf{x}^k$ into its matrix replica $\mathbf{X}^k$, then the real-valued matching matrix is further discretized using the Hungarian algorithm [19] to meet the constraint (4).

Given the pairwise matching $\mathbf{X}^k (k \in \{1, \ldots, K\})$, the group-wise matching $x_{i_0:i_K}$ is received by the formulation (5). With the global matching $x_{i_0:i_K}$, any pairwise matching $\mathbf{X}(k_1, k_2)$ between $\mathbb{G}^{k_1}$ and $\mathbb{G}^{k_2}$ is derived naturally by index searching.

4.4. Implementation Details

In real scenarios, a graph often has noisy structures with false/outlier nodes. These nodes should not be mapped to any real node. In the proposed algorithm, we supplement each graph with dummy nodes that are allowed to match with non-dummy nodes in the other graphs. The affinity involving the dummy node is set as $0.3 \sim 0.5$ to suppress the erroneous matches.

In the rest of this section, we give the definitions of affinities (i.e., $a_{i_0:i_K}$ and $s^k_{j_k l_k h_k}$) used in the optimization. We represent each node with the shape context [3] feature. Suppose the shape context of node $v^k_{i_k}$ is the column vector $\mathbf{y}^k_{i_k}$, then all features from the node set $\{v^0_{i_0}, v^1_{i_1}, \ldots, v^K_{i_K}\}$ are stacked into a matrix $\mathbf{Y}_{i_0:i_K} = \left[\begin{array}{cccc} \mathbf{y}^0_{i_0} & \mathbf{y}^1_{i_1} & \cdots & \mathbf{y}^K_{i_K} \end{array} \right]$, and the high-order node affinity $a_{i_0:i_K}$ is computed as

$$a_{i_0:i_K} = \frac{\text{eigen}\left(\mathbf{Y}_{i_0:i_K}, 1\right)}{\sum_d \text{eigen}\left(\mathbf{Y}_{i_0:i_K}, d\right)}, \tag{15}$$

where $\text{eigen}\left(\mathbf{Y}_{i_0:i_K}, d\right)$ denotes the d-th eigenvalue of the matrix $\mathbf{Y}_{i_0:i_K}$ in descending order. This affinity measures the compactness of the feature set.

Suppose $x^k_{j_k}$ is the matching between nodes $v^{k-1}_{i_{k-1}}$ and $v^k_{i_k}$, $x^k_{l_k}$ is the matching between nodes $v^{k-1}_{p_{k-1}}$ and $v^k_{p_k}$, $x^k_{h_k}$ matches nodes $v^{k-1}_{q_{k-1}}$ and $v^k_{q_k}$. Then the hyper-edge on the node triple $\{v^{k-1}_{i_{k-1}}, v^{k-1}_{p_{k-1}}, v^{k-1}_{q_{k-1}}\}$ has the triangle features as $\{\theta^{k-1}_1, \theta^{k-1}_2, \theta^{k-1}_3\}$. While the other hyper-edge on the nodes $\{v^k_{i_k}, v^k_{p_k}, v^k_{q_k}\}$ has its triangle features as $\{\theta^k_1, \theta^k_2, \theta^k_3\}$. This hyper-edge construction is illustrated in Figure 2. Then the hyper-edge affinity $s^k_{j_k l_k h_k}$ is defined as

$$s^k_{j_k l_k h_k} = \exp\left(- \frac{\sum_{i=1}^3 \left(\sin \theta^k_i - \sin \theta^{k-1}_i \right)^2}{2\sigma^2} \right), \tag{16}$$

[2] By supplementing dummy nodes in two graphs, we can always make the numbers of the nodes in two graphs the same.

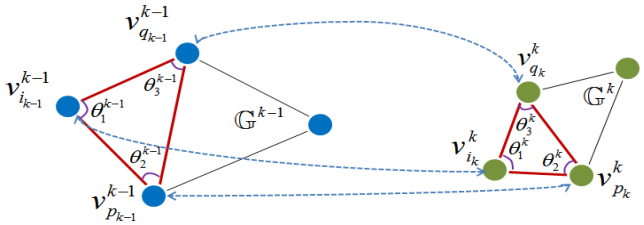

Figure 2. Hyper-edges and their triangle features. $\mathbb{G}^{k-1}$ has a hyper-edge that connects nodes $\{v^{k-1}_{i_{k-1}}, v^{k-1}_{p_{k-1}}, v^{k-1}_{q_{k-1}}\}$ and has the triangle features as θ^{k-1}_1, θ^{k-1}_2 and θ^{k-1}_3. While the hyper-edge on nodes $\{v^k_{i_k}, v^k_{p_k}, v^k_{q_k}\}$ in graph $\mathbb{G}^k$ has the triangle features as θ^k_1, θ^k_2 and θ^k_3. Note that the triangle structure of the hyper-edge is invariant to rotation and scaling.

where σ^2 is a regularized factor and set to 2 throughout the experiments. This hyper-edge affinity is widely used in hyper-graph matching, such as in [13, 20].

5. Experiments

Extensive experiments are conducted on a diverse set of public benchmarks including the CMU-House/Hotel dataset [1], the WILLOW-ObjectClass dataset [8] and the Repetitive Structure dataset [26]. Moreover, we demonstrate the effectiveness of the proposed approach by comparing with the state-of-the-arts [34, 26, 33].

5.1. Datasets and Settings

CMU House/Hotel Dataset. The house and hotel sequences contain 111 frames and 101 frames respectively, and each frame has thirty landmarks. Following the setting in [33], we extract 10 random landmarks from all as the inliers, and 3 random landmarks from the rest as the outliers.

WILLOW-ObjectClass Dataset. The object class dataset consists of five real world image sequences. Four sequences are used in the experiments including Duck (50 images), Car (40 images), Motorbike (40 images) and Winebottle (66 images). There are 10 manually annotated landmarks in each image, and the annotations are not much rigorous. With the same setting as [32], we use the 10 landmarks as the inliers and supplement 2 outliers detected from the background by the SIFT detector.

Repetitive Structure Dataset. This dataset consists of two sequences describing repetitive structures, which make image matching a difficult problem due to the ambiguous features. The Building sequence (16 images) is selected as the test sequence. For each image, we retain 10 landmarks as the inliers and randomly sample three landmarks from the rest as the outliers.

Graph sets with various sizes are utilized to validate the performances of the multi-graph matching algorithms. Generally, the experiments are conducted on 4-graph, 6-graph, 8-graph, 10-graph and 12-graph matching tasks. For the robust evaluation, 10 random tests are performed for each matching task, and the result is the average of all 10 tests.

The parameter λ in the optimization (9) is application-dependent, since the impacts of two basic components (i.e., the node affinity and hyper-edge affinity) vary in different scenarios. The stabler of the graph structure, the more confident is the component of hyper-edge affinity. In this way, λ is set to 8 for the CMU-House/Hotel and Building sequences, 4 for the Motorbike and Winebottle sequences, and 2 for the Duck and Car sequences. The number of the maximum iteration (i.e., max_iter) in Algorithm 1 is set as 100 throughout all the experiments.

5.2. Algorithms and Evaluations

The state-of-the-arts are compared in the experiments. They are the permutation synchronization approach (Match-Sync) [26], the alternative optimization method (Match-Opt) [34], the graduated consistency-regularized optimization algorithm (Match-Grad) [33]. The results of three algorithms on the CMU dataset and WILLOW-ObjectClass dataset are referred from the work [33, 32].

The proposed optimization objective consists of two components, the unary vertex affinity score and the hyper-edge affinity score, and each objective can be used in the optimization alone to solve the matching problem. The proposed algorithm framework is flexible to accommodate different kinds of optimizations. When the unary node affinity is exploited only, the optimization degenerates into the multi-dimensional assignment (MDA) problem, and the solution is termed as "Tensor-MDA". With the hyper-edge affinity used only, the problem degenerates into the hyper-graph matching, and the proposed algorithm is termed as "Tensor-HGM". While the algorithm for the optimization on both scores is termed as "Tensor-MGM". All three variants of the proposed algorithm are tested in the experiments.

There are two main measures involving the multi-graph matching: 1) accuracy: the number of correctly matched inliers divided by the total number of inliers, as popularly used in the related work [9, 39, 35]; 2) consistency: the number of consistent matches divided by the number of all possible matches, detailed definition can be referred to the work [33]. In this work, only the accuracy metric is applied, since our algorithm naturally guarantees 100% of consistency, which is another merit of our algorithm.

5.3. Results and Analyses

CMU-House/Hotel dataset. Quantitative results on CMU-House/Hotel sequences are presented in Table 1. It can be seen that the proposed approach performs the best on almost all tests, and there are the remarkable performance gaps between our algorithm and the state-of-the-arts. Surprisingly, both the two algorithm variants, Tensor-MDA and Tensor-HGM, have moderate results on this dataset.

Table 1. Matching accuracy on the CMU-House/Hotel dataset (%).

	CMU-House						CMU-Hotel					
	Match-Sync [26]	Match-Opt [34]	Match-Grad [33]	Tensor-MDA	Tensor-HGM	Tensor-MGM	Match-Sync [26]	Match-Opt [34]	Match-Grad [33]	Tensor-MDA	Tensor-HGM	Tensor-MGM
4-graph	85.2	**99.0**	84.0	90.5	89.4	**96.6**	87.7	**93.2**	90.0	88.7	86.7	**96.7**
6-graph	83.2	81.6	85.2	**90.0**	87.7	**96.2**	81.5	72.3	**87.9**	87.3	85.3	**97.7**
8-graph	76.4	82.4	87.2	81.7	**89.6**	**95.5**	60.5	62.8	**84.9**	78.5	80.7	**97.1**
10-graph	68.8	80.2	79.2	78.1	**83.6**	**94.3**	63.8	69.1	**84.5**	78.7	83.7	**93.7**
12-graph	75.4	80.0	82.6	75.2	**83.5**	**95.1**	70.0	68.5	**88.1**	73.6	86.1	**97.9**

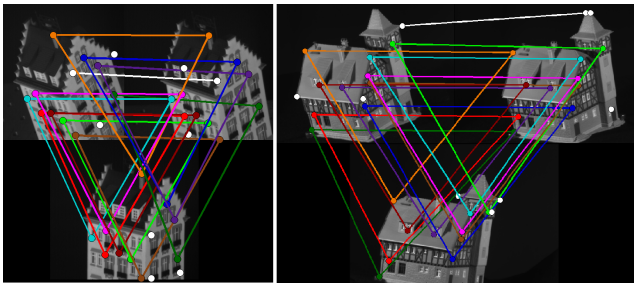

Figure 3. Matching results of the proposed approach across three graphs in the CMU-House/Hotel dataset. Left: hotel sequence; Right: house sequence. The nodes and matches are color-coded, and correct matches appear in the same color as the nodes they connect. White circles denote outliers. Best viewed in color.

The qualitative results on the hotel and house sequences are shown in Figure 3, our method has excellent performance with few mismatches.

WILLOW-ObjectClass dataset. Quantitative results on this dataset are presented in Table 2. Overall, the results on the WILLOW-Object dataset are much worse than that in the CMU-House/Hotel dataset, especially for the Duck and Car sequences. The large pose and viewpoint variations, flexible landmark annotations, noisy outliers make the matching on ObjectClass sequences extremely difficult. The proposed algorithm obtains the best results on the small and middle size multi-graph matching, such as the 4-graph, 6-graph and 8-graph matching tasks. It is observed that our approach gets lower performances as the numbers of input graphs increase, one reason is that the affinity (15) assumes node features lies in a low dimensional space, while the large variation in this noisy dataset shakes the assumption. It must be noted that the performance of the proposed algorithm can be improved by using more powerful high-order node affinity representation. The qualitative results on the WILLOW dataset are shown in Figure 4.

Repetitive Structure dataset. Quantitative results on this dataset are presented in Table 3, where only the approach [26] is compared. The building sequence has many repetitive patterns and viewpoint changes, but the annotations are stable. In this case, our approach achieves the favorable performance, and the qualitative matching is presented in Figure 5.

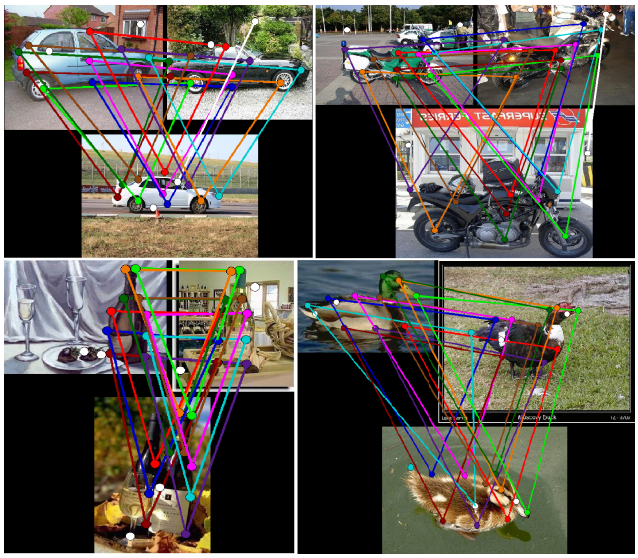

Figure 4. Matching results of the proposed approach across three graphs in the WILLOW-ObjectClass dataset. Top-left: Car; Top-right: Motorbike; Bottom-left: Winebottle; Bottom-right: Duck. The nodes and matches are color-coded, and correct matches appear in the same color as the nodes they connect. White circles denote outliers. Best viewed in color.

Table 3. Matching accuracy on the Building sequence (%)

	Match-Sync [26]	Tensor-MDA	Tensor-HGM	Tensor-MGM
4-graph	76.5	82.8	**87.3**	**92.8**
6-graph	**82.8**	75.3	58.7	**93.0**
8-graph	**77.9**	75.0	53.2	**88.3**
10-graph	**87.3**	73.5	66.3	**90.7**

5.4. Discussion

Consistency. The matching results of the proposed approach meet the full consistency, which is derived from the high-order matching naturally. This full consistency is also clearly observed from results such as Figure 3 $\sim$ Figure 5.

Convergence. To validate the convergence of the proposed iteration solution, we present the score variation curves in the iteration process. Two examples, one is the 4-graph matching on the CMU-House and the other is the 4-graph matching on the WILLOW-Motorbike, are shown in Figure 6. The total affinity scores along with the individual scores (i.e., the node/hyper-edge affinities) are drawn in

Table 2. Matching accuracy on the WILLOW dataset (%).

	WILLOW-Car						WILLOW-Motorbike					
	Match-Sync [26]	Match-Opt [34]	Match-Grad [33]	Tensor-MDA	Tensor-HGM	Tensor-MGM	Match-Sync [26]	Match-Opt [34]	Match-Grad [33]	Tensor-MDA	Tensor-HGM	Tensor-MGM
4-graph	54.2	57.1	63.3	62.2	**68.7**	**79.5**	75.9	**78.7**	78.4	78.3	69.7	**87.5**
8-graph	61.5	69.6	**74.3**	62.6	49.6	**75.7**	84.6	82.5	**86.3**	76.5	65.7	**85.6**
12-graph	55.8	66.0	**80.5**	55.4	40.3	**67.1**	81.3	**84.3**	**87.1**	70.5	58.7	80.3
	WILLOW-Winebottle						**WILLOW-Duck**					
	Match-Sync [26]	Match-Opt [34]	Match-Grad [33]	Tensor-MDA	Tensor-HGM	Tensor-MGM	Match-Sync [26]	Match-Opt [34]	Match-Grad [33]	Tensor-MDA	Tensor-HGM	Tensor-MGM
4-graph	49.8	71.2	64.2	81.2	**82.7**	**97.0**	35.3	42.3	40.0	**60.2**	58.2	**65.8**
8-graph	37.5	**91.2**	82.9	78.9	76.4	**94.3**	40.8	45.8	50.6	**60.6**	48.2	**61.3**
12-graph	69.0	92.7	**93.1**	71.3	63.2	**93.7**	45.9	56.6	**72.7**	49.4	33.8	**58.3**

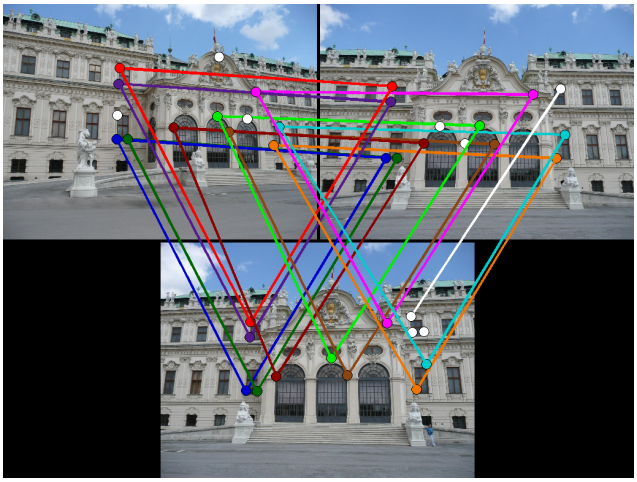

Figure 5. Matching results of the proposed approach across three graphs in the Building sequence. The nodes and matches are color-coded, and correct matches appear in the same color as the nodes they connect. White circles denote outliers. Best viewed in color.

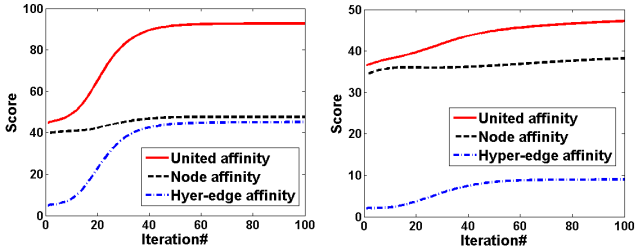

Figure 6. The curve of affinity score across iterations. Left: curves for matching the house images; Right: curves for matching the motorbike images.

the figure. It is clear that the optimization objective gradually climbs to the extreme with the iteration proceeding, and the proposed algorithm has the convergence property.

Optimization. It can be observed from the experiments that both Tensor-MDA and Tensor-HGM have moderate performances. By uniting the two complementary affinity scores, the proposed approach obtains a much better result, which suggests the necessity of incorporating high-order information across both multiple graphs and hyper-edges. Furthermore, the proposed algorithm is accommodative to diverse (hyper-)edge affinity scores, such as the pairwise edge similarity, the third or higher order hyper-edge affinity, and even the hybrid of different order hyper-edge affinities.

Affinity. The proposed approach has its advantage to explore the high-order node affinity which is not available in the state-of-the-arts. Currently, we take advantage of an oversimplified affinity measure which is sensitive to the factors such as large deformations of graphs. In the future, we will probe kinds of powerful high-order node affinities.

Complexity. The amount of group-wise matches grows exponentially with the number of graphs. For $K+1$ graphs, each graph has I nodes and each node N matching candidates, there are IN^K matchings and $K(IN)^3$ hyper-edge triples. In this way, the complexity is O($MIN^K K^2 + MK(IN)^3$) for M iterations. A divide-and-conquer strategy could be used for acceleration. In the experiments, all tests run on a laptop (2.1GHz Intel Core i7 with 8G memory) without code optimization, and the running times for matching 4 graphs, 8 graphs and 12 graphs are about 0.1 second, 3 seconds and 10 minutes respectively.

6. Conclusion

Starting from the pairwise graph matching (PGM) formulation, the work firstly derives the high-order optimization on multi-graph matching (MGM). The extension of the formulation from the PGM to MGM is self-evident, but surprisingly the insight is totally new to our knowledge. Furthermore, we formulate the optimization objective into the tensor products and propose an effective power iteration solution. The proposed approach is ready for exploring various kinds of high-order node affinities, accommodative to various edge/hyper-edge affinities, and has the full matching consistency. These merits make our approach much powerful, especially for small and the middle size graph matching. Finally, the experimental validations on diverse datasets illustrate the effectiveness of the proposed approach.

Acknowledgement. This work is partly supported by the 973 basic research program of China (No. 2014CB349303), the Natural Science Foundation of China (No. 61502492, 61472421, 61370185,61202327), the Guangdong Natural Science Foundation (No. S2013010013432,S2013010015940), the CAS Center for Excellence in Brain Science and Intelligence Technology, and the US NSF Grants No. 1449860, No. 1218156 and No. 1350521.

References

[1] http://vasc.ri.cmu.edu/idb/html/motion/. 6

[2] S. Agarwal, Y. Furukawa, N. Snavely, I. Simon, B. Curless, S. Seitz, and R. Szeliski. Building rome in a day. *Communications of the ACM*, 54, 2011. 1

[3] S. Belongie, J. Malik, and J. Puzicha. Shape matching and object recognition using shape contexts. *IEEE Trans. PAMI*, 24(4):509 – 522, 2002. 5

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

[5] W. Brendel and S. Todorovic. Learning spatiotemporal graphs of human activities. In *ICCV*, 2011. 1

[6] Y. Chen, L. Guibas, and Q. Huang. Nearly-optimal joint object matching via convex relaxation. In *ICML*, 2014. 1

[7] E. Cheng, Y. Pang, Y. Z. an J. Yu, and H. Ling. Curvilinear structure tracking by low rank tensor approximation with model propagation. In *CVPR*, 2014. 2

[8] M. Cho, K. Alahari, and J. Ponce. Learning graphs to match. In *ICCV*, 2013. 6

[9] M. Cho, J. Lee, and K. Lee. Reweighted random walks for graph matching. In *ECCV*, 2010. 6

[10] D. Conte, P. Foggia, C. Sansone, and M. Vento. Thirty years of graph matching in pattern recognition. In *IJPRAI*, 2004. 2

[11] T. Cour, P. Srinivasan, and J. Shi. Balanced graph matching. In *NIPS*, 2007. 2, 5

[12] L. De Lathauwer, B. De Moor, and J. Vandewalle. On the best rank-1 and rank-(r 1, r 2,..., rn) approximation of higher-order tensors. *SIAM J. Matrix Anal. Appl.*, 21(4):1324 – 1342, 2000. 4

[13] O. Duchenne, F. Bach, I. Kweon, and J. Ponce. A tensor-based algorithm for high-order graph matching. *IEEE Trans. PAMI*, 33(21):2383 – 2395, 2011. 2, 4, 5, 6

[14] O. Duchenne, A. Joulin, and J. Ponce. A graph-matching kernel for object categorization. In *ICCV*, 2011. 1

[15] M. Goesele, N. Snavely, B. Curless, H. Hoppe, and S. Seitz. Multi-view stereo for community photo collections. In *ICCV*, 2007. 1

[16] S. Gold and J. Rangarajan. A graduated assignment algorithm for graph matching. *IEEE Trans. PAMI*, 18(4):377–388, 1996. 2

[17] Q. Huang and L. Guibas. Consistent shape maps via semidefinite programming. 32(5):177–186, 2013. 1

[18] Q. Huang, G. Zhang, L. Gao, S. Hu, A. Butscher, and L. Guibas. An optimization approach for extracting and encoding consistent maps in a shape collection. *ACM Transactions on Graphics (TOG)*, 31(6):167, 2012. 1

[19] H. W. Kuhn. The hungarian method for the assignment problem. *Naval Research Logistics Quarterly*, 2(1-2):83–97, 1955. 5

[20] J. Lee, M. Cho, and K. Lee. Hyper-graph matching via reweighted random walks. In *CVPR*, 2011. 2, 6

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

[22] M. Leordeanu and M. Hebert. An integer projected fixed point method for graph matching and map inference. In *NIPS*, 2009. 2

[23] M. Leordeanu, M. Hebert, and R. Sukthankar. Beyond local appearance: Category recognition from pairwise interactions of simple features. In *CVPR*, 2007. 1

[24] H. Li, J. Huang, S. Zhang, and X. Huang. Optimal object matching via convexification and composition. In *ICCV*, 2011. 1

[25] Z. Liu, H. Qiao, X. Yang, and S. Hoi. Graph matching by simplified convex-concave relaxation procedure. *IJCV*, 109(3):169–186, 2014. 2

[26] D. Pachauri, R. Kondor, and S. Vikas. Solving the multi-way matching problem by permutation synchronization. In *NIPS*, 2013. 1, 2, 6, 7, 8

[27] X. Shi, H. Ling, W. Hu, C. Yuan, and J. Xing. Multi-target tracking with motion context in tensor power iteration. In *CVPR*, 2014. 2

[28] X. Shi, H. Ling, J. Xing, and W. Hu. Multi-target tracking by rank-1 tensor approximation. In *CVPR*, 2013. 1, 2, 4, 5

[29] R. Sinkhorn. A relationship between arbitrary positive matrices and doubly stochastic matrices. *Ann. Math. Statistics*, 35:876–879, 1964. 5

[30] A. Sole-Ribalta and F. Serratosa. Models and algorithms for computing the common labelling of a set of attributed graphs. *CVIU*, 115(7):929 – 945, 2011. 1, 2

[31] A. Sole-Ribalta and F. Serratosa. Graduated assignment algorithm for multiple graph matching based on a common labeling. In *IJPRAI*, 2013. 2

[32] J. Yan, M. Chu, H. Zha, and X. Yang. Multi-graph matching via affinity optimization with graduated consistency regularization. *10.1109/TPAMI.2015.2477832*. 6

[33] J. Yan, Y. Li, W. Liu, H. Zha, X. Yang, and M. Chu. Graduated consistency-regularized optimization for multi-graph matching. In *ECCV*, 2014. 1, 2, 6, 7, 8

[34] J. Yan, Y. Tian, H. Zha, X. Yang, and Y. Zhang. Joint optimization for consistent multiple graph matching. In *ICCV*, 2013. 2, 6, 7, 8

[35] J. Yan, J. Wang, H. Zha, X. Yang, and S. Chu. Consistency-driven alternating optimization for multigraph matching: A unified approach. *IEEE TIP*, 24(3):994 – 1009, 2015. 1, 6

[36] B. Yao and L. Fei-Fei. Action recognition with exemplar based 2.5d graph matching. In *ECCV*, 2012. 1

[37] M. Zaslavskiy, F. Bach, and J.-P. Vert. A path following algorithm for the graph matching problem. *IEEE Trans. PAMI*, 31(12):2227 – 2242, 2009. 2

[38] R. Zass and A. Shashua. Probabilistic graph and hypergraph matching. In *CVPR*, 2008. 2

[39] F. Zhou and D. la Torre. Factorized graph matching. In *CVPR*, 2012. 2, 6