Discrete Hyper-graph Matching

Junchi Yan[†‡], Chao Zhang[‡], Hongyuan Zha[§], Wei Liu[‡], Xiaokang Yang[†], Stephen M. Chu[‡]
[†] Shanghai Jiao Tong University
[‡] IBM Research
[§] Georgia Institute of Technology
{yanjunchi,xkyang}@sjtu.edu.cn bjzchao@cn.ibm.com zha@cc.gatech.edu {weiliu,schu}@us.ibm.com

Abstract

This paper focuses on the problem of hyper-graph matching, by accounting for both unary and higher-order affinity terms. Our method is in line with the linear approximate framework while the problem is iteratively solved in discrete space. It is empirically found more efficient than many extant continuous methods. Moreover, it avoids unknown accuracy loss by heuristic rounding step from the continuous approaches. Under weak assumptions, we prove the iterative discrete gradient assignment in general will trap into a degenerating case – an m-circle solution path where m is the order of the problem. A tailored adaptive relaxation mechanism is devised to detect the degenerating case and makes the algorithm converge to a fixed point in discrete space. Evaluations on both synthetic and real-world data corroborate the efficiency of our method.

1. Introduction

Graph matching has become a popular tool in a variety of areas in computer science, ranging from computer vision to machine learning to computational geometry. Graph matching aims at finding correspondence between two feature sets, with a wide spectrum of applications that require feature matching, as diverse as image recognition [36], shape matching [1] and object tracking [28], among others. Specifically, classical first-order methods [9, 12] only consider the node-wise unary compatibility between two point sets, which can be reduced to a linear assignment problem and solved in polynomial time by standard techniques such as the Hungarian algorithm [12]. One step further, in second-order graph matching, each feature set is formulated as a graph with nodes representing features and edge weights measuring the similarity between nodes, where the correspondence is established through preserving the structure similarity across two graphs. When involving both node and edge compatibilities, the objective of second-order graph matching is essentially linked to the quadratic assignment problem that is known to be NP hard [3, 10, 15].

More recently, the higher-order relations (mostly third-order) between graphs are explored to derive more sophisticated graph matching techniques, namely *hyper-graph matching*, which is the focus of this paper. Compared with conventional second-order approaches [7, 13, 14, 10, 38, 39], hyper-graph matching methods enjoy a better scale/rotation/deformation invariance thus tend to be more robust to outliers and noise. Due to the theoretical advance and empirical success, hyper-graph matching has attracted increasing attention and many methods have been developed *e.g.* [37, 8, 13, 4] and the references therein.

In this paper, we are particularly interested in devising a *pure discrete* method with theoretically guaranteed *convergence* property for hyper-graph matching. This is motivated by the observation that current hyper-graph/graph matching methods mostly relax the problem domain to continuous space, and adopt iterative deterministic annealing, such as soft-assignment used in [10] or iterative bistochastic normalization [13] to confine the variable in the convex hull of the discrete space. However this strategy might not be perfect: i) the graduated annealing procedure or iterative normalization is time-consuming and, in general no convergence can be theoretically guaranteed in *discrete domain* that leaves the optimality unclear; ii) at the end of iterations, an ad-hoc post-step is needed to round the continuous solution, which is implicitly based on the assumption that the continuous optimum is close to the discrete optimum of the original combinatorial problem. However so far no one can guarantee this assumption always hold thus it can incur unclear and perhaps arbitrary accuracy loss.

This paper adopts an iterative procedure that approximates the higher-order assignment problem into a first-order linear assignment one. This strategy is in line with the state-of-the-art for both second-order graph matching [14, 10, 7] and higher-order hyper-graph matching [8, 13]. In each iteration, we employ the gradient assignment method (*e.g.* Hungarian method) for discrete optimization as used in

[15, 26] to find the optimal assignment matrix in the integer domain. The relative performance-wise advantage of this "discrete" solver is its empirically observed efficiency (as shown in the experiment part of this paper) against the typical "soft-assignment" counterpart [10]. It also dispenses with the heuristic post-rounding step. Moreover, to our best knowledge, it is so far one of the *very few discrete methods*[1] for solving the hyper-graph matching problem.

Based on this general methodology, the key problem is how to devise an effective optimization mechanism in discrete domain. We propose a particular *adaptive discrete gradient assignment* method, which bears several desirable properties: i) no heuristic re-weighting step as used in [7, 13, 15] is imposed, being less dependent on parameter tuning; ii) inherently return integer solutions that avoids heuristic rounding; iii) converge to a fixed discrete point by theoretical guarantee and empirically show competitive overall efficiency compared with continuous methods.

Perhaps most importantly, at methodology level, this paper might raise the community's attention to pure discrete methods for hyper-graph matching, as currently this problem is often addressed in continuous space [37, 8, 13]. In the long run, we hope our work will trigger follow-up efforts for new discrete methods, and/or continues methods based on the proposed mechanism for hyper-graph matching.

At practice level, our method so far can be of help in places where efficiency is first priority as it can achieve a tradeoff between accuracy and efficiency, as shown later in the paper. Extensive empirical comparison with representative continuous hyper-graph matching methods suggests that the proposed algorithm produces competitive results.

2. Related Work

Graph matching has been previously mainly addressed by modeling its second-order affinity information[2] [7, 10, 14, 15, 25]. The more closely related work is for hyper-graph matching that incorporates higher-order affinity for matching, which typically extends the original algorithm for second-order problem to the higher-order one. Specifically, Duchenne *et. al.* extend the spectral matching method used for second-order graphs [14] to hyper-graphs via a multi-dimensional power method in continuous domain [8]. Re-weighted Random Walks Graph Matching method is developed for both the second-order [7] and higher-order [13] graph matching problems from a random walk perspective. The work [37] addresses the hyper-graph matching problem in a probabilistic setting solved by convex optimization. However, the probabilistic hyper-graph matching approach relies on the assumption of conditional independence that helps simplify the model by factorizing the higher-order interactions into first-order ones. Regardless of the formulations and objectives, except for few works [17], most of the aforementioned hyper-graph matching methods relax the original higher-order matching problem to a first-order assignment problem in continuous domain, followed by an iterative optimization procedure. The convergence of these methods are often *empirically* observed in [7, 10] and theoretically justified by [15, 20] under certain moderate conditions: for instance, when the affinity matrix is semi-positive definite for the second-order case. However in general cases when the affinity matrix is not semi-positive definite or the affinity tensor is arbitrary, the convergence is still unclear and cannot be theoretically guaranteed. Moreover, even if such an iterative strategy reaches convergence to a fixed point, it is often a continuous one, which calls for heuristic post-rounding whose influence on the accuracy is unclear. For instance, in the re-weighted hyper-graph random walk formulation [13], the final converged distribution has to be converted to a binary matrix by a certain means.

As the previous works commonly follow an iterative optimization procedure, it turns out the main difference is how the solution path is updated over iterations. As pointed out by the authors in [7, 13], their solvers RRWM [7] (RRWHM [13]) can be interpreted as a re-weighting of the power method [14, 8] and softassign that adopts Sinkhorn bistochastic normalization [23]. Accordingly, RRWHM is a natural extension from RRWM for hyper-graph matching. This extension is also applicable to IPFP [15] and GAGM [10], which are termed HIPFP and HGAGM [4] in this paper, and will be evaluated together in our experiments. In [24] candidate particles in permutation space are computed by reusing extant graph matching solvers, to sequentially reweight its sampling distribution. While no original discrete method is devised for generating particles.

There are other threads about graph matching in parallel with our work. Recent works leverage various learning algorithm to derive optimal graph matching, while refining the affinity matrix simultaneously or learning optimal affinity setting via labeled node correspondences [3, 16, 17, 6]. Such an affinity refinement process falls into either supervised [3] or unsupervised [16] or semi-supervised [17] learning paradigms. While [22] shows that point-wise information is often enough to provide good matchings and reduces the higher-order affinity matrix into a unary one. Recently, the problem of matching multiple graphs as a whole also attract research efforts such as [32, 30, 34, 33, 29]. Our work is learning-free and focuses on two-graph matching.

[1] Except for the hyper version of IPFP [15] we implemented in this paper termed as HIPFP, we are unaware of any other existing discrete method.

[2] Throughout the paper, several acronyms are used to term the related methods. RRWHM [13]: Re-weighted Random walk Hypergrph Matching; TSM [8]: Tensor Graph Matching; IPFP [15]: Integer Projected Fixed Point; GAGM [10]: Graduated Assignment Graph Matching; We further term HIPFP and HGAGM [4] as the hyper-graph matching extension for IPFP and GAGM respectively. In addition, the proposed two algorithms are termed as i) HDGA: *Hyper Discrete Gradient Assignment*; ii) HADGA: *Hyper Adaptive Discrete Gradient Assignment*.

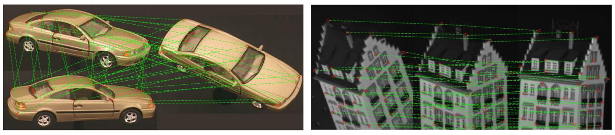


Figure 1: Image examples of the used dataset. Left: car from Pose dataset; right: hotel from CMU sequence.

3. Algorithms and Convergence Analysis

Under the general first-order approximate framework for hyper-graph matching as adopted by [8, 13], we show how to explore the capability of a discrete method, especially improving its cost-effectiveness compared with continuous methods. In the second-order cases, the extant popular continuous solvers like RRWM [7] and Factorized Graph Matching [39] usually achieve higher accuracy than the discrete solver IPFP [15] as reported in [39]. Our key technical point is an adaptive and dynamic objective relaxation mechanism for optimization in discrete domain directly, whose convergence is ensured by theoretical analysis.

3.1. Preliminaries and Notations

Given two hyper-graphs $\mathcal{G}_1$ and $\mathcal{G}_2$ with n_1 and n_2 nodes respectively, there exist affinity measurements between each subset of m nodes (a hyper-edge) in one graph and m nodes (also a hyper-edge) in the other, which together form a set of pairs of hyper-edges between two graphs. For a fixed assignment matrix used to account for node correspondence, by adding up all corresponding hyper-edge pairs determined by the assignment matrix, one can obtain an overall affinity objective score. This paper assumes the hyper-edge affinity is given, thus this score becomes a function *w.r.t.* the assignment matrix. The goal of hyper-graph matching is to establish node mapping of two hyper-graphs such that the objective is maximized.

We use the tuple $(i_1, i_2, \ldots, i_m)$ to denote the m-order hyper-edge correspondence where each element in the tuple $\{i_k\}_{k=1}^m$ indicates a node-to-node correspondence $(i_k^1 \leftrightarrow i_k^2)$, *i.e.* node i_k^1 (i_k^1=$1, \ldots, n_1$) in $\mathcal{G}_1$ matches node i_k^2 (i_k^2=$1, \ldots, n_2$) in $\mathcal{G}_2$. We allow $(i_k^1 \leftrightarrow i_k^2)$=$(i_s^1 \leftrightarrow i_s^2)$ for $k \neq s$, to encode lower-order ($<m$) hyper-edges. Formally, we use an mth-order super-symmetric tensor $\mathbf{H}$ to encode the hyper-edge to hyper-edge affinity score. For instance, $H_{i_1,i_2,\ldots,i_m}$ denotes the relation between two hyper-edges that is induced by the correspondence tuple $(i_1, i_2, \ldots, i_m)$.

More rigourously, let $\mathbf{p} \in \{0,1\}^{n_1 n_2}$ denote the vectorized form of the node correspondence matrix $\mathbf{P} \in \{0,1\}^{n_1 \times n_2}$ (or termed as assignment matrix) by its rows for one-to-one matching. Its nonzero element p_{i_k} whose position in $\mathbf{p}$ is indexed by i_k for (i_k=$1, 2, \ldots, n_1 n_2$), denotes the node-to-node correspondence $(i_k^1 \leftrightarrow i_k^2)$ and the index is calculated by $i_k = (i_k^1 - 1)n_2 + i_k^2$. Based on the above notations, by assuming $n_1 \leq n_2$ without loss of generality, the hyper-graph matching problem can be formulated as the following constrained combinatorial optimization problem which is widely adopted by extant works *e.g.* [8, 13]:

$$\mathbf{p}^* = \arg\max(\Sigma_{i_1,i_2,\ldots,i_m} H_{i_1,i_2,\ldots,i_m} p_{i_1} p_{i_2} \ldots p_{i_m}) \quad (1)$$

$$s.t. \quad \mathbf{P1}_{n_2} = \mathbf{1}_{n_1}, \mathbf{P}^T \mathbf{1}_{n_1} \leq \mathbf{1}_{n_2}, \mathbf{p} \in \{0,1\}^{n_1 n_2 \times 1}$$

Or using the tensor product [21] in a more compact form:

$$\mathbf{p}^* = \arg\max(\mathbf{H} \otimes_1 \mathbf{p} \otimes_2 \mathbf{p} \ldots \otimes_m \mathbf{p}) \quad (2)$$

$$s.t. \quad \mathbf{P1}_{n_2} = \mathbf{1}_{n_1}, \mathbf{P}^T \mathbf{1}_{n_1} \leq \mathbf{1}_{n_2}, \mathbf{p} \in \{0,1\}^{n_1 n_2 \times 1}$$

Note one can further add dummy nodes in $\mathcal{G}_1$ to make the number of nodes in two graphs become equal. Thus for discussion convenience, in the rest of this paper we further assume n_1=n_2 if not otherwise explicitly specified. In line with the convention used in [8], the tensor product $\otimes_k$ satisfies the property $\mathbf{B} = \mathbf{A} \otimes_k \mathbf{V}$: $B_{i_1,\ldots,i_{k-1},i_{k+1},\ldots,i_n}$=$\sum_{i_k} A_{i_1,\ldots,i_k,\ldots,i_n} V_{i_k}$ where $\mathbf{V}$ is a vector and $\mathbf{A}$, $\mathbf{B}$ is an n-th and $n-1$th order tensor, respectively. The index k in $\otimes_k$ indicates that we multiply on the k-th dimension. Readers are referred to Sec.3.1 in [8] for more expositions on tensor multiplication.

The constraints refer to two-way constraints for the one-to-one node matching. This formulation is also widely adopted by other work such as [8, 13], where the super-symmetric tensor $\mathbf{H}$ is assumed invariant under any permutation of the order in the tuple. One aims to find the solution that maximizes the matching score in the permutation matrix space. We describe our main results under m-order formulation, while the experiments focus on the third-order in line with previous work [37, 8, 5, 13, 17] since the third-order information has been shown able to handle the rotation and scale variance robustly for most applications. It is also a good compromise between efficiency and efficacy.

3.2. Proposed Convergent Algorithms

We first propose our baseline method *Hyper Discrete Gradient Assignment* (HDGA) which uses a series of iterative linear assignments to approximate the original objective by fixing m-1 subsequent variables $\{\mathbf{p}_j\}_{j=k+2-m}^k$ in the objective (2). Then it adopts the Hungarian method (denoted as $H_d(\cdot)$) to obtain the *global-optimal discrete* solution regarding the linearized objective per iteration: $\mathbf{p}_{k+1}$=$H_d(\mathbf{H} \otimes_1 \mathbf{p}_{k+2-m} \ldots \otimes_{m-1} \mathbf{p}_k)$ as stated in Alg.1. This is because the input to the Hungarian method $\mathbf{K} \triangleq \mathbf{H} \otimes_1 \mathbf{p}_{k+2-m} \ldots \otimes_{m-1} \mathbf{p}_k$ in fact is a vector after m-1 times of tensor product operations have been performed on the m-th order tensor $\mathbf{H}$. Thus $\mathbf{K} \otimes_m \mathbf{p}_{k+1}$ can be rewritten as $\mathbf{K}\mathbf{p}_{k+1}$ and reduced to a linear assignment problem. In this sense, our method follows the first-order approximate framework.

Note that HDGA forms a score-ascending solution sequence *w.r.t.* the relaxed objective function. Note the sequence generated by IPFP [15] for second-order matching is also score-ascending, but *w.r.t.* the original objective.

Algorithm 1 Hyper Discrete Gradient Assignment

```
Input: $\mathbf{p}_0 = [\frac{1}{n^2}, \cdots, \frac{1}{n^2}]^T$, $\mathbf{H}$, $K$
Output: vectorized assignment matrix solution $\mathbf{p}^*$
for k=1:K do
    $\mathbf{p}_{k+1} = H_d(\mathbf{H} \otimes_1 \mathbf{p}_{k+2-m} \otimes_2 \ldots \otimes_{m-1} \mathbf{p}_k)$
    if converge then
        return $\mathbf{p}^* = \mathbf{p}_{k+1}$;
    end if
end for
```

Algorithm 2 Hyper Adaptive Discrete Gradient Assignment

```
Input: $\mathbf{p}_0 = [\frac{1}{n^2}, \cdots, \frac{1}{n^2}]^T$, $\mathbf{p}^* = \mathbf{p}_0$, $\mathbf{H}$, $\alpha$, $K$
Output: vectorized assignment matrix solution $\mathbf{p}^*$
for k=1:K do
    $\mathbf{p}_{k+1} = H_d(\mathbf{H} \otimes_1 \mathbf{p}_{k+2-m} \otimes_2 \cdots \otimes_{m-1} \mathbf{p}_k)$
    if converge then
        return: $\mathbf{p}^* = \mathbf{p}_{k+1}$
    else if Fall into the $m$-cycling loop sequence then
        $\mathbf{H} = \mathbf{H} + \alpha_2\mathbf{H}^{(2)} + \alpha_3\mathbf{H}^{(3)} + \ldots + \alpha_m\mathbf{H}^{(m)}$
    end if
end for
```

One key issue of such an approximate strategy is the solution path over iterations might be degenerated and deviate from the original objective (2) when the solutions are divergent. Thus it is appealing to design a convergent algorithm which is able to find the optimal solution in discrete domain for the approximating objective. The following Theorem (1) shows under the moderate condition *i.e.* '*equal-score* equals *unique-solution*' as stated in Assumption (1), HDGA will converge to an m-point cycling solution path. Note that like many previous methods, the global-optimality for Alg.1 is not guaranteed as the original problem is in fact NP-hard.

Assumption 1. *Given* $\boldsymbol{p}_{i_1}, \boldsymbol{p}_{i_2}, \ldots, \boldsymbol{p}_{i_{m-1}}$*;* $\forall j, k, j \neq k$*, assume* $\boldsymbol{H} \otimes_1 (\boldsymbol{p}_j - \boldsymbol{p}_k) \otimes_2 \boldsymbol{p}_{i_1} \cdots \otimes_m \boldsymbol{p}_{i_{m-1}} \neq 0$*;* $\boldsymbol{p}_i, \boldsymbol{p}_j, \boldsymbol{p}_k$ *are vectorized permutation matrix in Alg.1's solution chain.*

Theorem 1. *Alg.1 will converge to an* m*-point circle:* $(\boldsymbol{p}_{k-m}, \boldsymbol{p}_{k+1-m}, \cdots, \boldsymbol{p}_{k-1}) \rightarrow (\boldsymbol{p}_{k+1-m}, \boldsymbol{p}_{k+2-m}, \cdots, \boldsymbol{p}_k) \rightarrow \cdots \rightarrow (\boldsymbol{p}_{k-m}, \boldsymbol{p}_{k+1-m}, \cdots, \boldsymbol{p}_{k-1}) \rightarrow \cdots$ *if Assumption (1) holds.*

Proof Given a super-symmetric affinity tensor $\mathbf{H}$, HDGA is score-ascending *w.r.t.* the *approximate* objective as it employs the gradient assignment approach *e.g.* Hungarian method iteratively. Since the score function is bounded due to the feasible domain is the assignment matrix space, there exists a certain k, such that after k rounds of iterations, the value of the *approximate* objective will stop ascending: $\mathbf{H} \otimes_1 \mathbf{p}_{k-m} \otimes_2 \mathbf{p}_{k+1-m} \cdots \otimes_m \mathbf{p}_{k-1} = \mathbf{H} \otimes_1 \mathbf{p}_{k+1-m} \otimes_2 \mathbf{p}_{k+2-m} \cdots \otimes_m \mathbf{p}_k$. On the other hand, Assumption (1) ensures the equal score value leads to the unique solution such that $\mathbf{p}_k = \mathbf{p}_{k-m}$, thus henceforth we are in the circled sequence $(\mathbf{p}_{k+1-m}, \mathbf{p}_{k+2-m}, \ldots, \mathbf{p}_k)$ of length m. Remember the tensor $\mathbf{H}$ is super-symmetric thus the score value is invariant to permutations of $\{\mathbf{p}_i\}_{i=k-m}^{k-1}$. □

It is desirable to devise an efficient discrete algorithm that bears strong convergence. The key idea is, one can monitor the solution path of Alg.1 and once the degenerating m-circle pattern is detected, an adaptive updating mechanism shall be launched to change the solution path to a fixed point. Theorem (2) shows by modifying the affinity tensor, the solution path will converge to a fixed point. To make the paper self-contained, two definitions for Theorem (2) are introduced. We leave proof details in Appendix.

Definition 1. *For an* m*-tuple* $(j_1, j_2, \ldots, j_m)$*, where* $\{j_k\}_{k=1}^m$ *is a positive integer, its multiplicity is defined as the largest number of duplicate elements in the tuple.*

Definition 2. *A* q*-multiplicity unit tensor is defined as:*

$$\boldsymbol{H}^{(q)} = \begin{cases} \boldsymbol{H}^{(q)}_{i_1,i_2,\ldots,i_m} = 1 & (i_1, i_2, \ldots, i_m)\text{'s multiplicity is } q \\ \boldsymbol{H}^{(q)}_{i_1,i_2,\ldots,i_m} = 0 & \text{otherwise} \end{cases}$$

Theorem 2. *If Assumption (1) holds, one can introduce* $\alpha_2, \alpha_3, \ldots, \alpha_m$ *to modify the affinity tensor by* $\boldsymbol{H} = \boldsymbol{H} + \alpha_2\boldsymbol{H}^{(2)} + \alpha_3\boldsymbol{H}^{(3)} + \ldots + \alpha_m\boldsymbol{H}^{(m)}$ s.t. *HDGA will converge to a fixed point in the permutation matrix space.*

This theorem stimulates the *adaptively regularized* gradient assignment algorithm as described in Alg.2. We use the term *adaptively regularized* because it gradually modifies the objective function that implicitly penalizes the deviation between two successive solutions during iteration.

Compared with Alg.1, its "adaptively regularized" counterpart Alg.2 obtains strong convergence at the cost of more iterations by gradually changing the original tensor. Note that the global optimal solution regarding the new objective function will still keep unchanged under the modified affinity tensor. This is because the added terms $\mathbf{H}^{(k)}$ as defined in Definition (2) have no discrimination to any particular solution in the assignment matrix space thus the added terms in the modified objective is constant to any assignment matrix. This is similar to the strategy for the second-order graph matching case discussed by [11, 8], wherein an identity matrix is added to the original affinity matrix *before iteration starts*, in order to make the affinity matrix positive-definite so as to obtain a convergent solution by their continuous approximating solvers[3]. Note that adding the identity matrix in the second-order graph matching formulation has neither no effect on the global optimum configuration in the permutation matrix space as discussed in [26] which further develops a convergent algorithm in *continuous* domain for the *second-order* case. While this work generalizes to the any-order case by a unified formulation and convergent algorithm in *discrete* domain.

[3] For second-order graph matching, positive definite affinity matrix induces a concave minimization problem leading to an integer point [2, 39].

We summarize the features of HADGA: i) solve the problem in discrete domain by theoretical convergence guarantee; ii) the convergence is fulfilled by adaptively modifying the affinity tensor which avoids the m-circle case, meanwhile induces a relaxed objective that keeps the global optimum unchanged for the original objective; iii) the approach is generalizable to any-order problem and no matter the affinity tensor is full or sparse, as its formulation is independent from the property of the affinity tensor.

3.3. Implementation Details

For practical implementation, there are two main aspects for consideration: i) effective m-circle degenerating case detection; ii) perturbation parameter α setting.

Effective cycling path detection One basic idea is comparing among the subsequent iterative solutions which brings about additional cost. More importantly, there exists a subtle case that breaks Assumption (1) when the global optimal solution of Hungarian method is not unique. Consider an extreme case when all elements in the tensor are equal, then any permutation matrix is a best solution. One important observation is that these two cases both lead to the score *w.r.t.* the relaxed objective function stop climbing. Based on this observation, we propose that instead of separately examining the cycling behavior of the solution path or verifying the uniqueness of the Hungarian method, a unified way is to monitor the score value over iterations. Once the objective score stops climbing and meanwhile does not converge to a fixed point, further relaxation can be launched.

Perturbation parameter α_q setting In general, a larger α will guide the iteration converge faster but hurts its exploration capability; Comparatively, a smaller α will enhance exploration capability, at the cost of more iterations. In line with the setting of the experiments, we conduct a concrete discussion on third-order matching. The idea can be generalized to any-order. As shown in the proof of Theorem 2 in Appendix, a loose estimation of the parameter α_q is $\frac{2n^{2m-2}\alpha_{q-1}}{n^{2(m-q-1)}}$. If the original $\mathbf{H}$ is sparse that only encodes the highest-order affinity $\mathbf{H}^{(m)}$ (as set in the experiment), the estimation can be tighten to $\frac{2C_n^{m-1}\alpha_q}{C_n^{m-q-1}}$. When m=3, for arbitrary $\mathbf{H}$, α_2=$2n^2\alpha_1$, α_3=$2n^4\alpha_2$. And for $\mathbf{H}$ only encoding third-order information, the estimation is tighten to α_2=$(n-1)\alpha_1$, α_3=$n(n-1)\alpha_2$. This bound is still very loose because we assume after one swapping operation (refer to Appendix for more details), the affected terms decrease their scores from the largest one to zero which in fact rarely happens. To better model the score difference after swapping, we suggest use the standard deviation s over the elements in $\mathbf{H}$ to approximate the upper bound α_q. This is another rough estimation α_2=$(n{-}1)s$, α_3=$n(n{-}1)s$ to have a good chance for converging. On the other hand, enforcing convergence too early may reduce its exploration ability thus we suggest the whole perturbation α is amortized into pieces and added separately by multiple times over iterations. Therefore, we use another piecewise parameter α^*=$\frac{\alpha}{IterMax}$ for each time when the cycling is detected. Concretely, given the original tensor $\mathbf{H}$ whose value is normalized within [0,1], α_2^* is estimated by $\frac{(n-1)s}{IterMax}$. Keep in mind that we assume $\mathbf{H}$ can be arbitrary, which renders the theoretical estimation not tightly bounded. In our tests for third-order hyper-graph matching, we set α_2^*=$\frac{1}{IterMax}$ and leave α_3^*=0. This setting is empirically found cost-effective and the results are insensitive to the ranges around it.

4. Experiments and Discussion

We conduct the experiments on a PC with dual cores at 3.02GHz for each. All methods are implemented by C++. By performing experiments on both synthetic point-set data and real images, we compare with several state-of-the-art hyper-graph matching approaches: the Higher Order Power Method [8] (TSM) and RRWHM [13]. For both TSM and RRWHM methods, we use the public codes from [13]. Note that some representative approaches like the probabilistic hyper-graph matching [37] underperforms other competing methods as shown in the previous evaluation [13], so we exclude them in the comparison for space saving. In addition, we implement two higher-order extensions of the GAGM [10] and IPFP [15] methods, which are termed as HGAGM and HIPFP. Note HIPFP (HGAGM) is generalized from IPFP (GAGM) in the way like [13] from [7] by iteratively using first order solver. For HIPFP, the first-order solver is the discrete gradient assignment, while HGAGM uses a soft-assignment strategy. In our implementation for HIPFP, we set the optimal line search step of λ for its iterative updating $\mathbf{x}_k = \eta\mathbf{x}_{ga} + (1-\eta)\mathbf{x}_{k-1}$ (see more details in [15]) using the binary search technique, as it is not straightforward to obtain a closed form for hyper problem as in its second-order case [15]. For HGAGM, we set only two rounds for outer annealing loop for efficiency otherwise its time cost is even higher. The goal is to focus on accuracy evaluation under comparable time cost.

The parameters are set based on the authors' original papers. For each trial, the same affinity tensor is used and the Hungarian algorithm is used as the postprocess if the discretization is required. The results of the synthetic experiments are acquired through averaging the results from 100 random trials. For our method, we set the perturbation parameter α_2^*=$1/IterMax$ (*i.e.* α_2=1) for HADGA uniformly in all experiments. For all methods, similar to [10], the iteration procedure will be terminated when the difference of normalized solutions of two consecutive iteration is smaller than $4/n$ where n is the number of nodes, or the iteration reaches the maximum $IterMax = 50$ rounds.

Synthetic Data The synthetic experiments follow a similar setup as [13]. Briefly, we first construct the model

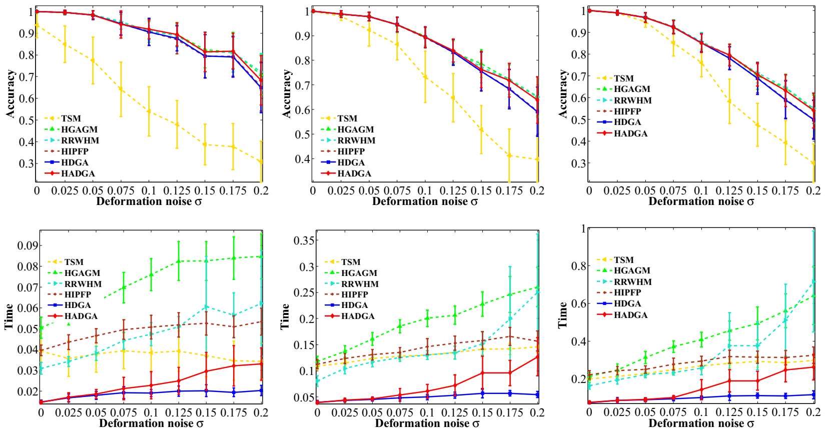


Figure 2: Deformation test on synthetic data by 100 trials. Left to right: # of inlier=20, 30, 40 and # of outlier is fixed to 0.

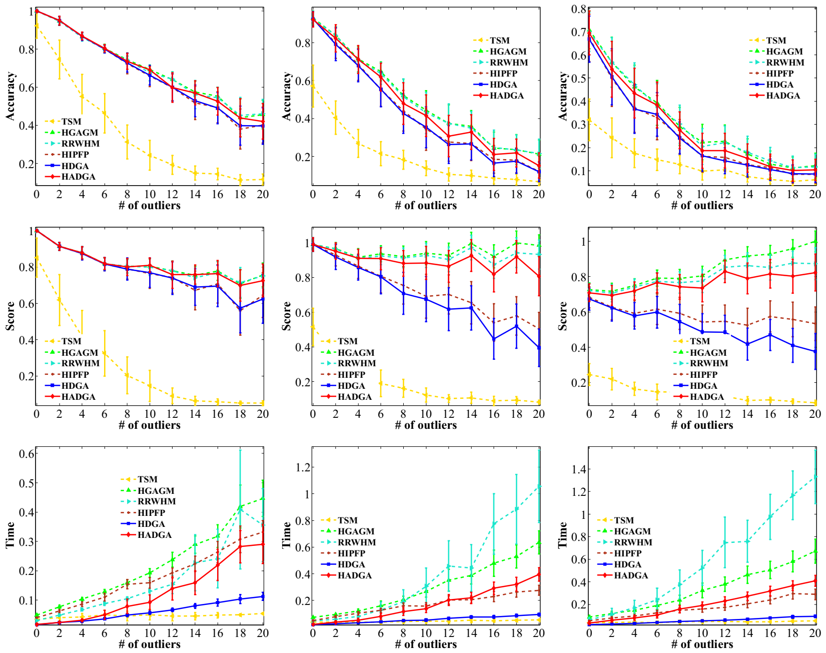
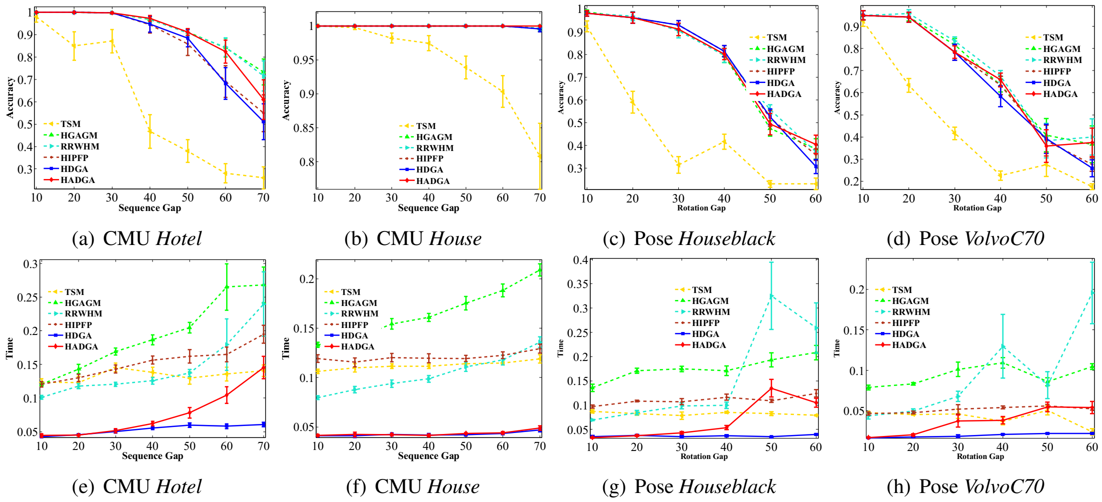


Figure 3: Outlier test on synthetic data by 100 trials. Left to right: deform ε=0, 0.1, 0.2 and # of inlier is fixed to 20.

graph $\mathcal{G}_m$ by randomly generating n_{in} inlier points under the Gaussian distribution $N(0;1)$. The test graph $\mathcal{G}_t$ is generated by perturbing the positions of the model points with Gaussian deformation noise σ in the interval [0,0.2]. n_{out} outliers are added to both sets, with positions from the same Gaussian distribution $N(0;1)$. The outliers are extra points in the test set having no correct correspondence in the model graph. Note that there is no unary descriptor for the synthetic points, and the point-to-point distance in the graph is susceptible to scale, thus making the unary or pairwise matching infeasible. Similar to [13], only third-order affinity is considered. Moreover, we use a triangle sampling strategy to construct the sparse affinity tensor for fair evaluation in line with [13]. The results of deformation and outlier tests are plotted in Fig.2 and Fig.3, which is measured by accuracy (fraction of correct correspondences), and the computational overhead. From these figures, we can observe that HADGA improves the performance of HDGA notably, and performs competitively compared with other methods, while being more efficient.

Image Sequence We further perform landmark point matching on the CMU House and Hotel sequences (see Fig.1 for illustration), which have been widely used as standard testbeds [7, 13, 26, 32]. Following the same setting in the literature, a total of 30 landmark feature points are manually tracked and labeled across all frames (101 frames



(a) CMU *Hotel* (b) CMU *House* (c) Pose *Houseblack* (d) Pose *VolvoC70*

(e) CMU *Hotel* (f) CMU *House* (g) Pose *Houseblack* (h) Pose *VolvoC70*

Figure 4: Accuracy (top row) and time cost (bottom row) on CMU and Pose sequences, by varying the frame gap.

for Hotel and 111 for House). The first 30 frames of each sequence is treated as one graph, and the other graph for matching is selected by forwarding 10, 20, . . ., 70 frames respectively. Moreover, two other image sequences (*houseblack* and *volvoC70* car) from [27] are tested. The results are shown in Fig.4 which suggests HADGA achieves comparable accuracy yet with reduced time cost.

Natural Images Following the experimental setting in [13], we test on 30 image pairs taken from the MSRC v2 dataset and Caltech 256 dataset. The MSER detector is used to extract the key points, and we adopt the similar point selection strategy as in [13] to prune redundant points. The results are shown in Table 1, where our methods have comparable accuracy and reduced time cost.

Further Discussions The above deformation and outlier testing results suggest that our discrete methods are competitive (even faster without sacrificing much accuracy). Although HGAGM and RRWHM achieve the best performance in most cases, they are the most inefficient ones. The empirical speed-up we think partly comes from the removal of the Sinkhorn step and softmax with an annealing procedure typically used in the comparing continuous methods. In particular, HADGA further improves the performance of the baseline HDGA especially when the deformation or outlier increases. RRWHM derives its solution through reweighting between HGAGM and TSM, converging faster than HGAGM in deformation test while slower in outlier test. Fig.5 shows the normalized score of the original objective as a function of the number of iterations along the solution path for three comparing methods. One can observe that HADGA further improves the HDGA score when HDGA is trapped into unsatisfactory solutions early due to the occurrence of the cycling behavior. This empirical observation further validates Theorem (1) which is based on Assumption (1), and makes this assumption more comprehensible than they appear in the theoretical statement.

Table 1: Evaluation on 30 pairs of real image data.

Method	RRWHM	HGAGM	TSM	**HDGA**	**HADGA**
Accuracy	55.38%	56.60 %	47.78%	**55.44%**	**54.86%**
Score	66.43	70.98	67.03	**63.06**	**70.75**
Time(S)	0.19	0.14	0.10	**0.02**	**0.05**

5. Conclusion and Future Work

This paper proposes an adaptive linearized mechanism in discrete domain that gradually modifies the affinity tensor to avoid the so-called "m-circle" degenerating case, which is also uncovered by this work. The future work is devoted to extend the discrete method to its continuous counterpart, in the hope of devising more effective hyper-graph matching methods especially for accuracy. We also would like to expand graph matching applications in scenarios such as facial analysis [19, 18] and saliency detection [35, 31].

Appendix: Proof of Theorem 2

Now we present Lemma 1 and Lemma 2 and prove their establishment. These two lemmas directly establish Theorem 2, which is the main result of this paper.

Let $(\mathbf{p}_1, \mathbf{p}_2, ..., \mathbf{p}_m)$ denote the m-circle sequence that

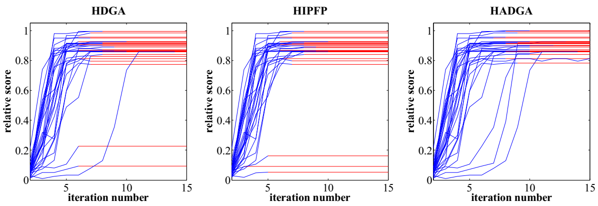


Figure 5: Solution path for normalized score of the original objective as a function of the number of iterations for three methods: 30 tests on CMU Hotel spaced by 50 frames. Red denotes for iterations when stoping criterion is satisfied.

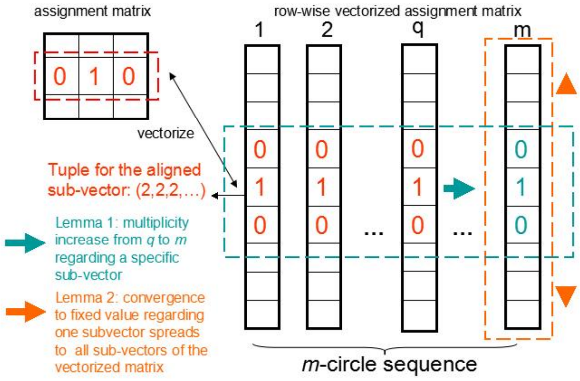

assignment matrix
row-wise vectorized assignment matrix
1
2
q
m
0 1 0
vectorize
0
1
0
Tuple for the aligned sub-vector: (2,2,2,...)
Lemma 1: multiplicity increase from q to m regarding a specific sub-vector
Lemma 2: convergence to fixed value regarding one subvector spreads to all sub-vectors of the vectorized matrix
...
m-circle sequence

Figure 6: The *dimension-spread* idea illustration regarding Lemma 1, Lemma 2: HADGA converges to a fixed point.

HDGA (Alg.1) converges to, where each $\mathbf{p}_s \in \{0,1\}^{n^2}$ by its definition is the vectorized permutation matrix of n unequal base vectors: $\mathbf{p}_s = [\mathbf{e}_{s_1}^T, \ldots, \mathbf{e}_{s_n}^T]^T$. The elements in $\{\mathbf{e}_s\}_{s=1}^n \in \{0,1\}^n$ are all zero except at the s-th position being 1. Given a fixed d, the sub-parts $\{\mathbf{p}_s^d \triangleq \mathbf{p}_s(nd-n+1 : nd)\}_{d=1}^n \in \{0,1\}^n$ in the sequence $\{\mathbf{p}_s\}_{s=1}^m$ also form a sequence $(\mathbf{p}_1^d, \mathbf{p}_2^d, ..., \mathbf{p}_m^d)$, and the total number of these sequences is n. One such sequence is illustrated by the green rectangle in Fig.6 which is the starting point for later proof.

The technical details are left in the constructive proof of Lemma 1 and 2. We first conceptually describe our key idea termed as *dimension-spread* as illustrated in Fig.6. Given n above defined sequences $\{(\mathbf{p}_1^d, \mathbf{p}_2^d, ..., \mathbf{p}_m^d)\}_{d=1}^n$, we can rewrite them to a more compact scalar form $(j_1^d, j_2^d, ..., j_m^d)$ where each integer $j^d \in [1, n]$ represents 1's position in $\mathbf{p}^d$. Assume the largest multiplicity by Definition 2 over all $\{(j_1^d, j_2^d, ..., j_m^d)\}_{d=1}^n$ is $q_k = \max\{q_d\}_{d=1}^n \in [1, m]$ where q_d is the multiplicity of $(j_1^d, j_2^d, ..., j_m^d)$. Lemma 1 first shows how to increase the multiplicity by $q_k = q_k + 1$, and further to m by distorting the affinity tensor $\mathbf{H}$. In other words, $(\mathbf{p}_1, \mathbf{p}_2, ..., \mathbf{p}_m)$ will converge *w.r.t.* its sub-part $(\mathbf{p}_1^k, \mathbf{p}_2^k, ..., \mathbf{p}_m^k)$. This step is visualized by the horizontally expanding green rectangle in Fig.6. Then Lemma 2 shows how to further enforce convergence for the whole $(\mathbf{p}_1, \mathbf{p}_2, ..., \mathbf{p}_m)$ to a fixed point in permutation space. Thus Lemma 2 establishes Theorem 2. This step is shown in Fig.6 by the vertically expanding orange rectangle.

Lemma 1. *For the* n *m-tuple* $\{(j_1^d, j_2^d, ..., j_m^d)\}_{d=1}^n$ *derived from the m-circle sequence* $(\boldsymbol{p}_1, \boldsymbol{p}_2, ..., \boldsymbol{p}_m)$ *generated by HDGA, by distorting the affinity tensor by HADGA, the largest multiplicity q_k can increase to m. In other words, the sub-part* $(\boldsymbol{p}_1^k, \boldsymbol{p}_2^k, ..., \boldsymbol{p}_m^k)$ *converges to a fixed point.*

Proof For the k-th sub-part $(\mathbf{p}_1^k, \mathbf{p}_2^k, ..., \mathbf{p}_m^k)$ associated with the largest multiplicity q_k, in its scalar form, suppose $j_1^k = j_2^k = \cdots = j_{q_k}^k \neq j_{q_k+1}^k \cdots j_m^k$ without loss of generality. We further suppose the element 1 in the first q_k sub-parts $\mathbf{p}_1^k, \mathbf{p}_2^k, ..., \mathbf{p}_{q_k}^k$ is at position a, *i.e.* $j_1^k = j_2^k = \cdots = j_{q_k}^k = a$, and $j_{q_k+1}^k = b$. Such settings indicate the first q_k+1 solutions out of the m consecutive ones assign node k in $\mathcal{G}_1$ to node a in $\mathcal{G}_2$ and the (q_k+1)-th solution assigns node k to node b (for $a \neq b$). Accordingly, for the (q_k+1)-th solution $\mathbf{p}_{q_k+1}$, suppose it assigns node l to node a, *i.e.* $j_{q_k+1}^l = a$ (for $l \neq a$). Note swapping from $\{l \leftrightarrow a, k \leftrightarrow b\}$ to $\{l \leftrightarrow b, k \leftrightarrow a\}$ for $\mathbf{p}_{q_k+1}$ will increase the multiplicity from q to q+1 *w.r.t.* $(\mathbf{p}_1^k, \mathbf{p}_2^k, ..., \mathbf{p}_m^k)$, while the current affinity tensor $\mathbf{H}$ disallows this swapped configuration as the score in Eq.2 will be smaller which contradicts the condition that the relaxed objective score is at its peak and stops climbing for the given m-circle sequence. We compare the function score difference before and after this swapping: in general, $2n^{2m-2}$ terms in the objective function that relate to the mapping $k \leftrightarrow a$ or $l \leftrightarrow b$ in the (q_k+1)-th solution will be affected. On the other hand, the new configuration for the q_k+1 solutions having the same match $k \leftrightarrow a$ will cover $n^{2(m-q_k-1)}$ terms. Thus a loose bound for the score decrease is $2n^{2m-2}$ given the tensor $\mathbf{H}$'s element is within [0,α_{q_k}]. In order to make the swapping becomes score ascending instead of decreasing, we perturb $\mathbf{H}$ by $\mathbf{H} = \mathbf{H} + \alpha_{q_k+1}\mathbf{H}^{q_k+1}$ and set $\alpha_{q_k+1} > \frac{2n^{2m-2}\alpha_{q_k}}{n^{2(m-q_k-1)}}$. Remember that in each iteration the Hungarian method in HDGA will return the global optimal solution, thus the swapping will be realized using the perturbed $\mathbf{H}$. Repeat such perturbation from q to m step by step along the same sub-part $(\mathbf{p}_1^k, \mathbf{p}_2^k, ..., \mathbf{p}_m^k)$ until q_k=m, which establishes the statement in Lemma 1. □

Lemma 2. *For the m-circle* $(\boldsymbol{p}_1, \boldsymbol{p}_2, ..., \boldsymbol{p}_m)$ *generated by HDGA, if it converges to a fixed point regarding the subpart* $(\boldsymbol{p}_1^k, \boldsymbol{p}_2^k, ..., \boldsymbol{p}_m^k)$*, then by distorting the affinity tensor* $\boldsymbol{H}$ *by HADGA,* $(\boldsymbol{p}_1, \boldsymbol{p}_2, ..., \boldsymbol{p}_m)$ *will converge to a fixed point in the permutation matrix space.*

Proof Given the k-th sub-part sequence whose multiplicity is m in its scalar form *i.e.* $j_1^k = j_2^k = \ldots = j_m^k = a$, it assigns node k in one graph to node a from the other. Suppose the largest multiplicity of the rest n-1 sub-parts *i.e.* $\{j_1^d, j_2^d, \ldots, j_m^d\}_{d=1, \neq k}^n$ is q. Disturbing the affinity tensor by $\mathbf{H} = \mathbf{H} + \alpha_{q+1}\mathbf{H}^{q+1}$ will not affect the fixed configuration $k \leftrightarrow a$ while increases q to q+1 by Lemma 1. This is because swapping out one of the m number of $k \leftrightarrow a$ will not increase the score after adding $\mathbf{H}^{q+1}$, thus the node matching for $k \leftrightarrow a$ will keep unchanged. Repeat the above step until the multiplicity of each sub-part sequence reaches m, which establishes the statement in Lemma 2. □

Acknowledgement

This work is partially supported by NSF IIS-1116886, IIS-1049694, NSFC (61025005, 61129001, 61221001), STCSM (14XD1402100, 13511504501) and the 111 Program (B07022). The authors are thankful to the anonymous reviewers for their valuable comments that improve the paper.

References

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

[2] I. M. Bomze and G. Danninger. A global optimization algorithm for concave quadratic programming problems. *SIAM Journal on Optimization*, pages 826–842, 1993. 4

[3] T. Caetano, J. McAuley, L. Cheng, Q. Le, and A. J. Smola. Learning graph matching. *IEEE Transaction on PAMI*, 31(6):1048–1058, 2009. 1, 2

[4] M. C. Chang and B. B. Kimia. Measuring 3d shape similarity by graph-based matching of the medial scaffolds. *Computer Vision and Image Understanding*, pages 707–720, 2011. 1, 2

[5] M. Chertok and Y. Keller. Efficient high order matching. *PAMI*, 2010. 3

[6] M. Cho, K. Alahari, and J. Ponce. Learning graphs to match. In *ICCV*, 2013. 2

[7] M. Cho, J. Lee, and K. M. Lee. Reweighted random walks for graph matching. In *ECCV*, 2010. 1, 2, 3, 5, 6

[8] O. Duchenne, F. Bach, I. Kweon, and J. Ponce. A tensor-based algorithm for high-order graph matching. In *CVPR*, 2009. 1, 2, 3, 4, 5

[9] G. Fielding and M. Kam. Weighted matchings for dense stereo correspondence. *Pattern Recognition*, pages 1151–1524, 2000. 1

[10] S. Gold and A. Rangarajan. A graduated assignment algorithm for graph matching. *IEEE Transaction on PAMI*, 1996. 1, 2, 5

[11] S. Gold and A. Rangarajan. Softmax to softassign: neural network algorithms for combinatorial opti-mization. *J. Artif. Neural Netw*, 1996. 4

[12] H. W. Kuhn. The hungarian method for the assignment problem. In *Export. Naval Research Logistics Quarterly*, pages 83–97, 1955. 1

[13] J. Lee, M. Cho, and K. M. Lee. Hyper-graph matching via reweighted randomwalks. In *CVPR*, 2011. 1, 2, 3, 5, 6, 7

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

[15] M. Leordeanu, M. Hebert, and R. Sukthankar. An integer projected fixed point method for graph matching and map inference. In *NIPS*, 2009. 1, 2, 3, 5

[16] M. Leordeanu, R. Sukthankar, and M. Hebert. Unsupervised learning for graph matching. *Int. J. Comput. Vis.*, pages 28–45, 2012. 2

[17] M. Leordeanu, A. Zanfir, and C. Sminchisescu. Semi-supervised learning and optimization for hypergraph matching. In *ICCV*, 2011. 2, 3

[18] C. Li, Q. Liu, W. Dong, X. Zhu, J. Liu, and H. Lu. Human age estimation based on locality and ordinal information. *IEEE Transaction on Cybernetics*, 2014. 7

[19] C. Li, Q. Liu, J. Liu, and H. Lu. Learning ordinal discriminative features for age estimation. In *CVPR*, 2012. 7

[20] A. Rangarajan, A. Yuille, and E. Mjolsness. Convergence properties of the softassign quadratic assignment algorithm. *Neural Computation*, 1999. 2

[21] P. A. Regalia and E. Kofidi. The higher-order power method revisited: Convergence proofs and effective initialization. In *ICASSP*, 2000. 3

[22] E. Rodolà, S. R. Bulà, T. Windheuser, M. Vestner, and D. Cremers. Dense non-rigid shape correspondence using random forests. In *CVPR*, 2014. 2

[23] R. Sinkhorn and A. Rangarajan. A relationship between arbitrary positive matrices and doubly stochastic matrices. *Ann. Math. Statistics*, 1964. 2

[24] Y. Suh, M. Cho, and K. M. Lee. Graph matching via sequential monte carlo. In *ECCV*, 2012. 2

[25] P. S. T. Cour and J. Shi. Balanced graph matching. In *NIPS*, 2006. 2

[26] Y. Tian, J. Yan, H. Zhang, Y. Zhang, X. Yang, and H. Zha. On the convergence of graph matching: Graduated assignment revisited. In *ECCV*, 2012. 2, 4, 6

[27] F. Viksten, P. Forssén, B. Johansson, and A. Moe. Comparison of local image descriptors for full 6 degree-of-freedom pose estimation. In *ICRA*, 2009. 7

[28] J. Xiao, H. Cheng, H. Sawhney, and F. Han. Vehicle detection and tracking in wide field-of-view aerial video. In *CVPR*, 2010. 1

[29] J. Yan, M. Cho, H. Zha, X. Yang, and S. Chu. A general multi-graph matching approach via graduated consistency-regularized boosting. *arXiv preprint arXiv:1502.05840*, 2015. 2

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

[31] J. Yan, J. Liu, Y. Li, Z. Niu, and Y. Liu. Visual saliency detection via rank-sparsity decomposition. In *ICIP*, 2010. 7

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

[33] J. Yan, J. Wang, H. Zha, X. Yang, and S. Chu. Multi-view point registration via alternating optimization. In *AAAI*, 2015. 2

[34] J. Yan, J. Wang, H. Y. Zha, and X. K. Yang. Consistency-driven alternating optimization for multi-graph matching: a unified approach. *IEEE Transactions on Image Processing*, 2015. 2

[35] J. Yan, M. Zhu, H. Liu, and Y. Liu. Visual saliency detection via sparsity pursuit. *Signal Processing Letters, IEEE*, 2010. 7

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

[37] R. Zass and A. Shashua. Probabilistic graph and hypergraph matching. In *CVPR*, 2008. 1, 2, 3, 5

[38] F. Zhou and F. D. Torre. Deformable graph matching. In *CVPR*, 2013. 1

[39] F. Zhou and F. D. L. Torre. Factorized graph matching. In *CVPR*, 2012. 1, 3, 4