A Flexible Tensor Block Coordinate Ascent Scheme for Hypergraph Matching

Quynh Nguyen Ngoc[1,2], Antoine Gautier[2], Matthias Hein[2]

[1]Max Planck Institute for Informatics, Saarbrücken, Germany
[2]Saarland University, Saarbrücken, Germany

Abstract

The estimation of correspondences between two images resp. point sets is a core problem in computer vision. One way to formulate the problem is graph matching leading to the quadratic assignment problem which is NP-hard. Several so called second order methods have been proposed to solve this problem. In recent years hypergraph matching leading to a third order problem became popular as it allows for better integration of geometric information. For most of these third order algorithms no theoretical guarantees are known. In this paper we propose a general framework for tensor block coordinate ascent methods for hypergraph matching. We propose two algorithms which both come along with the guarantee of monotonic ascent in the matching score on the set of discrete assignment matrices. In the experiments we show that our new algorithms outperform previous work both in terms of achieving better matching scores and matching accuracy. This holds in particular for very challenging settings where one has a high number of outliers and other forms of noise.

1. Introduction

Graph resp. hypergraph matching has been used in a variety of problems in computer vision such as object recognition [20], feature correspondences [7, 24], shape matching [11, 23, 25] and surface registration [27]. Given two sets of points, the task is to find the correspondences between them based on extracted features and/or geometric information. In general, the graph matching problem is NP-hard, therefore, many approximation algorithms have been proposed over the years. They can be grouped into second order and third order methods.

Among recent second order approaches, Leordeanu and Hebert [17] proposed the Spectral Matching (SM) algorithm, and Cour *et al*. [9] the Spectral Matching with Affine Constraint (SMAC). Both of these methods are based on the best rank-1 approximation of an affinity matrix. Leordeanu *et al*. later on proposed the Integer Projected Fixed Point (IPFP) algorithm [19], where they optimize the quadratic objective using a gradient-type algorithm interleaved with projection onto the discrete constraint set using e.g. the hungarian method. Lee *et al*. [15] tackle the graph matching problem using stochastic sampling, whereas Cho *et al*. [6] propose a reweighted random walk (RRWM) where the reweighting jumps are aiming at enforcing the matching constraints. Recently, Cho *et al*. [8] proposed a novel max pooling matching (MPM), where they tweak the power method to better cope with noise in the affinities. Although the algorithm comes without theoretical guarantees, it turns out to perform very well in practice, in particular, when one has a large number of outliers. Other second order methods include the work of Zhou and Torre [29] and Zaslavskiy *et al*. [25], where they propose deformable graph matching (DGM) and a path-following algorithm respectively.

As second order methods are limited to pairwise similarities between two correspondences, higher order methods can integrate better geometric information. While they have higher computational complexity, they can cope better with geometric transformations such as scaling or other forms of noise. Compared to second order methods, higher order approaches are less well studied in literature due to the difficulty in handling the third order optimization problem.

Duchenne *et al*. [10] formulated the hypergraph matching problem as a third order tensor optimization problem and proposed a higher order power method for solving the problem. This approach has shown significant improvements over second order ones which were proposed earlier. Zass and Shashua [26] introduced a probabilistic view to the problem, and proposed a Hypergraph Matching (HGM) algorithm. Their idea is to marginalize the tensor to a vector and solve again a lower dimensional problem. Chertok and Keller [4] extended this idea and marginalized the tensor to a matrix, resulting in a second order matching problem which is solved by spectral methods. Both methods are based on tensor marginalization, which leads to a loss of information. Moreover, they cannot effectively handle the one-to-one matching constraint during the iterations which is only considered at the final discretization step. Jungminlee *et al.* [16] extended the reweighted random walk approach of [15] to the third order setting. Their algorithm aims at enforcing the matching constraint via a bi-stochastic normalization scheme done for each iteration. In [27], Zeng *et al.* proposed to use pseudo-boolean optimization [1] for 3D surface matching, where higher order terms are decomposed into second order terms and then a quadratic pseudo-boolean optimization (QPBO) [13] algorithm is used to solve the problem.

In this paper, we propose an algorithmic framework for solving the hypergraph matching problem based on a tensor block coordinate scheme. The key idea of our framework is to use a multilinear reformulation of the original objective function. In particular, we solve the third order graph matching problem by solving an equivalent fourth order one. Based on this reformulation, we derive two new hypergraph matching algorithms. We can guarantee for both algorithms monotonic ascent in the third order matching score. In the experiments we show that our new algorithms outperform previous work both in terms of achieving better matching score and accuracy. This holds in particular for very challenging settings where one has a high number of outliers and other forms of noise. All proofs can be found in the supplement.

2. Hypergraph Matching Formulation

We formulate the correspondence problem as a hypergraph matching problem. Hypergraphs allow modeling of relations which are not only pairwise as in graphs but involve groups of vertices. In this paper, we consider 3-uniform hypergraphs, that is each hyperedge contains 3 vertices. k-uniform hypergraphs can be modelled as k-th order tensors which is the point of view we adopt in this paper.

Given two attributed point sets $G = (V, A)$ and $G' = (V', A')$ with $n_1 = |V| \leq n_2 = |V'|$, the matching problem can be formulated as finding a subset $\mathcal{X}$ in the set of correspondences $V \times V'$. The subset $\mathcal{X}$ can be represented by the binary assignment matrix $X \in \{0,1\}^{n_1 \times n_2}$ where $X_{ij} = 1$ if $v_i \in V$ matches $v'_j \in V'$ and $X_{ij} = 0$ else. A one-to-one matching X is an element of the set

$$M = \big\{X \in \{0,1\}^{n_1 \times n_2} \,\big|\, \sum_{i=1}^{n_1} X_{ij} \leq 1, \quad \sum_{j=1}^{n_2} X_{ij} = 1\big\},$$

that is we consider matchings which assign each vertex of V to exactly one vertex in V'. We further assume that we have a function $\mathcal{F}^3 : (V \times V')^3 \to \mathbb{R}_+$ which assigns to each chosen triple $\{v_{i_1}, v_{i_2}, v_{i_3}\} \subset V$ and $\{v'_{j_1}, v'_{j_2}, v'_{j_3}\} \subset V'$ its similarity weight $\mathcal{F}^3_{(i_1,j_1),(i_2,j_2),(i_3,j_3)}$. Finally, the score S of a matching X can then be defined as

$$S(X) = \sum_{i_1,i_2,i_3}^{n_1} \sum_{j_1,j_2,j_3}^{n_2} \mathcal{F}^3_{(i_1,j_1),(i_2,j_2),(i_3,j_3)} X_{i_1 j_1} X_{i_2 j_2} X_{i_3 j_3}.$$

In order to facilitate notation we introduce a linear ordering in $V \times V'$ and thus can rewrite X as a vector $\mathbf{x} \in \{0,1\}^n$ with $n = n_1 n_2$ and $\mathcal{F}^3$ becomes a tensor in $\mathbb{R}^{n \times n \times n}$ so that the matching score, $S^3 : \mathbb{R}^n \to \mathbb{R}$, is finally written as

$$S^3(\mathbf{x}) = \sum_{i,j,k=1}^{n} \mathcal{F}^3_{ijk}\, \mathbf{x}_i\, \mathbf{x}_j\, \mathbf{x}_k. \tag{1}$$

An m-th order tensor $\mathcal{G} \in \mathbb{R}^{n \times \cdots \times n}$ is called symmetric if its entries $\mathcal{G}_{i_1 \ldots i_m}$ are invariant under any permutation of their indices $\{i_1, \ldots, i_m\}$. Note that S^3 is the sum of the componentwise product of the tensor $\mathcal{F}^3$ with the symmetric tensor $\mathcal{G}^3_{ijk} = \mathbf{x}_i \mathbf{x}_j \mathbf{x}_k$. Thus any non-symmetric part of $\mathcal{F}^3$ is "averaged" out and we can without loss of generality assume that $\mathcal{F}^3$ is a symmetric tensor[1]. In principle, one can integrate unary terms on the diagonal $\mathcal{F}^3_{iii}$, $i = 1, \ldots, n$ and pairwise potentials $\mathcal{F}^3_{ijj}$, $i \neq j$. However, the tensors we consider in Section 5 have just terms of order 3, that is $\mathcal{F}^3_{ijk} = 0$ if $i = j$, $i = k$ or $j = k$.

3. Mathematical Foundations for the Tensor-Block-Coordinate Ascent Method

In this section we derive the basis for our tensor block coordinate ascent method to optimize S^3 directly over the set M of assignment matrices. The general idea is to optimize instead of a homogeneous polynomial function the associated multilinear form.

3.1. Lifting the Tensor and Multilinear Forms

Instead of optimizing the third order problem in Equation (1), we define a corresponding fourth order problem by lifting the third order tensor $\mathcal{F}^3$ to a fourth order tensor $\mathcal{F}^4$,

$$\mathcal{F}^4_{ijkl} = \mathcal{F}^3_{ijk} + \mathcal{F}^3_{ijl} + \mathcal{F}^3_{ikl} + \mathcal{F}^3_{jkl}. \tag{2}$$

This might be counter-intuitive at first in particular as previous work [26, 4] has done the opposite way by reducing the third order problem to a second order problem. However, lifting the tensor from a third order tensor to a fourth order tensor allows us to use structure of even order tensor which is not present for tensors of odd order. In particular, the score function S^3 is not convex.

Lemma 3.1 *Let S^3 be defined as in Equation (1). If S^3 is not constant zero, then $S^3 : \mathbb{R}^n \to \mathbb{R}$ is not convex.*

[1] If $\mathcal{F}^3$ is not already symmetric, then one can define $\tilde{\mathcal{F}}^3{}_{ijk} = \frac{1}{3!}\sum_{\sigma \in \mathfrak{S}_3} \mathcal{F}^3_{\sigma(i)\sigma(j)\sigma(k)}$, $i, j, k = 1, \ldots, n$ where $\mathfrak{S}_3$ is the set of permutations of three elements.

The convexity is crucial to derive our block coordinate ascent scheme. We do not work with the score function directly but with the multilinear form associated to it.

Definition 3.2 (Multilinear Form) *The multilinear form $F^m\colon \mathbb{R}^n \times \ldots \times \mathbb{R}^n \to \mathbb{R}$ associated to an m-th order tensor $\mathcal{F}^m$ is given by:*

$$F^m(\mathbf{x}^1, \ldots, \mathbf{x}^m) = \sum_{i_1,\ldots,i_m}^{n} \mathcal{F}^m_{i_1\ldots i_m} \mathbf{x}^1_{i_1} \ldots \mathbf{x}^m_{i_m},$$

and the score function $S^m\colon \mathbb{R}^n \to \mathbb{R}$ associated to $\mathcal{F}^m$ is defined by $S^m(\mathbf{x}) = F^m(\mathbf{x}, \ldots, \mathbf{x})$.

We write $F^4(\mathbf{x}, \mathbf{x}, \cdot, \cdot)$ to denote the matrix in $\mathbb{R}^{n\times n}$ such that $F^4(\mathbf{x}, \mathbf{x}, \cdot, \cdot)_{kl} = \sum_{i,j=1}^n \mathcal{F}^4_{ijkl}\mathbf{x}_i\mathbf{x}_j$ for all $1 \le k, l \le n$. Note that if F^4 is associated to a symmetric tensor, then the position of the dots do not matter.

It might seem at first that the lift from third to fourth order defined in (2) should lead to a huge computational overhead. However,

$$S^4(\mathbf{x}) = 4\, F^3(\mathbf{x}, \mathbf{x}, \mathbf{x}) \sum_{i=1}^n \mathbf{x}_i = 4\, S^3(\mathbf{x}) \sum_{i=1}^n \mathbf{x}_i \qquad (3)$$

and $\sum_{i=1}^n \mathbf{x}_i = n_1$, for all $\mathbf{x} \in M$, thus we have the following equivalence of the optimization problems,

$$\max_{\mathbf{x}\in M} F^4(\mathbf{x}, \mathbf{x}, \mathbf{x}, \mathbf{x}) \;\equiv\; \max_{\mathbf{x}\in M} F^3(\mathbf{x}, \mathbf{x}, \mathbf{x}) \qquad (4)$$

3.2. Convex Score Functions and Equivalence of Optimization Problems

The use of multilinear forms corresponding to convex score functions is crucial for the proposed algorithms. The following lemma shows that even if we optimize the multilinear form F^4 instead of S^4 we get ascent in S^4. We can either fix all but one argument or all but two arguments, which leads to the two variants of the algorithm in Sec. 4.

Lemma 3.3 *Let $\mathcal{F}^4$ be a fourth order symmetric tensor. If $S^4\colon \mathbb{R}^n \to \mathbb{R}$ is convex, then for all $\mathbf{x}, \mathbf{y}, \mathbf{z}, \mathbf{t} \in \mathbb{R}^n$:*

1. $F^4(\mathbf{x}, \mathbf{x}, \mathbf{y}, \mathbf{y}) \le \max_{\mathbf{u}\in\{\mathbf{x},\mathbf{y}\}} F^4(\mathbf{u}, \mathbf{u}, \mathbf{u}, \mathbf{u})$,
2. $F^4(\mathbf{x}, \mathbf{y}, \mathbf{z}, \mathbf{t}) \le \max_{\mathbf{u}\in\{\mathbf{x},\mathbf{y},\mathbf{z},\mathbf{t}\}} F^4(\mathbf{u}, \mathbf{u}, \mathbf{u}, \mathbf{u})$.

The following theorem shows that the optimization of the multilinear form F^4 and the score function S^4 are equivalent in the sense that there exists a globally optimal solution of the first problem which is also globally optimal for the second problem.

Theorem 3.4 *Let $\mathcal{F}^4$ be a fourth order symmetric tensor and suppose that $S^4\colon \mathbb{R}^n \to \mathbb{R}$ is convex. Then it holds for any compact constraint set $D \subset \mathbb{R}^n$,*

$$\begin{aligned}\max_{\mathbf{x}\in D} F^4(\mathbf{x}, \mathbf{x}, \mathbf{x}, \mathbf{x}) &= \max_{\mathbf{x},\mathbf{y}\in D} F^4(\mathbf{x}, \mathbf{x}, \mathbf{y}, \mathbf{y}) \qquad (5)\\ &= \max_{\mathbf{x},\mathbf{y},\mathbf{z},\mathbf{t}\in D} F^4(\mathbf{x}, \mathbf{y}, \mathbf{z}, \mathbf{t})\end{aligned}$$

As we cannot guarantee that S^4 is convex for our chosen affinity tensor, we propose a modification making it convex. Importantly, this modification turns out to be constant on the set of matchings M.

Proposition 3.5 *Let $\mathcal{F}^4$ be a fourth order symmetric tensor. Then for any $\alpha \ge 3\left\|\mathcal{F}^4\right\|_2$, where $\left\|\mathcal{F}^4\right\|_2 := \sqrt{\sum_{i,j,k,l=1}^n (\mathcal{F}^4_{ijkl})^2}$, the function*

$$S^4_\alpha(\mathbf{x}) := S^4(\mathbf{x}) + \alpha\, \|\mathbf{x}\|_2^4 \qquad (6)$$

is convex on $\mathbb{R}^n$ and, for any $\mathbf{x} \in M$, we have

$$S^4_\alpha(\mathbf{x}) = S^4(\mathbf{x}) + \alpha\, n_1^2. \qquad (7)$$

One of the key aspects of the algorithms we derive in the next section is that we optimize the multilinear form F^4 instead of the function S^4. This requires that we are able to extend the modification S^4_α resp. $\|\mathbf{x}\|_2^4$ to a multilinear form.

Proposition 3.6 *The symmetric tensor $\mathcal{G}^4 \in \mathbb{R}^{n\times n\times n\times n}$ with corresponding symmetric multilinear form defined as*

$$G^4(\mathbf{x}, \mathbf{y}, \mathbf{z}, \mathbf{t}) = \frac{\langle \mathbf{x}, \mathbf{y}\rangle\langle \mathbf{z}, \mathbf{t}\rangle + \langle \mathbf{x}, \mathbf{z}\rangle\langle \mathbf{y}, \mathbf{t}\rangle + \langle \mathbf{x}, \mathbf{t}\rangle\langle \mathbf{y}, \mathbf{z}\rangle}{3}$$

satisfies $G^4(\mathbf{x}, \mathbf{x}, \mathbf{x}, \mathbf{x}) = \|\mathbf{x}\|_2^4$.

Thus in the algorithm we optimize finally

$$F^4_\alpha(\mathbf{x}, \mathbf{y}, \mathbf{z}, \mathbf{t}) = F^4(\mathbf{x}, \mathbf{y}, \mathbf{z}, \mathbf{t}) + \alpha\, G^4(\mathbf{x}, \mathbf{y}, \mathbf{z}, \mathbf{t}).$$

Discussion: In [21] they proposed a general convexification strategy for arbitrary score functions where, similar to our modification, the added term is constant on the set of assignment matrices M. However, as the added term is inhomogeneous, it cannot be extended to a symmetric multilinear form and thus does not work in our framework. In second order graph matching also several methods use convexified score functions in various ways [25, 28, 29]. However, for none of these methods it is obvious how to extend it to a third order approach for hypergraph matching.

4. Tensor Block Coordinate Ascent for Hypergraph Matching

The key idea of both algorithms is that instead of directly optimizing the score function S^4 we optimize the corresponding multilinear form. Lemma 3.3 allows us then to connect the latter problem to the problem we are actually interested in. In both variants we directly optimize over the discrete set M of possible matchings, that is, there is no relaxation involved. Moreover, we show monotonic ascent for both methods. In most cases we achieve higher scores than all other higher order methods, which is reflected in significantly better performance in the experiments in Section 5.

The original graph matching problem is to maximize the score function S^3 over all assignment matrices in M. This combinatorial optimization problem is NP-hard. As stated above, instead of working with the original score function S^3, we use the lifted tensor and thus the lifted score function S^4. As we cannot guarantee that S^4 is convex, we optimize finally S^4_α resp. the associated multilinear form F^4_α. However, we emphasize that even though we work with the lifted objects, both algorithms prove monotonic ascent with respect to the original score function S^3 over the set M. The central element of both algorithms is Lemma 3.3 and the idea to optimize the multilinear form, instead of the score function directly, combined with the fact that for assignment matrices both modifications (lifting and convexification) are constant and thus do not change the problem. In order to simplify the notation in the algorithms we discard the superscript and write F_α instead of F^4_α as the algorithms only involve fourth order tensors.

4.1. Tensor Block Coordinate Ascent via Linear Assignment Problems

The first algorithm uses a block coordinate ascent scheme to optimize the multilinear form F_α. Fixing all but one argument and maximizing that over all assignment matrices leads to a linear assignment problem which can be solved globally optimal by the Hungarian algorithm [14, 2] - this is used for steps 1)-4) in Algorithm 1. However, optimization of the multilinear form is not necessarily equivalent to optimizing the score function. This is the reason why we use Lemma 3.3 in step 5) to come back to the original problem. The following theorem summarizes the properties of Algorithm 1.

Theorem 4.1 *Let $\mathcal{F}^4$ be a fourth order symmetric tensor. Then the following holds for Algorithm 1:*

1. *The sequence $F_\alpha(\mathbf{x}^k, \mathbf{y}^k, \mathbf{z}^k, \mathbf{t}^k)$ for $k = 1, 2, \ldots$ is strictly monotonically increasing or terminates.*
2. *The sequence of scores $S^4(\mathbf{u}^m)$ for $m = 1, 2, \ldots$ is strictly monotonically increasing or terminates. For every m, $\mathbf{u}^m \in M$ is a valid assignment matrix.*

Algorithm 1: BCAGM

Input: Lifted affinity tensor $\mathcal{F}^4$, $\alpha = 3\left\|\mathcal{F}^4\right\|_2$, $(\mathbf{x}^0, \mathbf{y}^0, \mathbf{z}^0, \mathbf{t}^0) \in M \times M \times M \times M$, $k = 0$, $m = 0$
Output: $\mathbf{x}^* \in M$
Repeat

1. $\tilde{\mathbf{x}}^{k+1} = \arg\max_{\mathbf{x}\in M} F_\alpha(\mathbf{x}, \mathbf{y}^k, \mathbf{z}^k, \mathbf{t}^k)$
2. $\tilde{\mathbf{y}}^{k+1} = \arg\max_{\mathbf{y}\in M} F_\alpha(\tilde{\mathbf{x}}^{k+1}, \mathbf{y}, \mathbf{z}^k, \mathbf{t}^k)$
3. $\tilde{\mathbf{z}}^{k+1} = \arg\max_{\mathbf{z}\in M} F_\alpha(\tilde{\mathbf{x}}^{k+1}, \tilde{\mathbf{y}}^{k+1}, \mathbf{z}, \mathbf{t}^k)$
4. $\tilde{\mathbf{t}}^{k+1} = \arg\max_{\mathbf{t}\in M} F_\alpha(\tilde{\mathbf{x}}^{k+1}, \tilde{\mathbf{y}}^{k+1}, \tilde{\mathbf{z}}^{k+1}, \mathbf{t})$
5. **if** $F_\alpha(\tilde{\mathbf{x}}^{k+1}, \tilde{\mathbf{y}}^{k+1}, \tilde{\mathbf{z}}^{k+1}, \tilde{\mathbf{t}}^{k+1}) = F_\alpha(\mathbf{x}^k, \mathbf{y}^k, \mathbf{z}^k, \mathbf{t}^k)$ **then**
   - $\mathbf{u}^{m+1} = \underset{\mathbf{u}\in\{\tilde{\mathbf{x}}^{k+1}, \tilde{\mathbf{y}}^{k+1}, \tilde{\mathbf{z}}^{k+1}, \tilde{\mathbf{t}}^{k+1}\}}{\arg\max} F_\alpha(\mathbf{u}, \mathbf{u}, \mathbf{u}, \mathbf{u})$
   - **if** $F_\alpha(\mathbf{u}^{m+1}, \mathbf{u}^{m+1}, \mathbf{u}^{m+1}, \mathbf{u}^{m+1}) = F_\alpha(\tilde{\mathbf{x}}^{k+1}, \tilde{\mathbf{y}}^{k+1}, \tilde{\mathbf{z}}^{k+1}, \tilde{\mathbf{t}}^{k+1})$ **then return**
   - $\mathbf{x}^{k+1} = \mathbf{y}^{k+1} = \mathbf{z}^{k+1} = \mathbf{t}^{k+1} = \mathbf{u}^{m+1}$
   - $m = m + 1$

   else $\mathbf{x}^{k+1} = \tilde{\mathbf{x}}^{k+1}$, $\mathbf{y}^{k+1} = \tilde{\mathbf{y}}^{k+1}$, $\mathbf{z}^{k+1} = \tilde{\mathbf{z}}^{k+1}$, $\mathbf{t}^{k+1} = \tilde{\mathbf{t}}^{k+1}$

   end
6. $k = k + 1$

---

3. *The sequence of original third order scores $S^3(\mathbf{u}^m)$ for $m = 1, 2, \ldots$ is strictly monotonically increasing or terminates.*
4. *The algorithm terminates after a finite number of iterations.*

We would like to note that all statements of Theorem 4.1 remain valid for $\alpha = 0$ if S^4 is convex. This is also the motivation why we run the algorithm first with $\alpha = 0$ until we get no further ascent and only then we set $\alpha = 3\|\mathcal{F}\|_2$. It turns out that in the experiments often the phase of the algorithm with $\alpha = 0$ leads automatically to a homogeneous solution and no further improvement is achieved when setting $\alpha = 3\|\mathcal{F}\|_2$. This suggests that the constructed score functions in the experiments are already convex or at least close to being convex.

4.2. Tensor Block Coordinate Ascent via a Sequence of Quadratic Assignment Problems

The second algorithm uses a block coordinate ascent scheme to optimize the multilinear form F_α where now two arguments are fixed and one maximizes over the other two.

The resulting problem is equivalent to a quadratic assignment problem (QAP) which is known to be NP-hard. Thus a globally optimal solution as for the linear assignment problem in Algorithm 1 is out of reach. Instead we require a sub-algorithm Ψ which delivers monotonic ascent for the QAP

$$\max_{\mathbf{x}\in M} \langle \mathbf{x}, A\mathbf{x}\rangle \,, \tag{8}$$

where $A \in \mathbb{R}^{n\times n}$ is a nonnegative symmetric matrix. As in Algorithm 1, we go back to the optimization of the score function in step 3) using Lemma 3.3. The following theorem summarizes the properties of Algorithm 2.

Algorithm 2: BCAGM-Ψ

Input: Lifted affinity tensor $\mathcal{F}^4$, $\alpha = 3\left\|\mathcal{F}^4\right\|_2$, $(\mathbf{x}^0, \mathbf{y}^0) \in M \times M,\ k = 0,\ m = 0,$ $\mathbf{z} = \Psi(A, \mathbf{x}^k)$ is an algorithm for the QAP in (8) which provides monotonic ascent, that is $\langle \mathbf{z}, A\mathbf{z}\rangle \geq \langle \mathbf{x}^k, A\mathbf{x}^k\rangle$

Output: $\mathbf{x}^* \in M$

Repeat

1. $\tilde{\mathbf{x}}^{k+1} = \Psi\big(F_\alpha(\cdot,\cdot,\mathbf{y}^k,\mathbf{y}^k),\ \mathbf{x}^k\big)$
2. $\tilde{\mathbf{y}}^{k+1} = \Psi\big(F_\alpha(\tilde{\mathbf{x}}^{k+1},\tilde{\mathbf{x}}^{k+1},\cdot,\cdot), \mathbf{y}^k\big)$
3. **if** $F_\alpha(\tilde{\mathbf{x}}^{k+1},\tilde{\mathbf{x}}^{k+1},\tilde{\mathbf{y}}^{k+1},\tilde{\mathbf{y}}^{k+1}) = F_\alpha(\mathbf{x}^k,\mathbf{x}^k,\mathbf{y}^k,\mathbf{y}^k)$ **then**
   - $\mathbf{u}^{m+1} = \underset{\mathbf{u}\in\{\tilde{\mathbf{x}}^{k+1},\tilde{\mathbf{y}}^{k+1}\}}{\arg\max}\ F_\alpha(\mathbf{u},\mathbf{u},\mathbf{u},\mathbf{u})$
   - **if** $F_\alpha(\tilde{\mathbf{x}}^{k+1},\tilde{\mathbf{x}}^{k+1},\tilde{\mathbf{y}}^{k+1},\tilde{\mathbf{y}}^{k+1}) = F_\alpha(\mathbf{u}^{m+1},\mathbf{u}^{m+1},\mathbf{u}^{m+1},\mathbf{u}^{m+1})$ **then return**
   - $\mathbf{x}^{k+1} = \mathbf{y}^{k+1} = \mathbf{u}^{m+1}$
   - $m = m + 1$

   else $\mathbf{x}^{k+1} = \tilde{\mathbf{x}}^{k+1}$, $\mathbf{y}^{k+1} = \tilde{\mathbf{y}}^{k+1}$.
   end
4. $k = k + 1$

Theorem 4.2 *Let $\mathcal{F}^4$ be a fourth order symmetric tensor and let Ψ be an algorithm for the QAP which yields monotonic ascent. Then the following holds for Algorithm 2:*

1. *The sequence $F_\alpha(\mathbf{x}^k,\mathbf{x}^k,\mathbf{y}^k,\mathbf{y}^k)$ for $k = 1, 2, \ldots$ is strictly monotonically increasing or terminates.*
2. *The sequence of scores $S^4(\mathbf{u}^m)$ for $m = 1, 2, \ldots$ is strictly monotonically increasing or terminates. For every m, $\mathbf{u}^m \in M$ is a valid assignment matrix.*
3. *The sequence of original third order scores $S^3(\mathbf{u}^m)$ for $m = 1, 2, \ldots$ is strictly monotonically increasing or terminates.*
4. *The algorithm terminates after a finite number of iterations.*

In analogy to Theorem 4.1 all statements of Theorem 4.2 remain valid for $\alpha = 0$ if S^4 is convex. Thus we use the same initialization strategy with $\alpha = 0$ as described above. There are several methods available which we could use for the sub-routine Ψ in the algorithm. We decided to use the recent max pooling algorithm [8] and the IPFP algorithm [19] and use the Hungarian algorithm to turn their output into a valid assignment matrix in M. It turns out that the combination of our tensor block coordinate ascent scheme using their algorithms as a sub-routine yields very good performance on all datasets.

5. Experiments

We evaluate our hypergraph matching algorithms on standard synthetic benchmarks and realistic image datasets by comparing them with state-of-the-art third order and second order approaches. In particular, we use the following third order approaches: Tensor Matching (TM) [10], Hypergraph Matching via Reweighted Random Walks (RRWHM) [16], Hypergraph Matching (HGM) [26]. For second order approaches, Max Pooling Matching (MPM) [8], Reweighted Random Walks for Graph Matching (RRWM) [6], Integer Projected Fixed Point (IPFP) [19], and Spectral Matching (SM) [17] are used. We denote our tensor block coordinate ascent methods as BCAGM from Algorithm 1 which uses the Hungarian algorithm as subroutine, and BCAGM+MP and BCAGM+IPFP from Algorithm 2 which uses MPM [8] and IPFP [19] respectively as subroutine. MPM has recently outperformed other second order methods in the presence of a large number of outliers without deformation. Thus, it serves as a very competitive candidate with third order approaches in this setting. For all the algorithms, we use the authors' original implementation.

In all experiments below, we use the all-ones vector as starting point for all the methods. Moreover, we apply the Hungarian algorithm to the output of every algorithm to turn it into a proper matching.

Generation of affinity tensor/matrix: To build the affinity tensor for the third order algorithms, we follow the approach of Duchenne *et al*. [10]: for each triple of points in the same image we compute a feature vector f based on the angles of the triangle formed by those three points. Let f_{i_1,i_2,i_3} and f_{j_1,j_2,j_3} denote two feature vectors for two triples (i_1, i_2, i_3) and (j_1, j_2, j_3) in two corresponding images. Then we compute the third order affinity tensor as:

$$\mathcal{F}^3_{(i_1,j_1),(i_2,j_2),(i_3,j_3)} = \exp(-\gamma \left\|f_{i_1,i_2,i_3} - f_{j_1,j_2,j_3}\right\|_2^2) \tag{9}$$

where γ is the inverse of the mean of all squared distances. As shown by Duchenne *et al*. [10] these affinities increase the geometric invariance compared to second order models making it more robust to transformations, like translations, rotations and scalings. This partially explains why third order approaches are preferred to second order ones in this regard. As the computation of the full third order affinity tensor is infeasible in practice, we adopt the sampling strategy proposed by [10, 16].

The affinity matrix for all the second order methods are built by following [8, 6], where the Euclidean distance of two point pairs is used to compute the pairwise similarity:

$$\mathcal{F}^2_{(i_1,j_1),(i_2,j_2)} = \exp(-|d^P_{i_1 i_2} - d^Q_{j_1 j_2}|^2/\sigma_s^2) \quad (10)$$

where $d^P_{i_1 i_2}$ is the Euclidean distance between two points i_1, $i_2 \in P$, $d^Q_{j_1 j_2}$ is the distance between the points $j_1, j_2 \in Q$ and σ_s is a normalization parameter specified below.

5.1. Synthetic Dataset

This section presents a comparative evaluation on matching point sets P and Q in $\mathbb{R}^2$. We follow the approach in [8] for the creation of the point sets. That is, for P we sample n_{in} inlier points from a Gaussian distribution $\mathcal{N}(0, 1)$. The points in Q are generated by adding Gaussian deformation noise $\mathcal{N}(0, \sigma^2)$ to the point set P. Furthermore, n_{out} outlier points are added to Q sampled from $\mathcal{N}(0, 1)$. We test all matching algorithms under different changes in the data: outliers, deformation and scaling (i.e. we multiply the coordinates of the points in Q by a constant factor).

In the outlier setting, we perform two test cases as follows. In the first test case, we vary the number of outliers from 0 to very challenging 200, which is 20-times more than the number of inliers n_{in}, while σ is set to 0.01 and no scaling is used. In the second case, we set σ to 0.03 and scale all the points in Q by a factor of 1.5, making the problem much more difficult. This test shows that third order methods are very robust against scaling compared to second order methods. In all the plots in this section, each quantitative result was obtained by averaging over 100 trials. The accuracy is computed as the ratio between the number of correct matches and the number of inlier points. We use $\sigma_s = 0.5$ in the affinity construction for the second order methods as suggested by [8]. The results in Figure 1 show that our algorithms are robust w.r.t. outliers even if deformation and scaling are involved. When the deformation is negligible as shown in Figure 1a, MPM and BCAGM+MP perform best followed by BCAGM, BCAM+IPFP and RRWHM. In a more realistic setting as shown in Figure 1b, where outlier, deformation and scaling are present in the data, our algorithms outperform all other methods. One can state that BCAGM+MP transfers the robustness of MPM to a third order method and thus is able to deal with scaling which is difficult for second order methods.

In the deformation setting, the deformation noise σ is varied from 0 to challenging 0.4 while the number of inliers

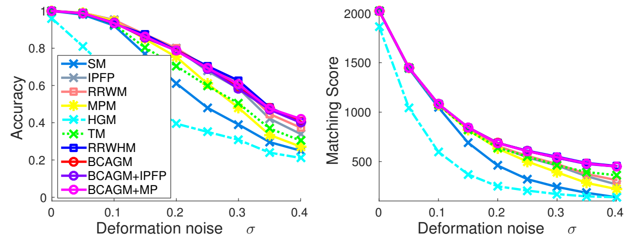
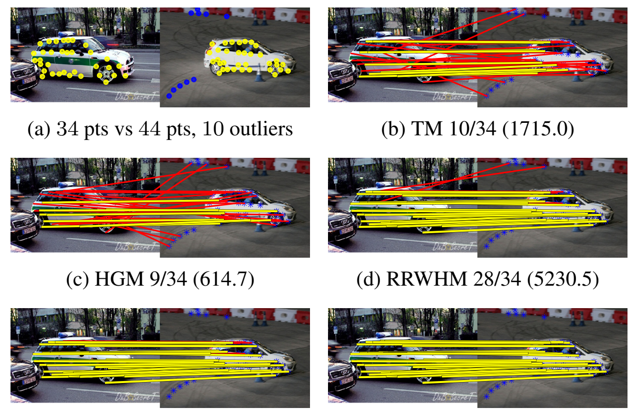


Figure 2: Robustness of matching algorithms w.r.t. deformation. ($n_{in} = 20, n_{out} = 0, scale = 1.$)



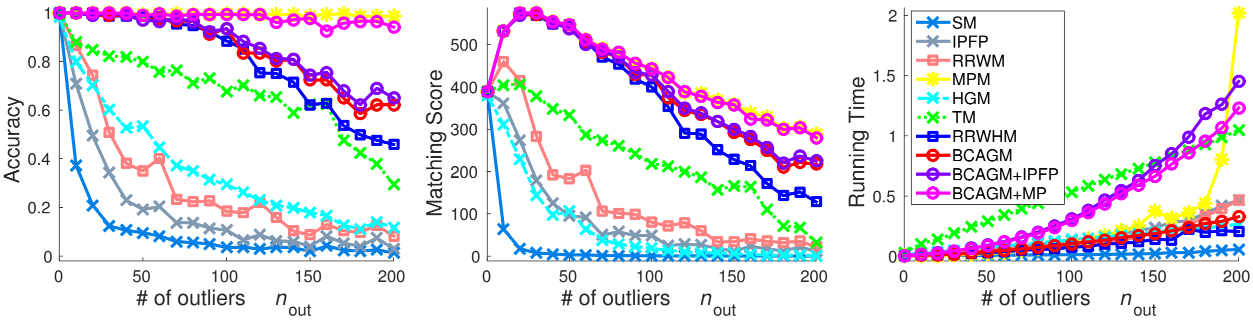
(a) 34 pts vs 44 pts, 10 outliers
(b) TM 10/34 (1715.0)
(c) HGM 9/34 (614.7)
(d) RRWHM 28/34 (5230.5)
(e) BCAGM 28/34 (5298.8)
(f) BCAGM+MP 34/34 (5377.3)

Figure 4: Car dataset: the number of correct matches and the objective score are reported. (Best viewed in color.)

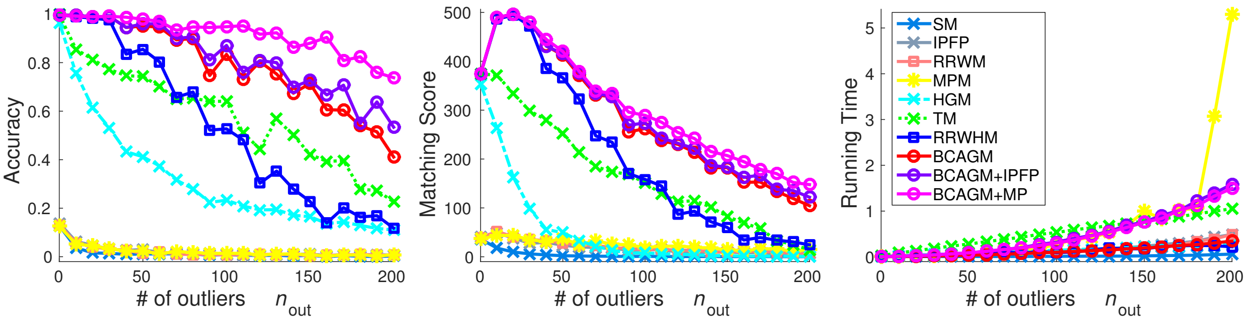
n_{in} is fixed to 20. There are no outlier and scale changes in this setting. This type of experiment has been used in previous works [8, 16]. where σ has been varied from 0 to 0.2. The result in Figure 2 shows that our algorithms are quite competitive with RRWHM which has been shown to be robust to deformation [16]. It is interesting to note that while MPM is very robust against outliers (without scaling) it is outperformed if the deformation is significantly increased (without having outliers). However, note that even though our BCAGM+MP uses MPM as a subroutine, it is not affected by this slight weakness. Considering all experiments our BCAGM+MP and to a slightly weaker extent BCAGM and BCAGM+IPFP outperform all other methods in terms of robustness to outliers, deformation and scaling.
Runtime: Figure 1 shows that the running time of our algorithms is competitive in comparison to other higher order and even second order methods. In particular, BCAGM is always among the methods with lowest running time (more results in the supplementary material).

5.2. CMU House Dataset

The CMU house dataset has been widely used in previous work [6, 16, 10, 29] to evaluate matching algorithms. In this dataset, 30 landmark points are manually tracked over a sequence of 111 images, which are taken from the same object under different view points. In this experiment, "base-

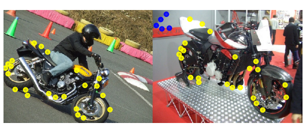


(a) $n_{in} = 10, \sigma = 0.01, scale = 1$

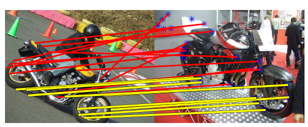


(b) $n_{in} = 10, \sigma = 0.03, scale = 1.5$

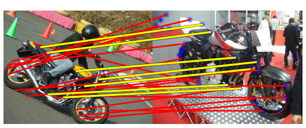
Figure 1: Matching point sets in $\mathbb{R}^2$: each row shows the accuracy, matching score and running time of all algorithms. The number of outliers has been varied from 0 to 200. (a) Increasing number of outliers with slight deformation and no scaling. (b) Increasing number of outliers with larger deformation and scaling. (Best viewed in color.)



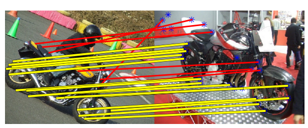
(a) 23 pts vs 28 pts, 5 outliers



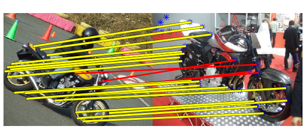
(b) TM 9/23 (2502.6)



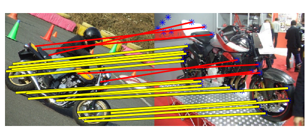
(c) HGM 7/23 (1435.8)



(d) RRWHM 16/23 (4059.4)



(e) BCAGM 19/23 (4016.2)



(f) BCAGM+MP 16/23 (4133.6)

Figure 5: Motorbike dataset: the number of correct matches and the objective score are reported. (Best viewed in color.)

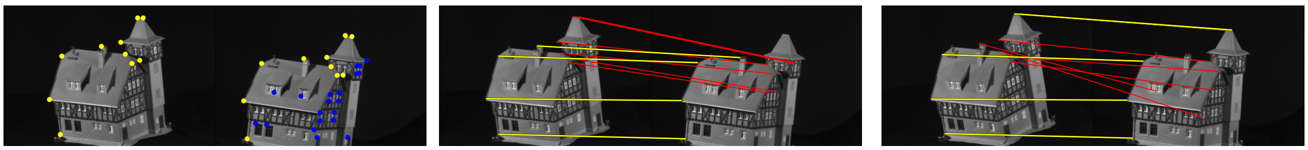
line" denotes the distance of the frames in the sequence and thus correlates well with the difficulty to match the corresponding frames.

We matched all possible image pairs with "baseline" of $10, 20, 30, \ldots, 100$ frames and computed the average matching accuracy for each algorithm. The algorithms are evaluated in three settings. In the first experiment, we match 30 points to 30 points. Then we make the problem significantly harder by randomly removing points from one image motivated by a scenario where one has background clutter in an image and thus not all points can be matched. This results in two matching experiments, namely 10 points to 30 points, and 20 points to 30 points. For the choice of σ_s in the affinity tensor for second order methods, we follow [6, 29] by setting $\sigma_s = 2500$.

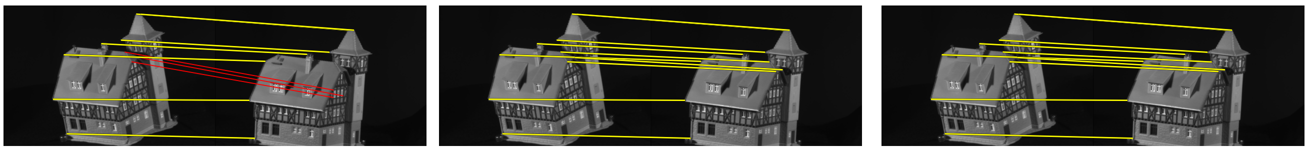
The experimental results are shown in Figure 3. While most algorithms perform rather well on the 30 to 30 task, our methods perform significantly better than all other methods in the more difficult tasks, thus showing as for the synthetic datasets that our methods are quite robust to different kind of noise in the matching problem.

5.3. Car Motorbike Dataset

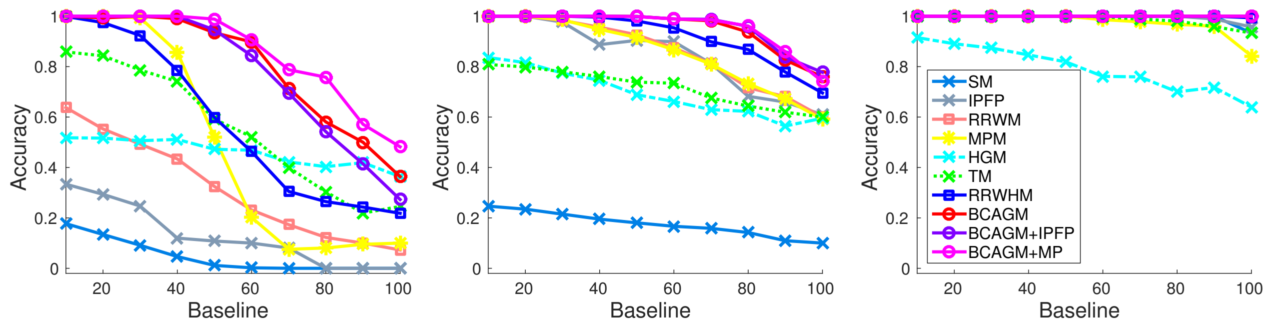
In this experiment, we compare our algorithms with other third order approaches on the car and motorbike dataset introduced in [18]. The dataset contains 30 resp. 20 pairs of car resp. motorbike images. Each image contains a number of inliers and outliers from the cluttered background. Ground-truth correspondences are given for the inlier points in both images. Figures 4 and 5 show some examples of matching results. To test the algorithms against noise, we kept all the inlier points in both images and randomly added a number of outliers from the cluttered background to the second image (between 0 to 15). For



(a) An input pair: 10 pts vs 30 pts, $baseline = 50$ (b) MPM 4/10 (15.6) (c) TM 5/10 (18.4)

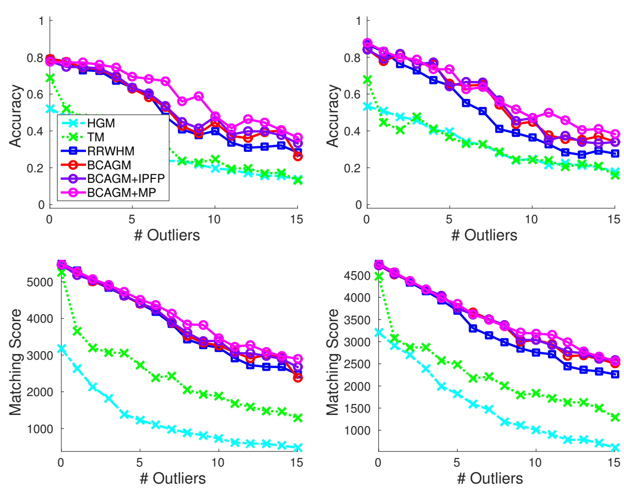


(d) RRWHM 7/10 (26.4) (e) BCAGM 10/10 (43.1) (f) BCAGM+MP 10/10 (43.1)



(g) 10 pts vs 30 pts (h) 20 pts vs 30 pts (i) 30 pts vs 30 pts

Figure 3: CMU house dataset: the first two rows show the matching output of several algorithms on an example pair of images where the baseline is 50. The yellow/red lines indicate correct/incorrect matches. The third row shows the average performance of algorithms with different number of points in the first image. (Best viewed in color.)



(a) Car (b) Motorbike

Figure 6: Evaluation of higher order GM algorithms on the car and motorbike dataset. (Best viewed in color.)

each number of outliers, we test all algorithms on all the image pairs of each dataset and report the average performance in Figure 6. It shows that our methods achieve better overall performance than other third order approaches in both datasets. However, we expect that the performance can be significantly improved if also other features are integrated rather than just relying on geometric information [28, 29], or graph learning methods can be also be integrated [5, 3, 18, 22, 24].

6. Conclusion and Outlook

We have presented a new optimization framework for higher order graph matching. Compared to previous methods for higher order matching we can guarantee monotonic ascent for our algorithms in the matching score on the set of assignment matrices. Our new algorithms achieve superior performance in terms of objective but also yield competitive or significantly better matching accuracy. This is particularly true for large number of outliers and other forms of noise. Moreover, both algorithms are also competitive in terms of runtime compared to other methods.

An interesting line of future research is the use of globally optimal solutions of relaxations of the hypergraph matching problem. We computed via the method of [12] the maximal singular vectors of the fourth order tensor and used these as initialization of Algorithm 1. This lead to an improvement in matching score of up to 2% and in accuracy of up to 3%. We will explore this further in future work.

Acknowledgment

The authors acknowledge support by the ERC starting grant NOLEPRO.

References

[1] E. Boros and P. L. Hammer. Pseudo-boolean optimization. *Discrete Applied Mathematics*, 123:155–225, 2002. 2

[2] R. E. Burkhard, M. Dell'Amico, and S. Martello. *Assignment problems*. SIAM, Philadelphia, 2012. 4

[3] T. S. Caetano, J. J. McAuley, L. Cheng, Q. V. Le, and A. J. Smola. Learning graph matching. *PAMI*, 31:1048–1058, 2009. 8

[4] M. Chertok and Y. Keller. Efficient high order matching. *PAMI*, 32:2205–2215, 2010. 1, 2

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

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

[7] M. Cho and K. M. Lee. Progressive graph matching: Making a move of graphs via probabilistic voting. In *CVPR*, 2012. 1

[8] M. Cho, J. Sun, O. Duchenne, and J. Ponce. Finding matches in a haystack: A max-pooling strategy for graph matching in the presence of outliers. In *CVPR*, 2014. 1, 5, 6

[9] T. Cour, P. Srinivasan, and J. Shi. Balanced graph matching. In *NIPS*, 2007. 1

[10] O. Duchenne, F. Bach, I. Kweon, and J. Ponce. A tensor-based algorithm for high-order graph matching. *PAMI*, 33:2383–2395, 2011. 1, 5, 6

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

[12] A. Gautier and M. Hein. Tensor norm and maximal singular vectors of non-negative tensors - a perron-frobenius theorem, a collatz-wielandt characterization and a generalized power method. *arXiv:1503.01273*, 2015. 8

[13] V. Kolmogorov and C. Rother. Minimizing nonsubmodular functions with graph cuts-a review. *PAMI*, 29:1274–1279, 2007. 2

[14] H. W. Kuhn. The hungarian method for the assignment problem. *Naval Research Logistics Quarterly*, 2:83–97, 1955. 4

[15] J. Lee, M. Cho, and K. M. Lee. A graph matching algorithm using data-driven markov chain monte carlo sampling. In *ICPR*, 2010. 1, 2

[16] J. Lee, M. Cho, and K. M. Lee. Hyper-graph matching via reweighted random walks. In *CVPR*, 2011. 2, 5, 6

[17] M. Leordeanu and M. Hebert. A spectral technique for correspondence problems using pairwise constraints. In *ICCV*, 2005. 1, 5

[18] M. Leordeanu and M. Hebert. Unsupervised learning for graph matching. In *CVPR*, 2009. 7, 8

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

[20] D. G. Lowe. Object recognition from local scale-invariant features. In *ICCV*, 1999. 1

[21] J. Maciel and J. P. Costeira. A global solution to sparse correspondence problems. *PAMI*, 25:187–199, 2003. 3

[22] D. Pachauri, M. Collins, V. Singh, and R. Kondor. Incorporating domain knowledge in matching problems via harmonic analysis. In *ICML*, 2012. 8

[23] A. Sharma, R. Horaud, J. Cech, and E. Boyer. Topologically-robust 3d shape matching based on diffusion geometry and seed growing. In *CVPR*, 2011. 1

[24] L. Torresani, V. Kolmogorov, and C. Rother. Feature correspondence via graph matching: Models and global optimization. In *ECCV*, 2008. 1, 8

[25] M. Zaslavskiy, F. Bach, and J. Vert. A path following algorithm for the graph matching problem. *PAMI*, 31:2227–2242, 2009. 1, 3

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

[27] Y. Zeng, C. Wang, Y. Wang, X. Gu, D. Samaras, and N. Paragios. Dense non-rigid surface registration using high-order graph matching. In *CVPR*, 2010. 1, 2

[28] F. Zhou and F. De la Torre. Factorized graph matching. In *CVPR*, 2012. 3, 8

[29] F. Zhou and F. De la Torre. Deformable graph matching. In *CVPR*, 2013. 1, 3, 6, 7, 8