Deformable Graph Matching

Feng Zhou **Fernando De la Torre**
Robotics Institute, Carnegie Mellon University, Pittsburgh, PA 15213
http://www.f-zhou.com ftorre@cs.cmu.edu

Abstract

Graph matching (GM) is a fundamental problem in computer science, and it has been successfully applied to many problems in computer vision. Although widely used, existing GM algorithms cannot incorporate global consistence among nodes, which is a natural constraint in computer vision problems. This paper proposes deformable graph matching (DGM), an extension of GM for matching graphs subject to global rigid and non-rigid geometric constraints. The key idea of this work is a new factorization of the pair-wise affinity matrix. This factorization decouples the affinity matrix into the local structure of each graph and the pair-wise affinity edges. Besides the ability to incorporate global geometric transformations, this factorization offers three more benefits. First, there is no need to compute the costly (in space and time) pair-wise affinity matrix. Second, it provides a unified view of many GM methods and extends the standard iterative closest point algorithm. Third, it allows to use the path-following optimization algorithm that leads to improved optimization strategies and matching performance. Experimental results on synthetic and real databases illustrate how DGM outperforms state-of-the-art algorithms for GM. The code is available at `http://humansensing.cs.cmu.edu/fgm`.

1. Introduction

Graph matching (GM) has been widely applied in computer vision to solve a variety of problems such as object categorization [10], feature tracking [13, 17], symmetry analysis [12], kernelized sorting [20] and action recognition [3]. From an optimization view-point, the GM problem is typically formulated as a quadratic assignment problem (QAP) [18]. Unlike the linear assignment problem, which can be efficiently solved with the Hungarian algorithm [4], the QAP is known to be NP-hard and exact optimal algorithms using variations of branch-and-bound [22] are only practical for very small graphs (*e.g.*, 30 nodes). Therefore, the main body of research in GM has focused on devising more accurate and faster algorithms to approximate it.

Although extensive research has been done on GM for decades, there are still two main challenges: (1) Many matching problems in computer vision naturally require global constraints among nodes in the graph. For instance, given two sets of coplanar points in two images, the matching between points should be constrained by an affine transformation (under orthographic projection). Similarly, when matching the deformations of non-rigid objects between two consecutive images that deformation is typically smooth in space and time. Existing GM algorithms do not constrain the nodes of both graphs to a given geometric transformation (*e.g.*, similarity, affine or non-rigid). (2) Optimizing GM is still difficult because the objective function is in general non-convex and the constraints are combinatorial. While there are a number of papers [6, 8, 11, 14, 24, 26, 15] addressing the second issue, the first has been rarely explored. This paper proposes DGM, an extension of GM that solves the first problem, and improves upon the second issue.

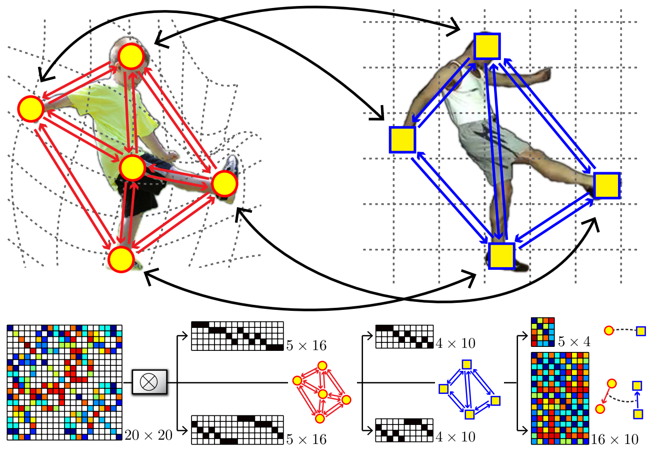

Figure 1. Matching two human poses with 5 and 4 features using DGM. DGM simultaneously estimates the correspondence and a smooth non-rigid transformation between nodes. DGM is able to factorize the 20×20 pair-wise affinity matrix as a Kronecker product of six smaller matrices. The first two groups of matrices of size 5×16 and 4×10 encode the structure of each of the graphs (i.e., adjacency matrix). The last two matrices encode the affinities for nodes (5×4) and edges (16×10).

In order to incorporate global transformations, the key idea of our method is to factorize the pairwise affinity matrix into matrices that preserve the local structure of each graph and matrices that encode the similarity between nodes and edges. This factorization is general and can be applied to both directed and undirected graphs. Consider the two graphs shown in Fig. 1 as an example. Using the factorization, we are able to factorize the large 20-by-20 pair-wise affinity matrix into six smaller matrices. Because we have decoupled the local structure for the nodes in each graph, it is easy to add global geometric constraints. Moreover, using this factorization has three additional benefits for GM. First, there is no need to compute the costly (in space and time) pair-wise affinity matrix. Second, it provides a unified view of many GM methods, which allows to understand the commonalities and differences between them. It also connects GM methods with the classical iterative closest point (ICP) algorithm, and provides a pair-wise generalization of ICP. Third, it allows the use of path-following optimization algorithms in general GM problems that leads to improved optimization strategies and matching performance. We illustrate the benefits of DGM in synthetic and real matching experiments on standard databases.

2. Previous works

2.1. Graph matching (GM)

We denote (see notation[1]) a graph with n nodes and m directed edges as a 4-tuple $\mathcal{G} = \{\mathbf{P}, \mathbf{Q}, \mathbf{G}, \mathbf{H}\}$. The features for nodes and edges are specified by $\mathbf{P} = [\mathbf{p}_1, \cdots, \mathbf{p}_n] \in \mathbb{R}^{d_p \times n}$ and $\mathbf{Q} = [\mathbf{q}_1, \cdots, \mathbf{q}_m] \in \mathbb{R}^{d_q \times m}$ respectively. The topology of the graph is encoded by two node-edge incidence matrices $\mathbf{G}, \mathbf{H} \in \{0,1\}^{n \times m}$, where $g_{ic} = h_{jc} = 1$ if the c^{th} edge starts from the i^{th} node and ends at the j^{th} node. For instance, Fig. 2a illustrates two synthetic graphs, whose edge connection between nodes is encoded by the corresponding matrices shown in Fig. 2b-c. A similar representation of graph was adopted in [29]. However, the work in [29] is only valid for undirected graphs. Our representation is more general and valid for directed and undirected graphs. Directed graphs typically occur when the features are asymmetrical such as the angle between an edge and the horizontal line. Our model incorporates directed graphs by encoding the starting and ending node in $\mathbf{G}$ and $\mathbf{H}$ respectively.

Given two graphs, $\mathcal{G}_1 = \{\mathbf{P}_1, \mathbf{Q}_1, \mathbf{G}_1, \mathbf{H}_1\}$ and $\mathcal{G}_2 = \{\mathbf{P}_2, \mathbf{Q}_2, \mathbf{G}_2, \mathbf{H}_2\}$, we compute two affinity matrices, $\mathbf{K}_p \in \mathbb{R}^{n_1 \times n_2}$ and $\mathbf{K}_q \in \mathbb{R}^{m_1 \times m_2}$, to measure the similarity of each node and edge pair respectively. More specifically, $\kappa^p_{i_1 i_2} = \phi_p(\mathbf{p}^1_{i_1}, \mathbf{p}^2_{i_2})$ measures the similarity between the i_1^{th} node of $\mathcal{G}_1$ and the i_2^{th} node of $\mathcal{G}_2$, and $\kappa^q_{c_1 c_2} = \phi_q(\mathbf{q}^1_{c_1}, \mathbf{q}^2_{c_2})$ measures the similarity between the c_1^{th} edge of $\mathcal{G}_1$ and the c_2^{th} edge of $\mathcal{G}_2$. For instance, Fig. 2d illustrates an example pair of $\mathbf{K}_p$ and $\mathbf{K}_q$ for the two synthetic graphs.

It is more convenient to encode the node and edge affinities in a global affinity matrix $\mathbf{K} \in \mathbb{R}^{n_1 n_2 \times n_1 n_2}$, whose element is computed as follows:

$$\kappa_{i_1 i_2 j_1 j_2} = \begin{cases} \kappa^p_{i_1 i_2}, & \text{if} \quad i_1 = j_1 \text{ and } i_2 = j_2, \\ \kappa^q_{c_1 c_2}, & \text{if} \quad i_1 \neq j_1 \text{ and } i_2 \neq j_2 \text{ and} \\ & g^1_{i_1 c_1} h^1_{j_1 c_1} g^2_{i_2 c_2} h^2_{j_2 c_2} = 1, \\ 0, & \text{otherwise.} \end{cases}$$

Given two graphs and $\mathbf{K}$, the problem of GM consists in finding the optimal correspondence $\mathbf{X}$ between nodes, such that the following score is maximized,

$$\max_{\mathbf{X}} \quad J_{gm}(\mathbf{X}) = \operatorname{vec}(\mathbf{X})^T \mathbf{K} \operatorname{vec}(\mathbf{X}), \quad \text{s.t. } \mathbf{X} \in \Pi. \tag{1}$$

where $\mathbf{X} \in \Pi$ is usually constrained to be a one-to-one mapping, *i.e.*, Π is the set of partial permutation matrices:

$$\Pi = \{\mathbf{X} | \mathbf{X} \in \{0,1\}^{n_1 \times n_2}, \mathbf{X}\mathbf{1}_{n_2} \leq \mathbf{1}_{n_1}, \mathbf{X}^T \mathbf{1}_{n_1} = \mathbf{1}_{n_2}\}.$$

The inequality in the above definition is used for the case when the graphs are of different sizes. Without loss of generality, we assume $n_1 \geq n_2$ throughout the rest of the paper.

Advances in GM: GM can be formulated as a quadratic assignment problem [18] and optimizing Eq. 1 is known to be NP-hard. Therefore, major research in GM has focused on finding better optimization strategies. Broadly speaking, most relaxations of the permutation constraints fall into two categories: spectral and doubly-stochastic.

The first group of methods approximates the permutation matrix with an orthogonal one, *i.e.*, $\mathbf{X}^T \mathbf{X} = \mathbf{I}$. Under the orthogonal constraint, optimizing $J_{gm}(\mathbf{X})$ can be solved in closed-form as an eigen-value problem [23, 21]. However, these methods can only work for a restricted case, where $\mathbf{K} = \mathbf{K}_1 \otimes \mathbf{K}_2$ is composed by two weighted adjacency matrices, $\mathbf{K}_1 \in \mathbb{R}^{n_1 \times n_1}$ and $\mathbf{K}_2 \in \mathbb{R}^{n_2 \times n_2}$, defined on each graph respectively. In order to handle more complex problems in computer vision, Leordeanu and Hebert [14] proposed to optimize Eq. 1 by relaxing the constraints on $\mathbf{X}$ to be of unit length, *i.e.*, $\|\operatorname{vec}(\mathbf{X})\|_2^2 = 1$. In this case, the optimal $\mathbf{X}$ can be simply computed as the leading eigenvector of $\mathbf{K}$. Cour *et al.* [8] incorporated additional affine constraints to solve a more general spectral problem.

The second group of methods relaxes $\mathbf{X} \in \mathcal{D}$ to be a doubly stochastic matrix, the convex hull of $\mathbf{X} \in \Pi$,

$$\mathcal{D} = \{\mathbf{X} \in \mathbb{R}^{n_1 \times n_2} | \mathbf{X}\mathbf{1}_{n_2} \leq \mathbf{1}_{n_1}, \mathbf{X}^T \mathbf{1}_{n_1} = \mathbf{1}_{n_2}, \mathbf{X} \geq \mathbf{0}\}.$$

[1]Bold capital letters denote a matrix $\mathbf{X}$, bold lower-case letters a column vector $\mathbf{x}$. $\mathbf{x}_i$ represents the i^{th} column of the matrix $\mathbf{X}$. x_{ij} denotes the scalar in the i^{th} row and j^{th} column of the matrix $\mathbf{X}$. All non-bold letters represent scalars. $\mathbf{1}_{m \times n}, \mathbf{0}_{m \times n} \in \mathbb{R}^{m \times n}$ are matrices of ones and zeros. $\mathbf{I}_n \in \mathbb{R}^{n \times n}$ is an identity matrix. $\|\mathbf{x}\|_p = \sqrt[p]{\sum |x_i|^p}$ denotes the p-norm. $|\mathbf{X}|$ represents the determinant of the square matrix $\mathbf{X}$. $\operatorname{vec}(\mathbf{X})$ denotes the vectorization of matrix $\mathbf{X}$. $\operatorname{diag}(\mathbf{x})$ is a diagonal matrix whose diagonal elements are $\mathbf{x}$. $\mathbf{X} \circ \mathbf{Y}$ and $\mathbf{X} \otimes \mathbf{Y}$ are the Hadamard and Kronecker products of matrices.

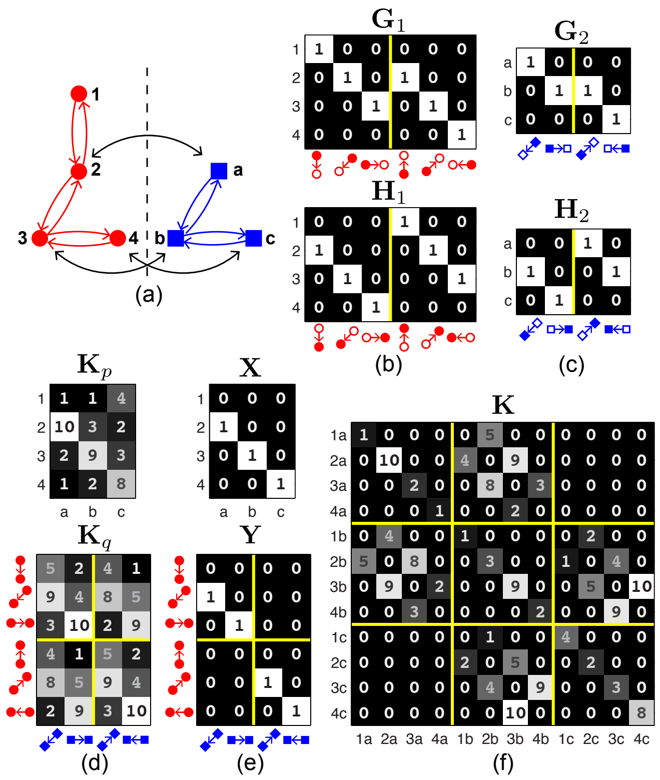

Figure 2. An example GM problem. (a) Two synthetic graphs. (b) The 1st graph's incidence matrices $\mathbf{G}_1$ and $\mathbf{H}_1$, where the non-zero elements in each column of $\mathbf{G}_1$ and $\mathbf{H}_1$ indicate the starting and ending nodes in the corresponding directed edge, respectively. (c) The 2nd graph's incidence matrices $\mathbf{G}_2$ and $\mathbf{H}_2$. (d) The node affinity matrix $\mathbf{K}_p$ and the edge affinity matrix $\mathbf{K}_q$ between graphs. (e) The node correspondence matrix $\mathbf{X}$ and the edge correspondence matrix $\mathbf{Y}$. (f) The global affinity matrix $\mathbf{K}$.

Under this constraint, optimizing Eq. 1 can be treated as a non-convex quadratic programming problem and various strategies have been proposed to find a local optima. For instance, Gold and Rangarajan [11] proposed the graduated assignment algorithm to iteratively solve a series of linear approximations of the cost function using Taylor expansions. Leordeanu *et al*. [15] proposed an integer projection algorithm to optimize the objective function in an integer domain. More recently, Zhou and De la Torre [29] used a path-following algorithm [25]. In addition to the optimization-based work, probabilistic frameworks [6, 26] were shown to be useful for interpreting and solving GM.

Our work is closely related to recent higher-order tensor factorization [5, 9, 26]. It has been noticed that $\mathbf{K}$ encoding the pairwise geometry is susceptible to scale and rotation differences between sets of points. In order to make GM invariant to rigid deformations, [5, 9, 26] extended the pairwise matrix $\mathbf{K}$ embedded into a tensor that encodes high-order geometrical relations. However, a small increment in the order of relations leads to a combinatorial explosion of the amount data needed to support the algorithm. Therefore, most of high-order GM methods can only work on very sparse graphs with no more than 3-order features. On the other hand, it is unclear on how to extend high-order methods to incorporate non-rigid deformations.

2.2. Iterative closest point (ICP)

Given two sets of points, $\mathbf{P}_1 = [\mathbf{p}_1^1, \cdots, \mathbf{p}_{n_1}^1] \in \mathbb{R}^{d \times n_1}$ and $\mathbf{P}_2 = [\mathbf{p}_1^2, \cdots, \mathbf{p}_{n_2}^2] \in \mathbb{R}^{d \times n_2}$, iterative closest point (ICP) algorithms (*e.g.*, [2, 27]) aim to find the correspondence and the geometric transformation between points such that the sum of distances is minimized:

$$\min_{\mathbf{X},\mathcal{T}} \quad J_{icp}(\mathbf{X},\mathcal{T}) = \sum_{i_1 i_2} x_{i_1 i_2} \|\mathbf{p}_{i_1}^1 - \tau(\mathbf{p}_{i_2}^2)\|_2^2 + \psi(\mathcal{T}), \quad (2)$$

$$\text{s.t.} \quad \mathbf{X} \in \Pi, \mathcal{T} \in \Psi,$$

where $\mathbf{X} \in \{0,1\}^{n_1 \times n_2}$ denotes the correspondence between points. Depending on the problem, $\mathbf{X}$ denotes either a one-to-one or many-to-one mapping. In this paper, we consider a one-to-one mapping between points and $\mathbf{X}$ is thus constrained to be a permutation matrix *i.e.*, $\mathbf{X} \in \Pi$. $\tau(\cdot) : \mathbb{R}^d \to \mathbb{R}^d$ denotes a geometric transformation and it is parameterized by $\mathcal{T}$. For instance, if $\tau(\cdot)$ is a 2-D similarity transformation, then $\tau(\mathbf{p}) = s\mathbf{R}\mathbf{p} + \mathbf{t}$ and $\mathcal{T} = \{s, \mathbf{R}, \mathbf{t}\}$, where $s \in \mathbb{R}$ is the scaling factor, $\mathbf{R} \in \mathbb{R}^{2\times 2}$ is the rotation matrix and $\mathbf{t} \in \mathbb{R}^2$ is the translation vector. In addition, the rotation matrix has to satisfy the constraint, $\Psi = \{\mathbf{R} | \mathbf{R}^T\mathbf{R} = \mathbf{I}_2, |\mathbf{R}| = 1\}$. If $\tau(\cdot)$ is chosen to be a non-rigid transformation, a penalization cost $\psi(\mathcal{T})$ is needed to further constrain the parameter. See [28] for a more comprehensive review of various transformations adopted in ICP.

To connect ICP with GM methods, we re-write Eq. 2 as:

$$J_{icp}(\mathbf{X},\mathcal{T}) = -\operatorname{tr}\left(\mathbf{K}_p(\mathcal{T})^T\mathbf{X}\right) + \psi(\mathcal{T}). \quad (3)$$

where $\mathbf{K}_p(\mathcal{T}) \in \mathbb{R}^{n_1 \times n_2}$ encodes the Euclidean distances between nodes, that is, $\kappa_{i_1 i_2}^p(\mathcal{T}) = -\|\mathbf{p}_{i_1}^1 - \tau(\mathbf{p}_{i_2}^2)\|_2^2$. Eq. 3 reveals two commonalities of ICP algorithms: (1) The optimization over $\mathbf{X}$ given the transformation $\mathcal{T}$ can be cast as a linear matching problem, which can be efficiently optimized by the Hungarian algorithm (if $\mathbf{X}$ is a one-to-one mapping) or the winner-take-all manner (if $\mathbf{X}$ is a many-to-one mapping). (2) In general, the joint optimization over $\mathbf{X}$ and $\mathcal{T}$ is non-convex, and no closed-form solution is known. Typically, some sort of alternated minimization (*e.g.*, EM, coordinate-descent) is needed to find a local optima.

3. Factorized graph matching

This section derives a new factorization of the pair-wise affinity matrix $\mathbf{K}$. As we will see in the following sections, this factorization allows the unification of GM methods, adding geometric constraints to GM and elaborating better optimization strategies.

To illustrate the intuition behind the factorization, let us consider the synthetic graph shown in Fig. 2. Notice that $\mathbf{K} \in \mathbb{R}^{n_1 n_2 \times n_1 n_2}$ is composed by two types of affinities: the node affinity ($\mathbf{K}_p$) on its diagonal and the pairwise edge affinity ($\mathbf{K}_q$) on its off-diagonals. Let's ignore the diagonal first. Then, $\mathbf{K}$ is a sparse block matrix with three unique structures: (1) $\mathbf{K}$ is composed by n_2-by-n_2 smaller blocks $\mathbf{K}_{ij} \in \mathbb{R}^{n_1 \times n_1}$. (2) Some of the $\mathbf{K}_{ij}$s are empty if there is no edge connecting the i^{th} and j^{th} nodes of $\mathcal{G}_2$. In another word, these empty blocks can be indexed by $\mathbf{G}_2\mathbf{H}_2^T$, *i.e.*, $\mathbf{K}_{ij} = \mathbf{0}_{n_1 \times n_1}$ if $[\mathbf{G}_2\mathbf{H}_2^T]_{ij} = 0$. (3) For the non-empty blocks, $\mathbf{K}_{ij}$ can be computed in a closed form as $\mathbf{G}_1 \operatorname{diag}(\mathbf{k}_c^q)\mathbf{H}_1^T$, where c is the index of the edge connecting the i^{th} and j^{th} nodes of $\mathcal{G}_2$, *i.e.*, $g_{ic}^2 = h_{jc}^2 = 1$. Based on these three observations, and after some linear algebra, it can be shown that $\mathbf{K}$ can be exactly factorized as:

$$\mathbf{K} = \operatorname{diag}(\operatorname{vec}(\mathbf{K}_p)) + (\mathbf{G}_2 \otimes \mathbf{G}_1) \operatorname{diag}(\operatorname{vec}(\mathbf{K}_q))(\mathbf{H}_2 \otimes \mathbf{H}_1)^T. \tag{4}$$

This factorization decouples the graph structure ($\mathbf{G}_1$, $\mathbf{H}_1$, $\mathbf{G}_2$ and $\mathbf{H}_2$) from the similarity ($\mathbf{K}_p$ and $\mathbf{K}_q$). It is important to notice that our factorization significantly differs from the one proposed in [29] in two aspects: (1) Eq. 4 is proposed for more general graphs composed by directed edges while [29] can be only applied for simpler graphs composed by undirected edges; (2) Unlike a joint factorization proposed in [29], Eq. 4 separates $\mathbf{K}_p$ and $\mathbf{K}_q$ in the factorization in two independent terms. This separation enables us to introduce geometric transformations on $\mathbf{K}_p$ and $\mathbf{K}_q$ in GM.

Eq. 4 is the key contribution of this work. Previous work in GM computed the computationally expensive (in space and time) $\mathbf{K}$. On the contrary, Eq. 4 offers an alternative framework by replacing $\mathbf{K}$ with six smaller matrices. For instance, plugging Eq. 4 into Eq. 1 leads to an equivalent objective function:

$$J_{gm}(\mathbf{X}) = \operatorname{tr}\left(\mathbf{K}_p^T\mathbf{X}\right) + \operatorname{tr}\left(\mathbf{K}_q^T\mathbf{Y}\right), \tag{5}$$

where $\mathbf{Y} = (\mathbf{G}_1^T\mathbf{X}\mathbf{G}_2 \circ \mathbf{H}_1^T\mathbf{X}\mathbf{H}_2) \in \{0,1\}^{m_1 \times m_2}$ is an auxiliary variable that encodes the correspondence between edges, *i.e.*, $y_{c_1c_2} = 1$ if c_1^{th} edge in $\mathcal{G}_1$ is matched to the c_2^{th} edge in $\mathcal{G}_2$. For instance, Fig. 2e illustrates the node and edge correspondence matrices for the matching defined in Fig. 2a. In addition, Eq. 5 reveals a connection between GM and ICP. In particular, maximizing the first term of Eq. 5 is equivalent to ICP (Eq. 3).

Observe that $\mathbf{K}_q$ can always be factorized (*e.g.*, SVD) as $\mathbf{K}_q = \mathbf{U}\mathbf{V}^T$, where $\mathbf{U} \in \mathbb{R}^{m_1 \times c}$ and $\mathbf{V} \in \mathbb{R}^{m_2 \times c}$. Taking advantage of the low-rank structure of $\mathbf{K}_q$, Eq. 5 can be further re-formulated as follows:

$$J_{gm}(\mathbf{X}) = \operatorname{tr}\left(\mathbf{K}_p^T\mathbf{X}\right) + \sum_{i=1}^{c} \operatorname{tr}\left(\mathbf{A}_i^1\mathbf{X}\mathbf{A}_i^2\mathbf{X}^T\right), \tag{6}$$

where $\mathbf{A}_i^1 = \mathbf{G}_1 \operatorname{diag}(\mathbf{u}_i)\mathbf{H}_1^T$ and $\mathbf{A}_i^2 = \mathbf{G}_2 \operatorname{diag}(\mathbf{v}_i)\mathbf{H}_2^T$.

The factorization (Eq. 4) and the two equivalent objectives (Eq. 5 and Eq. 6) allow to unify GM methods. For instance, Eq. 6 reveals the connection between two types of GM problems, the less general one [1, 23, 25] that maximizes $\operatorname{tr}(\mathbf{A}_1\mathbf{X}\mathbf{A}_2\mathbf{X}^T)$, versus the more general one [6, 8, 11, 14, 15, 24, 26] that maximizes $\operatorname{vec}(\mathbf{X})^T\mathbf{K}\operatorname{vec}(\mathbf{X})$. In particular, maximization of $\operatorname{vec}(\mathbf{X})^T\mathbf{K}\operatorname{vec}(\mathbf{X})$ is equivalently to the maximization of the sum of c traces $\operatorname{tr}(\mathbf{A}_i^1\mathbf{X}\mathbf{A}_i^2\mathbf{X}^T)$, where $\mathbf{A}_i^1$ and $\mathbf{A}_i^2$ can be interpreted as adjacency matrices.

3.1. A path-following algorithm

Given Eq. 6 we can optimize GM with the path-following algorithm proposed for the simplified GM problem ($\operatorname{tr}(\mathbf{A}_1\mathbf{X}\mathbf{A}_2\mathbf{X}^T)$) [1, 23, 25]. More specifically, we solved a series of concave-convex problems:

$$\max_{\mathbf{X}\in\mathcal{D}} \quad J_\alpha(\mathbf{X}) = (1-\alpha)J_{vex}(\mathbf{X}) + \alpha J_{cav}(\mathbf{X}), \tag{7}$$

where $\alpha \in [0,1]$ is a trade-off between the convex relaxation $J_{vex}(\mathbf{X})$ and the concave relaxation $J_{cav}(\mathbf{X})$ of the original objective $J_{gm}(\mathbf{X})$.

To employ the path-following algorithm, we need to find proper convex and concave relaxations of $J_{gm}(\mathbf{X})$. Fortunately, the factorization (Eq. 4) offers a principled way for deriving them:

$$\begin{aligned} J_{vex}(\mathbf{X}) &= J_{gm}(\mathbf{X}) - \frac{1}{2}J_{con}(\mathbf{X}) \\ &= \operatorname{tr}\left(\mathbf{K}_p^T\mathbf{X}\right) - \frac{1}{2}\sum_{i=1}^{c}\|\mathbf{X}^T\mathbf{A}_i^1 - \mathbf{A}_i^2\mathbf{X}^T\|_F^2, \qquad (8) \\ J_{cav}(\mathbf{X}) &= J_{gm}(\mathbf{X}) + \frac{1}{2}J_{con}(\mathbf{X}) \\ &= \operatorname{tr}\left(\mathbf{K}_p^T\mathbf{X}\right) + \frac{1}{2}\sum_{i=1}^{c}\|\mathbf{X}^T\mathbf{A}_i^1 + \mathbf{A}_i^2\mathbf{X}^T\|_F^2. \qquad (9) \\ J_{con}(\mathbf{X}) &= \sum_i \operatorname{tr}\left(\mathbf{A}_i^{1T}\mathbf{X}\mathbf{X}^T\mathbf{A}_i^1\right) + \operatorname{tr}\left(\mathbf{A}_i^2\mathbf{X}^T\mathbf{X}\mathbf{A}_i^{2T}\right), \end{aligned}$$

where $J_{con}(\mathbf{X}) = \gamma$ is a constant with respect to a permutation or orthogonal matrix $\mathbf{X}$ because $\mathbf{X}\mathbf{X}^T = \mathbf{X}^T\mathbf{X} = \mathbf{I}$. It is worth to point out that it not clear how to derive the relaxations ($J_{vex}(\mathbf{X})$ and $J_{cav}(\mathbf{X})$) and apply the path-following algorithm without the propose factorization of $\mathbf{K}$. Please refer [28] for details about the path-following optimization.

The advantages of the path-following algorithm over conventional GM algorithms are three-fold: (1) The algorithm starts with a convex problem ($\alpha = 0$) and it is guaranteed to find a globally optimal solution. (2) The algorithm ends at a concave problem ($\alpha = 1$) and the local optimal solution is always discrete; (3) By smoothly increasing α from $\alpha = 0$ to $\alpha = 1$, the path-following algorithm is more likely to find better local optima than gradient-based method.

4. Deformable graph matching (DGM)

This section describes how to incorporate rigid and non-rigid transformation into the GM framework. Moreover, we illustrate how the factorization can be used into the DGM to derive an improved optimization strategy.

4.1. Objective function

To simplify the discussion and to be consistent with ICP, we compute the node feature of each graph $\mathcal{G} = \{\mathbf{P}, \mathbf{Q}, \mathbf{G}, \mathbf{H}\}$ simply as the node coordinates, $\mathbf{P} = [\mathbf{p}_1, \cdots, \mathbf{p}_n] \in \mathbb{R}^{d \times n}$. Similarly, the edge features $\mathbf{Q} = [\mathbf{q}_1, \cdots, \mathbf{q}_m] \in \mathbb{R}^{d \times m}$ are computed as the coordinate difference between the connected nodes, *i.e.*, $\mathbf{q}_c = \mathbf{p}_i - \mathbf{p}_j$, where $g_{ic} = h_{jc} = 1$. In this case, the edge feature can be conveniently computed in a matrix form as, $\mathbf{Q} = \mathbf{P}(\mathbf{G} - \mathbf{H})$.

Suppose that we are given two graphs, $\mathcal{G}_1 = \{\mathbf{P}_1, \mathbf{Q}_1, \mathbf{G}_1, \mathbf{H}_1\}$ and $\mathcal{G}_2 = \{\mathbf{P}_2, \mathbf{Q}_2, \mathbf{G}_2, \mathbf{H}_2\}$, and a geometrical transformation defined on points by $\tau(\cdot)$. Similar to ICP, we compute the node affinity $\mathbf{K}_p(\mathcal{T}) \in \mathbb{R}^{n_1 \times n_2}$ and the edge affinity $\mathbf{K}_q(\mathcal{T}) \in \mathbb{R}^{m_1 \times m_2}$ as a function of the Euclidean distance, *i.e.*:

$$\begin{aligned} \kappa^p_{i_1 i_2}(\mathcal{T}) &= -\|\mathbf{p}^1_{i_1} - \tau(\mathbf{p}^2_{i_2})\|_2^2, \\ \kappa^q_{c_1 c_2}(\mathcal{T}) &= \beta - \| \underbrace{(\mathbf{p}^1_{i_1} - \mathbf{p}^1_{j_1})}_{\mathbf{q}^1_{c_1}} - \underbrace{(\tau(\mathbf{p}^2_{i_2}) - \tau(\mathbf{p}^2_{j_2}))}_{\tau(\mathbf{q}^2_{c_2})} \|_2^2, \end{aligned} \quad (10)$$

where β is chosen to be reasonably large to ensure that the pairwise affinity is greater than zero.

Recall that the factorization (Eq. 4) reveals that the goal of GM (Eq. 5) is similar to ICP (Eq. 3). In order to make the GM more robust to geometric deformations, DGM aims to find the optimal correspondence $\mathbf{X}$ as well as the optimal transformation $\mathcal{T}$ such that the global consistency can be maximized:

$$\begin{aligned} \max_{\mathbf{X}, \mathcal{T}} \quad & J_{dgm}(\mathbf{X}, \mathcal{T}) = \operatorname{tr}\left(\mathbf{K}_p(\mathcal{T})^T \mathbf{X}\right) + \lambda \operatorname{tr}\left(\mathbf{K}_q(\mathcal{T})^T \mathbf{Y}\right) \\ & \qquad - \psi(\mathcal{T}), \quad (11) \\ \text{s.t.} \quad & \mathbf{X} \in \Pi, \mathcal{T} \in \Psi, \end{aligned}$$

where $\lambda \geq 0$ is used to balance between the importance of the node and edge consistency. Similar to ICP, $\psi(\mathcal{T})$ and Ψ are used to constrain the transformation parameter. Eq. 11 unifies GM and ICP. In particular, if $\lambda = 0$, solving DGM is equivalent to ICP. In other case when $\lambda > 0$ and $\mathcal{T}$ is known, solving DGM is identical to a GM problem.

Due to the non-convex nature of the objective, we optimize DGM by alternatively solving the correspondence ($\mathbf{X}$) and the transformation parameter ($\mathcal{T}$). The initialization is important for the performance of DGM. However, the way of choosing a good initialization is beyond the scope of this paper and we simply set the initial transformation as an identity one, *i.e.*, $\tau(\mathbf{p}) = \mathbf{p}$.

4.2. Optimization

Optimizing Eq. 11 will alternate between optimizing for the correspondence and the geometric transformation.

Optimization for the correspondence: Given the transformation $\mathcal{T}$, DGM is equivalent to a traditional GM problem. To find the node correspondence $\mathbf{X}$, we adopt the path-following algorithm by optimizing Eq. 7.

Optimization for the geometric transformation: Given the correspondence matrix $\mathbf{X}$, the optimization over the transformation parameter $\mathcal{T}$ is similar to ICP. The main difficulty lies in the fact that the transformation parameter $\mathcal{T}$ appears not only in the node affinity $\mathbf{K}_p(\mathcal{T})$, but also in the edge affinity $\mathbf{K}_q(\mathcal{T})$. After some linear algebra, however, it can be shown that for certain choices of transformations in 2-D (*e.g.*, similarity, affine, RBF non-rigid), the parameter can be computed in closed-form. For instance, let $\bar{\mathbf{P}}_1 = \mathbf{P}_1 - \bar{\mathbf{p}}_1 \mathbf{1}^T_{n_1} \in \mathbb{R}^{2 \times n_1}$ and $\bar{\mathbf{P}}_2 = \mathbf{P}_2 - \bar{\mathbf{p}}_2 \mathbf{1}^T_{n_2} \in \mathbb{R}^{2 \times n_2}$ be the centralized point sets, where $\bar{\mathbf{p}}_1 = \frac{\mathbf{P}_1 \mathbf{X} \mathbf{1}_{n_2}}{\mathbf{1}^T_{n_1} \mathbf{X} \mathbf{1}_{n_2}} \in \mathbb{R}^2$ and $\bar{\mathbf{p}}_2 = \frac{\mathbf{P}_2 \mathbf{X}^T \mathbf{1}_{n_1}}{\mathbf{1}^T_{n_1} \mathbf{X} \mathbf{1}_{n_2}} \in \mathbb{R}^2$ are the mean vectors of the two point sets respectively. Then the parameters for the 2-D similarity transformation could be computed as:

$$\mathbf{t} = \bar{\mathbf{p}}_1 - s\mathbf{R}\bar{\mathbf{p}}_2, \quad \mathbf{R} = \mathbf{U} \operatorname{diag}(1, \cdots, |\mathbf{U}\mathbf{V}^T|)\mathbf{V}^T,$$

$$s = \frac{\operatorname{tr}(\mathbf{\Sigma})}{\operatorname{tr}\left(\mathbf{1}_{n_1 \times 2}(\bar{\mathbf{P}}_2 \circ \bar{\mathbf{P}}_2)\mathbf{X}^T\right) + \lambda_q \operatorname{tr}\left(\mathbf{1}_{m_1 \times 2}(\mathbf{Q}_2 \circ \mathbf{Q}_2)\mathbf{Y}^T\right)},$$

where $\mathbf{U}\mathbf{\Sigma}\mathbf{V}^T = \bar{\mathbf{P}}_1 \mathbf{X} \bar{\mathbf{P}}_2^T + \lambda_q \mathbf{Q}_1 \mathbf{Y} \mathbf{Q}_2^T$ is computed by SVD. Please refer [28] for the derivation of the optimal affine and non-rigid transformations.

It is well known that the performance of ICP algorithms largely depends on the effectiveness of the initialization step. In the following example, we empirically illustrate how by adding additional pair-wise constrains, DGM is less sensitive to the initialization. Fig. 3a illustrates the problem of aligning two fish shapes under varying values for the initial rotation and scale parameters. As shown in Fig. 3b, ICP gets trapped into a local optima if the orientation gap is larger than $\frac{1}{3}\pi$ (the error should be 0). Similarly, DGM fails for large orientation gap after two iterations (the left column of Fig. 3c). However, as the number of iterations increases, DGM is able to match shapes with very large deformation in rotation and scales. After 24 iterations, DGM ultimately finds the optimal matching for all the initializations (the right column of Fig. 3c). This experiment shows that adding pairwise constraints can make the ICP algorithm more robust to the problem of local optima.

5. Experiments

This section reports experimental results on three benchmark datasets and compares FGM for Directed graphs (FGM-D) and DGM to several state-of-the-art methods for GM and ICP respectively. The first two experiments compare the path-following algorithm to other GM approaches

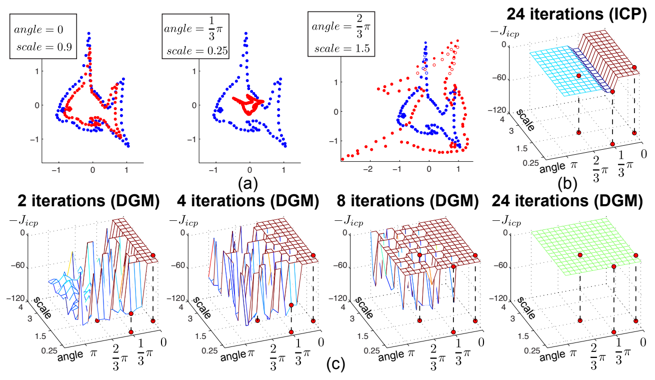

Figure 3. Comparison between ICP and DGM to align shapes for several initial values of rotation and scale parameters. (a) Examples of initializations. (b) Objective surfaces obtained by ICP for different initializations. (c) Objective surfaces obtained by DGM.

using undirected and directed edges. In the third experiment we add a known geometrical transformation between graphs and compare it with ICP algorithm on the problem of matching non-rigid shapes.

5.1. CMU house image dataset

The CMU house image dataset consists of 111 frames of a house, each of which has been manually labeled with 30 landmarks. We connected the landmarks via Delaunay triangulation. In this experiment, we focused on the simple case, where the edge is undirected and the edge feature is symmetric. In particular, the edge feature q_c was computed as the pairwise distance between the connected nodes. Given an image pair, the edge-affinity matrix $\mathbf{K}_q$ was computed by $k^q_{c_1 c_2} = \exp(-\frac{(q^1_{c_1} - q^2_{c_2})^2}{2500})$ and the node-affinity $\mathbf{K}_p$ was set to zero.

This experiment tested the performance of the path-following algorithm. We compared FGM-D against eight state-of-the-art algorithms: GA [11], SM [14], SMAC [8], IPFP [15] initialized with a uniform correspondence (IPFP-U) and spectral matching (IPFP-S), PM [26], RRWM [6] and FGM for Undirected graphs (FGM-U) [29]. We tested the performance of all methods as a function of the separation between frames. We matched all possible image pairs, spaced exactly by $0:10:90$ frames and computed the average matching accuracy and objective ratio ($\frac{J_{gm}(\mathbf{X}_{alg})}{J_{gm}(\mathbf{X}_{dgm})}$) per gap. Fig. 4a demonstrates an example pair of two frames.

We tested the performance of GM methods under two scenarios. In the first case (Fig. 4b) we used all 30 landmarks and in the second one (Fig. 4c) we matched subgraphs by randomly picking 25 landmarks. It can be observed that in both cases, FGM-U and FGM-D consistently achieved the best performance. The results demonstrate the advantages of the path-following algorithm over other state-of-the-art methods in solving general GM problems. In addition, it is interesting to notice that FGM-U and FGM-D performed similarly in both cases. This is because FGM-U can be considered as a special case of FGM-D when the graph only has undirected edges.

5.2. Car and motorbike image dataset

The car and motorbike image dataset was created in [16]. This dataset consists of 30 pairs of car images and 20 pairs of motorbike images. Each pair contains $30 \sim 60$ ground-truth correspondences. We computed for each node the feature, p_i, which is the orientation of the normal vector to the contour. We adopted the Delaunay triangulation to build the graph. In this experiment, we consider the most general graph where the edge is directed and the edge feature is asymmetrical. More specifically, each edge was represented by a couple of values, $\mathbf{q}_c = [d_c, \theta_c]^T$, where d_c is the pairwise distance between the connected nodes and θ_c is the angle between the edge and the horizontal line. Thus, for each pair of images, we computed the node affinity as $k^p_{ij} = \exp(-|p_i - p_j|)$ and the edge affinity as $k^q_{c_1 c_2} = \exp(-\frac{1}{2}|d_{c_1} - d_{c_2}| - \frac{1}{2}|\theta_{c_1} - \theta_{c_2}|)$. Fig. 5a and Fig. 5b demonstrate example pairs of car and motorbike images respectively. To test the performance against noise, we randomly selected $0 \sim 20$ outlier nodes from the background. Similarly, we compared FGM-D against eight state-of-the-art methods. However, we were unable to directly use FGM-U to match directed graphs. Therefore, we ran FGM-U on an approximated undirected graph, where for each pair of directed edges, we computed its new edge affinity as the average value of the original ones.

As observed in Fig. 5c-d, the proposed FGM-D consistently outperformed other methods in both datasets. As we show in the previous experiment, the path-following algorithm used by FGM-D provides a better optimization strategy than existing approaches. On the other hand, although FGM-U has a similar path-following strategy, it did not perform well because it is only applicable to undirected edges. Finally, it is important to remind the reader that without the factorization proposed in this work it is not possible to apply the path-following method to general graphs.

5.3. Fish and character shape dataset

The UCF shape dataset [7] has been widely used for comparing ICP algorithms. In our experiment, we used two different templates. The first one has 91 points sampled from the outer contour of a tropical fish. The second one consist of 105 points sampled from a Chinese character. For each template, we designed two series of experiments to measure the robustness of an algorithm under different deformations and outliers. In the first series of experiments, we rotated the template with a varying degree (between 0 and π). In the second set of experiments, a varying amount of outliers (between 0 and 20) were randomly added in the bounding box of template. For instance, Fig. 6a-b illustrate two pairs of example shapes with 20 outliers. We repeated

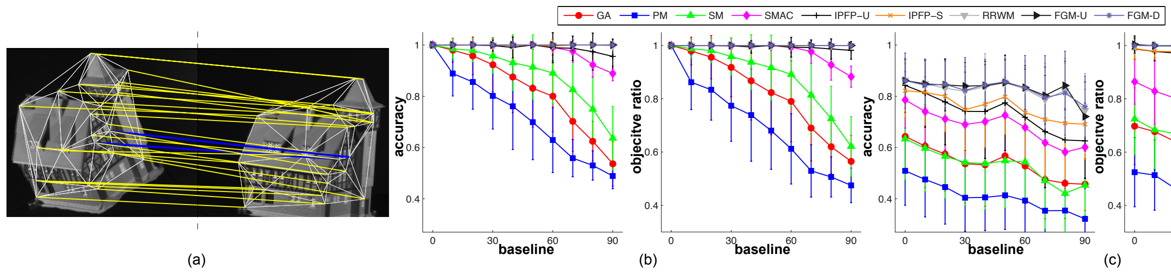
Figure 4. Comparison of GM methods on the CMU house datasets. (a) An example pair of frames with the correspondence generated by our method, where the blue lines indicate incorrect matches. (b) Performance of several algorithms using 30 nodes. (c) Performance using 25 nodes.

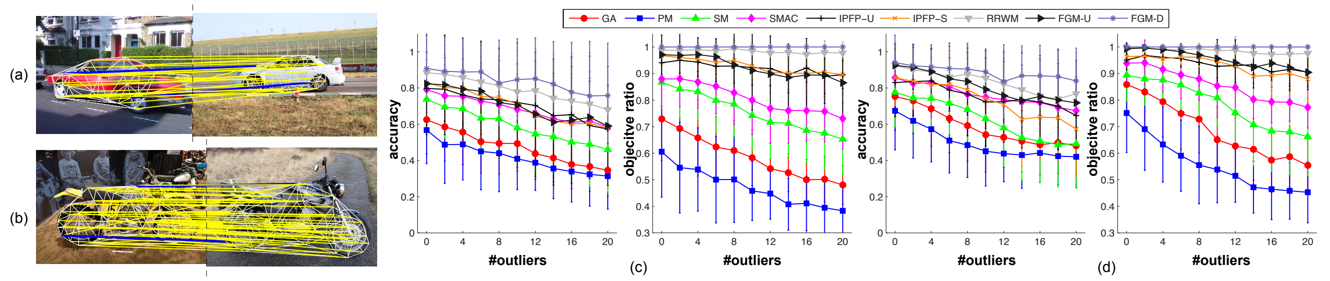
Figure 5. Comparison of GM methods on the car and motorbike dataset. (a) An example pair of car images with the correspondence generated by our method, where the blue lines indicate incorrect matches. (b) An example pair of motorbike images. (c) Performance as a function of the outlier number for the car images. (d) Performance as a function of the outlier number for the motorbike images.

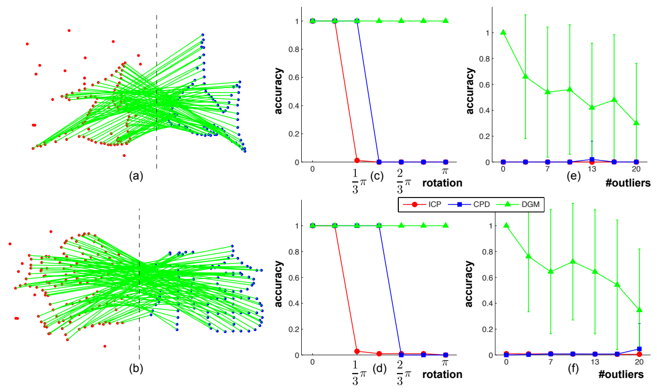
Figure 6. Comparison between DGM and ICP on the UCF shape datasets. (a-b) Two example pairs of shapes aligned using DGM. The red shape (left) is a rotated version of the blue one (right) by $\frac{2}{3}\pi$ and 20 random outliers were added. (c-d) Matching performance as a function of the initial rotations. (e-f) Matching performance as a function of the number of outliers.

the random generation 50 times for different levels of noise and compared DGM with the standard ICP algorithm and the coherent point drifting (CPD) [19]. The ICP algorithm was implemented by ourselves and CPD implementation was taken from the authors' website. We initialized all the algorithms with the same transformation, *i.e.*, $\tau(\mathbf{p}) = \mathbf{p}$. In DGM, Delaunay triangulation was employed to compute the graph structure. Recall that DGM simultaneously computes the correspondence and the rotation.

As shown in Fig. 6c-d, the proposed DGM can perfectly match the shapes across all the rotations without outliers, whereas both ICP and CPD get trapped in the local optimal when the rotation is larger than $\frac{2}{3}\pi$. When the number of outliers increases, DGM can still match most points under large rotation at $\frac{2}{3}\pi$. In contrast, ICP and CPD drastically failed in presence of outliers and large rotations (Fig. 6e-f).

In addition to a similarity transform, DGM can also incorporate non-rigid transformations in GM. Similar to the rigid case described in the main submission, we synthesized the non-rigid shape from the UCF shape dataset [7]. To generate the nonrigid transformation, we followed a similar setting in [19], where the domain of the point set was parameterized by a mesh of control points. The deformation of the mesh was modeled as an spline-based interpolation of the perturbation of the control points. We repeated the random generation 50 times. Fig. 7a illustrates a synthetic pair of graphs.

We compared DGM with other two state-of-the-art GM methods: SM [14] and RRWM [6]. In addition, we tested the performance of our algorithm (FGM-D) only using the path-following algorithm for computing the correspondence but without estimating the transformation. As shown in Fig. 7b-c, FGM-D performed better than the other two GM methods. This is due to the path-following algorithm that is more accurate in optimizing GM problems. DGM significantly improved FGM-D by estimating the transformation.

5.4. Conclusions

This paper proposes DGM, an extension of GM for matching points under a global geometric transformation for directed and undirected graphs. The key idea for DGM is a novel factorization of the pairwise affinity matrix. Sev-

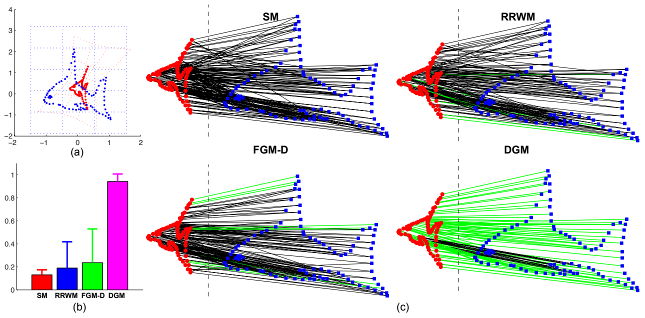

Figure 7. Comparison between DGM and GM methods for aligning non-rigidly deformed shapes. (a) An example of two fishes, where the red one is generated by a non-rigid transformation from the blue one. (b) Accuracy. (c) Results of GM methods, where the green and black lines indicate correct and incorrect correspondence respectively.

eral benefits follow from the factorization. First, it avoids the expensive (in space and time) computation of the pairwise affinity matrix. Second, it allows for a unification of GM methods and provides a clean connection with existing ICP algorithms. Finally, the decomposition enables the use of path-following algorithms that improve the performance of GM methods.

Acknowledgments The first author was supported by the National Science Foundation (NSF) under Grant No. EEEC-0540865 and CPS-0931999. The second author was partially supported by the NSF grant RI-1116583. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the NSF.

References

[1] H. A. Almohamad and S. O. Duffuaa. A linear programming approach for the weighted graph matching problem. *IEEE Trans. Pattern Anal. Mach. Intell.*, 15(5):522–525, 1993. 4

[2] P. J. Besl and N. D. McKay. A method for registration of 3-d shapes. *IEEE Trans. Pattern Anal. Mach. Intell.*, 14(2):239–256, 1992. 3

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

[4] R. Burkard, M. DellAmico, and S. Martello. *Assignment Problems*. SIAM, 2009. 1

[5] M. Chertok and Y. Keller. Efficient high order matching. *IEEE Trans. Pattern Anal. Mach. Intell.*, 32(12):2205–2215, 2010. 3

[6] M. Cho, J. Lee, and K. M. Lee. Reweighted random walks for graph matching. In *ECCV*, 2010. 1, 3, 4, 6, 7

[7] H. Chui and A. Rangarajan. A new point matching algorithm for non-rigid registration. *Comput. Vis. Image Underst.*, 89(2-3):114–141, 2003. 6, 7

[8] T. Cour, P. Srinivasan, and J. Shi. Balanced graph matching. In *NIPS*, 2006. 1, 2, 4, 6

[9] O. Duchenne, F. Bach, I.-S. Kweon, and J. Ponce. A tensor-based algorithm for high-order graph matching. *IEEE Trans. Pattern Anal. Mach. Intell.*, 33(12):2383–2395, 2011. 3

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

[11] S. Gold and A. Rangarajan. A graduated assignment algorithm for graph matching. *IEEE Trans. Pattern Anal. Mach. Intell.*, 18(4):377–388, 1996. 1, 3, 4, 6

[12] J. Hays, M. Leordeanu, A. A. Efros, and Y. Liu. Discovering texture regularity as a higher-order correspondence problem. In *ECCV*, 2006. 1

[13] H. Jiang, S. X. Yu, and D. R. Martin. Linear scale and rotation invariant matching. *IEEE Trans. Pattern Anal. Mach. Intell.*, 33(7):1339–1355, 2011. 1

[14] M. Leordeanu and M. Hebert. A spectral technique for correspondence problems using pairwise constraints. In *ICCV*, 2005. 1, 2, 4, 6, 7

[15] M. Leordeanu, M. Hebert, and R. Sukthankar. An integer projected fixed point method for graph matching and MAP inference. In *NIPS*, 2009. 1, 3, 4, 6

[16] M. Leordeanu, R. Sukthankar, and M. Hebert. Unsupervised learning for graph matching. *Int. J. Comput. Vis.*, 95(1):1–18, 2011. 6

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

[18] E. M. Loiola, N. M. De Abreu, P. O. Boaventura, P. Hahn, and T. M. Querido. A survey for the quadratic assignment problem. *Eur. J. Oper. Res.*, 176(2):657–690, 2007. 1, 2

[19] A. Myronenko and X. B. Song. Point set registration: Coherent point drift. *IEEE Trans. Pattern Anal. Mach. Intell.*, 32(12):2262–2275, 2010. 7

[20] N. Quadrianto, A. J. Smola, L. Song, and T. Tuytelaars. Kernelized sorting. *IEEE Trans. Pattern Anal. Mach. Intell.*, 32(10):1809–1821, 2010. 1

[21] L. S. Shapiro and M. Brady. Feature-based correspondence: an eigenvector approach. *Image Vision Comput.*, 10(5):283–288, 1992. 2

[22] J. R. Ullmann. An algorithm for subgraph isomorphism. *J. ACM*, 23:31–42, 1976. 1

[23] S. Umeyama. An eigendecomposition approach to weighted graph matching problems. *IEEE Trans. Pattern Anal. Mach. Intell.*, 10(5):695–703, 1988. 2, 4

[24] B. J. van Wyk and M. A. van Wyk. A POCS-based graph matching algorithm. *IEEE Trans. Pattern Anal. Mach. Intell.*, 26(11):1526–1530, 2004. 1, 4

[25] M. Zaslavskiy, F. R. Bach, and J.-P. Vert. A path following algorithm for the graph matching problem. *IEEE Trans. Pattern Anal. Mach. Intell.*, 31(12):2227–2242, 2009. 3, 4

[26] R. Zass and A. Shashua. Probabilistic graph and hypergraph matching. In *CVPR*, 2008. 1, 3, 4, 6

[27] Z. Zhang. Iterative point matching for registration of free-form curves and surfaces. *Int. J. Comput. Vis.*, 13(2):119–152, 1994. 3

[28] F. Zhou and F. De la Torre. Factorized graph matching. *IEEE Trans. Pattern Anal. Mach. Intell.* Under review, `http://humansensing.cs.cmu.edu/fgm`. 3, 4, 5

[29] F. Zhou and F. De la Torre. Factorized graph matching. In *CVPR*, 2012. 2, 3, 4, 6