Consistency-driven Alternating Optimization for Multi-Graph Matching: a Unified Approach

Junchi Yan, Jun Wang *Member, IEEE*, Hongyuan Zha, Xiaokang Yang *Senior Member, IEEE* and Stephen Chu

***Abstract*—The problem of graph matching in general is NP-complete and many approximate pairwise matching techniques have been proposed. For a general setting in real applications, it typically requires to find the consistent matchings across a batch of graphs. Sequentially performing pairwise matching is prone to error propagation along the pairwise matching sequence, and the sequences generated in different pairwise matching orders can lead to contradictory solutions. Motivated by devising a robust and consistent multiple-graph matching model, we propose a unified alternating optimization framework for multi-graph matching. In addition, we define and use two metrics related to graph-wise and pairwise consistencies. The former is used to find an appropriate reference graph which induces a set of basis variables and launches the iteration procedure. The latter defines the order in which the considered graphs in the iterations are manipulated. We show two embodiments under the proposed framework that can cope with the non-factorized and factorized affinity matrix, respectively. Our multi-graph matching model has two major characters: i) the affinity information across multiple graphs are explored in each iteration by fixing part of the matching variables via a consistency-driven mechanism; ii) the framework is flexible to incorporate various existing pairwise graph matching solvers in an "out-of-box" fashion, and also can proceed with the output of other multi-graph matching methods. The experimental results on both synthetic data and real images empirically show that the proposed framework performs competitively with the state-of-the-art.**

I. Introduction

GRAPH matching (GM) plays a critical role in both theory and application for computer science community. Briefly speaking, graph matching aims to find correspondence between two feature sets, with a wide spectrum of applications that require feature matching in image processing and computer vision, as diverse as image registration [2], object recognition [3], shape matching [4], object tracking [5], and action recognition [6], among others. Different from the point based matching methods such as RANSAC [7] and Iterative Closet Point (ICP) [8], GM methods incorporate both the unary *node-to-node*, as well as the second-order *edge-to-edge* similarity as structural information. As discussed in [9], by encoding the additional second-order edge similarity in the graph representation and matching process, GM methods can usually lead to better node correspondence solutions. Although extensive research has been done for decades [10], deriving optimal GM is still a challenging problem in both theory and practice since the GM problem can be formulated as a quadratic assignment problem (QAP) [11], being well-known NP-complete [12][13].

Graph matching is mostly considered under the two-graph scenario [9], [14], [15], [16], [17], [18] etc. However, in many real applications, given a batch of graphs referring to identical or related structures, it is required to find the global node mappings among all graphs, which is essentially related to graph clustering [19], [20], [21], [22], classification [23], [24], [25] and indexing [26]. Due to the advance of modern imaging and scanning technologies, there is an increasing need for multi-graph matching in various applications. In [27], objects are scanned using infrared, optical, cartographic and Synthetic Aperture Radar (SAR) that are desired to be combined to improve representations. In addition, 3D shape analysis often requires to model objects by multi-view assembly [28].

From the methodology perspective, pairwise graph matching methods aim to find the correspondences that preserve the structural invariance between two graphs. Though being successful when the structural invariance property is roughly hold, pairwise matching methods tend to fail when the differences between the input graphs are significant or the graph noises are nontrivial. This is because the optimization solver may be trapped in local optima and/or far from the semantic ground truth especially when large noises exist. However, a set of graphs open the possibility to explore the global information and matching consistency regularization that help improve the matching accuracy. For instance, directly producing node correspondences between two significantly deformed graphs via pairwise matching techniques is extremely error-prone; while ideally, an indirect matching with a sequence of interpolating

A preliminary version of the presented paper appeared in [1]. The current paper makes several further extensions and improvements: i) incorporate the factorized graph matching in our formulation, and solve it via a path-following algorithm; ii) propose two metrics for graph-wise and pairwise consistency, which are used for the effective finding of basis variables and updating order for the alternating optimization procedure. In particular, the preliminary work does not address these two problems as only a few graphs are tested in [1]. However, they become more critical when a relatively large number of graphs are considered; iii) perform extensive evaluations that involves more graph samples, additional datasets, and more state-of-the-arts for comparison.

J. Yan (correspondence author) is with the Department of Electronic Engineering, Shanghai Jiao Tong University, Shanghai, 200240 China. Meanwhile, he is also secondarily affiliated with IBM Research – China, Shanghai, 201203 China. e-mail: yanjunchi@sjtu.edu.cn

J. Wang is with Institute of Data Science and Technology, Alibaba Group, Seattle, WA, USA. e-mail: j.wang@alibaba-inc.com

H. Zha is with Software Engineering Institute, East China Normal University, Shanghai, 200062 China, and the School of Computational Science and Engineering, College of Computing, Georgia Institute of Technology, Atlanta, Georgia, 30332, USA. e-mail: zha@cc.gatech.edu

X. Yang is with Shanghai Key Laboratory of Media Processing and Transmission, Department of Electronic Engineering, Shanghai Jiao Tong University, Shanghai, 200240 China. e-mail: xkyang@sjtu.edu.cn

S. Chu is with IBM Research – China, Shanghai, 201203 China. e-mail: schu@cn.ibm.com

The work is supported by NSF IIS-1116886, NSF IIS-1049694, NSFC 61129001, NSFC 61025005 and the 111 Program (B07022).

graphs yields more robust results. For arbitrary graphs, simply applying the existing pairwise matching algorithms on each pair of graph is far from satisfaction. Such a pairwise matching based scheme encodes no global reasoning and regularization due to the separated and independent matching process, resulting in information-loss and vulnerability to local noises. Moreover, different orders of sequential pairwise matching would usually lead to inconsistent matching results, which means $\mathbf{X}_{ia}^T\mathbf{X}_{aj} \neq \mathbf{X}_{ib}^T\mathbf{X}_{bj}$ for matching graph i and j, and the anchor graph is a and b respectively. Here $\mathbf{X}$ is the node-mapping assignment matrix whose formal definition would be given later in the paper. This fact has been illustrated and discussed in our conference paper [1].

Compared with the vast existing work on the pairwise graph matching problem [9], [14], [15], [16], [17], [18], [29], [30] etc., the topic of multi-graph matching is in fact a relatively less-investigated one in the image processing and computer vision community [27], [31], [32]. Motivated by the limitations of the existing approaches, we aim to design a robust method to obtain consistent node-to-node correspondence across a collection of loosely related objects or weighted graphs. Despite the fact that no theoretical guarantee is provided for obtaining a globally optimal solutions (note the problem is NP-complete), in this paper, we do take account the global information of the entire set of graphs to generate approximated solutions that tend to satisfy the structural information and matching consistency across all the graphs. To evaluate the effectiveness of the proposed multi-graph matching approach, we perform extensive empirical studies on both synthetic datasets that are built using the popular protocols [15], [29], as well as the well-known benchmark datasets including the CMU motion sequence, the POSE-sequence, and the WILLOW-ObjectClass data.

The major contributions of our work are:

First, we propose a unified framework, which has two alternating optimization algorithm variants to cope with non-factorized and factorized affinity matrix in the objective function, respectively. Our method bears several merits: i) the pairwise affinities over multiple graphs are jointly explored, in the hope of being more robust against local noises, which is empirically observed in our extensive tests; ii) the framework is flexible to incorporate various existing pairwise matching techniques in an "out-of-box" fashion, in two stages for both before and during the iteration; iii) our method does not impose any consistency requirement on the initial matching inputs. By contrast, other methods [32], [33] require the initial solutions must be inconsistent, which allows them to improve the overall accuracy by enforcing consistency. Therefore, our method can start with, and further improve the results of [32], [33] also in an "out-of-box" fashion.

Second, we define and use the metrics related to graph-wise/pairwise consistency to address two key challenges in our framework: i) appropriate selection of the reference graph which induces a set of basis variables to proceed the iterative optimization procedure effectively; ii) adaptive setting of the iterative updating order, which improves the convergence speed by a notable margin. Compared with the preliminary version [1], the consistency-driven framework outperforms in extensive empirical tests.

II. Related Work

As a general problem for matching structural data, graph matching has been extensively studied for decades not only in image processing and computer vision, but also in computer science and mathematics [10], [34]. Here we view the problem from several key aspects that account for the main threads of the related work, mostly in the area of image processing and pattern recognition. We will discuss pairwise matching and multiple graph matching separately.

A. Advances on pairwise graph matching

Machine Learning Methodologies: Conventional graph matching methods first compute an affinity matrix and keep the affinity metrics unchanged during the entire matching process. Recent work leverage various leaning algorithms for estimating the optimal affinity matrix [9], [17], [35], [36], [37], and the methods can fall into either supervised [17], unsupervised [9], or semi-supervised [35] learning paradigm. The problem of estimating the optimal setting of the affinity metric is out of the scope of this paper. We assume the affinity matrix is prior known which is the same with most of related work [4], [14], [15], [16], [29], [38], [39] etc.

Higher-Order Affinity Modeling: Combing the unary and second-order edge information has been heavily investigated since such a type of matching schemes plays a good tradeoff between computational complexity and representation capability [4], [14], [15], [38], [39]. Recently, the higher-order information has been encoded to achieve more robust matching paradigms. Several representative hyper-graph matching methods have been proposed for pattern recognition and image processing applications [35], [40], [41], [42], [43] that encode higher-order information to enhance the matching distinctiveness. However, a slight increase in the order would lead to a combinatorial explosion of the state space. As a result, most of higher-order methods, such as the related work we mentioned here, are typically applied on very sparse graphs with no more than third-order features. In addition, it is worth noting that the term *pairwise* used in this paper refers to matching two graphs, while in the context of hyper-graph matching, *pairwise* sometimes refers to the second-order edge affinity.

Optimization Methods: Most approaches first formulate an objective function, and then employ certain optimization methods to derive optimal matching results [4], [14], [38], which can vary among a wide spectrum of optimization techniques and strategies. Several recent work first relax the objective function to a convex-concave formulation [29], [44]. Then the optimal solutions are obtained using the so-called path following strategy and a modified version of the Frank-Wolfe algorithm [45]. Probabilistic matching paradigms are also developed, which have shown power in interpreting and addressing hyper-graph matching problems [15], [42], [43].

B. Advances on multiple graph matching

The multi-graph matching problem is also referred as *Common Labeling* [31] or *Multiple Isomorphism* [46]. In [31], a

common labeling is defined as a bijective mapping between all graph nodes in the considered graphs to a virtual node set. The common labeling is constructed by a consistent multiple isomorphism [46], where an isomorphism involves assigning each node from one graph to one of the other graphs.

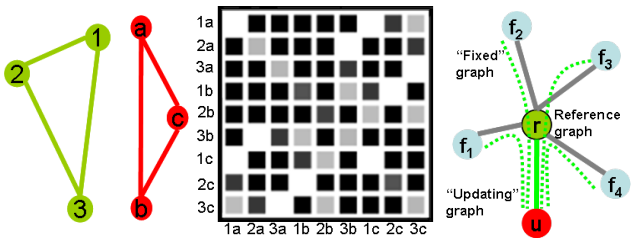
Compared with the pairwise matching problem, the multiple graph matching has not been extensively studied, and only a few address this problem using principled formulations or techniques. A loosely related early work [47] aims at finding a maximum spanning tree on a *super graph* whose vertices represent graphs, and its edges represent matching correspondences between two graphs. The edges on the super-graph are weighted by their "quality" such as matching affinity score as we will describe later in this paper. Very recently, Sole-Ribalta and Serratosa [48] (and its journal version [31]) generalize the classical Graduated Assignment Graph Matching (GAGM) algorithm [14], [16] from two-graph to multi-graph case, which generates the common labeling by matching all graph nodes to a virtual node set. This method is assumed to inherit the sound performance of the original method, while being more time-consuming as it repeatedly applies GAGM across graph pairs iteratively. Other two recent work [33], [32] start their procedure on an inconsistent pairwise matching configuration that covers all graph pairs. Given N graphs with n nodes per each, the former method first builds an N-dimensional probabilistic hyper-cube with size n in each dimension, whereby each cell indicates the probability for the N-node correspondence from N graphs respectively. Then the consistency and binarization are fulfilled in a post-processing step. The latter employs spectral approximation to eigenvector decomposition on the matching configuration matrix stacked by all raw pairwise matching solutions (assignment matrix), and recover the consistent matching solutions. There are also several other more recent work on matching a batch of graphs by self-boosting [49] and multi-view point registration [50].

III. Preliminaries for Graph Matching

In this section, several notations used in this paper and two definitions related to the proposed framework will be introduced, followed by a retrospection on existing formulations for pairwise graph matching under two paradigms: non-factorized and factorized representations of the affinity matrix.

A. Notations and definitions

Throughout the paper, $\mathbb{R}$ denotes the real number domain. Bold capital letters denote for a matrix $\mathbf{X}$, bold lower-case letters for a column vector $\mathbf{x}$, and hollow bold letters for a set $\mathbb{X}$. All non-bold letters represent scalars. $\mathbf{X}^T$ is the transpose of $\mathbf{X}$; diag($\mathbf{x}$) is a diagonal matrix whose diagonal elements are $\mathbf{x}$; vec($\mathbf{X}$) is the column-wise vectorized matrix $\mathbf{X}$. tr($\mathbf{X}$) is the trace of matrix $\mathbf{X}$. $\mathbf{I}_n \in \mathbb{R}^{n\times n}$ is an identity matrix and the subscript n will be omitted when it can be inferred from context. $\mathbf{1}_{m\times n}, \mathbf{0}_{m\times n} \in \mathbb{R}^{m\times n}$ are matrices of all elements being ones and zeros. $\mathbf{1}_n$ is the abbreviation for $\mathbf{1}_{n\times 1}$. $\mathbf{X}\circ\mathbf{Y}$ and $\mathbf{X}\otimes\mathbf{Y}$ denote the Hadamard and Kronecker products of matrices. $|\mathbb{X}|$ is the cardinality of the set $\mathbb{X}$, $\|\mathbf{X}\|_F = \text{tr}(\mathbf{X}^T\mathbf{X})$ for the Frobenious norm and $\|\cdot\|_p$ is the p-norm.



(a) two graphs and the associated affinity matrix (b) the star tree structure

Fig. 1: Illustrations for: a) the pairwise matching affinity matrix induced by two sample graphs. Darker cells in the matrix denote smaller value within the normalized value range $[0, 1]$; b) the star tree structure for the reference graph together with the specified "updated" graph and the resting "fixed" graphs in one iteration of the proposed two methods. Five pairwise terms are considered: $\mathbf{K}_{ru}$, $\mathbf{K}_{f_1u}$, $\mathbf{K}_{f_2u}$, $\mathbf{K}_{f_3u}$, $\mathbf{K}_{f_4u}$.

Specifically, his paper intensively uses $\mathbf{X}_{ij}$ to denote the pairwise matching matrix between graph i, j, and $\mathbb{X} = \{\mathbf{X}_{ij}, i, j = 1, \ldots, N\}$ for the set of pairwise matching matrix over a graph set denoted by $\mathbb{G} = \{\mathcal{G}_1, \ldots, \mathcal{G}_N\}$. We call $\mathbb{X}$ the matching *configuration* w.r.t. $\mathbb{G}$. If not explicitly stated, we follow the convention that using N for the cardinality of $\mathbb{G}$, n for the number of nodes in a graph, and m for edges.

Now we introduce two definitions regarding with matching consistency induced by the initial pairwise matching results from any pairwise matching solver. As would be shown later in this paper, these two definitions play a key role in the proposed alternating optimization framework. In general, they are used to solve two common problems associated with an alternating optimization method, respectively: i) how to choose a set of base pairwise matching variables to proceed the iteration; ii) how to set an optimal rotating order for alternating updating.

Definition 1. *Given N graphs $\mathbb{G}=\{\mathcal{G}_k, k = 1, \ldots, N\}$ and the pairwise matching configuration $\mathbb{X} = \{\boldsymbol{X}_{ij}, i, j = 1, \ldots, N\}$, the graph-wise consistency of graph $\mathcal{G}_k$ is defined as $C_g(\mathcal{G}_k, \mathbb{X})=1-\frac{\sum_{i=1}^{N-1}\sum_{j=i+1}^{N}\|\boldsymbol{X}_{ij}-\boldsymbol{X}_{ik}\boldsymbol{X}_{kj}\|_F/2}{nN(N-1)/2} \in [0, 1]$ where $\boldsymbol{X}_{ij}$ is the pairwise permutation matrix over the graph set $\mathbb{G}$.*

Definition 2. *Given a set of graphs $\mathbb{G}=\{\mathcal{G}_k, k = 1, \ldots, N\}$, matching configuration $\mathbb{X}$, and a reference graph $\mathcal{G}_r$, for any other graph $\mathcal{G}_u$, the pairwise consistency between $\mathcal{G}_u$ and $\mathcal{G}_r$ is defined as $C_p(\mathcal{G}_u, \mathcal{G}_r, \mathbb{X}) = 1 - \frac{\sum_{i=1}^{N}\|\boldsymbol{X}_{ri}-\boldsymbol{X}_{ru}\boldsymbol{X}_{ui}\|_F/2}{nN} \in [0, 1]$*

We leave the details of using the two definitions to Section IV in the context of the proposed multi-graph matching framework. In the rest of this section, we depict the two existing pairwise graph matching formulations, which are the origins of the proposed multi-graph matching algorithms.

B. Non-factorized pairwise graph matching

Given two graphs $\mathcal{G}_1(\mathcal{V}_1, \mathcal{E}_1, \mathcal{A}_1)$ and $\mathcal{G}_2(\mathcal{V}_2, \mathcal{E}_2, \mathcal{A}_2)$, where $\mathcal{V}$ denotes nodes, $\mathcal{E}$, edges and $\mathcal{A}$, attributes. There is an affinity matrix $\mathbf{K} \in \mathbb{R}^{n_1n_2\times n_1n_2}$ defined such that $K_{ia;jb}, (i, j = 1, \ldots, n_1), (a, b = 1, \ldots, n_2)$ measures the edge pair affinity (v_i, v_j) vs. (v_a, v_b) from two graphs. The diagonal term $K_{ia;ia}$ describes the unary affinity of a node match (v_i, v_a). Rigorously, the affinity value $K_{ia;jb}$ for the edge pair (v_i, v_j) vs. (v_a, v_b) is located at the $((a\text{-}1)n_1+i)$-th row and $((b\text{-}1)n_2+j)$-th column of $\mathbf{K}$. One example of $\mathbf{K}$ is illustrated in Fig.1(a). By introducing an assignment matrix $\mathbf{X} \in \{0, 1\}^{n_1\times n_2}$ for $n_1 = |\mathcal{V}_1|, n_2 = |\mathcal{V}_2|$, we set $X_{ia} = 1$ if

node v_i matches node v_a (0 otherwise). The problem of GM involves finding the optimal correspondence $\mathbf{X}$, such that the sum of the node and edge compatibility between two graphs is maximized. Without loss of generality, we assume $n_1 \geq n_2$ for different sizes of graphs. This leads to the following two-way constrained quadratic assignment problem (QAP):

$$\begin{aligned} \mathbf{X}^* &= \arg\max_{\mathbf{X}} \mathrm{vec}(\mathbf{X})^T \mathbf{K} \mathrm{vec}(\mathbf{X}) \\ s.t. \quad & \mathbf{X}\mathbf{1}_{n_2} \leq \mathbf{1}_{n_1} \quad \mathbf{1}_{n_1}^T \mathbf{X} = \mathbf{1}_{n_2}^T \quad \mathbf{X} \in \{0,1\}^{n_1 \times n_2} \end{aligned}$$

The constraints refer to the two-way one-to-one node mapping: a node from graph $\mathcal{G}_1$ can match at most one node in $\mathcal{G}_2$ and every node in $\mathcal{G}_2$ is corresponding to one node in $\mathcal{G}_1$. There is no (one/many)-to-many matchings between two graphs.

The above formulation is general as it allows two graphs having unequal number of nodes ($n_1 \neq n_2$). Similar to the previous work [12], [29], we follow the protocol that converts $\mathbf{X}$ from an assignment matrix ($\mathbf{X}\mathbf{1}_{n_2} \leq \mathbf{1}_{n_1}$) to a permutation matrix ($\mathbf{X}\mathbf{1}_{n_2} = \mathbf{1}_{n_1}$) by adding dummy nodes to one graph (i.e. adding slack variables to the assignment matrix and augment the affinity matrix by zeros) in case $n_1 \neq n_2$. This is a standard technique from linear programming [51] and is adopted by [12], [29], [52] etc., being able to handle superfluous nodes in a statistically robust manner. By taking this step, the graphs are of equal sizes. This preprocessing opens up the applicability of existing multi-graph methods [31], [32], [46] as they are all based on the assumption that all graphs are of equal sizes. Therefore, the following formulation is used throughout the paper which assumes $n_1 = n_2 = n$.

$$\begin{aligned} \mathbf{X}_{12}^* &= \arg\max_{\mathbf{X}_{12}} \mathrm{vec}(\mathbf{X}_{12})^T \mathbf{K}_{12} \mathrm{vec}(\mathbf{X}_{12}) \\ s.t. \quad & \mathbf{X}_{12}\mathbf{1}_{n_2} = \mathbf{1}_{n_1} \quad \mathbf{1}_{n_1}^T \mathbf{X}_{12} = \mathbf{1}_{n_2}^T \quad \mathbf{X}_{12} \in \{0,1\}^{n_1 \times n_2} \end{aligned} \tag{1}$$

Here $\mathbf{X}_{12}$ is a permutation matrix which is constructed by augmenting the assignment matrix with slack columns in case $n_1 > n_2$, such that there is always a bijection between graphs.

The above QAP formulation is used by most existing GM methods [4], [14], [15], [16], [38], which directly deals with the large pairwise affinity matrix $\mathbf{K}$. The affinity matrix plays a central role in GM because it encodes all the first-order and second-order relations between graphs. The two common properties of $\mathbf{K}$, full-rankness and indefiniteness, pose key challenges to conventional GM methods such as spectral methods [4], [38] and gradient-based approaches [14], [15], [16]. In addition, its relatively large size also impedes the applicability when large-size graphs are considered. Thus the factorized graph matching formulation is studied and introduced by [29] in parallel which avoids involving the whole $\mathbf{K}$ for optimization.

C. Factorized pairwise graph matching

The authors in [29] show that the affinity matrix can be factorized as a Kronecker product of smaller matrices, which decouples the graph structure from the affinity. Specifically a graph $\mathcal{G}$ can be denoted by $\{\mathbf{P}, \mathbf{Q}, \mathbf{G}\}$, where $\mathbf{P} = [\mathbf{p}_1, \ldots, \mathbf{p}_n] \in \mathbb{R}^{d_p \times n}$, $\mathbf{Q} = [\mathbf{q}_1, \ldots, \mathbf{q}_m] \in \mathbb{R}^{d_q \times m}$ are the feature matrices computed for n nodes and m edges. Here d_p and d_q are the number of dimensions for unary features and edge features. The topology of the graph is specified by the node-edge incidence matrix $\mathbf{G} \in \mathbb{R}^{n \times m}$ such that the non-zero elements in each column of $\mathbf{G}$ indicate the starting and ending nodes in the corresponding edge. Refer to [29] for more details. More specifically, given two graphs $\{\mathbf{P}_1, \mathbf{Q}_1, \mathbf{G}_1\}$ and $\{\mathbf{P}_2, \mathbf{Q}_2, \mathbf{G}_2\}$, let $\mathbf{K}_{12}^p \in \mathbb{R}^{n_1 \times n_2}$ denote the node affinity matrix, and $\mathbf{K}_{12}^q \in \mathbb{R}^{m_1 \times m_2}$ for the edge affinity matrix, which is used to measure the similarity of each pair of nodes and edges, respectively. As shown in [29], then the affinity matrix for pairwise matching can be factorized to:

$$\begin{aligned} \mathbf{K}_{12} &= (\mathbf{H}_2 \otimes \mathbf{H}_1)\mathrm{diag}(\mathrm{vec}(\mathbf{L}_{12}))(\mathbf{H}_2 \otimes \mathbf{H}_1)^T \\ \text{where} \quad \mathbf{H}_i &= [\mathbf{G}_i, \mathbf{I}_{n_i}] \in \{0,1\}^{n_i \times (m_i + n_i)}, \quad i = 1, 2 \\ \mathbf{L}_{12} &= \begin{bmatrix} \mathbf{K}_{12}^q & -\mathbf{K}_{12}^q \mathbf{G}_2^T \\ -\mathbf{G}_1 \mathbf{K}_{12}^q & \mathbf{G}_1 \mathbf{K}_{12}^q \mathbf{G}_2^T + \mathbf{K}_{12}^p \end{bmatrix} \end{aligned}$$

Based on the above factorization of $\mathbf{K}_{12}$, the original pairwise graph matching objective function can be written as [29]:

$$\begin{aligned} J_{12} =& \mathrm{vec}(\mathbf{X}_{12})^T (\mathbf{H}_2 \otimes \mathbf{H}_1)\mathrm{diag}(\mathrm{vec}(\mathbf{L}_{12}))(\mathbf{H}_2 \otimes \mathbf{H}_1)^T \mathrm{vec}(\mathbf{X}_{12}) \\ =& \mathrm{tr}\Big(\mathbf{L}_{12}(\mathbf{H}_1^T \mathbf{X}_{12} \mathbf{H}_2) \circ (\mathbf{H}_1^T \mathbf{X}_{12} \mathbf{H}_2\Big) \end{aligned}$$

By factorizing $\mathbf{L}_{12}$ to $\mathbf{L}_{12} = \mathbf{U}_{12}\mathbf{V}_{12}^T = \sum_{i=1}^{c} \mathbf{u}_{12}^i \mathbf{v}_{12}^{iT}$, where $\mathbf{U}_{12} = [\mathbf{u}_{12}^1, \mathbf{u}_{12}^2, \cdots, \mathbf{u}_{12}^c] \in \mathbb{R}^{(n_1+m_1) \times c}$, $\mathbf{V} = [\mathbf{v}_{12}^1, \mathbf{v}_{12}^2, \cdots, \mathbf{v}_{12}^c] \in \mathbb{R}^{(n_2+m_2) \times c}$, J_{12} can be derived as [29]:

$$\begin{aligned} J_{12} &= \sum_i \mathrm{tr}\Big(\mathbf{A}_{12}^i \mathbf{X}_{12} \mathbf{B}_{12}^i \mathbf{X}_{12}^T\Big) \\ \text{where} \quad \mathbf{A}_{12}^i &= \mathbf{H}_1 \mathrm{diag}(\mathbf{u}_{12}^i) \mathbf{H}_1^T, \quad \mathbf{B}_{12}^i = \mathbf{H}_2 \mathrm{diag}(\mathbf{v}_{12}^i) \mathbf{H}_2^T \end{aligned} \tag{2}$$

Note the above factorization formulation does not require two graphs being of equal sizes. In the rest of paper, similar to the non-factorization case, we will add dummy nodes in case of unequal sizes of graph, to make the two-way constraint become a bijection. This step will transform $\mathbf{X}_{12}$ to a strict permutation matrix, which leads to the following convex-concave relaxation formulation as discussed in [29].

D. Convex-concave relaxations for factorized graph matching

By ensuring $\mathbf{X}_{12}$ a permutation matrix such that $\mathbf{X}_{12}^T \mathbf{X}_{12} = \mathbf{I}_{nn}$, the objective can be relaxed to the convex formulation, by adding a constant term $\mathbf{C}_{12}$ as used in [29]. Thus we have:

$$\begin{aligned} J_{12}^{vex}(\mathbf{X}_{12}) =& J_{12}(\mathbf{X}_{12}) - \frac{1}{2}\mathbf{C}_{12}(\mathbf{X}_{12}) \\ \text{where} \quad \mathbf{C}_{12} =& \sum_i \Big(\mathrm{tr}(\mathbf{A}_{12}^i \mathbf{A}_{12}^i \mathbf{X}_{12} \mathbf{X}_{12}^T) + \mathrm{tr}(\mathbf{B}_{12}^i \mathbf{B}_{12}^i \mathbf{X}_{12}^T \mathbf{X}_{12})\Big) \end{aligned} \tag{3}$$

On the other hand, as shown in [29], the concave relaxation J_{12}^{cav} can be written as:

$$\begin{aligned} J_{12}^{cav} =& \mathrm{tr}\Big(\mathbf{K}_{12}^{qT}(\mathbf{G}_1^T \mathbf{X}_{12} \mathbf{G}_2 \circ \mathbf{G}_1^T \mathbf{X}_{12} \mathbf{G}_2)\Big) \\ & - \mathrm{tr}\Big((\mathbf{G}_1 \mathbf{K}_{12}^q \mathbf{G}_2^T)^T \mathbf{X}_{12}\Big) + \mathrm{tr}\Big(\mathbf{K}_{12}^{pT} \mathbf{X}_{12}\Big) \end{aligned} \tag{4}$$

IV. Multiple Graph Matching by Consistency-driven Alternating Optimization

First of all, as the same with the preprocessing used in pairwise graph matching that makes graphs of equal size, we add dummy nodes to make all input graphs of the same size. Note this protocol is also used by [31], [32], [46] for multi-graph matching. Based on this consensus, in what follows we will present our algorithm with two variants, by factorizing the affinity matrix and not, respectively. The non-factorized formulation is a more dominant representation used in most existing work [4], [39], [15], [16], while the other avoids using the whole affinity matrix in optimization [29], [30]. We will provide a unified approach to cope with these two cases.

A. Non-factorized multiple graph matching

Given N graphs and the pairwise affinity matrix $\mathbf{K}_{ij}$ for each pair of graphs $\mathcal{G}_i, \mathcal{G}_j$, without loss of generality, the multi-graph matching objective function can be written as follows:

$$\begin{aligned} \mathbb{X}^* = \arg\max_{\mathbb{X}} \sum_{i,j=1,i\neq j}^{N} \mathrm{vec}(\mathbf{X}_{ij})^T \mathbf{K}_{ij} \mathrm{vec}(\mathbf{X}_{ij}) \qquad (5) \\ s.t. \quad \mathbf{X}_{ij}\mathbf{1}_n = \mathbf{1}_n \quad \mathbf{1}_n^T \mathbf{X}_{ij} = \mathbf{1}_n^T \quad \mathbf{X}_{ij} = \mathbf{X}_{ji}^T \in \{0,1\}^{n\times n} \\ \forall i,j = 1,\ldots,N; \quad |\mathbb{X}| = N(N-1) \end{aligned}$$

where $\mathbb{X} = \{\mathbf{X}_{ij}, i,j = 1,\ldots,N\}$ is the permutation (augmented assignment) matrix set such that $\mathbf{X}_{ij} = \mathbf{X}_{ji}^T$. Note that by the definition of affinity matrix as introduced earlier in the paper, in general $\mathbf{K}_{ij} \neq \mathbf{K}_{ji}$, $\mathbf{K}_{ij} \neq \mathbf{K}_{ji}^T$. Though they encode redundant information, we include all $\mathbf{K}_{ij}$ in the objective function, which would be used in the optimization procedure as we will show in the following.

One obvious observation on the above function is that the pairwise matching variables $\mathbf{X}_{ij}$ are redundant and can be determined by a compact basis variable set $\{\mathbf{X}_{rk}, k = 1,\ldots,N, k \neq r\}$ (a star tree rooted at the reference graph) such that any $\mathbf{X}_{ij}$ can be computed by $\mathbf{X}_{ki}^T\mathbf{X}_{kj}$. Consequently, we design our non-factorized multi-graph matching algorithm as follows. First, in order to induce a basis matching set to proceed the optimization procedure, a reference graph $\mathcal{G}_r$ is set by a certain means. Then in each iteration, one can choose a certain graph $\mathcal{G}_u$ in rotation by a certain order O_{udt}, and update its mapping with $\mathcal{G}_r$ i.e. $\mathbf{X}_{ur}$, by fixing the other $N-2$ $\mathbf{X}_{rf}$ regarding graphs $\mathcal{G}_f$ $(f = 1,\ldots,N, f \neq r, u)$ against $\mathcal{G}_r$. This idea is illustrated in Fig.1(b). Now, we reach the following objective function by dropping the constant terms[1] for $\mathbf{X}_{rf}$:

$$\mathrm{vec}(\mathbf{X}_{ur})^T \mathbf{K}_{ur} \mathrm{vec}(\mathbf{X}_{ur}) + \sum_{f=1,f\neq r,u}^{N} \mathrm{vec}(\mathbf{X}_{uf})^T \mathbf{K}_{uf} \mathrm{vec}(\mathbf{X}_{uf})$$

Note in the permutation matrix form we have $\mathbf{X}_{uf} = \mathbf{X}_{ur}\mathbf{X}_{rf}$, this leads to the following equation in the vectorized form:

$$\mathrm{vec}(\mathbf{X}_{uf}) = (\mathbf{X}_{fr} \otimes \mathbf{I})\mathrm{vec}(\mathbf{X}_{ur}) \qquad (6)$$

From now on we will use $\mathbf{F}_{fr} \in \mathbb{R}^{n^2\times 1}$ to denote $\mathbf{X}_{fr} \otimes \mathbf{I}$, by replacing $\mathrm{vec}(\mathbf{X}_{uf})$ via $\mathrm{vec}(\mathbf{X}_{uf}) = \mathbf{F}_{fr}\mathrm{vec}(\mathbf{X}_{ur})$ we rewrite the objective function more compactly as follows:

$$J(\mathbf{X}_{ur}) = \mathrm{vec}(\mathbf{X}_{ur})^T (\mathbf{K}_{ur} + \sum_{f=1,f\neq r,u}^{N} \mathbf{F}_{fr}^T \mathbf{K}_{uf} \mathbf{F}_{fr}) \mathrm{vec}(\mathbf{X}_{ur}) \quad (7)$$

It becomes clear that in each iteration the sub-problem (7) is a standard pairwise graph matching problem, which can be solved by various pairwise graph matching techniques based on the QAP formulation in an "out-of-the-box" manner[2]

So far, we have proposed our alternating optimization framework that directly deals with the affinity matrix. Note that in the above discussion, we assume the reference graph $\mathcal{G}_r$ and the updating order o_{udt} are both pre-given. We would still hold this assumption for presenting the factorized variant in the next paragraph. In the end of this section, we would address these two common problems using a principled paradigm. As we will show later, it is closely related to the graph-wise/pairwise consistency metrics as defined in Section III.

[1]For efficiency, we omit the terms regarding $\mathbf{K}_{fu}$ as they have been encoded by $\mathbf{K}_{uf}$. This policy is used thought the paper for the proposed algorithms.

[2]Several matching methods are *not* or *not explicitely* based on QAP: e.g. the graph edit distance based methods [53] and the recent advances [54], [55] can enjoy better error-tolerance against missing nodes and edges.

Algorithm 1 Consistency-driven Non-factorized Alternating Optimization for Multi-Graph Matching

Input:
N graphs with n nodes of each graph;
Pairwise affinity matrix $\mathbf{K}_{ij}(i,j = 1,\ldots,N)$;
Maximum iteration count: T_{max};
Output:
Consistent assignment matrix $\mathbf{X}_{ij}$ $(i,j = 1,\ldots,N)$;
Procedure:
Obtain the raw matching configuration $\mathbb{X} = \{\mathbf{X}_{ij}\}$ by exhaustively performing pairwise graph matching over N graphs;
Set reference graph $\mathcal{G}_r = \max_{\mathcal{G}_k} C_g(\mathcal{G}_k, \mathbb{X}), \{k = 1,\ldots,N\}$;
Set updating list O_{udt} in ascending order w.r.t. $C_p(\mathcal{G}_u, \mathcal{G}_r, \mathbb{X})$
Initialize the best solutions $\mathbf{X}_{kr}^* = \mathbf{X}_{kr}, k = 1,\ldots,N, k \neq r$;
for $t = 1 : T_{max}$ **do**
  for u in O_{udt} **do**
    Fix N-2 $\mathbf{X}_{uf}^t$, for $f = 1,\ldots,N, f \neq r, u$, update $\mathbf{X}_{ur}^t$ by solving the two-graph matching problem (7);
    Compute the updated objective score $J(\mathbf{X}_{ur})$ using the updated $\mathbf{X}_{ur}^t$ for the objective (7);
    Compute the so-far best objective score $J^*(\mathbf{X}_{ur})$ using the so-far-best $\mathbf{X}_{ur}^*$ for the objective (7);
    if $J_{ur} > J_{ur}^*$, $\mathbf{X}_{ur}^* = \mathbf{X}_{ur}^t$;
  end for
end for

Algorithm 2 Consistency-driven Factorized Alternating Optimization for Multi-Graph Matching

Input:
N graphs with n nodes of each graph;
Graph-wise node-edge incidence matrix $\mathbf{G}_i$, node and edge affinity matrix $\mathbf{K}_{ij}^p$, $\mathbf{K}_{ij}^q$ $(i,j = 1,\ldots,N)$;
Maximum iteration count: T_{max}, path-following step size δ
Output:
Consistent assignment matrix $\mathbf{X}_{ij}(i,j = 1,\ldots,N)$;
Procedure:
Obtain the raw matching configuration $\mathbb{X} = \{\mathbf{X}_{ij}\}$ by exhaustively performing two-graph matching over N graphs;
Set reference graph $\mathcal{G}_r = \max_{\mathcal{G}_k} C_g(\mathcal{G}_k, \mathbb{X}), \{k = 1,\ldots,N\}$;
Set updating list O_{udt} in ascending order w.r.t. $C_p(\mathcal{G}_u, \mathcal{G}_r, \mathbb{X})$
Initialize the best solutions $\mathbf{X}_{rk}^* = \mathbf{X}_{rk}, k = 1,\ldots,N, k \neq r$;
for $t = 1 : T_{max}$ **do**
  for u in O_{udt} **do**
    Fix N-2 $\mathbf{X}_{fu}^t$, for $f = 1,\ldots,N, f \neq r, u$, update $\mathbf{X}_{ru}^t$ by using the following path-following procedure:
    for $\alpha = 0 : \delta : 1$ **do**
      update $\mathbf{X}_{ru}^t$ via applying MFW on the convex-concave relaxation: $J = (1-\alpha)J^{vex} + \alpha J^{cav}$ by formula (10).
    end for
    Compute the updated objective score $J(\mathbf{X}_{ru})$ using the updated $\mathbf{X}_{ru}^t$ for the objective (9);
    Compute the so-far best objective score $J^*(\mathbf{X}_{ru})$ using the so-far-best $\mathbf{X}_{ru}^*$ for the objective (9);
    if $J_{ru} > J_{ru}^*$, $\mathbf{X}_{ru}^* = \mathbf{X}_{ru}^t$;
  end for
end for

B. Factorized multiple graph matching

Recent work [29] exploits the underlying structure of affinity matrix $\mathbf{K}$, which is not necessarily negative definite, in the hope of designing better optimization scheme for addressing the non-convex issue. The presented algorithm in the previous section cannot be directly applied to the original factorized graph matching formulation in [29] as the re-weighted affinity matrix in (7) lacks a convenient interpretation as the original affinity matrix that can be factorized (decoupled) into pairwise similarity and self-structure. In this section, we present an alternating optimization approach to solve the multiple graph

matching problem based on the factorized formulation. As pointed out by [29], the main advantage of factorized model is avoiding computing and storing the large-sized affinity matrix by decoupling it into smaller edge-wise and node-wise matrices as we have shown in the pairwise graph matching formulation. Another notable difference from our non-factorized multi-graph matching algorithm, is that it derives a specific convex-concave path-following algorithm as we will present in this section, while the former can re-use any non-factorized graph matching solvers [15], [16], [38], [39].

In general, the factorized multiple graph matching problem can be expressed by maximizing the following objective function, where J_{ij} is in the form of formula (2):

$$\begin{aligned} \mathbb{X}^* = \arg\max_{\mathbb{X}} \sum_{i,j=1,i\neq j}^{N} J_{ij}(\mathbf{X}_{ij}) \\ s.t. \quad \mathbf{X}_{ij}\mathbf{1}_n = \mathbf{1}_n \quad \mathbf{1}_n^T\mathbf{X}_{ij} = \mathbf{1}_n^T \quad \mathbf{X}_{ij} = \mathbf{X}_{ji}^T \in \{0,1\}^{n\times n} \\ \forall i,j = 1,\ldots,N; \quad |\mathbb{X}| = N(N-1) \end{aligned} \tag{8}$$

For the factorized graph matching formulation, we will directly use the matrix form $\mathbf{X}$ instead of the vectorized form $\mathbf{x} = \text{vec}(\mathbf{X})$ as used in the non-factorized case. As the same with the non-factorized multi-graph matching problem, a reference graph $\mathcal{G}_r$ is first selected, followed by an alternating optimization procedure with a certain sequential updating order. Similar to the non-factorized case, we reach the following objective function without using the affinity matrix:

$$J(\mathbf{X}_{ru}) = J_{ru}(\mathbf{X}_{ru}) + \sum_{f=1,f\neq r,u}^{N} J_{fu}(\mathbf{X}_{fu}) \tag{9}$$

Note J is a function of $\mathbf{X}_{ru}$ as $\mathbf{X}_{fu} = \mathbf{X}_{fr}\mathbf{X}_{ru}$ for fixed $\mathbf{X}_{fr}$. Its convex-concave relaxation form, which is weighted by $\alpha \in$[0,1], can be written as follows:

$$\begin{aligned} J^\alpha(\mathbf{X}_{ru}) =& (1-\alpha)\Big(J_{ru}^{vex}(\mathbf{X}_{ru}) + \sum_{f=1,f\neq r,u}^{N} J_{fu}^{vex}(\mathbf{X}_{fu})\Big) \\ &+ \alpha\Big(J_{ru}^{cav}(\mathbf{X}_{ru}) + \sum_{f=1,f\neq r,u}^{N} J_{fu}^{cav}(\mathbf{X}_{fu})\Big) \end{aligned} \tag{10}$$

where J^{vex} is written by formula (3), and J^{cav} by formula (4). As α increases from 0 to 1, this path-following strategy enjoys the global optimal solution when $\alpha = 0$ due to the convex form, and finally leads to an integer solution when $\alpha = 1$, for the concave form. Thus it is insensitive to the initial point, and usually no post-binarization is needed [56], [57] which otherwise may cause additional performance loss.

For each $\alpha \in [0,1]$, the sub-problem can be solved using the Frank Wolfe's algorithm (FW) [45], which is widely used constrained nonlinear programming. In implementation, we use the modified Frank Wolfe's algorithm (MFW) [58] to speed up its convergence which probably finds a better searching direction $\mathbf{Y}$ by a convex combination of previously obtained solutions. In the experiments, we used the directions computed in 2 previous steps. The FW (MFW) algorithm first computes the optimal direction $\mathbf{Y}$ and then determines the optimal step size $\lambda \in [0,1]$, with respect to the objective function (10). The optimal $\mathbf{Y}$ with respect to $\mathbf{X}_{ru}$ in iteration k can be calculated by solving the following linear programming problem by the Hungarian method [59]:

$$\begin{aligned} \max_{\mathbf{Y}} \quad & \text{tr}\Big(\nabla_{\mathbf{X}_{ru}} J^\alpha(\mathbf{X}_{ru}^k)^T(\mathbf{Y} - \mathbf{X}_{ru}^k)\Big) \\ s.t. \quad & \mathbf{Y}\mathbf{1}_n = \mathbf{1}_n, \mathbf{Y}^T\mathbf{1}_n = \mathbf{1}_n, \mathbf{Y} \geq \mathbf{0}_{n\times n} \end{aligned} \tag{11}$$

where the gradient can be obtained as follows:

$$\begin{aligned} \nabla_{\mathbf{X}_{ru}} J^\alpha =& (1-\alpha)\nabla_{\mathbf{X}_{ru}}(J_{ru}^{vex} + \sum_{f=1,f\neq r,u}^{N} J_{fu}^{vex}) \\ &+ \alpha\nabla_{\mathbf{X}_{ru}}(J_{ru}^{cav} + \sum_{f=1,f\neq r,u}^{N} J_{fu}^{cav}) \end{aligned} \tag{12}$$

Having obtained the optimal $\mathbf{Y}$, the optimal step size λ can be found at the optimal point of the following parabola:

$$\begin{aligned} J^\alpha =& (1-\alpha)J^{vex}(\mathbf{X}_{ru} + \lambda\mathbf{Y}) + \alpha J^{cav}(\mathbf{X}_{ru} + \lambda\mathbf{Y}) \\ =& a\lambda^2 + b\lambda + \text{const} \end{aligned} \tag{13}$$

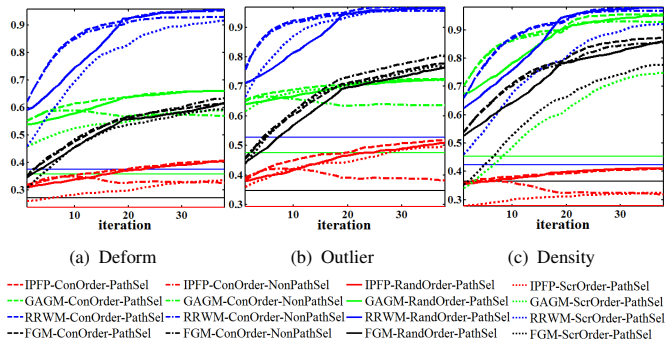
Next, we would focus on solving the two common issues for both non-factorized and factorized variants: i) finding the optimal reference graph $\mathcal{G}_r$ that induces the basis variable set $\{\mathbf{X}_{rk}, k = 1,\ldots,N, k \neq r\}$ to initialize the iterative optimization procedure; ii) setting the optimal updating order O_{udt} to speed up convergence and to suppress possible error propagation over alternating optimization. This is because although in each iteration the objective (7) or (9) contains affinities over multiple graphs to explore jointly, nevertheless, this new objective is coupled with the estimated $\mathbf{X}_{rf}$ from the previous iteration thus cannot fully avoid error propagation.

C. Consistency-driven alternating optimization

First we address the problem related to finding the reference graph. Our iterative alternating optimization involves a set of basis mappings between a reference graph and others in an expectation maximization manner. It is critical to generate reasonably accurate initial solutions to avoid trapping into an unsatisfactory local optimum. Specifically, one is given a raw matching configuration $\mathbb{X}^{raw}$ obtained from independent pairwise matching without knowing the ground truth. Under this condition, one is seeking an "appropriate" reference graph $\mathcal{G}_r$ and its associated basis set that span a new matching configuration $\mathbb{X}^{span}$ by $\mathbf{X}_{ij}^{span} = \mathbf{X}_{ri}^{rawT}\mathbf{X}_{rj}^{raw}$. One intuitive metric to approximate the accuracy is the graph-wise affinity score by $\sum_{k=1,k\neq r}^{N} \text{vec}(\mathbf{X}_{kr})^T\mathbf{K}_{kr}\text{vec}(\mathbf{X}_{kr})$ given $\mathcal{G}_r$. However, due to outliers and local deformation, as well as the difficulty in setting up the perfect affinity matrix in a parametric manner[3], the ground truth may not correspond to the highest score modeled by the parametric objective function. Thus graph-wise score is not a robust indicator to accuracy as shown in Fig.2, which will be discussed in details in our experiments.

Our embodiment is setting $\mathcal{G}_r = \max_{\mathcal{G}_k} C_g(\mathcal{G}_k, \mathbb{X})$ by Definition (1). The rationale is that the erroneous correspondences are at random yet correct correspondences concur thus consistent across pairwise matchings. This observation motivates us choosing $\mathcal{G}_r$ as the one maximizing the graph-wise consistency – Definition (1), in the hope that it would induce the most accurate basis set for initialization, or equally: $\min_r \sum_{i,j=1}^{N} \|\mathbf{X}_{ij}^{span} - \mathbf{X}_{ri}^{rawT}\mathbf{X}_{rj}^{raw}\|_p$, where $\|\|_p$ is the p-norm for a matrix. Since the resultant matching matrices are

[3]Currently the affinity function is mostly modeled by parametric functions, the fixed parameters by whatever manual setting [16], or learned from training samples [17], [36] etc. are still unable to perfect fit the score with accuracy, leading to the existence of discrepancy between score and accuracy.



(a) Deform (b) Outlier (c) Density

Fig. 2: Comparison of four strategies for setting reference graph, updating order and enforcing score-non-descending path selection constraint, refer to Table I and Table II for structured descriptions. Accuracy over iteration path by four solvers in different colors: RRWM (blue), GAGM (green), FGM (black), IPFP (red). The corresponding four horizontal solid lines in each plot indicate the performance of a baseline by randomly choosing a graph as the reference one to derive consistent matching configuration. The curves are generated by averaging 50 random tests over $T_{max} = 2$ iterations, i.e. the number of inner iteration rounds is $(N-1) * T_{max}$=38. Zoom in for better display.

TABLE I: Settings of reference graph, alternating updating order, and path selection strategy for the solution paths as illustrated in Fig.2.

reference graph	updating order	path selection	curve
graphwise consistency	pairwise consistency	enforced	dashed
graphwise consistency	random order	enforced	solid
graphwise consistency	pairwise consistency	non-enforced	dash-dot
graphwise score	pairwise score	enforced	dotted

all binary ones, thus the setting of the reference graph is insensitive to p. Without loss of generality, we set $p = 2$.

Given the basis variable set for rotating updating, the next problem is finding an "optimal" updating order. The pairwise consistency $C_p(\mathcal{G}_u, \mathcal{G}_r, \mathbb{X})$ – Definition (2) is used to define the order in which the sample graphs in the iterations are considered, so that the least accurate ones are updated first, thus avoiding error propagation and speeding up the convergence. Motivated by the similar observation in the case of finding the reference graph, the pairwise consistency measure is considered to approximate the pairwise matching accuracy. As a result, the updating variables are ranked in ascending order with respect to $C_p(\mathcal{G}_u, \mathcal{G}_r, \mathbb{X})$. Similarly, $C_p(\mathcal{G}_u, \mathcal{G}_r, \mathbb{X})$ is also insensitive to which matrix norm $\|\|_p$ (we set $p = 2$ here) is used as only binary matching matrices are involved.

Note the authors in [31], [33] also define an inconsistency index for the *overall* pairwise matching configuration $\mathbb{X}$, or, *multiple isomorphism* as termed in their papers. However, that index is not related to the fine-grained graph-wise/pairwise consistency, and unable to define the reference graph and updating order. Perhaps more importantly, the two proposed definitions in general provide the sketch for deriving other consistency metrics, which can be tailored in particular applications. For instance, the graph-wise consistency can be calculated by weighting each term $\|\mathbf{X}_{ij} - \mathbf{X}_{ir}\mathbf{X}_{rj}\|$ in Definition (1) by a parameter w_{ij} if the prior estimation on the quality of the initial pairwise matching solution is given. In summary, the presented two measures themselves already bear some generalities since the binary matrices are insensitive to the choice of p-norm and they can allow weighted variants as discussed above. Moreover, our general consistency-driven mechanism can also benefit from other newly designed metrics.

TABLE II: Parameter settings for solution path illustration as shown in Fig.2.

noise type	parameter settings	results
deform	N=20,n_{in}=12,n_{out}=0,ε=.15,ρ=1,σ^2=.05	Fig.2(a)
outlier	N=20,n_{in}=8,n_{out}=4,ε=.05,ρ=1,σ^2=.05	Fig.2(b)
density	N=20,n_{in}=12,n_{out}=0,ε=.05,ρ=.5,σ^2=.05	Fig.2(c)

D. Implementation details and convergence discussion

Another consideration is due to the sub-optimality nature of existing pairwise matching techniques for solving the NP-hard sub-problem of Eq.(7) or (9) per iteration. Therefore the newly obtained solution cannot guarantee to achieve higher affinity score than the previous one, although often empirically observed. To avoid the score degenerating case, we suggest enforce a score-non-descending path selection strategy by comparing the objective score calculated by the new solution $\mathbf{X}_{ur}^t$ and the one from the currently maintained solution $\mathbf{X}_{ur}^*$. The new solution would be dropped if it decreases the score. As such, we obtain a score-non-descending solution path. When this strategy is not imposed, $\mathbf{X}_{ur}$ would always be overwritten. We would compare both the "path selection" and "non-path selection" strategies in our experiments.

Now, we summarize the above discussion and analysis into two variants of our alternating optimization algorithms, which are depicted in Alg.1 and Alg.2 for the non-factorized and factorized model respectively. The score-non-descending path selection step is plugged in the two algorithm charts.

Fig.2 depicts a comparison under different settings by using the proposed framework, regarding with the way of setting i) the reference graph, ii) the updating order and iii) the path selection strategy. Four pairwise matching solvers are used to study if the solution path is impacted by the specific pairwise matching technique or if a more general pattern exists. We will give a detailed analysis to Fig.2 later in our experiments.

In the last, we provide a general discussion about the convergence behavior of the proposed methods. Both proposed algorithms involve updating a set of basis assignment solutions of length N: $\{\mathbf{X}_{rk}^t, k = 1, \ldots, N, k \neq r\}$ w.r.t. the reference graph $\mathcal{G}_r$ in iteration t. Now we concatenate all individual $\mathbf{X}_{rk}^t$ into a single matrix $\boldsymbol{X}_r^t = [\mathbf{X}_{r1}^t, \mathbf{X}_{r2}^t, \ldots, \mathbf{X}_{rN}^t]$. Since each $\mathbf{X}_{rk}^t$ is a discrete binary permutation matrix thus $\boldsymbol{X}_r^t$ is also exhaustively enumerable. As a result, as iteration continues, there exist t and s ($t < s$), such that the t-th iteration solution $\boldsymbol{X}_r^t$ would equal to a previous value $\boldsymbol{X}_r^s$ (Otherwise it implies the solution space is innumerable thus contradicts with the fact). Furthermore, because both algorithms are deterministic, thus the iterative solution path would restart the solution path: $\mathbf{X}_{rk}^t \rightarrow \mathbf{X}_{rk}^s = \mathbf{X}_{rk}^t \rightarrow \mathbf{X}_{rk}^s \rightarrow \cdots$ in a looping manner. Therefore, both algorithms would converge to a looping sequences when $t < s$ or to a fixed point when $t = s$.

V. Experiments and Discussion

The experiments are performed on both synthetic and real-image datasets. The synthetic test is controlled by quantitatively varying the disturbance of deformation, outlier and edge density. The real-image datasets are tested with varying viewing angles, scales, shapes, and spurious outliers.

The matching accuracy over all graphs, is calculated by averaging all pairwise matching accuracy $\frac{\sum_{i=1}^{N-1}\sum_{j=i+1}^{N}\text{Acc}_{ij}}{N(N-1)/2}$. Each Acc_{ij} computes the matches between the correspondence matrix $\mathbf{X}_{ij}^{alg}$ given by the ground truth $\mathbf{X}_{ij}^{tru}$: $\text{Acc}_{ij} = \frac{\text{tr}(\mathbf{X}_{ij}^{alg}\mathbf{X}_{ij}^{tru})}{\text{tr}(\mathbf{1}_{n_j\times n_i}\mathbf{X}_{ij}^{tru})}$. Note we only calculate the accuracy for common inliers and ignore the matching results over outliers.

The above testing methodologies follow a standard protocol widely adopted by many related works such as [15], [16], [29].

A. Dataset description and affinity setting

Synthetic data The synthetic test follows the widely used protocol of [1], [14], [15], [16], [29], [38]. For each trial, a reference graph with n_{in} nodes is created by assigning a random attribute to its edge, which is uniformly sampled from the interval [0,1]. Then the "perturbed" graphs are created by adding a Gaussian deformation disturbance to the edge attribute q_{ij}^r, which is sampled from $N(0,\varepsilon)$ i.e. q_{ij}^p = q_{ij}^r+$N(0,\varepsilon)$ where the superscript 'p' and 'r' denotes for "perturb" and "reference" respectively. Each "perturbed" graph is further added by n_{out} outliers, which can also be helpful to make the graphs of equal sizes when the input graphs are different sizes. Its edge density is controlled by the density parameter $\rho \in [0,1]$ via random sampling. The edge affinity is computed by $K_{ij,ab} = \exp(-\frac{(q_{ij}-q_{ab})^2}{\sigma^2})$ where σ^2 is the edge similarity sensitivity parameter. No single-node feature is used and the unary affinity $K_{ii,aa}$ is set to zero.

CMU-POSE Sequence This data contains three sequences. The first two are from the CMU house (30 marks, 101 frames), hotel (30 marks, 111 frames) sequence (http://vasc.ri.cmu.edu//idb/html/motion/) which are widely used in [1], [15], [16], [17], [29], [36]. The third sequence is sampled from the sedan (VolvoC70) sequence (19 marks, 225 frames) which is viewed from various angles covering a range of 70 degrees from the POSE dataset [60]. For each test, an image sequence is sampled which is spaced evenly by three frames. We utilize this dataset for the outlier test. We select n_{in}=10 landmarks out of all n_{ant} annotated points (n_{ant}=30 for the two CMU sequences, n_{ant}=19 for Pose sedan sequence), and randomly chose n_{out}=4 nodes from the rest $n_{ant}-n_{in}$ nodes as outliers. The graphs are fully connected to encode all information to suppress outliers.

The affinity matrix is constructed by the edge length similarity $K_{ij,ab}^{len} = \exp(-\frac{(d_{ij}-d_{ab})^2}{\sigma^2})$ where d_{ij}, d_{ab} are the Euclidean distance between two points that are further normalized to [0,1] by dividing the largest edge length. The unary affinity is also set to zero in line with [15], [16], [29].

WILLOW-ObjectClass The object class dataset released in [36] is constructed by images from Caltech-256 and PASCAL VOC2007. Each object category contains different number of images: 109 Face, 50 Duck, 66 Wine bottle, 40 Motorbike, and 40 Car images. For each image, 10 feature points were manually labeled on the target object. The edge sampling for the affinity matrix follows the same way as [29] by constructing the sparse delaunay triangulation among the marks, since the triangulation can efficiently encode the object structure when no outlier exists. For defining the edge affinity, we follow the protocol of [29], [36] that set the final affinity matrix re-weighted by the edge length affinity and angle affinity: $K_{ij,ab} = \beta K_{ij,ab}^{len} + (1-\beta)K_{ij,ab}^{ang}$, where $\beta \in [0,1]$ is the weighting parameter. This is because this dataset is more geometrically ambiguous than the sequence data if only edge length is used. The angle for each edge is computed by the absolute angle between the edge and the horizontal line as used in [29]. The edge affinity and angle affinity are calculated in the same way in the CMU-POSE test. For node-wise affinity, we use the 128-dim SIFT feature [61] $\mathbf{s} \in \mathbb{R}^{128}$ associated with each single landmark by $K_{ia;ia} = \exp(-\|\mathbf{s}_i - \mathbf{s}_a\|_2)$.

B. Comparing methods

First we give a short description for the used pairwise matching solvers, which serve as building-blocks for our and other multi-graph matching algorithms [32], [33]. Other state-of-the-art multi-graph methods [31], [32], [33] are briefly described in the sequel. The tests run on a laptop with 2.9G Intel Core I7 and 8G memory with a single thread. The code of the comparing methods are all from original authors, apart from [14], which is implemented by the author of [29].

We select the following widely used pairwise graph matching methods as the pairwise solvers in our methods. In particular, the parameter settings of these methods as reported below mostly follow the original settings from the authors. We further slightly tune the parameters to balance efficiency and efficacy as matching multiple graphs is more time-consuming.

Graduated assignment (GAGM) GAGM [14] performs gradient ascent on a relaxed QAP objective driven by the deterministic annealing procedure. The relaxation level is controlled by a continuation parameter β, which is updated by $\alpha\beta_t \rightarrow \beta_{t+1} \leq \beta_{max}$. We set α=1.1, β_0 = 0.5 and β_{max}=50. In the CMU-POSE data test, we also replace GAGM with its variant, i.e. Soft Constrained Graduated Assignment [16] which is a recent improvement based on GAGM.

Integer projected fixed point method (IPFP) IPFP [39] approximates the original objective function via transforming it into a series of linear assignments over iterations, and in each iteration finds the optimal solution in the discrete domain via Hungarian method. Then the method finds the re-weighted solution in the continuous domain along the direction from the current solution to the optimal discrete solution. We set the maximum round of iterations as 10 in all experiments, which is in accordance with the empirical observation by [39].

Re-weighted random walk matching (RRWM) RRWM [15] introduces a random walk view on the problem and to some extent can be regarded as a re-weighted version for GAGM [14] and Spectral Matching [4] methods. We fix its parameters α=0.2 and β=30 in all experiments.

Fast Bipartite graph matching (FBP) FBP [54] is a recently proposed graph edit distance based method. It approximates the second-order graph matching problem to a first-order assignment problem by considering the local edge structure via fast bipartite matching. It requires to build the node-to-node cost matrix $\mathbf{C} \in \mathbb{R}^{n\times n}$ which encodes both node-wise and local structure information around each node. However, in each iteration of our framework, only the (re-weighted)

affinity matrix $\mathbf{K}$ is given while the attribute matrix (this paper is dealing with attributed graphs) for each graph is unknown and cannot be recovered from $\mathbf{K}$. Thus the attribute/adjacency matrix based steps for computing $\mathbf{C}$ as used in [54], [55] are inapplicable in our framework. To compute the element $\mathbf{C}_{ia}$ when only $\mathbf{K}$ is given, we employ the Hungarian method to compute the assignment cost for the rest of nodes, by fixing the mapping $i \to a$. Specifically, the input to the Hungarian method is the row/column of $\mathbf{K}$ (by inverting the value so as to change it from affinity to cost) that can be re-shapen into the cost matrix w.r.t. $i \to a$ i.e. vec2mat($\max(\mathbf{K}) - \mathbf{K}_{ia,:}$) – see Fig.1(a) for illustration. Note this adaption may weaken the performance of FBP which is originally designed for effective exploring the individual graph attribute/adjancey information.

Factorized graph matching (FGM) FGM [29] factorizes the affinity matrix which allows for relaxing the objective to a convex function and a concave one respectively. We set the increment step $\delta = 0.2$ for the path-following algorithm, and the iteration count in the Modified Frank-Wolfe method as 5.

On the other hand, the following peer multi-graph matching methods are evaluated and compared with our methods:

Graduated assignment based common labeling (GACL) GACL [48], [31] is a more recent work that extends the graduated assignment method to solve the multi-graph matching problem, by matching all graph nodes to a virtual node set. It continuously updates the set of probabilities for the mapping between the nodes in each graph to the virtual node set by optimizing the objective function until converge. The final solution is consistent and discrete. Here we use the exponential objective function as the same form with ours, but inverse affinity to cost $\mathbf{K}^{gacl} = \max(\mathbf{K}^{ours}) - \mathbf{K}^{ours}$ element-wise, as GACL concerns with minimizing the cost function. We set the parameter in GACL (refer to the original paper) $\beta_0 = .5$, $\beta_{max} = 20$, $\alpha = 1.1$ for the annealing outer loop $\alpha\beta_t \to \beta_{t+1} \leq \beta_{max}$, and set the maximum iteration round of the inner loop $T_{in} = 5$ for cost efficiency.

Modified hype-cube based common labeling (MHCL) the original hypercube framework is used in [33], [46] while specifically tested in the three-graph matching case, due to exponential memory overhead. The algorithm has two main steps: first, it performs a pairwise matching via existing pair solvers and averages either discrete assignment solutions or probabilistic ones into an N-dimensional assignment hypercube, as an extension to the permutation matrix for two graphs. Each element of the hypercube represents the matching probability for the tuple of N nodes from N graphs respectively. Second, a post-binarization step is applied to to obtain the consistent common labeling. While the original hypercube data structure can only work with a very small number of graphs, not scalable to large N due to its memory space overhead is n^N. Thus we modify the original method to a more light-weighted one including the following four steps: i) adopt a pairwise graph matching solver to obtain all possible pair assignment solutions $\mathbf{X}_{ij}^0$; ii) re-calculate the assignment matrix $\mathbf{X}_{ij}$ by averaging the indirect mapping $\mathbf{X}_{ij} = \sum_{k=1}^{N} \mathbf{X}_{ik}\mathbf{X}_{kj}/N$; iii) use Hungarian method to binarize $\mathbf{X}_{ij}$; iv) use the SYNC method (when $N > n$) or Maximum Spanning Tree (MST) [62] over the raw configuration[4] (when $N \leq n$) to obtain consistent matchings.

Permutation synchronization (SYNC) SYNC [32] employs spectral analysis and approximation to eigenvector decomposition on the matching configuration matrix comprised of all initial pairwise matching solutions, and recover the consistent matching solutions. Note that the SYNC method only can work when the number of nodes is less than the number of graphs, thus similar to the sub-step in the MHCL method, Maximum Spanning Tree (MST) is used when $n \leq N$.

C. Incorporating both pairwise and multi-matching solvers

Having described the above multi-graph matching methods, we provide a further comparison for the advantage of our framework, which covers two aspects: i) *better* reusing various pairwise matching techniques in an "out-of-the-box" fashion; ii) proceeding with the output of other multi-graph matching methods to improve the solution.

For the first aspect, our framework not only leverages the pairwise matching solver to compute the putative matchings in the first stage for initialization and consistency estimation, but also reuses the solver during the iterative alternating optimization. In this second stage, the affinity matrix is renewed, containing the global information beyond two graphs. In contrast, the peer methods MHCL/SYNC use the pairwise matching technique in one-shot: only for the purpose of obtaining the initial pairwise matching configuration, while the post-procedure is immune from any pairwise matching solver. For the above reason, in the following plots when our methods are tested, we use the naming convention "X-Y" to term the different pairwise matching solvers used in two stages. While the compared other multi-graph matching methods are set to always use the first stage solver as termed by "X".

For the second aspect, the proposed framework can either use the proposed reference graph driven mechanism, or reuse any other multi-graph matching methods to generate an initial matching configuration $\mathbb{X}$. As we will show later, these two approaches perform similarly in accuracy, while using the reference graph based initialization only involves one-shot O(N^3) times of *multiplication of permutation matrix* which can be very fast, compared with the spectral analysis based method SYNC related to *SVD computing* with the complexity of $O(N^2n^3)$ and the tedious three-layer looping procedure for GACL whose time complexity would be further discussed later in this section. Moreover, SYNC/MHCL are both unable to proceed their processing given the output of our algorithms, because they require the input solutions are inconsistent and then transform the inconsistent solution to a consistent one.

To make the plots in the rest of the paper more digestible, the methods with their acronyms are listed for cross-reference.

- **PAIR** The baseline which performs exhaustive **PAIR**wise graph matching over the whole graph set. Note the consistency over pairwise matchings is not guaranteed;
- **AREF** Abbreviated for **A**daptive **REF**erence Graph Selection, which consists of three steps: i) perform exhaustive pairwise matching; ii) select the most "consistent"

[4]The raw configuration induces a fully connected *super graph* for each node is a graph, and the edge is the pairwise matching score. A Maximum Spanning Tree (MST) can be found which spans a new consistent configuration [47].

graph as the reference $\mathcal{G}_r$ according to the metric of graph-wise consistency; iii) use its pairwise matching solutions $\mathbf{X}_{rk}$ with other graphs to span the new pairwise matching solutions by $\mathbf{X}_{ij} = \mathbf{X}_{ri}^T \mathbf{X}_{rj}$;

- **FREF** Abbreviated for **F**ixed (random) **REF**erence Graph Selection, which replaces the first two steps of AREF by randomly selecting a reference graph while keeping the third step the same. It is very efficient since only a linear number of pairwise matchings are performed w.r.t. the number of graphs N;
- **GACL** Abbreviated for **G**raduated **A**ssignment based **C**ommon **L**abeling as proposed in [31], [48];
- **SYNC** Abbreviated for **SYNC**hronization in [32];
- **MHCL** Abbreviated for **M**odified **H**ype-Cube based **C**ommon **L**abeling, which is introduced earlier in this paper as an modified version of [33], [46] to handle the memory bottleneck when more graphs are involved;
- **FREF+/GACL+/SYNC+/MHCL+** Perform Alg.1/2 using FREF/GACL/SYNC/MHCL's output as initial input. The difference to Alg.1/2 is setting $\mathbf{X}_{kr}^*$ in step (8) as the solutions from other methods instead of pairwise matchings. All "plus-sign" cases also use the consistency-driven method to set the reference graph and updating order, and this can be done in a free-rider fashion since other methods also require computing pairwise matchings. It is worth noting that FREF+ in fact corresponds to the method in our conference version [1] since either the reference graph or the updating order is randomly set;
- **ALG$_2^1$** One of our two methods, being Alg.1 when non-factorized graph matching solvers are adopted including RRWM/IPFP/GAGM, or Alg.2 when factorized solvers are used like FGM. Note our method in fact can also be termed as AREF+ based on the above naming convention. This is because the result of AREF in fact initialize both Alg.1 and Alg.2 in step (8). For this reason, in the related plots, AREF and ALG$_2^1$ are in the same red color;
- **ALG$_2^1$*** same as ALG$_2^1$, but does not enforce the score-non-descending path selection strategy.

D. Time complexity analysis

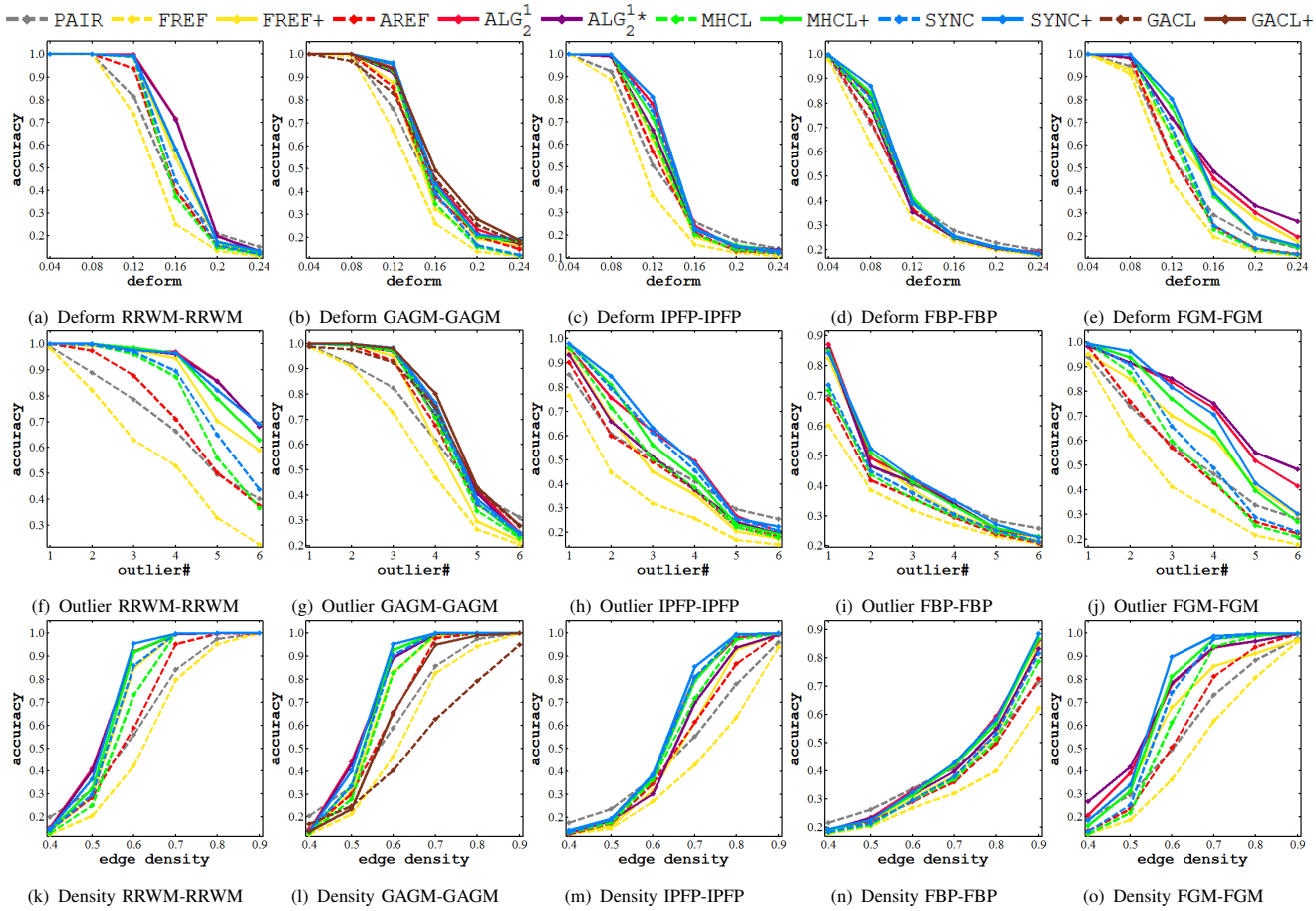
Since the existing multi-graph matching methods SYNC [32], MHCL [33], [46] start with the pairwise matching solutions, and our methods also use pairwise matching solvers for initialization and optimization. Thus we will consider a specific pairwise matching solver and estimate the time complexity when applying it for the different methods.

Non-factorized multi-graph matching Without loss of generality, we choose RRWM [15] as the pairwise matching solver, and study the overall time complexity when applying it in our non-factorized method for Alg.1. RRWM involves an iterative procedure, in each iteration, the power iteration method [63] and Sinkhorn method [64] are employed. The cost of the former is $O(m^2)$ and the latter, which is denoted as $\tau_{\text{Sh}} = O(n^2)$ largely depends on the convergence speed, thus we have $\tau_{\text{rrwm}} = T_{\text{rrwm}}(O(m^2) + \tau_{\text{Sh}})$ where T_{rrwm} refers to the number of iteration rounds. For our method, first RRWM is performed on each pair of graphs whose cost is $O(N^2 \tau_{\text{rrwm}})$, then the cost of setting the reference graph and updating order is $O(N^3 n^3)$ where $O(n^3)$ refers to the permutation matrix multiplication that in fact can be signification speeded up at the code level typically in a linear time w.r.t. n. In each iteration, as shown in formula (7), it adds up $N-2$ indirect pairwise affinities in addition with the direct affinity objective. Building this whole objective function as the input to RRWM costs $O(Nn^4)$ per iteration. The whole overhead is $N(\tau_{\text{rrwm}} + O(Nn^4)) + O(N^3 n^3) + N^2 \tau_{\text{rrwm}}$ as our method stops when all assignment matrix are updated for one time, i.e. after $N-1$ round of iterations. The time complexity estimation for applying other pairwise graph matching solvers including IPFP/GAGM can also be derived in a similar manner.

We further study the time complexity of other peer multi-graph matching methods. i) SYNC: it involves a one-shot SVD to find the n leading eigenvectors on the grouped assignment matrix of size $nN \times nN$, whose complexity is $O(N^2 n^3)$, and N iterations of Hungarian method with cost $O(N\tau_{Hun}) = O(Nn^3)$. Thus the total cost is $\tau_{\text{sync}} = O(N^2 n^3) + N^2 \tau_{\text{rrwm}}$; ii) MHCL: it performs the Hungarian method to binarize each matrix which is averaged by the putative assignment matrices from $\frac{N(N-1)}{2}$ pairwise matchings. The cost is $O(N^2 n^3) + O(N^3 n^3)$, where the first term is concerned with the Hungarian method, and the second is related to the averaging step involving N^3 times of permutation matrix multiplication, which can usually be optimized to $O(N^3 n)$ at the code level. Then SYNC is performed to obtain consistency. Thus its time complexity is $O(N^3 n^3) + N^2 \tau_{\text{rrwm}}$; iii) GACL: according to the analysis by the authors in [31], its complexity is $O(N^2 m^2) + \tau_{\text{Sh}}$ per iteration which is further surrounded by two layers of loops. Specifically, the outer loop concerns a continuation method which gradually increases the control parameter β. The inner loop concerns iteratively updating the assignment matrix towards the virtual node set by given a fixed β. Thus the total complexity is $T_{\text{ga}} O(N^2 m^2)$ where T_{ga} is the total number of iterations over two layers of loops, which in fact is significantly larger than N or n.

Factorized multi-graph matching Due to currently we are unable to identify any other factorized multi-graph matching method to our best knowledge, thus we only present the result for our method. Specifically, Besides the step for initial pairwise matching, reference graph and updating order setting, Alg.2 further involves i) factorizing the $N(N-1)$ pairs of graphs that involves SVD decomposition at the cost of $O((n+m)^3)$ per each; and ii) using the MFW method used in each iteration that includes calculating the gradient of the objective function $\nabla^2_{\mathbf{X}_{ur}} J_{ij}$ at the cost of $O((n+m)^2)$ for each pair (i,j), which is repeatedly performed over $N-1$ terms $\sum_{f=1,\neq r,u}^{N} J_{fu} + J_{ru}$ in the objective function; iii) performing the Hungarian method at the cost of $O(n^3)$ to obtain the optimal line search direction $\mathbf{Y}$ given the calculated gradient $\nabla^2_{\mathbf{X}_{ur}} J$; iv) the step-size search given the optimal search direction that is at the same cost of computing gradient. Thus the total complexity is $O(N^2(n+m)^3) + T_{\text{fw}} O(N((n+m)^2 + n^3) + N^3 n^3 + N^2 \tau_{\text{rrwm}}$ where T_{fw} is the number of iterations for the MFW method.

Table III summarizes the time complexity by the general theoretical analysis. In general, Alg.1 would be relatively more



(a) Deform RRWM-RRWM (b) Deform GAGM-GAGM (c) Deform IPFP-IPFP (d) Deform FBP-FBP (e) Deform FGM-FGM

(f) Outlier RRWM-RRWM (g) Outlier GAGM-GAGM (h) Outlier IPFP-IPFP (i) Outlier FBP-FBP (j) Outlier FGM-FGM

(k) Density RRWM-RRWM (l) Density GAGM-GAGM (m) Density IPFP-IPFP (n) Density FBP-FBP (o) Density FGM-FGM

Fig. 3: Accuracy on synthetic dataset by varying the disturbance level. The pairwise matching solvers are the same for initial pairwise matching and alternating optimization.

TABLE III: Time complexity comparison, where $\tau_{\text{rrwm}} = O(m^2 + n^2)$.

method	time complexity
Alg.1	$O(N^2n^4 + N^3n^3) + N^2\tau_{\text{rrwm}}$
Alg.2	$O(N^2(n+m)^3 + N^3n^3) + N^2\tau_{\text{rrwm}}$
SYNC [32]	$O(N^2n^3) + N^2\tau_{\text{rrwm}}$
MHCL [33]	$O(N^3n^3) + N^2\tau_{\text{rrwm}}$
GACL [31]	$T_{\text{ga}}O(N^2m^2)$

efficient when N is large and n small. Alg.2 can be accelerated when there are only a few edges. Fig.8 depicts a more specific comparison where one can find our methods bring slightly more overhead compared with other methods when non-factorized model is used (RRWM/GAGM/IPFP/FBP). While the factorized variant FGM is even more costive.

E. Results and further discussion

First, we use Fig.2 to compare our graph-wise/pairwise consistency-driven strategies for setting the reference graph and the alternating updating order, with the score-driven alternative and the baseline by using a random updating order. Moreover, the impact of score-non-descending path selection strategy is also tested. In terms of the score-driven reference graph selection, the graph-wise score is defined as $S_g(\mathcal{G}_r, \mathbb{X}) = \sum_{k=1,k\neq r}^{N} \text{vec}(\mathbf{X}_{kr})^T \mathbf{K}_{kr} \text{vec}(\mathbf{X}_{kr})$; regarding the score-driven updating order setting, the pairwise-score for each graph $\mathcal{G}_u$, to $\mathcal{G}_r$, is defined by $S_p(\mathcal{G}_u, \mathcal{G}_r, \mathbb{X}) = \text{vec}(\mathbf{X}_{ur})^T \mathbf{K}_{ur} \text{vec}(\mathbf{X}_{ur})$.

To comprehensively study the behavior of different strategies, the four pairwise matching solvers tested in the paper (RRWM/IPFP/GAGM/FGM) are tested and plot in different colors in Fig.2. Each curve is an average over 50 random synthetic trials that involves 20 graphs by adding three types of random noises: i) deformation noise on the edge attributes; ii) a half ratio of random outliers; and iii) down-sampled edge density, respectively. Readers are referred to and Table I and Table II for the details of the testing settings. We present several observations together with our analysis as follows:

i) As the iteration continues, accuracy grows especially in the first half rounds of iterations i.e. iter#= $N-1$. Note this is the first time when the algorithm finishes the updating for all $N-1$ basis matchings $\mathbf{X}_{rf}$. Specifically, RRWM/FGM

TABLE IV: Parameter settings for synthetic test in Fig.3, Fig.4. The settings in Fig.5 are the same with Fig.4 for deformation, outlier and density tests.

x-axis	fixing parameter	results
ε=.04-.24	N=20,n_{in}=12,n_{out}=0,ρ=1,σ^2=.05	Fig.3(abcde)
n_{out}=1-6	N=20,ε=.05,n_{in}=8,ρ=1,σ^2=.05	Fig.3(fghij)
ρ =.4-.9	N=20,n_{in}=12,n_{out}=0,ε=.05,σ^2=.05	Fig.3(klmno)
N=4-32	n_{in}=10,n_{out}=0,ε=.15,ρ=1,σ^2=.05	Fig.4(abcde)
N=4-32	n_{in}=6,n_{out}=4,ε=.05,ρ=1,σ^2=.05	Fig.4(fghij)
N=4-32	n_{in}=10,n_{out}=0,ε=0,ρ=.5,σ^2=.05	Fig.4(klmno)

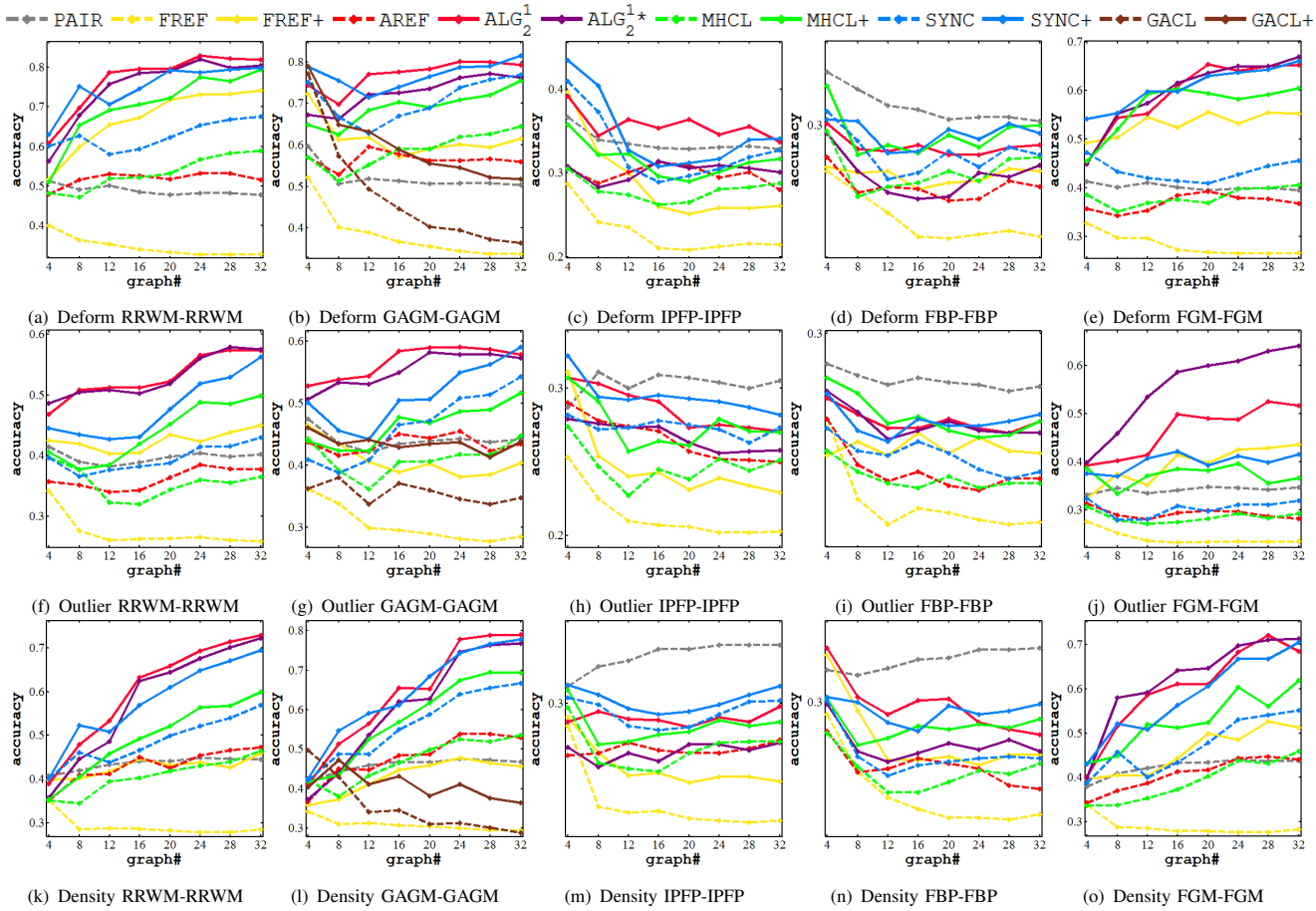


Fig. 4: Accuracy on synthetic dataset by varying the graph number. The pairwise matching solvers are the same for initial pairwise matching and alternating optimization.

significantly lift the accuracy which is initialized by the basis set induced by the selected reference graph. In contrast IPFP/GAGM are less effective in seeking quality solutions over iterations. This is consistent with their proved capability by [15], [29] in the context of pairwise graph matching.

ii) Regarding the proposed consistency-driven optimization method, the path selection strategy (dashed curve) in general improves the robustness and in particular avoids the degeneration cases when IPFP/GAGM are used. Nevertheless, as for FGM, enforcing path selection leads to slightly worse results compared with no path selection constraint is enforced. This also suggests IPFP/GAGM are less robust in finding a quality solution path thus require additional score-non-descending constraint, which meanwhile reduces its exploration capability.

iii) The graph-wise consistency-driven mechanism of setting the reference graph for initialization (dashed curve), show advantages to the random selection baseline (horizontal solid curve), as well as the score-driven approach (dotted curve).

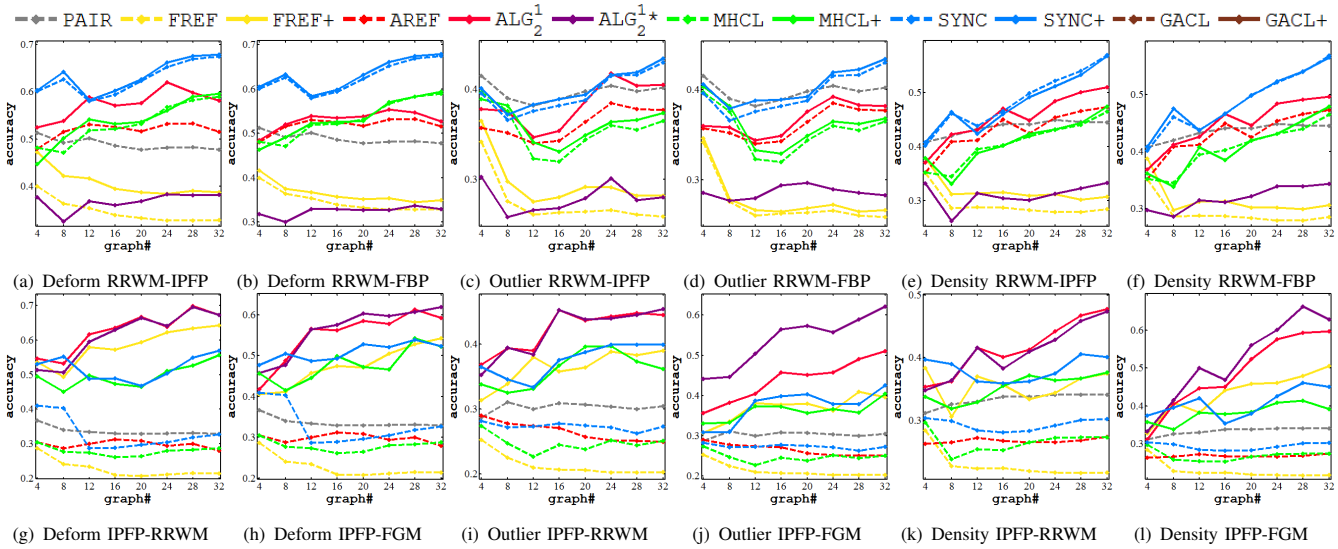
iv) The pairwise consistency-driven mechanism for setting the updating order, speeds up the performance improvement over iterations. The score-driven method exhibits similar speeding up pattern (dotted curve), while the former curve is more steep in general. In contrast, the random updating order (solid curve) results in a linear improvement in the first half of iterations, and all methods slow down their accuracy-climbing trends in the second half of iterations. Thus in the following experiments, we employ the graph-wise/pairwise consistency mechanisms and fix the iteration parameter $T_{max} = 1$ for cost-efficiency. In another word, all $\mathbf{X}_{rk}$ are updated once.

In addition, extensive evaluations are performed on the synthetic test by varying the noise level (Fig.3) and graph number (Fig.4, Fig.5), and on the real image by varying the graph number (Fig.6, Fig.7). The time costs on several tests are depicted in Fig.8. The parameter settings of these tests can be found in Table IV and Table V for synthetic test and real image test. Further discussions are presented as follows:

i) In Fig.3 we present our synthetic evaluation by varying the noise level related to deformation, outlier and edge density. On one hand, as the noise grows, our method consistently performs competitively and boosts performance in two cases: a) used as an independent solver (red & purple solid curves); b) coupled with other solvers by using their output as the initial input (solid curve in corresponding colors whose names

TABLE V: Parameter settings for real-image test in Fig.6 and Fig.7. The settings in Fig.8 regarding time cost are the same with Fig.6 and Fig.7 which show accuracy.

object	graph #	σ^2	β	$n_{in,out}$	edge ρ	results
car	4-24	0.1	.9	10,0	delaunay	Fig.6(abcde)
duck	4-24	0.1	.9	10,0	delaunay	Fig.6(fghij)
bike	4-24	0.1	.9	10,0	delaunay	Fig.6(klmno)
hotel	4-24	.05	0	10,4	full	Fig.7(abcde)
house	4-24	.05	0	10,4	full	Fig.7(fghij)
sedan	4-25	.05	0	10,4	full	Fig.7(klmno)



(a) Deform RRWM-IPFP (b) Deform RRWM-FBP (c) Outlier RRWM-IPFP (d) Outlier RRWM-FBP (e) Density RRWM-IPFP (f) Density RRWM-FBP

(g) Deform IPFP-RRWM (h) Deform IPFP-FGM (i) Outlier IPFP-RRWM (j) Outlier IPFP-FGM (k) Density IPFP-RRWM (l) Density IPFP-FGM

Fig. 5: Accuracy on synthetic dataset by varying the graph number. The pairwise matching solvers are different: the former for initial matching, the latter for alternating optimization.

end with a '+' sign). The observations from Fig.2 is cross-verified here: IPFP/GAGM are less effective in lifting the initial input, so for the newly added solver FBP. While more notable improvements are attained by RRWM/FGM. On the other hand, our non-factorized model (IPFP/GAGM/RRWM) that enforces the score-non-descending path selection strategy (ALG^1 in red) performs similarly with the counterpart that does not enforce such a strategy (ALG^{1*} in purple). Especially when IPFP is used, ALG^1 outperforms notably to ALG^{1*}. However, when the factorized model (FGM) is used, the score-non-descending path selection strategy affects the performance adversely: ALG$_2$ in red is worse than ALG$_2*$ in purple, especially when heavy noises are imposed. This is perhaps due to the side-effect of the conservative score non-descending strategy that blindly confines the effective exploration capability of FGM especially when the objective score deviates from semantic similarity due to large noises.

ii) Fig.4 and Fig.5 are used to evaluate the impact when a growing number of graphs are involved. When plugged with RRWM/GAGM/FGM, our methods tend to exhibit a syngeneic effect as the number of graphs increases. While IPFP/FBP shows less exploration capability, which has also been evidenced by Fig.3. Our methods also consistently and notably lift the initial input obtained by other methods. Note in Fig.5 RRWM/FGM boosts the less accurate initial solutions which is generated by the independent pairwise matching using IPFP (top row), but not vice versa (bottom row). Table IV depicts the detailed configuration of the synthetic tests, where 50 random trials are performed for all tests.

iii) For real images, the results on Willow-Object are shown in Fig.6. It involves related objects (car, duck, motorbike) that are viewed by different angles and scales. Moreover, the outliers are added in the CMU-POSE sequence (house, hotel and volvoC70), whereby the performance is plot in Fig.7. Readers are referred to Table V for the related settings. Multiple trials are performed such that it exhaustively traverses all images. Similar to the synthetic test, our methods perform competitively. Fig.8 plots the time costs. Our methods boost MHCL/SYNC at the expense of more run-time. Moreover, compared with RRWM/IPFP, the FGM based solver is less efficient. GACL tends to be even more inefficient due to its deterministic annealing mechanism (see more details in time complexity analysis and the paper [31]). Similar observations are made from the synthetic test while the plots are omitted.

iv) Last but not least, in all above experiments, ALG$_2^1$ notably outperforms its preliminary version FREF+ [1] where the consistency-driven mechanism is not used. This clearly shows the direct advantage of the presented work against [1].

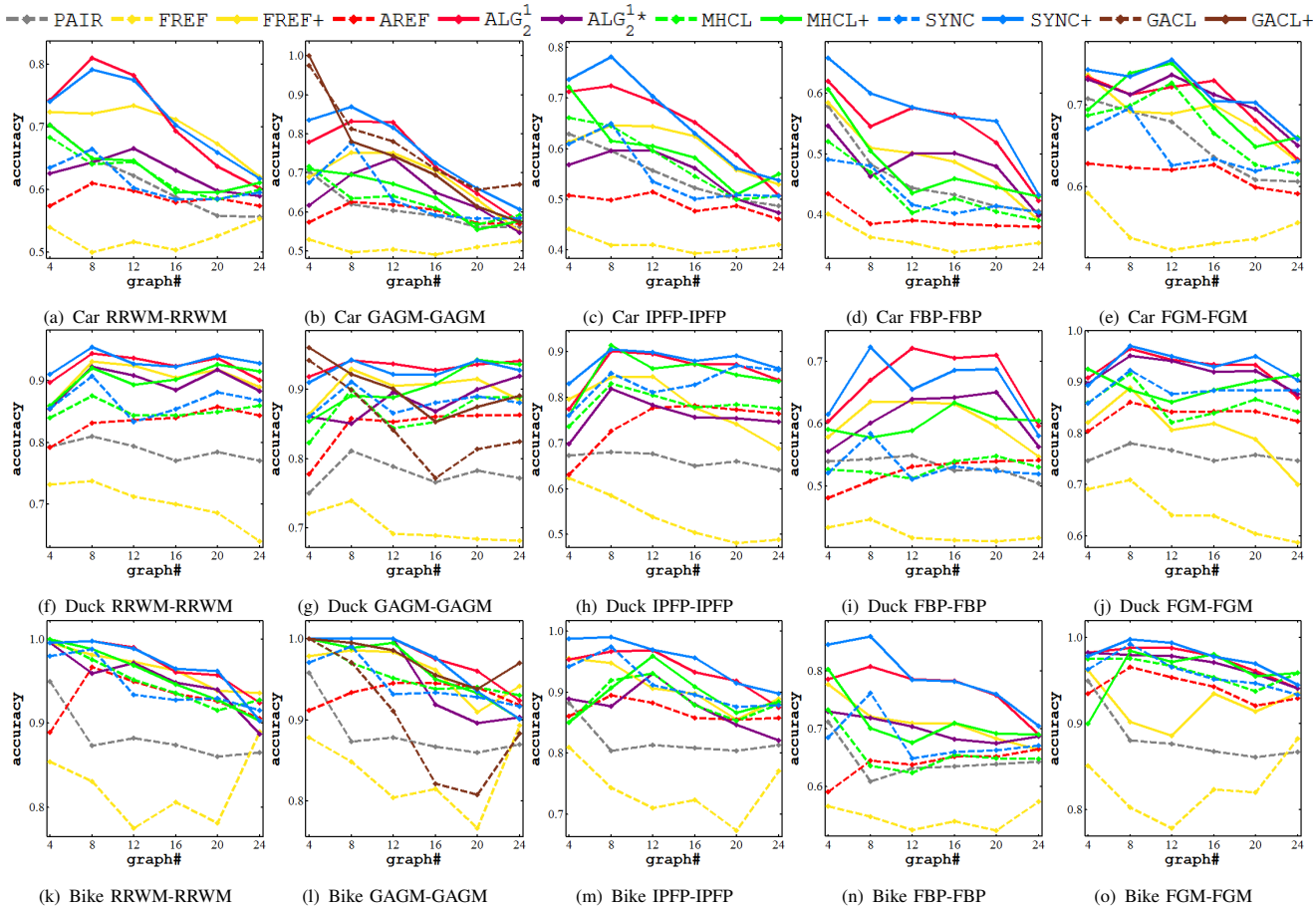
VI. Conclusion

We have proposed both non-factorized and factorized formulations and an alternating optimization framework for joint multiple graph matching. For the non-factorized variant, it is reduced to a pairwise graph matching problem over iterations, which can be solved by various existing non-factorized pairwise graph matching solvers based on the QAP formulation. For the factorized one, we also enable the reuse of the existing convex-concave relaxation based pairwise graph matching solver. Meanwhile, the graph-wise and pairwise consistency metrics are proposed to set the reference graph and the order of sequential variable updating respectively, which are two key aspects associated with our algorithms. Extensive experimental results on synthetic and real-images show competitive performance of the proposed approaches.

Acknowledgement The authors are thankful to Dr. Francesc Serratosa and Dr. Albert Solé-Ribalta who shared the source code of their work [54], [31] to facilitate our evaluation.

References

[1] J. Yan, Y. Tian, H. Zha, X. Yang, Y. Zhang, and S. Chu, "Joint optimization for consistent multiple graph matching," in *ICCV*, 2013.
[2] D. Shen and C. D. Hammer, "Hierarchical attribute matching mechanism for elastic registration," *TMI*, 2002.
[3] O. Duchenne, A. Joulin, and J. Ponce, "A graph-matching kernel for object categorization," in *ICCV*, 2011.



(a) Car RRWM-RRWM (b) Car GAGM-GAGM (c) Car IPFP-IPFP (d) Car FBP-FBP (e) Car FGM-FGM

(f) Duck RRWM-RRWM (g) Duck GAGM-GAGM (h) Duck IPFP-IPFP (i) Duck FBP-FBP (j) Duck FGM-FGM

(k) Bike RRWM-RRWM (l) Bike GAGM-GAGM (m) Bike IPFP-IPFP (n) Bike FBP-FBP (o) Bike FGM-FGM

Fig. 6: Accuracy evaluation on WILLOW-ObjectClass by varying the number of graphs. The selection of a set of graphs covers all possible combinations from that image set.

[4] M. Leordeanu and M. Hebert, "A spectral technique for correspondence problems using pairwise constraints," in *ICCV*, 2005.

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

[6] B. Yao and F. Li, "Action recognition with exemplar based 2.5 d graph matching," in *ECCV*, 2012.

[7] M. A. Fischler and R. C. Bolles, "Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography," *Commun. ACM*, 1981.

[8] Z. Zhang, "Iterative point matching for registration of free-form curves and surfaces," *IJCV*, 1994.

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

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

[11] E. M. Loiola, N. M. de Abreu, P. O. Boaventura-Netto, P. Hahn, and T. Querido, "A survey for the quadratic assignment problem," *EJOR*, pp. 657–90, 2007.

[12] S. Gold and A. Rangarajan, "Softmax to softassign: neural network algorithms for combinatorial optimization," *J. Artif. Neural Netw*, 1996.

[13] M. R. Garey and D. S. Johnson, *Computers and Intractability; A Guide to the Theory of NP-Completeness*. New York, USA: Freeman and Co., 1990.

[14] S. Gold and A. Rangarajan, "A graduated assignment algorithm for graph matching," *IEEE Transaction on PAMI*, 1996.

[15] M. Cho, J. Lee, and K. M. Lee, "Reweighted random walks for graph matching," in *ECCV*, 2010.

[16] 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.

[17] T. Caetano, J. McAuley, L. Cheng, Q. Le, and A. J. Smola, "Learning graph matching," *PAMI*, 2009.

[18] A. Egozi, Y. Keller, and H. Guterman, "A probabilistic approach to spectral graph matching," *IEEE Transactions on PAMI*, pp. 18–27, 2013.

[19] A. Sanfeliu, F. Serratosa, and R. Alquézar, "Clustering of attributed graphs and unsupervised synthesis of function-described graphs," in *ICPR*, 2000.

[20] F. Serratosa, R. Alquézar, and A. Sanfeliu, "Synthesis of function-described graphs and clustering of attributed graphs," *IJPRAI*, 2002.

[21] M. Ferrer, E. Valveny, F. Serratosa, I. Bardají, and H. Bunke, "Graph-based k-means clustering: A comparison of the set median versus the generalized median graph," in *CAIP*, 2009.

[22] A. Torsello and E. Hancock, "Graph embedding using tree edit-union," *Pattern Recogn.*, 2007.

[23] F. Serratosa, R. Alquézar, and A. Sanfeliu, "Function-described graphs for modelling objects represented by sets of attributed graphs," *Pattern Recognition*, vol. 36, no. 3, 2003.

[24] M. Ferrer, E. Valveny, F. Serratosa, K. Riesen, and H. Bunke, "Generalized median graph computation by means of graph embedding in vector spaces," *Pattern Recogn.*, 2010.

[25] A. Sanfeliu, F. Serratosa, and R. Alquezar, "Second-order random graphs for modelling sets of attributed graphs and their application to object learning and recognition," *Int. J. Pattern Recogn. Artif. Intell.*, 2004.

[26] F. Serratosa, A. Solé-Ribalta, and E. Vidiella, "Graph indexing and retrieval based on median graphs," pp. 311–321, 2010.

[27] M. L. Williams, R. C. Wilson, and E. Hancock, "Multiple graph matching with bayesian inference," *PRL*, pp. 1275–1281, 1997.

[28] T. Funkhouser, M. Kazhdan, P. Shilane, P. Min, W. Kiefer, A. Tal, and D. Dobkin, "Modeling by example," *ACM TOG*, 2004.

[29] F. Zhou and F. D. Torre, "Factorized graph matching," in *CVPR*, 2012.

[30] ——, "Deformable graph matching," in *CVPR*, 2013.

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

[32] D. Pachauri, R. Kondor, and S. Vikas, "Solving the multi-way matching problem by permutation synchronization," in *NIPS*, 2013.

[33] A. Solé-Ribalta and F. Serratosa, "Models and algorithms for computing the common labelling of a set of attributed graphs," *CVIU*, 2011.

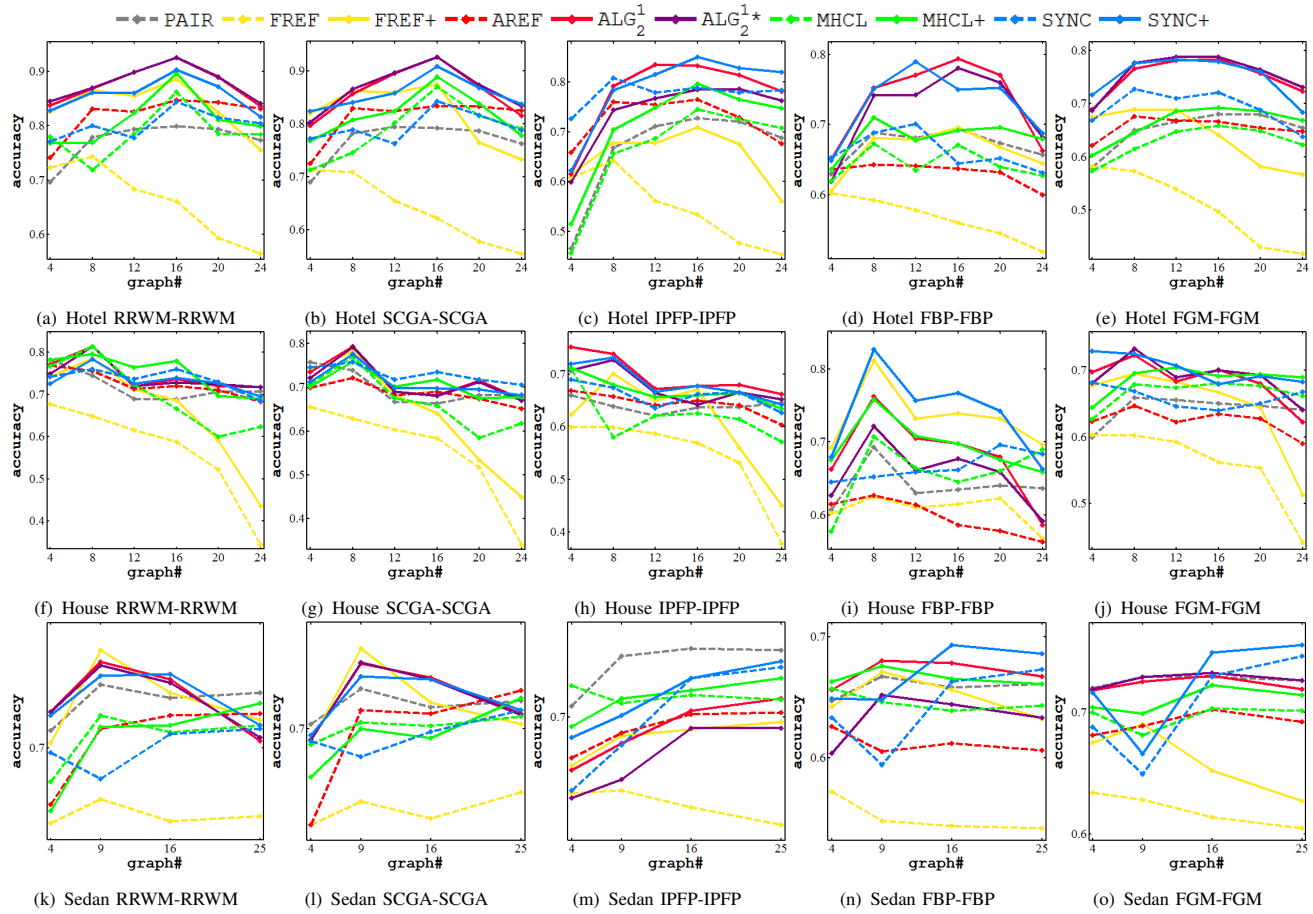


Fig. 7: Accuracy evaluation on CMU-POSE sequence by varying the number of graphs. The selection of a set of graphs covers all possible combinations from that image set.

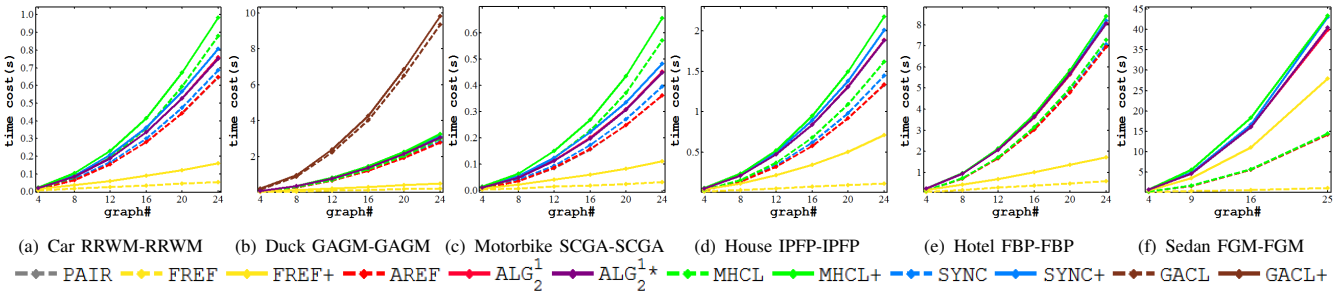


Fig. 8: Time cost comparison on real image dataset for CMU-POSE sequence and WILLOW-ObjectClass.

[34] P. Foggia, G. Percannella, and M. Vento, "Graph matching and learning in pattern recognition in the last 10 years," *IJPRAI*, 2014.

[35] M. Leordeanu, A. Zanfir, and C. Sminchisescu, "Semi-supervised learning and optimization for hypergraph matching," in *ICCV*, 2011.

[36] M. Cho, K. Alahari, and J. Ponce, "Learning graphs to match," in *ICCV*, 2013.

[37] N. Hu, R. M. Rustamov, and L. Guibas, "Graph matching with anchor nodes: a learning approach," in *CVPR*, 2013.

[38] P. S. T. Cour and J. Shi, "Balanced graph matching," in *NIPS*, 2006.

[39] M. Leordeanu and M. Herbert, "An integer projected fixed point method for graph matching and map inference," in *NIPS*, 2009.

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

[41] O. Duchenne, F. Bach, I. Kweon, and J. Ponce, "A tensor-based algorithm for high-order graph matching," in *CVPR*, 2009.

[42] J. Lee, M. Cho, and K. M. Lee, "Hyper-graph matching via reweighted randomwalks," in *CVPR*, 2011.

[43] R. Zass and A. Shashua, "Probabilistic graph and hypergraph matching," in *CVPR*, 2008.

[44] M. Zaslavskiy, F. R. Bach, and J.-P. Vert, "A path following algorithm for the graph matching problem," *PAMI*, 2009.

[45] M. Frank and P. Wolfe, "An algorithm for quadratic programming," *Naval Research Logistics Quarterly*, 1956.

[46] A. Solé-Ribalta and F. Serratosa, "On the computation of the common labelling of a set of attributed graphs," in *CIARP*, 2009.

[47] Q. X. Huang, S. Flory, N. Gelfand, M. Hofer, and H. Pottmann, "Reassembling fractured objects by geometric matching," *ACM Trans. Graph.*, pp. "569–578", 2006.

[48] A. Solé-Ribalta and F. Serratosa, "Graduated assignment algorithm for finding the common labeling of a set of graphs," in *Int. Conf. Structural, Syntactic, and Statistical Pattern Recognition*, 2010.

[49] J. Yan, Y. Li, W. Liu, H. Zha, X. Yang, and S. Chu, "Graduated consistency-regularized optimization for multi-graph match," in *ECCV*, 2014.

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

[51] V. Chvatal, *Linear programming*. Freeman and Company, 1983.

[52] A. Wong and M. You, "Entropy and distance of random graphs with application to structural pattern recognition," *PAMI*, 1985.

[53] A. Solé-Ribalta, F. Serratosa, and A. Sanfeliu, "On the graph edit distance cost: properties and applications," *IJPRAI*, 2012.

[54] F. Serratosa, "Fast computation of bipartite graph matching," *Pattern Recognition Letters*, 2014.
[55] K. Riesen and H. Bunke, "Approximate graph edit distance computation by means of bipartite graph matching," *Image and Vision Computing*, 2008.
[56] I. M. Bomze and G. Danninger, "A global optimization algorithm for concave quadratic programming problems," *SIAM Journal on Optimization*, pp. 826–842, 1993.
[57] J. Maciel and J. Costeira, "A global solution to sparse correspondence problems," *IEEE Trans. Pattern Anal. Mach. Intell.*, pp. 187–199, 2003.
[58] M. Fukushima, "A modified frank-wolfe algorithm for solving the traffic assignment problem," *Transportation Research Part B*, 1984.
[59] H. W. Kuhn, "The hungarian method for the assignment problem," in *Export. Naval Research Logistics Quarterly*, 1955, pp. 83–97.
[60] F. Vikstn, P. Forssn, B. Johansson, and A. Moe, "Comparison of local image descriptors for full 6 degree-of-freedom pose estimation," in *ICRA*, 2009.
[61] D. Lowe, "Distinctive image features from scale-invariant keypoints," *IJCV*, 2004.
[62] F. Gavril, "Generating the maximum spanning trees of a weighted graph," *Journal of Algorithms*, pp. "592–597", 1987.
[63] T. E. Booth, "Power iteration method for the several largest eigenvalues and eigenfunctions," *Nuclear science and engineering*, 2006.
[64] R. Sinkhorn and A. Rangarajan, "A relationship between arbitrary positive matrices and doubly stochastic matrices," *Ann. Math. Statistics*, 1964.

Junchi Yan is currently a Ph.D. candidate at the Department of Electronic Engineering of Shanghai Jiao Tong University, Shanghai, China. Before that he received the M. S. degree from the Department of Automation of the same university in 2011. He is also a Research Staff Member in IBM Research – China, and has been entitled as the IBM Master Inventor since 2014 for his contribution to the patent portfolio of IBM. He has been the recipient of several awards, including the national champion of AMD China University Accelerated Computing Application Contest 2011 for the work on 3-D human motion tracking system, IBM Research Accomplishment Award and Outstanding Accomplishment Award for the work on machine learning-driven sales analytics in 2013 and 2014, respectively. His work on visual saliency detection was featured on the cover of IEEE Signal Processing Letters in 2010. Junchi's research interests are computer vision, machine learning applications and business analytics.

Jun Wang (Member, IEEE) received the Ph.D. degree from Columbia University, NY, in 2011. Currently, he is a staff researcher at the Institute of Data Science and Technology, Alibaba Group, Seattle, WA. He is also an adjunct faculty at Columbia University. From 2010 to 2014, he was a research staff member in the data science group at IBM T. J. Watson Research Center, Yorktown Heights, NY. He also worked as an intern at Google Research in 2009 and as a research assistant at Harvard Medical School, Harvard University in 2006. He has been the recipient of several awards and scholarships, including an Outstanding Technical Achievement Award from IBM Corp. in 2013, the Jury Thesis Award from the Department of Electrical Engineering at Columbia University in 2011, the Google Global Intern Scholarship in 2009, and the Chinese Government Scholarship for Outstanding Self-Financed Students Abroad in 2009. His research interests include machine learning, business analytics, information retrieval, and data mining.

Hongyuan Zha is a Professor at Software Engineering Institute, East China Normal University and the School of Computational Science and Engineering, College of Computing, Georgia Institute of Technology. He earned his PhD degree in scientific computing from Stanford University in 1993. Since then he has been working on information retrieval, machine learning applications, and numerical algorithms. Before joining Georgia Tech, Hongyuan Zha was a Professor at the Department of Computer Science and Engineering at Pennsylvania State University from 1992 to 2006, and worked from 1999 to 2001 at Inktomi Corporation.

Xiaokang YANG (M'00-SM'04) received the B. S. degree from Xiamen University, Xiamen, China, in 1994, the M. S. degree from Chinese Academy of Sciences, Shanghai, China, in 1997, and the Ph.D. degree from Shanghai Jiao Tong University, Shanghai, China, in 2000. He is currently a Distinguished Professor of School of Electronic Information and Electrical Engineering, and the deputy director of the Institute of Image Communication and Information Processing, Shanghai Jiao Tong University, Shanghai, China. From September 2000 to March 2002, he worked as a Research Fellow in Centre for Signal Processing, Nanyang Technological University, Singapore. From April 2002 to October 2004, he was a Research Scientist in the Institute for Infocomm Research (I2R), Singapore. From August 2007 to July 2008, he visited the Institute for Computer Science, University of Freiburg, Germany, as an Alexander von Humboldt Research Fellow. He has published over 200 refereed papers, and has filed 40 patents. His current research interests include visual signal processing and communication, media analysis and retrieval, and pattern recognition. He is associate editor of IEEE Signal Processing Letters, Serias Editor of Springer CCIS, and was a member of Editorial Board of Digital Signal Processing. He is a member of APSIPA, a senior member of IEEE, a member of VSPC Technical Committee of IEEE Circuits and Systems Society, Chair of Multimedia Big Data Interest Group of MMTC Technical Committee of IEEE Communication Society.

Stephen Chu is a Research Scientist at IBM. He leads the worldwide Sales Science program in IBM Global Labs. Before his current assignment, Stephen was a Research Staff Member at the IBM T. J. Watson Research Center in New York for over ten years, and conducted research in the Automatic Speech Recognition field. His team won the top prize in the DARPA-organized speech recognition contest for five consecutive times, beating top competing teams from SRI, BBN, and Cambridge University. His pioneering work in Audio-Visual Speech recognition was featured in the PBS program Scientific American Frontiers with Alan Alda. Stephen's research interests include: Machine Learning, Pattern Recognition, Computer Vision, and Signal Processing. He has published over 50 papers in relevant journals and conferences. Stephen studied Physics at Peking University in Beijing, China, and received his M.S. and Ph.D. degrees in Electrical and Computer Engineering form University of Illinois at Urbana-Champaign.