# Joint Cuts and Matching of Partitions in One Graph

Tianshu Yu
Arizona State University
85281 Tempe AZ USA
tianshuy@asu.edu

Junchi Yan (✉)
Shanghai Jiao Tong University
IBM Research – China
yanesta13@163.com

Jieyi Zhao
University of Texas at Houston
77030 Houston TX USA
jieyi.zhao@uth.tmc.edu

Baoxin Li
Arizona State University
85281 Tempe AZ USA
baoxin.li@asu.edu

## Abstract

*As two fundamental problems, graph cuts and graph matching have been investigated over decades, resulting in vast literature in these two topics respectively. However the way of jointly applying and solving graph cuts and matching receives few attention. In this paper, we first formalize the problem of simultaneously cutting a graph into two partitions i.e. graph cuts and establishing their correspondence i.e. graph matching. Then we develop an optimization algorithm by updating matching and cutting alternatively, provided with theoretical analysis. The efficacy of our algorithm is verified on both synthetic dataset and real-world images containing similar regions or structures.*

## 1. Introduction

Over the past few decades, exploration on graphs has brought remarkable advances for computer vision. Images often have strong structural correlation and are modeled as connected graphs with nodes and edges. While nodes correspond to specific feature points, edges reveal the spatial relation or interaction between neighboring nodes. In this sense, graph is capable of encoding the local and global structural information, and is a natural representation of visual input. We consider two important tasks associated with graph structure: **graph cuts (GC)** and **graph matching (GM)**. Applications of these tasks can be found in stereo [23], video synthesis [16], image segmentation [21] (for graph cut/partition) and object categorization [9], action recognition [32], and feature matching [24], etc.

**Graph cuts**, as the name suggests, is a task to cut one graph into two or more partitions, so as to aggregate the nodes with higher similarities and separate the ones without. Conventionally, the edges is associated with flow weights measuring the transportation capacity between end nodes. Thus cutting the graph can be casted as finding a collection of edges, whose end nodes form a bipartite separation such that the overall transportation on these edges is minimized. For computer vision, pixels and their neighboring relation are often regarded as nodes and edges, respectively. In this fashion, graph cuts is equivalent to grouping the pixels into two clusters taking into account their appearance and adjacency properties. With different weights or transportation measurements, various objectives are proposed.

**Graph matching** aims to find correspondence among graphs. Both unary and second-order (or higher-order) affinity are considered for matching. Since graph matching (GM) typically involves structural information beyond unary/point-wise similarity, it is in general robust against noise and local ambiguities. However, compared with linear assignment with polynomial-time global optimum solvers such as the Hungarian method [19], GM in general involves quadratic assignment which is known NP-hard.

This paper is motivated primarily by the interest of exploring how the above two key tasks on graphs may be jointly considered so as to attain certain optimality not possible with each task considered independently. There are real applications of image segmentation where graph cut alone would have difficulty, while GM seems to be a natural part of the problem, and hence a joint approach is called for.

Fig. 1 illustrates the key potential benefits of such a joint approach (more explanations in the caption): (balanced) graph cuts [1] fails to produce correct partition for the image containing two swans, even though the size balance between the partition is considered. This can be explained by the fact that GC encourages the clustered points (also called local connectivity in this paper) to be assigned to the same partition[1]. On the other hand, if we enforce exist-

[1]We follow a traditional setting by using the point distance as the cut

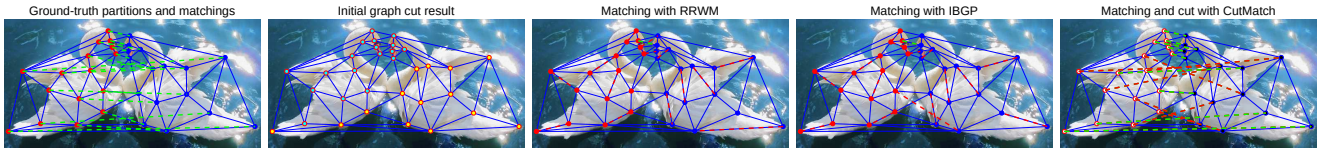



Figure 1. Comparison of the proposed CutMatch v.s. graph cut and graph matching on image with two swans: **a)** ground truth matching and partition; **b)** graph cuts which aims to minimize the overall cut loss (here defined as node distance) tends to group nearby nodes into the same partition (two heads are close to each other); **c)** (modified) graph matching focuses on find similar nodes based the appearance and local structural similarity while gets completely wrong matchings when the partition is unknown; **d)** CutMatch combines the best of the two by considering both local closeness and overall correspondences and thus produce the best results for similar object partition from a single image; **e)** Green (red) dashed lines refer to the matchings that agree (disagree) with the ground-truth. Zoom in for better view.

ing graph matching solvers such as reweighed random walk (RRWM) [6] or our proposed IBGP (a side product of our main method, see more details in Sect. 4) with a brute-force modification to make its input compatible with only one input graph, the result would be completely meaningless.

We draw important observation from the above example: In many scenarios, separating identical/similar objects in an image requires a new mechanism to assign two correspondence nodes (e.g., those similar in appearance and local structure) into two different partitions. This conflicts with the general assumption of graph cut whereby similar nodes should be grouped into the same partition. Importantly, we posit that such a misalignment can be to some extent alleviated by adopting graph matching as it encourages similar nodes to be assigned into separate partitions by finding their correspondence. In general, by combining cut and matching, one task may be facilitated through the extra information/constraint provided by the other (e.g., nearby nodes shall be with the same group while two nodes with similar local structure should be with different partitions), and thus algorithms may be developed to achieve cut and matching alternately in reaching a balanced optima. This inspires us to take a joint approach to achieve the best of the two worlds. To our best knowledge, this is a new problem that has not been addressed.

**Contribution** This paper makes two-fold contributions: 1) a novel model for jointly incorporating and solving graph cuts and graph matching in a unified framework; 2) a novel approach for the joint cut and matching problem whose key component is an effective optimization algorithm named Iterative Bregman Gradient Projection (IBGP). IBGP is theoretically ensured to find a stationary solution. One side-product of IBGP is its simplified version that can be adapted as a new approach for standard two-graph matching.

## 2. Related Work

**Graph partitions and cuts** Graph partition and cuts have been a long-standing research topic. The general idea is to obtain high similarity among nodes within partition and low across partitions. The minimum weight $k$-cut problem aims to divide the graph into $k$ disjoint non-empty partitions such that the cut metric is minimized, with no balance constraint enforced. As shown in [12], this problem can be solved optimally in $O(n^{k^2})$ when $k$ is given. Without knowing $k$, the problem becomes NP-complete. When the (roughly) equal size constraint is involved, early work have proved it is NP-complete. Extended theoretical study is presented in [1]. To limit the scope, we focus in this paper on two-cut partition and forgo a more generalized treatment of (relaxed) $k$-cut problem for future study.

Apart from the above general theoretical studies, more efficient algorithms are devised in practice. To the computer vision community, graph cut based approaches are popular due to their empirically observed promising performance as well as solid theoretical foundations [15].

**Graph matching** Graph matching is mostly considered in the two-graph setting. Since the problem can in general be reformulated as a quadratic assignment task which is NP-hard, different approximate solvers [11, 18, 6] are devised with empirical success. A line of work [20, 30, 29, 4, 22] moves from the two-graph setting to a collection of graphs. While all these work assume each graph is known and focus on finding the node correspondences among graphs. There are also end-to-end approaches [7, 8] on joint node detection (in their respective candidate sets) and matching in an iterative fashion. However, none of these work involves graph partition as they still assume the two graphs are separate in advance. Another orthogonal area is graph structure learning [3, 5] whereby the edge weights on each separate graph is learned to improve the matching accuracy. Readers are referred to [31] for a comprehensive survey.

Though there have been a large amount of literature on graph cuts and graph matching, we are unable to identify any prior work for combing the both lines of research.

## 3. Problem Formulation

**Notations** Lower-case bold $\mathbf{x}$ and upper-case bold $\mathbf{X}$ represent a vector and a matrix, respectively. Lower-case letter such as $n$ corresponds to scalar. Specifically $\mathbf{x}_i$ and $\mathbf{X}_{ij}$ returns the scalar values of $i$-th element of $\mathbf{x}$ and el-

cost whereby the goal is to minimize the overall cut cost.

ement $(i, j)$ of $\mathbf{X}$, respectively. Calligraphic letters $\mathcal{G}$ and $\mathcal{E}$ represent set of nodes and edges. $\mathbb{R}$ and $\mathbb{S}^+$ denote the real-number domain and $n$-th order non-negative symmetric matrices. Function $\mathbf{K} = \text{diag}(\mathbf{x})$ spans a vector into a matrix such that $\mathbf{K}_{ij} = \mathbf{x}_i$ if $i = j$, and $\mathbf{K}_{ij} = 0$ otherwise. Denote $\mathbf{I}$ and $\mathbf{O}$ ($\mathbf{0}$) the identity matrix and all-zeros matrix (vector), respectively. $[\cdot]_+$ is the element-wise ramp function, which keeps the input if it is positive, and 0 otherwise.

**Preliminaries on graph cuts** Consider graph $\mathcal{G}$ associated with weight matrix $\mathbf{W} \in \mathbb{R}^{n\times n}$, where $n$ is the number of nodes in the graph and $\mathbf{W}_{ij}$ corresponds to the weight (similarity) assigned to edge $(i, j) \in \mathcal{E}$. Graph cuts amounts to finding binary partition $\{\mathcal{P}_i\}_{i\in\{1,2\}}$ with $\mathcal{P}_1 \cup \mathcal{P}_2 = \mathcal{N}$ and $\mathcal{P}_1 \cap \mathcal{P}_2 = \emptyset$, such that the energy $\sum_{i\in\mathcal{P}_1, j\in\mathcal{P}_2} \mathbf{W}_{ij}$ is minimized. By introducing variable $\mathbf{y} \in \{-1, 1\}^n$ as indication vector, such that $\mathbf{y}_i = -1$ if $i \in \mathcal{P}_1$ and $\mathbf{y}_i = 1$ for $i \in \mathcal{P}_2$, this problem can be casted as minimizing $\sum_{i,j} \mathbf{W}_{ij}(\mathbf{y}_i - \mathbf{y}_j)^2$ [21]. This energy function encourages the node pair $(i, j)$ belonging to the same partition if the corresponding edge weight is large.

Optimizing this energy is NP-hard. Hence continuous relaxation is often adopted by letting $\mathbf{y} \in [-1, 1]^n$. A popular reformulation of this problem in the context of spectral theory is $\mathbf{y}^T\mathbf{L}^g\mathbf{y} = \sum_{i,j} \mathbf{W}_{ij}(\mathbf{y}_i-\mathbf{y}_j)^2$, where $\mathbf{L}^g = \mathbf{D}-\mathbf{W}$ and $\mathbf{D} = \text{diag}(\sum_i \mathbf{W}_i)$ [21]. Here $\mathbf{L}^g$ is conventionally called graph Laplacian in terms of weight $\mathbf{W}$. This reformulation yields the following problem:

$$\min_{\mathbf{y}} \frac{\mathbf{y}^T\mathbf{L}^g\mathbf{y}}{\mathbf{y}^T\mathbf{y}} \tag{1}$$

where the imposed normalization $\|\mathbf{y}\|_2^2 = 1$ is encoded into the objective by dividing the denominator $\mathbf{y}^T\mathbf{y}$. Graph cuts based methods such as Ratio Cut [28] and NCut (normalized cut) [21] are popular solvers to this problem, which are devised based on the Rayleigh-Ritz theorem [25].

**Adapting standard graph matching to one input graph setting** Standard, namely conventional graph matching (GM) usually involves two given graphs for establishing their node correspondence. In this paper, we consider a more challenging case aiming to find matching between two *implicit* partitions with equal size $m$ from a whole graph of size $n = 2m^2$. Consider a graph $\mathcal{G} = \langle\mathcal{N}, \mathcal{E}\rangle$ with $n = 2m$ nodes, where $\mathcal{N}$ and $\mathcal{E}$ are the nodes and edges, respectively. We establish the node-to-node correspondences within this graph using a matrix $\mathbf{X} \in \{0, 1\}^{n\times n}$, where $\mathbf{X}_{ij} = 1$ implies there is a matching between node $i$ and $j$, and $\mathbf{X}_{ij} = 0$ otherwise. We set $\mathbf{X}_{ii} = 0$ by eliminating the matching feasibility between a node $i$ and itself. Given matrix $\mathbf{A} \in \mathbb{R}^{n^2\times n^2}$ encoding the first (on diagonal) and second order (off diagonal) affinity, graph matching can be formulated as finding binary solution to maximize the overall score $\text{vec}(\mathbf{X})^T\mathbf{A}\text{vec}(\mathbf{X})$ [31], where $\text{vec}(\mathbf{X})$ is the vectorized replica of $\mathbf{X}$. This problem, however, is notoriously NP-hard with combinatorial complexity. In line with standard graph matching methods, we relax $\mathbf{X}$ into the continuous interval $[0, 1]$ and derive the following model:

$$\begin{aligned} &\max_{\mathbf{X}} \text{vec}\,(\mathbf{X})^T\,\mathbf{A}\text{vec}\,(\mathbf{X}) \\ &\text{s.t.} \sum_i \mathbf{X}_{ij} = \mathbf{1}, \sum_j \mathbf{X}_{ij} = \mathbf{1}^T, \mathbf{X}_{ii} = 0, \mathbf{X} \in \mathbb{S}^+ \end{aligned} \tag{2}$$

where $\mathbf{1}$ is a vector with all 1 values. The first two constraints, ensuring $\mathbf{X}$ to be doubly stochastic, indicate one-to-one matching in line with the widely used (relaxed) formulation for two-graph matching [6].

In fact, matching two partitions from one input graph is more challenging compared with standard two-graph matching: i) there are additional constraints i.e. $\mathbf{X}_{ii} = 0$, $\mathbf{X} \in \mathbb{S}^+$ as the model seeks the matching between two partitions from *a single input graph*; ii) the involved variables in Eq. 2 for single graph's partition matching are of larger size. In fact, when the two partitions are given for standard two-graph matching, the affinity matrix and matching matrix become $\mathbf{A}_{gm}^{standard} \in \mathbb{R}^{m^2\times m^2}$ and $\mathbf{X}_{gm}^{standard} \in \{0, 1\}^{m\times m}$ for $n = 2m$. Hence the new problem cannot be addressed by existing graph matching solvers e.g. [11, 6, 18].

**Joint cuts and partition matching in one graph** Our model aims to unify cut and matching, to cut a graph into two components and establish their correspondence simultaneously. As discussed in Section 1 our method is based on the observation that both **closeness** (**for graph cuts**) and **correspondence** (**for graph matching**), no matter measured by appearance or local structure, shall be considered to find meaningful partitions in particular scenarios, e.g. finding identical objects/structures from an input image. Specifically we assume if $\mathbf{X}_{ij}$ is large for matching, then partition $\mathbf{y}_i \neq \mathbf{y}_j$ is more likely to be true. Similar to the graph cuts objective form, the above discussion can be quantified by $\sum_{i,j} \mathbf{X}_{ij}(\mathbf{y}_i - \mathbf{y}_j)^2$. Moreover the doubly stochastic property guarantees that $\sum_i \mathbf{X}_{ij} = \mathbf{1}$ is constant during iterative optimization. Note while graph cuts seeks a minimizer, this term is to be maximized. Hence we must have the Laplacian of matching $\mathbf{I} - \mathbf{X}$, and the coupled energy measuring how much a matching and partition agree with each other becomes:

$$\mathbf{y}^T\,(\mathbf{I} - \mathbf{X})\,\mathbf{y} \tag{3}$$

In summary, adding up Eq. (1), (2), (3) and letting $\mathbf{x} = \text{vec}(\mathbf{X})$ for short, we have the joint objective:

$$\begin{aligned} &\max_{\mathbf{X},\mathbf{y}} \mathbf{x}^T\mathbf{A}\mathbf{x} - \lambda_1\mathbf{y}^T\mathbf{L}^g\mathbf{y} + \lambda_2\mathbf{y}^T\,(\mathbf{I} - \mathbf{X})\,\mathbf{y} \\ &\text{s.t.} \sum_i \mathbf{X}_{ij} = \mathbf{1}, \sum_j \mathbf{X}_{ij} = \mathbf{1}^T, \mathbf{X}_{ii} = 0, \mathbf{X} \in \mathbb{S}^+, \|\mathbf{y}\|_2^2 = 1 \end{aligned} \tag{4}$$

where $\lambda_1$ and $\lambda_2$ balance the cut energy and strength of coupling, respectively. Denote $\mathcal{X}$ the set of variables satisfying the four constraints in Eq. (4). Though Normalized Cut

[2]We use the term 'implicit' to denote the partitions are unknown before matching, and leave the more ill-posed and challenging case of unequal sizes of partitions and matching for future work.

[21] can produce more balanced partitions, it is not suitable in our case. This is because Laplacian $\lambda_2(\mathbf{I}-\mathbf{X})-\lambda_1\mathbf{L}^g$ is not necessarily positive semidefinite. Thus the normalization matrix may have negative diagonal elements.

## 4. Proposed Solver

We devise an optimization procedure which involves updating graph cuts and matching alternatively.

**Initialization** There are two variables for initialization: $\mathbf{X}^{\text{init}}$ and $\mathbf{y}^{\text{init}}$. ii) For initial cut $\mathbf{y}^{\text{init}}$, it is obtained by performing Rayleigh Quotient over the graph Laplacian; ii) For initial matching $\mathbf{X}^{\text{init}}$, firstly we employ on-the-shelf graph matching solver e.g. Reweighted Random Walk Matching (RRWM) [6] to obtain the raw matching $\mathbf{X}^{\text{raw}} \in \{0,1\}^{n\times n}$ with respect to the affinity matrix $\mathbf{A} \in \mathbb{R}^{n^2\times n^2}$, then the symmetry constraint is fulfilled by averaging $\mathbf{X}^{\text{raw}}$ with its transpose: $\mathbf{X}^{\text{raw}} = \frac{\mathbf{X}^{\text{raw}}+\mathbf{X}^{\text{raw}T}}{2}$. Then we continue to set the matchings $\mathbf{X}^{\text{raw}}_{ij} = 0$ if $i$ and $j$ are in the same partition according to $\mathbf{y}$. As the obtained matching $\mathbf{X}^{\text{raw}}$ may violate the constraint $\mathbf{X}_{ii} = 0$, we further project it onto the convex set $\mathcal{X}$. To this end, we draw inspiration from the Bregmanian Bi-Stochastication algorithm [26] and devise a modified version to obtain $\mathbf{X}^{\text{init}}$ satisfying $\mathbf{X}_{ii} = 0$. Different from the algorithm in [26] to find a bi-stochastic projection, the modified version also seeks to obtain the solution with diagonal elements all 0s. The modified projection algorithm denoted by $\mathfrak{P}(\cdot)$ is depicted in Algorithm 1. Note the projection is performed regarding with Euclidian distance. We show how to update $\mathbf{y}$, $\mathbf{X}$ alternatively until converge.

**Update cuts** $\mathbf{y}$ Given current matching $\mathbf{X}^{\text{cur}}$, by peeling off the terms involving $\mathbf{y}$ from Eq. (4), we have:

$$\max_{\mathbf{y}} \mathbf{y}^T \left\{\lambda_2(\mathbf{I}-\mathbf{X}^{\text{cur}}) - \lambda_1(\mathbf{D}-\mathbf{W})\right\}\mathbf{y} \tag{5}$$

Optimizing this objective is straightforward. Firstly the graph Laplacian, which is the second term within the curly braces, is positive semidefinite by its definition. For the first term in the curly braces, we notice that as $\mathbf{X}^{\text{cur}}$ is a doubly stochastic matrix with the largest eigenvalue no larger than 1, then $\mathbf{I}-\mathbf{X}^{\text{cur}}$ must also be positive semidefinite. Thus solving $\mathbf{y}$ yields to calculate the eigenvector corresponding to the largest non-zero algebraic eigenvalue.

**Update matching** $\mathbf{X}$ One natural idea on updating matching $\mathbf{X}$ is to employ a standard graph matching solver. However, this is not suitable for our case. This fact is caused by the difficulty to encode the linear part coupling cut and matching $\mathbf{y}^T(\mathbf{I}-\mathbf{X})\mathbf{y}$ into the quadratic term. Instead we devise a gradient projection based method to obtain stationary solution as described in the following.

First, we compute the partial derivative of $\mathbb{E}$ w.r.t. $\mathbf{X}$:

$$[\nabla\mathbb{E}]_{ij} = \left[\left(\mathbf{A}+\mathbf{A}^T\right)\mathbf{x}\right]_{\tau(i,j)} - \lambda_2\left[\mathbf{y}\mathbf{y}^T\right]_{ij} \tag{6}$$

**Algorithm 1** Bregmanian Projection with Zero Constraint

**Input:** $\mathbf{X}^{\text{raw}}$ by a standard GM solver e.g. RRWM [6]

$\mathbf{X} = \mathbf{X}^{\text{raw}}$
**repeat**
  $\mathbf{X} \leftarrow \left[\mathbf{X} + \frac{\mathbf{1}\mathbf{1}^T-\mathbf{X}\mathbf{1}\mathbf{1}^T-\mathbf{1}\mathbf{1}^T\mathbf{X}}{n} + \frac{\mathbf{1}\mathbf{1}^T\mathbf{X}\mathbf{1}\mathbf{1}^T}{n^2}\right]_+$
  $\mathbf{X}_{jj} \leftarrow 0$
**until** Converge
**return** $\mathbf{X}^{\text{init}} \leftarrow \mathbf{X}$

**Algorithm 2** Iterative Bregman Gradient Projection **IBGP**

**Input:** $\mathbf{X}^{\text{prev}}$, $\mathbf{y}^{\text{prev}}$, $\mathbf{A}$, $\epsilon$

**repeat**
  Compute partial derivative w.r.t. $\mathbf{X}^{\text{prev}}$ by Eq. (6)
  $\mathbf{V} \leftarrow \frac{\nabla\mathbb{E}+\nabla\mathbb{E}^T}{2}$ *// remove for standard GM*
  **repeat**
    $\mathbf{V} \leftarrow \mathbf{V} - \frac{1}{n}\mathbf{V}\mathbf{1}\mathbf{1}^T - \frac{1}{n}\mathbf{1}\mathbf{1}^T\mathbf{V} + \frac{1}{n^2}\mathbf{1}\mathbf{1}^T\mathbf{V}\mathbf{1}\mathbf{1}^T$
    $\mathbf{V}_{jj} \leftarrow 0$ *// remove for standard GM*
    **if** *solving CutMatch* **then**
      **if** $\mathbf{V}_{jk} < \max\{-\frac{\mathbf{X}^{\text{prev}}_{jk}}{\epsilon}, -\frac{\mathbf{X}^{\text{prev}}_{kj}}{\epsilon}\}$ **then**
        $\mathbf{V}_{jk} \leftarrow \max\{-\frac{\mathbf{X}^{\text{prev}}_{jk}}{\epsilon}, -\frac{\mathbf{X}^{\text{prev}}_{kj}}{\epsilon}\}$
      **end if**
      **if** $\mathbf{V}_{jk} > \min\{\frac{1-\mathbf{X}^{\text{prev}}_{jk}}{\epsilon}, \frac{1-\mathbf{X}^{\text{prev}}_{kj}}{\epsilon}\}$ **then**
        $\mathbf{V}_{jk} \leftarrow \min\{\frac{1-\mathbf{X}^{prev}_{jk}}{\epsilon}, \frac{1-\mathbf{X}^{\text{prev}}_{kj}}{\epsilon}\}$
      **end if**
    **else if** *solving standard GM* **then**
      **if** $\mathbf{V}_{jk} < -\frac{\mathbf{X}^{\text{prev}}_{jk}}{\epsilon}$ **then**
        $\mathbf{V}_{jk} \leftarrow -\frac{\mathbf{X}^{\text{prev}}_{jk}}{\epsilon}$
      **end if**
      **if** $\mathbf{V}_{jk} > \frac{1-\mathbf{X}^{\text{prev}}_{jk}}{\epsilon}$ **then**
        $\mathbf{V}_{jk} \leftarrow \frac{1-\mathbf{X}^{\text{prev}}_{jk}}{\epsilon}$
      **end if**
    **end if**
  **until** Converge
  $\mathbf{X}^{\text{prev}} \leftarrow \mathbf{X}^{\text{prev}} + \epsilon\mathbf{V}$
**until** Converge
**return** $\mathbf{X}^{\text{cur}} \leftarrow \mathbf{X}^{\text{prev}}$

where $\tau(i,j)$ is a mapping from matrix subscript $(i,j)$ to its corresponding vector index – elements in two terms on the right hand side are from vector and matrix respectively.

Given $\mathbf{X}^{\text{prev}}$ obtained in previous iteration, using gradient $\nabla\mathbb{E}$ can not guarantee the updated $\mathbf{X}^{\text{cur}}$ lying in the feasible convex set $\mathcal{X}$. To address this issue, we develop an objective to find an optimal updating direction:

$$\begin{aligned} &\min_{\nabla\mathbf{X}} \|\nabla\mathbf{X} - \nabla\mathbb{E}\|_F^2 \\ &\text{s.t. } \nabla\mathbf{X}\mathbf{1} = \mathbf{0}, \nabla\mathbf{X} = \nabla\mathbf{X}^T, \nabla\mathbf{X}_{ii} = 0 \\ &\quad \mathbf{X}^{\text{prev}}_{ij} + \epsilon\nabla\mathbf{X}_{ij} \geq 0, \mathbf{X}^{\text{prev}}_{ij} + \epsilon\nabla\mathbf{X}_{ij} \leq 1 \end{aligned} \tag{7}$$

where $\nabla\mathbf{X}$ is the optimal update direction starting from $\mathbf{X}^{\text{prev}}$, and $\epsilon > 0$ is a pre-defined step length. The convex objective $\min_{\nabla\mathbf{X}}\|\nabla\mathbf{X} - \nabla\mathbb{E}\|_F^2$ is to find an optimal

ascending direction within the feasible set, which is defined by the constraints in Eq. (7) claiming the updated $\mathbf{X}^{\text{cur}} = \mathbf{X}^{\text{prev}} + \epsilon\nabla\mathbf{X}$ still falling into $\mathcal{X}$. The verification of this claim is trivial. To obtain the solution to Eq. (7), we perform *Dykstra Algorithm* [10], whereby we first split the constraints into two sets $\mathcal{C}_1$ and $\mathcal{C}_2$:

$$\mathcal{C}_1 : \nabla\mathbf{X}\mathbf{1} = \mathbf{0}, \nabla\mathbf{X} = \nabla\mathbf{X}^T$$
$$\mathcal{C}_2 : \nabla\mathbf{X}_{ii} = 0, \mathbf{X}_{ij}^{\text{prev}} + \epsilon\nabla\mathbf{X}_{ij} \geq 0, \mathbf{X}_{ij}^{\text{prev}} + \epsilon\nabla\mathbf{X}_{ij} \leq 1$$

where $\mathcal{C}_1$ is a subspace and $\mathcal{C}_2$ is a bounded convex set. By denoting $\mathbf{V} = \nabla\mathbf{X}$ and $\mathbf{U} = \nabla\mathbb{E}$ for short, we first optimize the objective with constraints in $\mathcal{C}_1$:

$$\min_{\mathbf{V}} \|\mathbf{V} - \mathbf{U}\|_F^2, \quad \text{s.t. } \mathbf{V}\mathbf{1} = \mathbf{0}, \mathbf{V} = \mathbf{V}^T \tag{8}$$

The Lagrangian function of this problem is:

$$\mathcal{L} = \text{Tr}(\mathbf{V}^T\mathbf{V} - 2\mathbf{V}^T\mathbf{U} + \mathbf{U}^T\mathbf{U}) - \boldsymbol{\mu}^T\mathbf{V}\mathbf{1} - \boldsymbol{\mu}^T\mathbf{V}^T\mathbf{1} \tag{9}$$

where $\boldsymbol{\mu} \in \mathbb{R}^n$ is the corresponding Lagrangian multiplier. There is only one multiplier because $\mathbf{V}$ is symmetric. Taking the partial derivative with respect to $\mathbf{V}$ and letting $\partial\mathcal{L}/\partial\mathbf{V} = 0$, we have:

$$\mathbf{V} = \mathbf{U} + \frac{1}{2}\boldsymbol{\mu}\mathbf{1}^T + \frac{1}{2}\mathbf{1}\boldsymbol{\mu}^T \tag{10}$$

After right multiplying $\mathbf{1}$ on both sides and noticing the constraint $\mathbf{V}\mathbf{1} = \mathbf{0}$ in Eq. (8), we have:

$$\mathbf{0} = \mathbf{U}\mathbf{1} + \frac{n}{2}\boldsymbol{\mu} + \frac{1}{2}\mathbf{1}\mathbf{1}^T\boldsymbol{\mu} \tag{11}$$

Applying Woodbury formula [14] for the inverse, we can derive a closed form of multiplier:

$$\boldsymbol{\mu} = -\frac{2}{n}\left(\mathbf{I} - \frac{1}{2n}\mathbf{1}\mathbf{1}^T\right)\mathbf{U}\mathbf{1} \tag{12}$$

Substituting Eq. (12) back to Eq. (10) we obtain:

$$\mathbf{V} = \mathbf{U} - \frac{1}{n}\mathbf{U}\mathbf{1}\mathbf{1}^T - \frac{1}{n}\mathbf{1}\mathbf{1}^T\mathbf{U} + \frac{1}{n^2}\mathbf{1}\mathbf{1}^T\mathbf{U}\mathbf{1}\mathbf{1}^T \tag{13}$$

We then consider the partial objective involving $\mathcal{C}_2$:

$$\begin{aligned}&\min_{\mathbf{V}} \|\mathbf{V} - \mathbf{U}\|_F^2 \\ &\text{s.t. } \mathbf{V}_{ii} = 0, \mathbf{X}_{ij}^{\text{prev}} + \epsilon\mathbf{V}_{ij} \geq 0, \mathbf{X}_{ij}^{\text{prev}} + \epsilon\nabla\mathbf{X}_{ij} \leq 1\end{aligned} \tag{14}$$

This problem can be readily solved by letting $\mathbf{V}_{ii} = 0$, truncating $\mathbf{V}_{ij} = -\mathbf{X}_{ij}^{\text{prev}}/\epsilon$ if $\mathbf{V}_{ij} < -\mathbf{X}_{ij}^{\text{prev}}/\epsilon$ and $\mathbf{V}_{ij} = (1 - \mathbf{X}_{ij}^{\text{prev}})/\epsilon$ if $\mathbf{V}_{ij} > (1 - \mathbf{X}_{ij}^{\text{prev}})/\epsilon$. By alternating the procedure between Eq. (8) and Eq. (14), the optimal direction $\nabla\mathbf{X}$ can be found. Hence the update rule over $\mathbf{X}$ is (recall $\mathbf{V} = \nabla\mathbf{X}$ by definition):

$$\mathbf{X}^{\text{cur}} = \mathbf{X}^{\text{prev}} + \epsilon\nabla\mathbf{X} \tag{15}$$

**Algorithm 3** CutMatch

**Input:** $\mathbf{X}^{\text{init}}$ by Algorithm 1, $\mathbf{y}^{\text{init}}$, $\mathbf{A}$, $\mathbf{D}$, $\mathbf{W}$, $\lambda_1$, $\lambda_2$
$\mathbf{X}^{\text{cur}} \leftarrow \mathbf{X}^{\text{init}}$, $\mathbf{y}^{\text{cur}} \leftarrow \mathbf{y}^{\text{init}}$
$\mathbf{E} \leftarrow \lambda_1\mathbf{D} + \lambda_2\mathbf{I}$
**repeat**
  $\mathbf{L}^{\text{cur}} \leftarrow \lambda_2(\mathbf{I} - \mathbf{X}^{\text{cur}}) - \lambda_1(\mathbf{D} - \mathbf{W})$
  *// update cut:*
  $\mathbf{y} \leftarrow \arg\min_{\mathbf{y}} \frac{\mathbf{y}^T\mathbf{L}^{\text{cur}}\mathbf{y}}{\mathbf{y}^T\mathbf{y}}$
  **if** First iteration **then**
    *// correlate starting point of matching:*
    $\mathbf{X}_{ij}^{\text{cur}} \leftarrow 0$ if $\text{sign}(\mathbf{y}_i) = \text{sign}(\mathbf{y}_j)$
    $\mathbf{X}^{\text{cur}} \leftarrow \mathfrak{P}(\mathbf{X}^{\text{cur}})$ using Algorithm 1
  **end if**
  *// update matching:*
  $\mathbf{X}^{\text{cur}} \leftarrow \arg\max_{\mathbf{X}} \mathbb{E}$ according to Algorithm 2
**until** Converge
**return** $\mathbf{X}^{\text{opt}} \leftarrow \mathbf{X}^{\text{cur}}$, $\mathbf{y}^{\text{opt}} \leftarrow \mathbf{y}$

Fixing $\mathbf{y}$ and repeating the update rule until convergence, one can reach the stationary $\mathbf{X}$ to Problem (4).

We call the above algorithm **Iterative Bregman Gradient Projection** (**IBGP**) and summarize it in Algorithm 2. **IBGP** can easily adapt to standard graph matching problem (with or without linear part) by removing constraint $\nabla\mathbf{X}_{ii} = 0$ and $\nabla\mathbf{X} = \nabla\mathbf{X}^T$ from Eq. (7) – fortunately the corresponding update rule is still as Eq. 13. See supplementary material for derivation. Furthermore, IBGP can be integrated with popular non-convex optimization framework, such as convex-concave relaxation [33] and path following [27]. We leave this to future work. The following proposition ensures the convergence of the update strategy.

**Proposition 1.** *The optimal solution $\nabla\mathbf{X}^{opt}$ to Eq. (7) must be a non-decreasing direction.*

*Proof.* As objective 7 is convex, the global optima is reachable. First note that matrix $\mathbf{O}$ is in the feasible set of 7 as $0 \leq \mathbf{X}_{ij}^{\text{prev}} \leq 1$. Thus if an optimal solution $\mathbf{V}$ is with $\text{Tr}(\mathbf{V}^T\nabla\mathbb{E}) < 0$, we must have $\|\mathbf{V} - \nabla\mathbb{E}\|_F^2 = \text{Tr}(V^TV) - 2\,\text{Tr}(\mathbf{V}^T\nabla\mathbb{E}) + \text{Tr}(\nabla\mathbb{E}^T\nabla\mathbb{E}) > 0 + 0 + \text{Tr}(\nabla\mathbb{E}^T\nabla\mathbb{E}) = \|\mathbf{O} - \nabla\mathbb{E}\|_F^2$. Then $\mathbf{V}$ cannot be optimal. This implies that for any optimal $\nabla\mathbf{X}^{\text{opt}}$ we have $\text{Tr}(\nabla\mathbb{E}^T\nabla\mathbf{X}^{\text{opt}}) \geq 0$, thus $\nabla\mathbf{X}^{\text{opt}}$ must be a non-decreasing direction. QED. □

We summarize **Initialization**, **Update cut** and **Update matching** in Algorithm 3 which is termed as **CutMatch**.

**Discretization** We apply Hungarian method [2] over $\mathbf{X}$ to calculate the discrete solution to graph matching subproblem. To obtain the discrete solution to partition, we simply refer to the sign of each element in $\mathbf{y}$. Though more complicated discretization method can be devised to avoid the conflict between matching $\mathbf{X}$ and $\mathbf{y}$, an easier way is adopted as we observe that the aforementioned discretization methods yield high performance in experiments.

**Convergence** According to the four constraints in Eq. (4), $\mathbf{X}$ lies in a closed convex set. Similarly, under the latter three constraints, $\mathbf{y}$ is in a closed ellipsoid with dimension at most $m-1$, which is also convex. Therefore the objective in Eq. (4) should be the upper bounded. The updating rule in Algorithm 3, on the other hand, ensures that the energy $\mathbb{E}$ ascends at each iteration. In summary, Algorithm 3 is guaranteed to converge to a local stationary point.

# 5. Experiment and Discussion

## 5.1. Graph matching on synthetic data

Since we have devised a new graph matching solver – IBGP along with CutMatch (see Algorithm 1), we first test it on synthetic data by the protocol in [6]. Four popular graph matching solvers are compared, including graduated assignment (GAGM) [11], integer projection fixed point (IPFP) [18], reweighed random walk (RRWM) [6] and spectral matching (SM) [17]. The four peer methods and IBGP are all initialized with a uniform doubly stochastic matrix. Various levels of deformation, outlier and density change are randomly generated and casted to the affinity matrix. Figure 2 demonstrates the experimental results. In deformation test, we generate 30 pairs of graphs with 20 nodes, then varying noise from 0 to 0.4 with increment 0.05 is added to the affinity. In outlier test, up to 10 outlier points by increments 2 are further generated to disturb the matching. In density test, we test the matching performance by varying density from 0.2 to 1 as well as noise 0.25 and 5 extra outlier points. We see that in deformation and density test, IBGP performs competitively compared to state-of-the-art algorithms in most settings in terms of accuracy and affinity score. Though IBGP is relatively sensitive to outlier, we notice that it has high stability in combinations of various disturbance. From numerical perspective, we further observe that when affinity matrix is negative definite (or contains many negative elements), SM, RRWM and GAGM always report error, due to the spectral limitation.

## 5.2. Joint cut and matching on synthetic data

**Dataset** We construct the benchmark by randomly generating synthetic graphs with different types of disturbances including noise, randomly displacement and feature permutation. For each partition, there are 20 nodes from normal distribution, thus 40 nodes in total for the final test graph. Specifically we first generate partition $\mathcal{P}_1$ and translate it to obtain partition $\mathcal{P}_2$, while the translation retains overlapping $\gamma$ measuring the horizontal distance between the most left point of $\mathcal{P}_2$ and the most right point of $\mathcal{P}_1$. We further add level $\sigma$ Gaussian noise to partition $\mathcal{P}_2$. Having obtained the nodes, we perform Delaunay triangulation [13] to obtain the edges. For an element $(ia : jb)$ in affinity matrix (for **Graph Matching**) $\mathbf{A}$, we calculate it as $\mathbf{A}_{ia:jb} = \exp(-(d_{ij} - d_{ab})^2/\delta_1)$, where $d_{ij}$ is the Euclidean distance between node $i$ and $j$. For each corresponding node pair $i$ and $j$ in $\mathcal{P}_1$ and $\mathcal{P}_2$, respectively, we assign a randomly generated 128-D feature $\mathbf{f}_i = \mathbf{f}_j$ (to simulate SIFT feature) from normal distribution and further add level $\mu$ of Gaussian noise to $\mathbf{f}_j$. $\rho$ level of random permutation is also performed to disturb the order of the features in the whole graph. For any node $i \in \mathcal{P}_1 \cup \mathcal{P}_2$, we denote its 2-D location $\mathbf{l}_i$. Thus similarity matrix (for **Graph Cuts**) $\mathbf{W}_{ij} = \exp(-\|\mathbf{f}_i - \mathbf{f}_j\|_2^2/\delta_2) + \exp(-\|\mathbf{l}_i - \mathbf{l}_j\|_2^2/\delta_3)$ and we set $\delta_1 = 0.5$, $\delta_2 = 5$, $\delta_3 = 0.5$ in all tests.

**Evaluation metrics** We calculate the average $Accuracy$ for both graph cut and graph matching. For graph cut, $Accuracy$ represents the portion of correctly clustered nodes w.r.t. the sum of all nodes. As the final output of cuts is a $sign$ function, thus given a ground-truth label vector $\mathbf{C} \in \{-1, 1\}^{2n}$ and an output label vector $\mathbf{C}_{cut} \in \{-1, 1\}^{2n}$, we calculate as $Accuracy = \frac{\max\{\text{number of } \mathbf{C}=\mathbf{C}_{out}, \text{number of } \mathbf{C}\neq\mathbf{C}_{cut}\}}{2n}$. For graph matching, $Accuracy$ corresponds to the portion of correctly matched nodes w.r.t. the sum of all nodes. The metric is also adopted in real-image test in Section 5.3. Note in the graph cuts experiments, the minimal $Accuracy$ is 0.5.

**Results** We evaluate the performance of CutMatch by varying $\gamma$, $\sigma$, $\mu$ and $\rho$. For each value of parameters, we randomly generate 80 graphs and calculate the average cuts and matching accuracy. Figure 3 shows the statistical results, and the first and second columns correspond to the matching and the cuts accuracy, respectively. We also present the performance with three pairs of $\lambda_1$ and $\lambda_2$ values: $(200, 50)$, $(150, 30)$ and $(100, 20)$. Specifically, Fig. 1 shows an example with combined disturbance, where $\gamma = 0.2$, $\sigma = 0.1$, $\mu = 0.3$ and $\rho = 0.1$. Figure 3 presents the raw accuracy by vanilla matching: RRWM, IPFP and IBGP, together with the raw cuts accuracy with vanilla spectral graph cuts [21]. One can observe that, with the joint objective, CutMatch significantly outperforms vanilla cuts and matching algorithms. As well as the performance enhancement, CutMatch is also robust to all kinds of disturbances. We also found that matching nodes (RRWM, IPFP and IBGP) within one graph is extremely sensitive to the initial $\mathbf{X}$. The CutMatch approach, however, is capable of correlating the starting points of cuts and matching alternatively in each iteration. In all the experiments, we observe that 5 iterations are sufficient to yield satisfactory performance.

## 5.3. Joint cut and matching on real images

**Dataset** We collect 15 images from internet containing two similar objects with the same category (**see supplemental material for the dataset**). For each image, we manually select 10 to 20 landmark nodes and establish the ground-truth correspondences. To obtain the connectivity, we first apply Delaunay triangulation on each partition, and then all

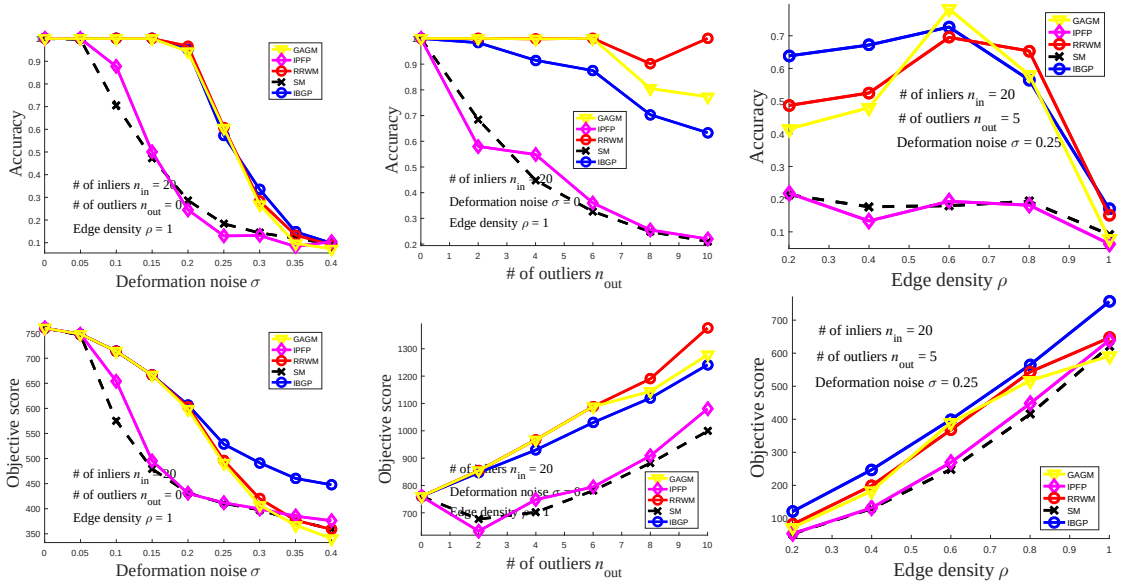



Figure 2. Evaluation of the proposed IBGP and peer methods on synthetic data for standard two-graph matching problem.

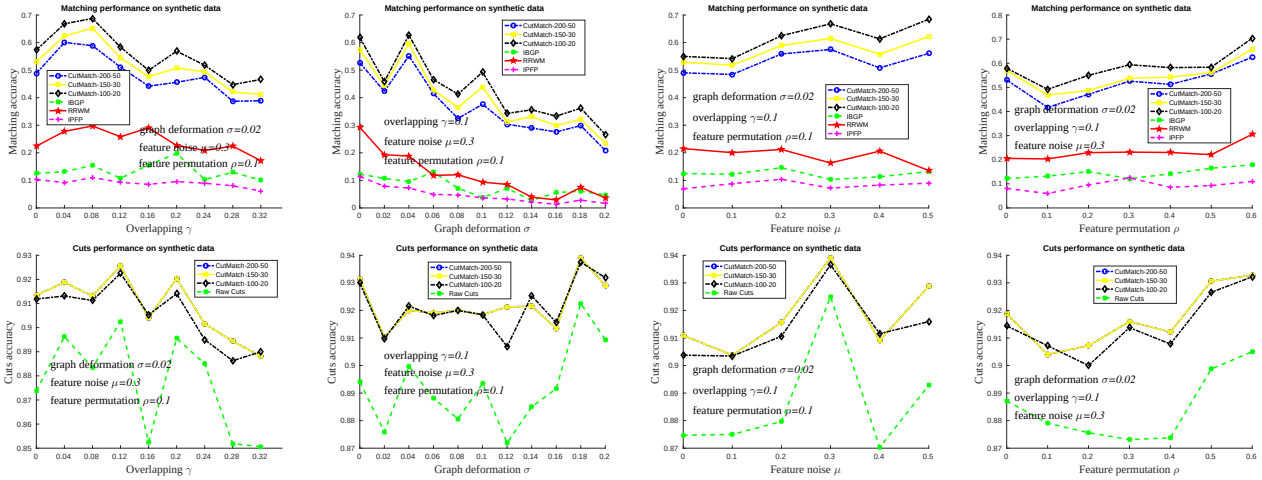



Figure 3. Evaluation of CutMatch and vanilla graph cuts and graph matching solvers on synthetic data for the joint cut and matching task.

the landmarks. The union of the edges in the three triangulations is regarded as the baseline connectivity. We further calculate the SIFT feature of each node. The same strategy on generating affinity $\mathbf{A}$ and similarity $\mathbf{W}$ is applied. For generating matrix $\mathbf{W}$, we set $\lambda_2 = 2 \times 10^5$ and $\lambda_3 = 100$, which is to balance the un-normalized SIFT features. Because we observe that sometimes the deformation contained in real images is so large and arbitrary that local connectivity obtained from triangulation changes drastically, we test the performance under varying edge sampling rate $\eta$, where $\eta = 1$ and $\eta = 0$ correspond to fully connected and baseline graphs, respectively. As we believe that real images already contain natural contaminations (e.g., noises, deformation and outliers), we don't add extra degradations in this experiment as in section 5.2. This dataset will be published.

**Results** To this end, we randomly generate edges corresponding to varying edge density from 0 to 0.2 with step 0.02. For each density value other than 0, 10 graphs are generated to avoid the bias, and the average performance is calculated regarding all the graphs. Fig. 4 demonstrates the cuts and matching results. IBGP, RRWM and raw cuts are selected one again for comparison. As CutMatch shows its parametric stability in synthetic test, we fix parameters $\lambda_1 = 200$ and $\lambda_2 = 50$. We also let $\delta_1 = 150$ and $\delta_2 = 400,000$ to generate affinity and flow in proper range. As can be seen from Fig. 4, though the task on real images is challenging, CutMatch is superior to all the selected counterparts in terms of cuts and matching accuracy. In general, matching partitions is relatively more difficult than cuts, thus needs more concentration in our future

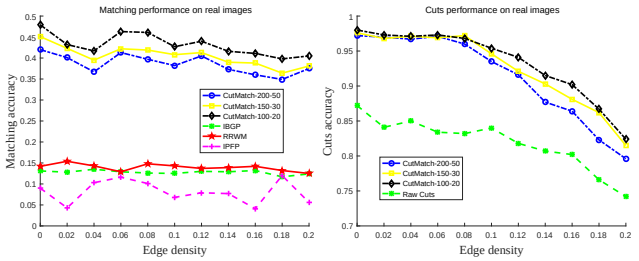



Figure 4. Joint cuts and matching results on real-world images. **Left**: CutMatch with different parameter settings $(\lambda_1, \lambda_2) = \{(200, 50), (150, 30), (100, 20)\}$ vs. three vanilla graph matching solver IBGP, RRWM [6], IPFP [18]. **Right**: CutMatch $(\lambda_1, \lambda_2) = \{(200, 50), (150, 30), (100, 20)\}$ vs. vanilla graph cuts [21].

work. Fig. 5 shows some results on real-world images. The cuts performance is observed to be enhanced by using CutMatch. When the connectivity is stable, e.g. in the first row (for rigid bike), the matching with CutMatch is reliable and very close to the ground-truth. When objects contain salient partial deformation, e.g. in the second and the third row (for deformable human body), CutMatch reaches significant matching performance compared to vanilla IBGP for graph matching. However, when disturbance is severe as in the last row (more large transformation), the matching will mostly fail. In either case, the performance of raw IBGP matching is extremely low.

## 6. Conclusion and Future Work

This paper presents a solver to the joint graph cuts and matching problem on a single input graph, in order to better account for both the closeness and correspondences in the presence of two similar objects appearing in one image. Despite the illustrated potential of CutMatch, there is space for future work: i) CutMatch can possibly trap into a local optima. This issue can be possibly addressed with annealing strategies; ii) the large amount of variables and gradient-based update scheme involved in MatchCut makes it currently not scalable for dealing with dense correspondences. We empirically observe that CutMatch tends to deliver sparse solution $\mathbf{X}$, hence we believe MatchCut can be accelerated especially considering affinity $\mathbf{A}$ is also sparse; iii) the generalized case: cut graph into $k$ partitions and establish their correspondence.

## Appendix

### Update of gradient for GM

For standard graph matching, we seek to solve:

$$\begin{aligned} &\min_{\mathbf{V}} \|\mathbf{V} - \mathbf{U}\|_F^2 \\ &\text{s.t. } \mathbf{V}\mathbf{1} = \mathbf{0}, \mathbf{V}^T\mathbf{1} = \mathbf{0} \end{aligned} \tag{16}$$

where $\mathbf{V}$ and $\mathbf{U}$ are the optimal update direction and the objective gradient, respectively. Refer to the main body for the CutMatch version. Similar to the previous procedure, the Lagrangian function is:

$$\mathcal{L} = \|\mathbf{V} - \mathbf{U}\|_F^2 - \mathbf{a}^T\mathbf{V}\mathbf{1} - \mathbf{b}^T\mathbf{V}^T\mathbf{1} \tag{17}$$

Zeroing the partial derivative w.r.t. $\mathbf{V}$ we have:

$$\frac{\partial \mathcal{L}}{\partial \mathbf{V}} = 2(\mathbf{V} - \mathbf{U}) - \mathbf{a}\mathbf{1}^T - \mathbf{1}\mathbf{b}^T = \mathbf{O} \tag{18}$$

Then we have:

$$\mathbf{V} = \mathbf{U} + \frac{1}{2}\mathbf{a}\mathbf{1}^T + \frac{1}{2}\mathbf{1}\mathbf{b}^T \tag{19}$$

Right multiplying $\mathbf{1}$, we have:

$$\begin{aligned} \mathbf{0} &= \mathbf{U}\mathbf{1} + \frac{1}{2}\mathbf{a}\mathbf{1}^T\mathbf{1} + \frac{1}{2}\mathbf{1}\mathbf{b}\mathbf{1}^T \\ &= \mathbf{U}\mathbf{1} + \frac{n}{2}\mathbf{a} + \frac{1}{2}\mathbf{1}\mathbf{1}^T\mathbf{b} \end{aligned} \tag{20}$$

Transposing Eq. (19) and right multiplying $\mathbf{1}$ we have:

$$\mathbf{0} = \mathbf{U}^T\mathbf{1} + \frac{1}{2}\mathbf{1}\mathbf{1}^T\mathbf{a} + \frac{n}{2}\mathbf{b} \tag{21}$$

Then we have:

$$\frac{\mathbf{1}\mathbf{1}^T \cdot \text{Eq.20} - 21}{n} \tag{22}$$

$$= \frac{\mathbf{1}\mathbf{1}^T\mathbf{U}\mathbf{1}}{n} - \mathbf{U}^T\mathbf{1} + \frac{\mathbf{1}\mathbf{1}^T\mathbf{1}\mathbf{1}^T\mathbf{b}}{2n} - \frac{n\mathbf{b}}{2} \tag{23}$$

$$= \mathbf{0} \tag{24}$$

Then,

$$\frac{1}{2}(\mathbf{1}\mathbf{1}^T - n\mathbf{I})\mathbf{b} = \mathbf{U}^T\mathbf{1} - \frac{1}{n}\mathbf{1}\mathbf{1}^T\mathbf{U}\mathbf{1} \tag{25}$$

As we have the pseudo inverse (see next section):

$$(n\mathbf{I} - \mathbf{1}\mathbf{1}^T)^{-1} = \frac{1}{n}(\mathbf{I} - \frac{1}{n}\mathbf{1}\mathbf{1}^T) \tag{26}$$

We then have:

$$\mathbf{b} = -\frac{2}{n}(\mathbf{U}^T\mathbf{1} - \frac{1}{n}\mathbf{1}\mathbf{1}^T\mathbf{U}^T\mathbf{1}) \tag{27}$$

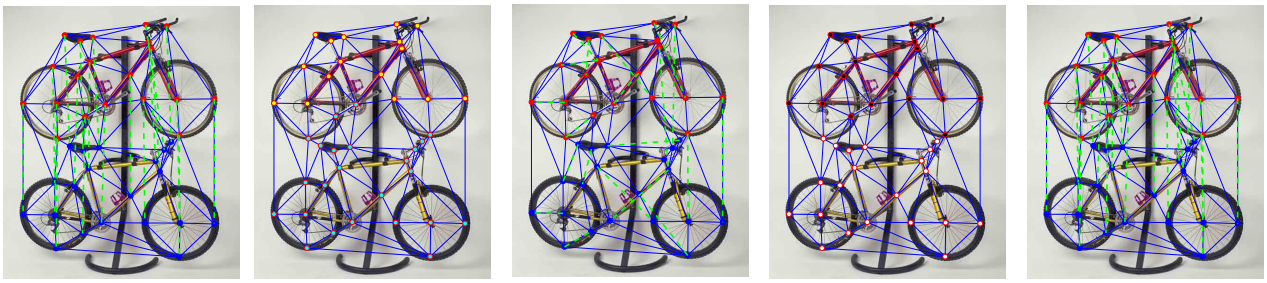
Analogously, we can obtain:

$$\mathbf{a} = -\frac{2}{n}(\mathbf{U}\mathbf{1} - \frac{1}{n}\mathbf{1}\mathbf{1}^T\mathbf{U}\mathbf{1}) \tag{28}$$

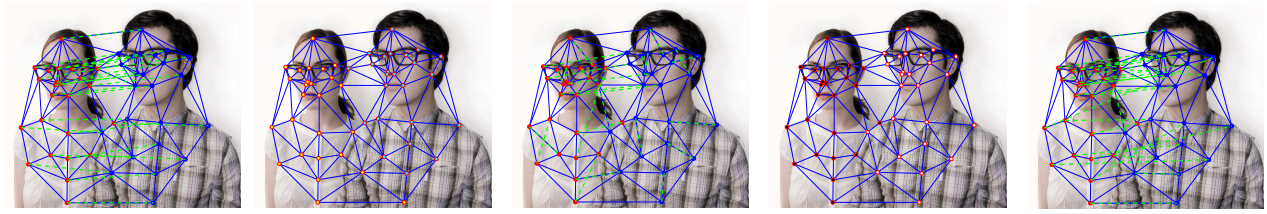
Substituting Eq. (27) and (28) back to (19), we obtain:

$$\mathbf{V} = \mathbf{U} - \frac{1}{n}\mathbf{U}\mathbf{1}\mathbf{1}^T - \frac{1}{n}\mathbf{1}\mathbf{1}^T\mathbf{U} + \frac{2}{n^2}\mathbf{1}\mathbf{1}^T\mathbf{U}\mathbf{1}\mathbf{1}^T \tag{29}$$

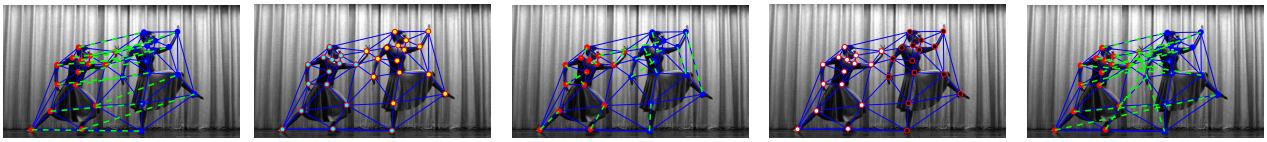
which is the same form as the update rule in the main text.



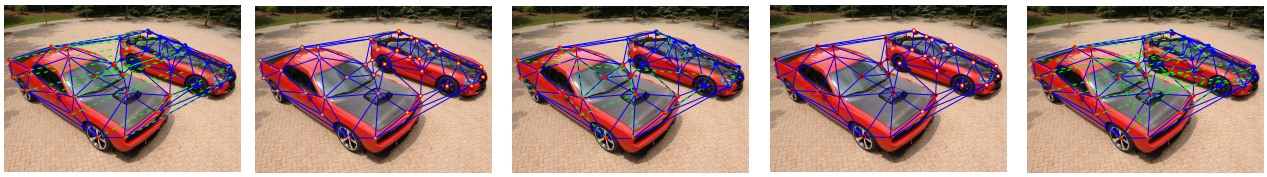
(a) Ground-truth: **20/20** (e) Standard cuts: **16/20** (i) Match by IBGP: **3/20** (m) CutMatch-cuts: **19/20** (q) CutMatch-match: **20/20**



(b) Ground-truth: **25/25** (f) Standard cuts: **21/25** (j) Match by IBGP: **5/25** (n) CutMatch-cuts: **25/25** (r) CutMatch-match: **16/25**



(c) Ground-truth: **15/15** (g) Standard cuts: **13/15** (k) Match by IBGP: **1/15** (o) CutMatch-cuts: **15/15** (s) CutMatch-match: **7/15**



(d) Ground-truth: **18/18** (h) Standard cuts: **15/18** (l) Match by IBGP: **0/18** (p) CutMatch-cuts: **15/18** (t) CutMatch-match: **4/18**

Figure 5. Results on real-world image by joint CutMatch and vanilla graph cuts or graph matching alone.

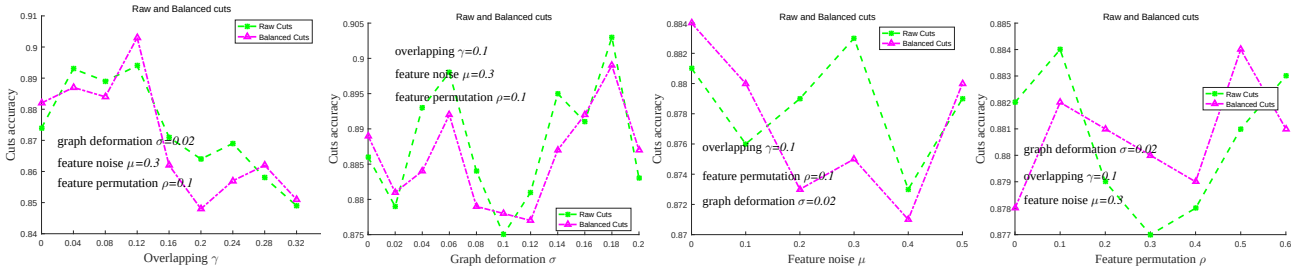



Figure 6. Performance of raw cuts and balanced cuts.

## Derivation of pseudo inverse

Consider the matrix $\mathbf{K} = \mathbf{I} - \frac{1}{n}\mathbf{1}\mathbf{1}^T$, and the corresponding SVD is $\mathbf{K} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T$. As $\mathbf{K}$ is symmetric, we have $\mathbf{U} = \mathbf{V}$. Note that $\mathbf{K}$ corresponds to the Laplacian of a complete graph with $n$ nodes, divided by $n$. Recall that Laplacian of a complete graph must have one 0 eigenvalue, and $n$ for all the remaining ones. Thus the scaled matrix $\mathbf{K}$ must have one 0 eigenvalue, and 1 for the rest. Denoting $\text{pinv}(\cdot)$ the psuedo inverse function, we have:

$$\begin{aligned}
\text{pinv}(\mathbf{K}) &= \text{pinv}(\mathbf{I} - \frac{1}{n}\mathbf{1}\mathbf{1}^T) \\
&= \text{pinv}\left(\mathbf{U}\begin{bmatrix}\mathbf{I} & \mathbf{0}\\ \mathbf{0}^T & 0\end{bmatrix}\mathbf{U}^T\right) \\
&= \left(\mathbf{U}\begin{bmatrix}\mathbf{I}^{-1} & \mathbf{0}\\ \mathbf{0}^T & 0\end{bmatrix}\mathbf{U}^T\right) \\
&= \left(\mathbf{U}\begin{bmatrix}\mathbf{I} & \mathbf{0}\\ \mathbf{0}^T & 0\end{bmatrix}\mathbf{U}^T\right) \\
&= \mathbf{K}
\end{aligned} \tag{30}$$

which gives the analytical form of the inverse.

## Supplementary experiment on balanced cuts

We present performance of balanced cuts on synthetic graphs. To enforce each partition has the same size, we calculate the balanced cuts as follows. Given the graph Laplacian $\mathbf{L}^g$, we first obtain the optimal solution to the Rayleigh problem (the same as in the main body):

$$\mathbf{y}^* = \arg\min_{\mathbf{y}} \frac{\mathbf{y}^T \mathbf{L}^g \mathbf{y}}{\mathbf{y}^T \mathbf{y}} \tag{31}$$

Then for each element of vector $\mathbf{y}^*$, we find its median value $\mathbf{y}_m$. For an element $\mathbf{y}_i$, we category node $i$ to the first partition if $\mathbf{y}_i \geq \mathbf{y}_m$, and to the second partition if $\mathbf{y}_i < \mathbf{y}_m$. For comparison, we only additionally present the performance of raw graph cuts which is sufficient to illustrate that balanced cuts is no more superior than raw cuts in our settings. In synthetic test, we generate 50 graphs for each parameter value combination, and calculate the average cuts performance. All the other setting follow the ones from the manuscript. Fig 6 summarizes the experimental result, and we can observe that the performance of balanced cuts is very similar to that of raw cuts. Note the range of the accuracy on the vertical axis.

## References


[1] K. Andreev and H. Räcke. Balanced graph partitioning. In *Proceedings of the sixteenth annual ACM symposium on Parallelism in algorithms and architectures*, pages 120–124. ACM, 2004.

[2] D. Bruff. The assignment problem and the hungarian method. *Notes for Math*, 20:29–47, 2005.

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

[4] Y. Chen, G. Leonidas, and Q. Huang. Matching partially similar objects via matrix completion. In *ICML*, 2014.

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

[6] M. Cho, J. Lee, and K. Lee. Reweighted random walks for graph matching. *ECCV*, pages 492–505, 2010.

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

[8] T. Collins, P. Mesejo, and A. Bartoli. An analysis of errors in graph-based keypoint matching and proposed solutions. In *ECCV*, 2014.

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

[10] R. L. Dykstra. An algorithm for restricted least squares regression. *Journal of the American Statistical Association*, 78(384):837–842, 1983.

[11] S. Gold and A. Rangarajan. A graduated assignment algorithm for graph matching. *TPAMI*, 18(4):377–388, 1996.

[12] O. Goldschmidt and D. S. Hochbaum. A polynomial algorithm for the k-cut problem for fixed k. *Mathematics of operations research*, 19(1):24–37, 1994.

[13] L. Guibas and J. Stolfi. Primitives for the manipulation of general subdivisions and the computation of voronoi. *ACM transactions on graphics (TOG)*, 4(2):74–123, 1985.

[14] W. W. Hager. Updating the inverse of a matrix. *SIAM review*, 31(2):221–239, 1989.

[15] V. Kolmogorov and R. Zabin. What energy functions can be minimized via graph cuts? *TPAMI*, 26(2):147–159, 2004.

[16] V. Kwatra, A. Schödl, I. Essa, G. Turk, and A. Bobick. Graphcut textures: image and video synthesis using graph cuts. In *ACM Transactions on Graphics (ToG)*, volume 22, pages 277–286. ACM, 2003.

[17] M. Leordeanu and M. Hebert. A spectral technique for correspondence problems using pairwise constraints. In *ICCV*, volume 2, pages 1482–1489. IEEE, 2005.

[18] M. Leordeanu, M. Hebert, and R. Sukthankar. An integer projected fixed point method for graph matching and map inference. In *NIPS*, pages 1114–1122, 2009.

[19] J. Munkres. Algorithms for the assignment and transportation problems. *Journal of the Society for Industrial & Applied Mathematics*, 5(1):32–38, 1957.

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

[21] J. Shi and J. Malik. Normalized cuts and image segmentation. *TPAMI*, 22(8):888–905, 2000.

[22] X. Shi, H. Ling, W. Hu, J. Xing, and Y. Zhang. Tensor power iteration for multi-graph matching. In *CVPR*, 2016.

[23] M. F. Tappen and W. T. Freeman. Comparison of graph cuts with belief propagation for stereo, using identical mrf parameters. In *ICCV*, volume 2, pages 900–906, 2003.

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

[25] L. N. Trefethen and D. Bau, III. Numerical linear algebra. In *p. 254. ISBN 978-0-89871-957-4.* SIAM, 1997.

[26] F. Wang, P. Li, and A. C. Konig. Learning a bi-stochastic data similarity matrix. In *ICDM*, pages 551–560. IEEE, 2010.

[27] T. Wang, H. Ling, C. Lang, and J. Wu. Branching path following for graph matching. In *ECCV*, pages 508–523. Springer, 2016.

[28] Y.-C. Wei and C.-K. Cheng. Towards efficient hierarchical designs by ratio cut partitioning. In *Computer-Aided Design, 1989. ICCAD-89. Digest of Technical Papers., 1989 IEEE International Conference on*, pages 298–301. IEEE, 1989.

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

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

[31] J. Yan, X. Yin, W. Lin, C. Deng, H. Zha, and X. Yang. A short survey of recent advances in graph matching. In *ICMR*, 2016.

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

[33] F. Zhou and F. De la Torre. Factorized graph matching. In *CVPR*, pages 127–134. IEEE, 2012.