Branching Path Following for Graph Matching

Tao Wang[1,2], Haibin Ling[1,3], Congyan Lang[2], and Jun Wu[2]

[1]Meitu HiScene Lab, HiScene Information Technologies, Shanghai, China
[2]School of Computer &Information Technology, Beijing Jiaotong University, China
[3]Computer & Information Sciences Department, Temple University, USA
twang@bjtu.edu.cn, hbling@temple.edu, {cylang,shfeng,wuj}@bjtu.edu.cn

Abstract. Recently, graph matching algorithms utilizing the path following strategy have exhibited state-of-the-art performances. However, the paths computed in these algorithms often contain singular points, which usually hurt the matching performance. To deal with this issue, in this paper we propose a novel path following strategy, named *branching path following* (BPF), which consequently improves graph matching performance. In particular, we first propose a singular point detector by solving an KKT system, and then design a branch switching method to seek for better paths at singular points. Using BPF, a new graph matching algorithm named *BPF-G* is developed by applying BPF to a recently proposed path following algorithm named GNCCP (Liu & Qiao 2014). For evaluation, we compare BPF-G with several recently proposed graph matching algorithms on a synthetic dataset and four public benchmark datasets. Experimental results show that our approach achieves remarkable improvement in matching accuracy and outperforms other algorithms.

Keywords: Graph Matching, Path Following, Numerical Continuation, Singular Point, Branch Switching

1 Introduction

Graph matching is a fundamental problem in computer science and closely relates to many computer vision problems including feature registration [1–3], shape matching [4–6], object recognition [7, 8], visual tracking [9], activity analysis [10], etc. Despite decades of research effort devoted to graph matching, it remains a challenging problem due to the non-convexity in the objective function and the constraints over the solutions. A typical way is to utilize relaxation to harness the solution searching. Popular algorithms include, but not limited to, three categories: spectral relaxation [11–13], continuous optimization [14–18] and probabilistic modeling [19, 20].

Among recently proposed graph matching algorithms, the ones utilizing the path-following strategy have exhibited state-of-the-art performances [15–18]. These *path following algorithms* reformulate graph matching as a *convex-concave relaxation procedure* (CCRP) problem, which is solved by interpolating between two simpler approximate formulations, and they use the *path following* strategy to recast iteratively the bistochastic matrix solution in the discrete domain. The path following algorithms can be viewed as special cases of the *numerical continuation method* (NCM) [21], which computes approximate solutions of parameterized nonlinear equation systems. These algorithms succeed at *regular points* but may fail at *singular points* (details in Sec. 4). It therefore demands research attention on how to address this issue to improve matching performance.

Motivated by above discussion, we propose a novel path following strategy, *branching path following* (BPF), to improve path following graph matching algorithms. In particular, BPF extends the traditional path following strategy by branching new paths at singular points. It first discovers singular points on the original path by determining the Jacobian of the associated KKT system, and then branches a new path at each singular point using the *pseudo-arclength continuation* method [22, 23]. After searching along all branching paths, BPF chooses the best one in terms of the objective function as the final solution. Since the original path is always searched, BPF is guaranteed to achieve better or the same optimization solution, and thus the matching performance. Using the BPF strategy, we develop a new graph matching algorithm, named BPF-G, by applying BPF to the GNCCP (*graduated nonconvexity and concavity procedure*) algorithm [17]. Note that GNCCP is chosen since it is one of the latest path following algorithms, while BPF is by no means limited to working with GNCCP.

For a thorough evaluation, we test the proposed BPF-G algorithm on four popular benchmarks and a synthetic dataset. Experimental results show that, the proposed algorithm significantly improves the path following procedure and outperforms state-of-the-art graph matching algorithms in comparison.

In summary, our main contribution lies in the new path following strategy for graph matching, and the contribution is three-fold: (1) we discuss the pitfalls of path following algorithms at singular points, and propose an efficient singular point discovery method; (2) we design a novel branching path following strategy to bypass these pitfalls and thus improve matching performance; and (3) we develop a new graph matching algorithm by applying the proposed BPF strategy to the GNCCP algorithm, and demonstrate the effectiveness of the algorithm in a thorough evaluation.

In the rest of the paper, Sec. 2 summarizes related work; then Sec. 3 reviews path following algorithms and Sec. 4 discusses the numerical continuation interpretation; after that, Sec. 5 introduces the proposed branching path following strategy, followed by experimental validation in Sec. 6 and conclusion in Sec. 7.

2 Related work

Graph matching has been investigated for decades and many algorithms have been invented, as summarized in [24, 25]. In general, graph matching has a combinatorial nature that makes the global optimum solution hardly available. As a result, approximate solutions are commonly applied to graph matching. In this section we review studies that relate the most to ours, and leave general graph matching research to the surveys mentioned above. Some sampled latest studies include [26] that uses discrete methods in the linear approximation framework, and [27] that adapts discrete tabu search for graph matching.

A popular way to approximate graph matching is based on spectral relaxation with notable work by Leordeanu and Hebert [11], who model graph matching with spectral relaxation and propose an eigen-analysis solution. Later, the work is extended by Cour et al. [12] by first encoding affine constraints into the spectral decomposition and then applying bistochastic normalization. Cho et al. [13] reformulate graph matching as a vertex selection problem and introduce an affinity-preserving random walk algorithm. From a different perspective, Zass and Shashua [19] present a probabilistic framework for (hyper-)graph matching. The two lines somewhat merge in Egozi et al. [20], where a probabilistic view of the spectral relaxation scheme is presented.

Being inherently a discrete optimization problem, graph matching is often relaxed to continuous domain and many important algorithms have been designed on top of the relaxation. For example, Gold and Rangarajan [14] propose the graduated assignment algorithm to iteratively solve a series of linear approximations of the cost function using Taylor expansion. Leordeanu and Hebert [28] develop an integer projection algorithm to optimize the objective function in the integer domain. The studies that related most to ours are the so-called *path following* one. In particular, Zaslavskiy et al. [15] reformulate graph matching as a *convex-concave relaxation procedure* (CCRP) problem and then solve it by interpolating between simpler relaxed formulations. More specifically, the *path following* algorithm proposed by them iteratively searches a solution by tracing a path of local minima of a series of functions that linearly interpolate between the two relaxations. Later, Zhou and Torre [16] apply the similar strategy, and factorize an affinity matrix into a Kronecker product of smaller matrices, each of them encodes the structure of the graphs and the affinities between vertices and between edges. Liu and Qiao [17] propose the *graduated nonconvexity and concavity procedure* (GNCCP) to equivalently realize CCRP on partial permutation matrix, and GNCCP provides a much simpler way for CCRP without explicitly involving the convex or concave relaxation. Wang and Ling [29] propose a novel search strategy with adaptive path estimation to improve the computational efficiency of the path following algorithms.

Our work falls in the group using path following algorithms, but focuses on improving the path following strategy itself, which is not fully explored in previous studies. For this, we propose a novel *branching path following* (BPF) strategy, which is shown to effectively boost the graph matching performance as demonstrated in thorough evaluation (Sec. 6).

3 Path following for graph matching

3.1 Problem formulation

An undirected graph of n vertices can be represented by $\mathbb{G} = (\mathbb{V}, \mathbb{E})$, where $\mathbb{V} = \{v_1, \ldots, v_n\}$ and $\mathbb{E} \subseteq \mathbb{V} \times \mathbb{V}$ denote the vertex and edge sets, respectively.

A graph is often conveniently represented by a symmetric adjacency matrix $A \in \mathbb{R}^{n\times n}$, such that $A_{ij} > 0$ if and only if there is an edge between v_i and v_j.

For graph matching, given two graphs $\mathbb{G}^{(i)} = (\mathbb{V}^{(i)}, \mathbb{E}^{(i)})$ of size n_i, $i = 1, 2$, the problem is to find a vertex correspondence $X \in \{0,1\}^{n_1\times n_2}$ between $\mathbb{G}^{(1)}$ and $\mathbb{G}^{(2)}$ in favor of the following global consistency:

$$\mathcal{E}_1(X) = \sum_{i_1,i_2} c_{i_1 i_2} X_{i_1 i_2} + \sum_{i_1,i_2,j_1,j_2} d_{i_1 j_1 i_2 j_2} X_{i_1 i_2} X_{j_1 j_2}, \tag{1}$$

where $c_{i_1 i_2}$ measures the consistency between the i_1-th vertex in $\mathbb{G}^{(1)}$ and the i_2-th vertex in $\mathbb{G}^{(2)}$, and $d_{i_1 j_1 i_2 j_2}$ the the consistency between edge (i_1, j_1) in $\mathbb{G}^{(1)}$ and edge (i_2, j_2) in $\mathbb{G}^{(2)}$. The correspondence matrix X denotes matching result, i.e., $X_{i_1 i_2} = 1$ if and only if $v_{i_1} \in \mathbb{V}^{(1)}$ corresponds to $v_{i_2} \in \mathbb{V}^{(2)}$. In practice, the matching is often restricted to be one-to-one, which requires $X\mathbf{1}_{n_2} \leq \mathbf{1}_{n_1}$ and $X^\top \mathbf{1}_{n_1} \leq \mathbf{1}_{n_2}$, where $\mathbf{1}_n$ denotes a vector of n ones.

Let $A^{(i)}$ be the adjacency matrix for $\mathbb{G}^{(i)}$, $i = 1, 2$, a more commonly used formulation for graph matching is defined as

$$\mathcal{E}_2(X) = \mathrm{tr}(C^\top X) + \alpha \|A^{(1)} - XA^{(2)}X^\top\|_F^2, \tag{2}$$

where $C = (c_{i_1 i_2}) \in \mathbb{R}^{n_1\times n_2}$ is the vertex consistency matrix, $\alpha \geq 0$ the weight balancing between the vertex and edge comparisons, and $\|\cdot\|_F$ the Frobenius norm.

A more general formulation of Eq. (1) is formulated in a pairwise compatibility form

$$\mathcal{E}_3(\mathbf{x}) = \mathbf{x}^\top K \mathbf{x}, \tag{3}$$

where $\mathbf{x} \doteq \mathrm{vec}(X) \in \{0,1\}^{n_1 n_2}$ is the vectorized version of matrix X and $K \in \mathbb{R}^{n_1 n_2 \times n_1 n_2}$ is the corresponding affinity matrix defined as:

$$K_{\mathrm{ind}(i_1,i_2)\mathrm{ind}(j_1,j_2)} = \begin{cases} c_{i_1 i_2} & \text{if } i_1 = j_1 \text{ and } i_2 = j_2, \\ d_{i_1 j_1 i_2 j_2} & \text{if } A^{(1)}_{i_1 j_1} A^{(2)}_{i_2 j_2} > 0, \\ 0 & \text{otherwise.} \end{cases} \tag{4}$$

while $\mathrm{ind}(\cdot,\cdot)$ is a bijection mapping a vertex correspondence to an integer index.

In this paper, we mainly discuss and test graph matching algorithms for Eq. (3) since it encodes not only the difference of edge weight but also many complex graph compatibility functions.

3.2 The path following algorithm

In [15], Zaslavskiy et al. introduce the *convex-concave relaxation procedure* (CCRP) into the graph matching problem by reformulating it as interpolation between two relaxed and simpler formulations. The first relaxation is obtained by expanding the convex quadratic function $\mathcal{E}_2(X)$ from the set of permutation matrices $\mathcal{P}$ to the set of doubly stochastic matrices $\mathcal{D}$. The second relaxation is a concave function

$$\min_{X\in\mathcal{D}} \mathcal{E}_4(X) = -\mathrm{tr}(\Delta X) - 2\mathrm{vec}(X)^\top \big((L^{(1)})^\top \otimes (L^{(2)})^\top\big)\mathrm{vec}(X), \tag{5}$$

where Δ is a matrix with element $\Delta_{ij} = \big(D^{(1)}_{ii} - D^{(2)}_{jj}\big)^2$; $D^{(i)}$ and $L^{(i)}$ represent respectively the diagonal degree matrix and the Laplacian matrix of an adjacency matrix $A^{(i)}$, $i = 1, 2$; and $\otimes$ denotes the Kronecker product. A key property is that the optimum solution of $\mathcal{E}_4(X)$ over $\mathcal{P}$ is the solution of the original graph matching problem.

The *path following* strategy proposed in [15] can be interpreted as an iterative procedure that smoothly projects an initial solution of $\mathcal{E}_2$ in the continuous space $\mathcal{D}$ to the discrete space $\mathcal{P}$ by tracking a path of local minima of a series of functionals $\mathcal{E}_\lambda$ over $\mathcal{D}$

$$\mathcal{E}_\lambda = (1-\lambda)\mathcal{E}_2 + \lambda\mathcal{E}_4, \tag{6}$$

for $0 \le \lambda \le 1$. Each local minimum of $\mathcal{E}_{\lambda+d_\lambda}$ is gained by the Frank-Wolfe algorithm [30] given the local minimum of $\mathcal{E}_\lambda$ as the start point. Increasing λ from 0 to 1, this approach searches toward a local minimum of $\mathcal{E}_4$ from the unique local minimum of $\mathcal{E}_2$, and takes it as the final solution. For more details about the path following algorithm please see the literature [15].

Recently, Liu and Qiao [17] proposed the *graduated nonconvexity and concavity procedure* (GNCCP) to equivalently realize CCRP on partial permutation matrix without explicitly involving the convex or concave relaxation. As the latest work following the path following strategy, GNCCP provides a general optimization framework for the combinatorial optimization problems defined on the set of partial permutation matrices. In Section 5.3, we improve GNCCP by integrating the proposed branching path following strategy.

4 Numerical continuation method interpretation

In this section, we interpret the path following algorithms in a numerical continuation view, and then discuss their pitfalls due singular points. The discussion will guide the subsequent extension on these algorithms.

4.1 KKT system

According to the path following strategy described above and by converting $\mathcal{D}$ into constraints, we need to solve a series of optimization problems with equality and inequality constraints parameterized by λ:

$$\begin{aligned} \mathbf{x}^* &= \arg\max_{\mathbf{x}} \mathcal{E}_\lambda(\mathbf{x}), \\ \text{s.t.} & \begin{cases} B\mathbf{x} = \mathbf{1}_{2n}, \\ \quad x \ge \mathbf{0}_{n^2}. \end{cases} \end{aligned} \tag{7}$$

where $B\mathbf{x} = \mathbf{1}_{2n}$ encodes the one-to-one matching constraints ($B \in \mathbb{R}^{2n\times n^2}$).

Using Lagrange multipliers α_i and KKT multipliers μ_i, the above constrained problem can be converted to the following unconstrained one

$$\mathbf{x}^* = \arg\max_{\mathbf{x}} \Big(\mathcal{E}_\lambda(\mathbf{x}) + \sum_{i=1}^{2n} \alpha_i h_i(\mathbf{x}) - \sum_{i=1}^{n^2} \mu_i g_i(\mathbf{x})\Big), \tag{8}$$

$$\text{where } \begin{cases} h_i(\mathbf{x}) = B(i,:)\mathbf{x} - 1, \\ g_i(\mathbf{x}) = -\mathbf{x}_i. \end{cases} \tag{9}$$

This results in the following system of Karush-Kuhn-Tucker (KKT) equations [31]:

$$\begin{cases} \nabla\mathcal{E}_\lambda(\mathbf{x}) + B^\top\alpha + \mu = \mathbf{0}_{n^2}, & \\ h_i(\mathbf{x}) = 0, & 1 \le i \le 2n, \\ \mu_i g_i(\mathbf{x}) = 0, & 1 \le i \le n^2, \\ \mu_i \ge 0, & 1 \le i \le n^2. \end{cases} \tag{10}$$

In the next subsection we formulate the KKT system as a constrained nonlinear system $F(\lambda, \mathbf{x}, \alpha, \mu) = \mathbf{0}$, s.t. $\mu \ge \mathbf{0}$.

4.2 Numerical continuation method

The existing path following algorithms (PATH [15], FGM [16], GNCCP [17]) can be viewed as special cases of the *numerical continuation method* (NCM) [21]. In general, NCM computes approximate solutions of parameterized nonlinear equation systems, and it estimates curves given in the following implicit form:

$$F(\lambda, u) = \mathbf{0}_m, \text{ where } F \text{ is a mapping: } \mathbb{R}^{m+1} \to \mathbb{R}^m. \tag{11}$$

This method works as well in the presence of constraints on any or all of the variables [32, 33]. In particular, for graph matching, we have $u = [\mathbf{x}^\top, \alpha^\top, \mu^\top]^\top$ and $m = 2n^2 + 2n$.

Most solutions of nonlinear equation systems are iterative methods. For a particular parameter value λ_0, a mapping is repeatedly applied to an initial guess u_0. In fact, the existing PATH, FGM and GNCCP algorithms correspond to a particular implementation of the so-called *generic predictor corrector* (GPC) approach [21]. The solution at a specific λ is used as the initial guess for the solution at $\lambda + \Delta\lambda$. With $\Delta\lambda$ sufficiently small the iteration applied to the initial guess converges [21].

4.3 Pitfalls at singular points

A *solution component* $\Gamma(\lambda_0, u_0)$ of the nonlinear system F is a set of points (λ, u) such that $F(\lambda, u) = 0$ and these points are connected to the initial solution (λ_0, u_0) by a path of solutions. A *regular point* of F is a point (λ, u) at which the Jacobian of F is of full rank, while a *singular point* of F is a point (λ, u) at which the Jacobian of F is rank deficient. As discussed in [23], near a regular point the solution component is an isolated curve passing through the regular point. By contrast, for a singular point, there may be multiple curves passing through it. The local structure of a point in Γ is determined by high-order derivatives of F.

An advantage of the GPC approach is that it uses the solution for the original problem as a black box where all that required is an initial solution. However, this approach may fail at singular points, where the branch of solutions turns around [23]. In general, solution components Γ of a nonlinear system are branching curves where the branching points are singular [34]. Therefore, for problems with singular points, more sophisticated handling is desired.

5 Branching path following

In this section, we propose the *branching path following* (BPF) strategy that branches new curves at singular points toward potentially better matching results. BPF contains two main steps: singular point discovery and branch switching, as described in the following subsections. In the last subsection, we apply BPF to GNCCP to develop a new graph matching algorithm.

5.1 Singular points discovery

The first step in BPF is to discover singular points. Theoretically, these points can be detected by checking whether the Jacobian of F is of full rank. However, it is impractical to check discrete samples by sampling λ_i, since these samples are rarely able to cover the exact singular points.

Denote J_λ the Jacobian of F parameterized by λ. A singular point (λ, u) should have $|J_\lambda| = 0$. A reasonable assumption is that the curve formed by $(\lambda, |J_\lambda|)$ over λ is continuous. This implies there is at least one singular point between two points (λ_1, u_1) and (λ_2, u_2) if $|J_{\lambda_1}|$ and $|J_{\lambda_2}|$ have different signs.

Thus inspired, we design a simple yet effective way for singular point discovery by checking the signs of determinants of Jacobian on consecutive sampled points. Specifically, denote (λ_t, u_t) the point at iteration t in the path, we mark (λ_t, u_t) as a singular point if $|J_{\lambda_t}||J_{\lambda_{t+1}}| \leq 0$.

It is computationally expensive to decide determinants of large Jacobian matrices. Since we are only interested in the signs of these Jacobian matrices, we develop an efficient solution that first decompose the Jacobian matrices using the LU decomposition and then accumulate the signs of the diagonal elements of decomposed matrices.

5.2 Branch switching

Finding the solution curves passing a singular point is called branch switching. Once a singular point (λ_t, u_t) is discovered, we branch a new curve using the *pseudo-arclength continuation* (PAC) algorithm [22, 23].

PAC is based on the observation that an ideal parameterization of a curve is through arclength s. With the parameterization, we extend equations in (10) to the following form

$$G(\lambda, u) = \begin{Bmatrix} F(\lambda, u) \\ N(\lambda, u, s) \end{Bmatrix} = \begin{Bmatrix} 0 \\ 0 \end{Bmatrix}, \tag{12}$$

where the normalization equation $N(.) = 0$ approximates the statement that s represents arclength. Denote $(\dot{\lambda}, \dot{u})$ the tangent vector at point (λ_t, u_t), we have

$$N(\lambda, u, s) = \dot{u}^\top (u - u_t) + \dot{\lambda}^\top (\lambda - \lambda_t) - s = 0. \tag{13}$$

According to the implicit function theorem [35], the tangent vector $(\dot{\lambda}, \dot{u})$ can be computed as

$$(\dot{\lambda}, \dot{u}) = \big(\Delta\lambda, \Delta\lambda \frac{\partial u}{\partial \lambda}(\lambda_t, u_t)\big) = \big(\Delta\lambda, \Delta\lambda (F_u(\lambda_t, u_t))^{-1} F_\lambda(\lambda_t, u_t)\big). \tag{14}$$

However, it is inapplicable to the bifurcation point and is computationally expensive. We therefore propose approximation of the tangent vector using previous points for computational efficiency. The tangent vector at a previous iteration i ($i < t$) is approximated as $(\varDelta\lambda, u_{i+1} - u_i)$, and the tangent vector $(\dot{\lambda}, \dot{u})$ at t be estimated as

$$(\dot{\lambda}, \dot{u}) = \Big(\varDelta\lambda, \frac{\sum_{i=1}^{k} (k-i+1)(u_{t-i+1} - u_{t-i})}{\sum_{i=1}^{k} i}\Big), \tag{15}$$

where k controls the size of the window used for approximation. The motivation behind this approximation is the smoothness of the path, which implies the similarity between tangent vectors of consecutive iterations.

The Jacobian of the pseudo-arclength system is the bordered matrix $\begin{bmatrix} F_u & F_\lambda \\ \dot{u} & \dot{\lambda} \end{bmatrix}$. Appending the tangent vector as the last row can be seen as determining the coefficient of the null vector in the general solution of the Newton system [36] (particular solution plus an arbitrary multiple of the null vector).

Finally, we solve Eq. (12) using the *trust-region-reflective* algorithm [37], and then branch a new curve using the solution (λ^*, u^*) as the initial solution.

5.3 Applying BPF to GNCCP

Now we apply the BPF strategy to GNCCP [17] to develop our new graph matching algorithm named BPF-G.

Note that, in both steps of singular point discovery and branch switching, we need to compute the Jacobian of F in advance, which includes a parameter-dependent sub-matrix, the Jacobian of $\nabla\mathcal{E}_\lambda(\mathbf{x})$ (denoted as $J(\lambda, \mathbf{x})$). In GNCCP

$$\mathcal{E}_\lambda(\mathbf{x}) = \begin{cases} (1-\lambda)\mathbf{x}^\top K\mathbf{x} + \lambda \mathrm{tr}(\mathbf{x}^\top\mathbf{x}), \text{ if } \lambda \geq 0, \\ (1+\lambda)\mathbf{x}^\top K\mathbf{x} + \lambda \mathrm{tr}(\mathbf{x}^\top\mathbf{x}), \text{ if } \lambda < 0. \end{cases} \tag{16}$$

Applying BPF here, we have

$$J(\lambda, \mathbf{x}) = \begin{cases} (1-\lambda)(K^T + K) + 2\lambda\mathbf{I}, \text{ if } \lambda \geq 0, \\ (1+\lambda)(K^T + K) + 2\lambda\mathbf{I}, \text{ if } \lambda < 0. \end{cases} \tag{17}$$

The pseudo-code of BPF-G is shown in Algorithm 1. Note that, since the solution of the original algorithm is always in set T, the new algorithm is guaranteed to achieve the same or better solution in terms of objectives.

The computational complexity of GNCCP is $O(n^3)$ [17], where n is the vertex number of the graph. Thus, the computational complexity of our algorithm is $O(kn^3)$, where k is the number of explored additional branches. As a result, the complexity of the proposed algorithm roughly equals to $O(n^3)$ because k is a small bounded integer.

Algorithm 1 Branching Path Following.

```
{$T$ is the set of candidate solutions (end points of branching paths).}
Compute path $P$ by GNCCP.
Push the end point of $P$ into set $T$.
Discover the set of singular points $S$ from $P$.
for each point $(\lambda, \mathbf{x})$ in $S$ do
    Compute the solution $(\lambda^*, u^*)$ of Eq. (12).
    Compute the new path $P^*$ using $(\lambda^*, u^*)$ as initial point.
    Push the end point of $P^*$ int set $T$.
end for
Select the best solution from $T$ in terms of objectives.
```

6 Experiments

We compare the proposed BPF-G algorithm with four state-of-the-art graph matching algorithms including GNCCP [17], IPFP [28], RRWM [13] and PSM [20], and report experimental results on a synthetic dataset and four benchmark datasets. Two indicators, matching accuracy and objective ratio, are used to evaluate algorithms. Specifically, denote f_i the objective achieved by the i-th algorithm g_i, the objective ratio r_i of g_i is computed as $r_i = f_i / \max_k f_k$.

6.1 Synthetic dataset

We first perform a comparative evaluation of the algorithms on graphs randomly synthesized following the experimental protocol in [13]. For each trial, we construct two graphs, $\mathbb{G}^{(1)}$ and $\mathbb{G}^{(2)}$, with 20 inlier nodes and later add n_{out} outlier nodes to both graphs. The edges between nodes are randomly generated with respect to an edge density parameter ρ. Each edge (i, j) in the first graph $\mathbb{G}^{(1)}$ is assigned with a random edge weight $A^{(1)}_{ij}$ distributed uniformly in [0,1], and $A^{(2)}_{ab} = A^{(1)}_{ij} + \epsilon$ the edge weight of the corresponding edge (a, b) in $\mathbb{G}^{(2)}$ is perturbed by adding a random Gaussian noise $\epsilon \sim \mathcal{N}(0, \sigma^2)$. The edge affinity is computed as $K_{\mathrm{ind}(i,a)\mathrm{ind}(j,b)} = \exp\big(-(A^{(1)}_{ij} - A^{(2)}_{ab})^2/0.15\big)$ and the node affinity is set to zero.

We compare the performance of the algorithms under three different settings by varying the number of outliers n_{out}, edge density ρ and edge noise σ, respectively. For each setting, we construct 100 different pairs of graphs and evaluate the average matching accuracy and objective ratio. In the first setting (Fig. 1((a)), we fix edge density $\rho = 0.5$ and edge noise $\sigma = 0$, and increase the number of outliers n_{out} from 0 to 10. In the second setting (Fig. 1(b)), we change the edge noise parameter σ from 0 to 0.2 while fixing $n_{out} = 0$ and $\rho = 0.5$. In the last case (Fig. 1(c)), the edge density ρ ranges from 0.3 to 1, and the other two parameters are fixed as $n_{out} = 0$ and $\sigma = 0.1$.

It can be observed that in almost all of cases under varying parameters, our approach achieves the best performance in terms of both objective ratio and

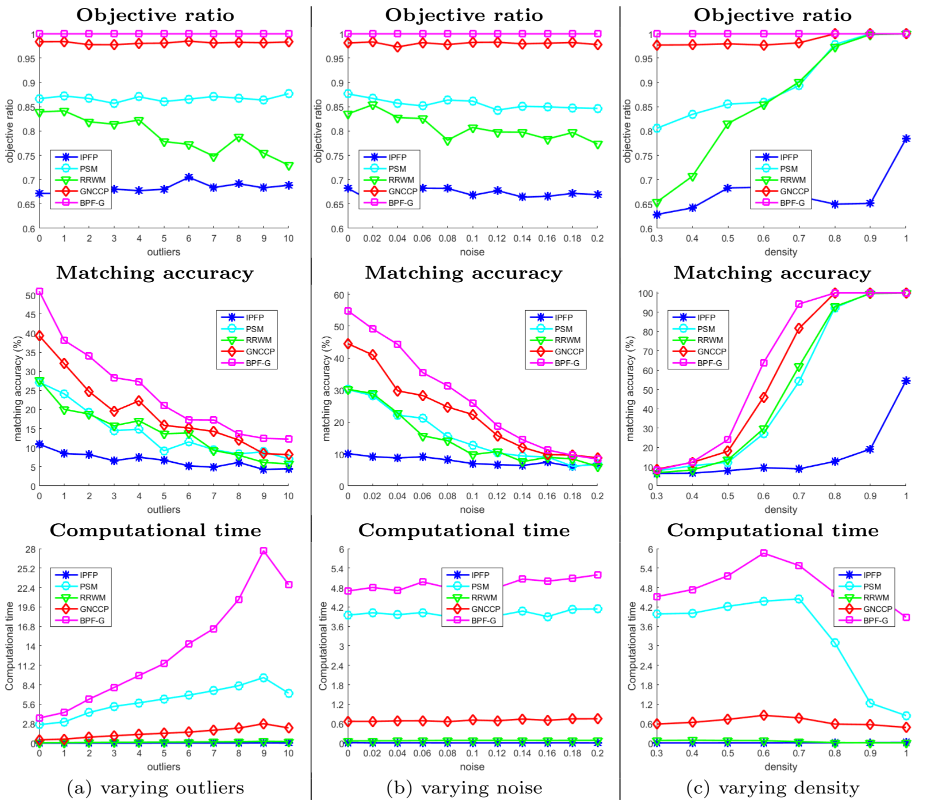

Fig. 1. Evaluation on the synthetic dataset by different settings.

matching accuracy. From Fig. 1(c), the PSM, RRWM and GNCCP algorithms are comparable to our approach when the graphs are close to full connections (the density parameter ρ near to 1). All algorithms fail to achieve reasonable solutions when graph pairs present extreme deformation or sparsity. The comparison on the running time is also provided in Fig. 1. Our approach spends several times of running time comparing to the GNCCP algorithm of which the multiple depends on the number of explored additional branches.

6.2 CMU house dataset

The CMU house dataset includes 111 frames of image sequences, where all sequences contain the same house object with transformation cross sequence gaps. In order to assess the matching accuracy, following [38, 2], 30 landmarks were manually tracked and labeled across all frames. We matched all possible image pairs, in total 560 pairs gapped by 10, 20, ..., 100 frames, where increasing sampling gaps implies the increase of deformation degree. To evaluate graph matching algorithms against noise, we use two different settings of nodes (n_1, n_2)

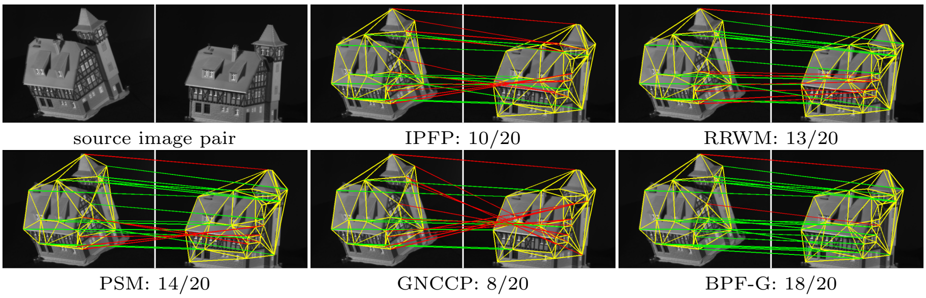

Fig. 2. An example of graph matching on the CMU house dataset. The algorithm, the number of true matches per ground truths for each subfigure are captioned. Graph edges are represented by yellow lines, true matches by green lines and false matches by red lines (best viewed in color, and the same style is also used for figures 5, 6 and 7).

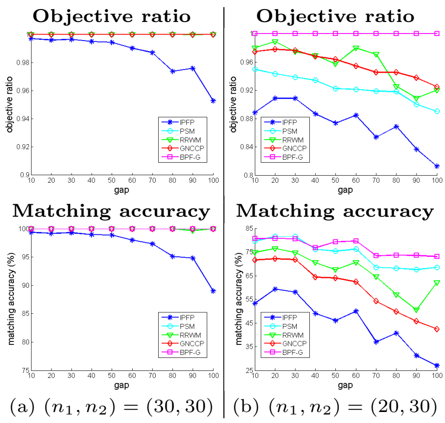

Fig. 3. Comparison of graph matching on the CMU house dataset.

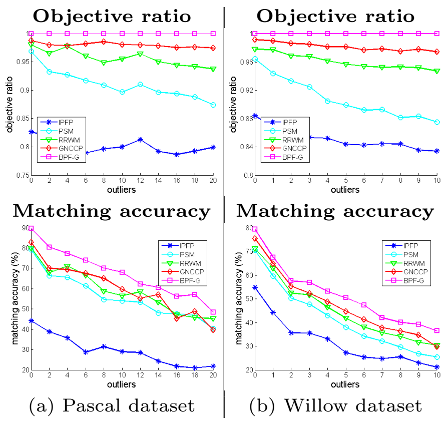

Fig. 4. Evaluation on (a) the Pascal dataset, and (b) the Willow dataset.

=(30,30) and (20, 30), where decreasing n_1 implies the increase of outlier. In the setting where $n_1 < 30$, n_1 points are randomly chosen out of the 30 landmark points.

We model each landmark as a graph node, and then build graph edges by Delaunay triangulation [39]. Each edge (i, j) is associated with a weight A_{ij} which is computed as the Euclidean distance between the connected nodes v_i and v_j. The node affinity is set to zero, and the edge affinity between edges (i, j) in $\mathbb{G}^{(1)}$ and (a, b) in $\mathbb{G}^{(2)}$ is computed as $K_{\text{ind}(i,a)\text{ind}(j,b)} = \exp(-(A_{ij}^{(1)} - A_{ab}^{(2)})^2/2500)$.

Fig. 2 presents an example for graph matching with 10 outliers and significant deformation. Fig. 3 shows the performance curves for $n_1 = 30$ and 20 with respect to variant sequence gaps. All algorithms except IPFP achieve perfect matching when no outliers existing ($n_1 = 30$). When we increase the number of outliers

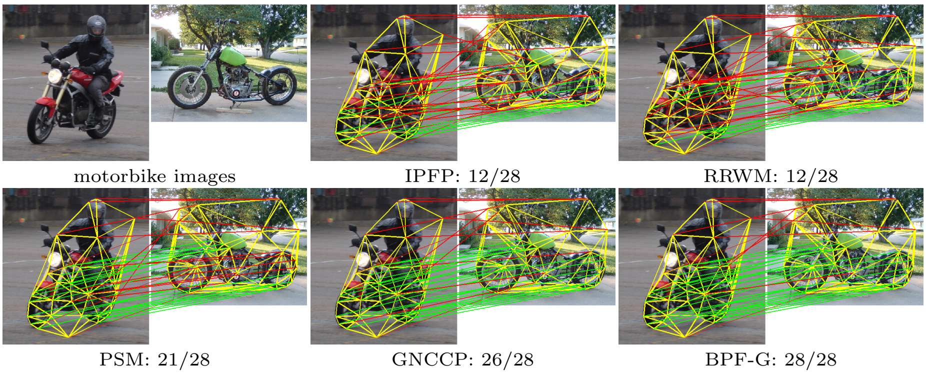

Fig. 5. A matching example of motorbike images in the Pascal dataset.

to 10 ($n_1 = 20$), our approach gains remarkable improvement in both accuracy and objective compared with the original GNCCP algorithm. It is interesting to see that PSM gains comparable matching accuracy to our approach with certain sequence gaps but achieves lower objectives.

6.3 Pascal dataset

The Pascal dataset [40] consists of 30 pairs of car images and 20 pairs of motorbike images selected from Pascal 2007. The authors provide detected feature points and manually labeled ground-truth correspondences for each pair of images. To evaluate the performance of each algorithm against noise, we randomly select $0 \sim 20$ outlier nodes from the background.

For each node v_i, we associate it with a feature p_i which is computed as its orientation of the normal vector at that point to the contour where the point was sampled. The node affinity between nodes v_i and v_j is consequently computed as $\exp(-|p_i - p_j|)$. We use Delaunay triangulation to build graph edges, and associate each edge (i, j) with two features d_{ij} and θ_{ij}, where d_{ij} is the pairwise distance between the connected nodes v_i and v_j, and θ_{ij} is the absolute angle between the edge and the horizontal line. Consequently, the edge affinity between edges (i, j) in $\mathbb{G}^{(1)}$ and (a, b) in $\mathbb{G}^{(2)}$ is computed as $K_{\text{ind}(i,a)\text{ind}(j,b)} = \exp(-(|d_{ij} - d_{ab}| + |\theta_{ij} - \theta_{ab}|)/2)$.

Fig. 5 presents an example for graph matching of motorbike images (with 10 outliers). The matching accuracy and objective ratio of each algorithm with respect to the outlier number was summarized in Fig. 4(a). It can be observed that our approach outperforms other algorithms remarkably in both matching accuracy and objective ratio. In the case when no outliers exist, our method achieves near 90% matching rate which is much higher than other ones.

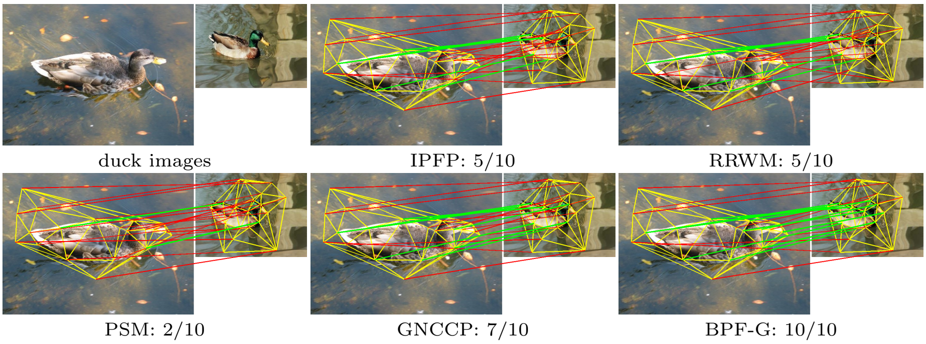

Fig. 6. A matching example of duck images on the Willow dataset.

6.4 Willow object dataset

In this experiment, we create 500 pairs of images using Willow object class dataset [41]. This dataset provides images of five classes, namely car, duck, face, motorbike and winebottle. Each class contains at least 40 images with different instances and 10 distinctive landmarks were manually labeled on the target object across all images in each class. We randomly select 100 pairs of images from each class respectively.

We use Hessian detector [42] to extract interesting points and SIFT descriptor [43] to represent the node attributes. To test the performance against noise, we randomly select $0 \sim 10$ outlier nodes from the background. We utilize the Delaunay triangulation to connect nodes and compute the affinity between nodes via their appearance similarity. Edge affinity is computed following the method used in Sec. 6.3.

Fig. 6 shows a representative example for graph matching selected from the duck class (with 10 outliers). The comparison on matching accuracy and objective ratio of each algorithm is summarized in Fig. 4(b). Our approach achieves remarkable improvement compared with GNCCP and outperforms other algorithms.

6.5 Caltech image dataset

The Caltech image dataset provided by Cho et al. [13] contains 30 pairs of real images. The authors provide detected MSER keypoints [44], initial matches, affinity matrix, and manually labeled ground-truth correspondences for each image pair. In [13], the low-quality candidate matches are filtered out according to the distance between SIFT features [43]. The affinity matrix is consequently computed by the mutual projection error function [45].

Fig. 7 shows a representative example for graph matching with significant deformation and plenty of repeated patterns. The matching accuracy and objective ratio of each algorithm was summarized in Table 1. Our approach gains remarkable improvement compared with the original GNCCP algorithm in both

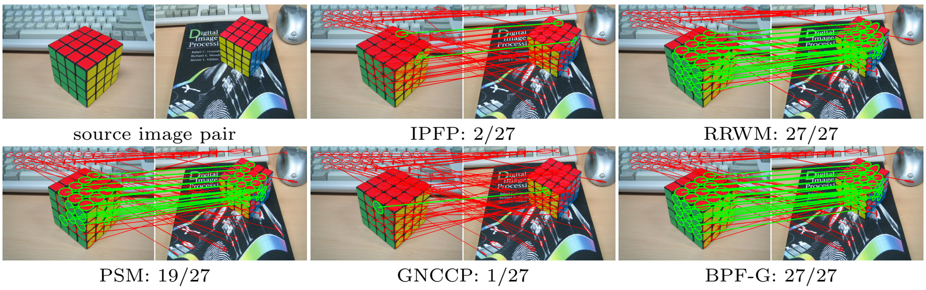

Fig. 7. A matching example on the Caltech dataset.

Table 1. Comparison of graph matching on the Caltech dataset (The top 1 result is indicated in **red** and top 2 in **blue**).

Algorithm	IPFP[28]	RRWM[13]	PSM[20]	GNCCP[17]	BPF-G (ours)
Objective ratio	0.91	0.88	0.75	**0.92**	**1**
Accuracy(%)	68.83	**75.49**	66.26	68.72	**73.56**

matching accuracy and objective ratio. It is interesting to see that RRWM performs better in terms of accuracy, whereas our approach obtains much higher objectives.

7 Conclusion

In this paper, we proposed a novel branching path following strategy for graph matching aiming to improve the matching performance. To avoid the pitfalls at singular points in the original path following strategy, our new strategy first discovers singular points and subsequently branches new paths from them seeking for potentially better solutions. We integrated the strategy into a state-of-the-art graph matching algorithm that utilizes the original path following strategy. Experimental results reveal that, our approach gains remarkable improvement on matching performance compared to the original algorithm, and also outperforms other state-of-the-art algorithms.

Acknowledgments. This work is supported by the National Nature Science Foundation of China (nos. 61300071, 61272352, 61472028, and 61301185), the National Key Research and Development Plan under Grant(No.2016YFB1001200), Beijing Natural Science Foundation (nos. 4142045 and 4162048), the Fundamental Research Funds for the Central Universities (no. 2015JBM029), and in part by the US National Science Foundation under Grants 1407156 and 1350521.

References

1. Serradell, E., Pinheiro, M.A., Sznitman, R., Kybic, J., Moreno-Noguer, F., Fua, P.: Non-rigid graph registration using active testing search. PAMI **37**(3) (2015) 625–638
2. Torresani, L., Kolmogorov, V., Rother, C.: Feature correspondence via graph matching: Models and global optimization. In: ECCV. (2008) 596–609
3. Wang, J., Li, S.: Query-driven iterated neighborhood graph search for large scale indexing. In: ACM MM. (2012) 179–188
4. Bai, X., Yang, X., Latecki, L.J., Liu, W., Tu, Z.: Learning context-sensitive shape similarity by graph transduction. PAMI **32**(5) (2010) 861–874
5. Michel, D., Oikonomidis, I., Argyros, A.A.: Scale invariant and deformation tolerant partial shape matching. Image Vision Comput. **29**(7) (2011) 459–469
6. Wang, T., Ling, H., Lang, C., Feng, S.: Symmetry-aware graph matching. Pattern Recognition **60** (2016) 657–668
7. Duchenne, O., Joulin, A., Ponce, J.: A graph-matching kernel for object categorization. In: ICCV. (2011) 1792–1799
8. Wu, J., Shen, H., Li, Y., Xiao, Z., Lu, M., Wang, C.: Learning a hybrid similarity measure for image retrieval. Pattern Recognition **46**(11) (2013) 2927–2939
9. Cai, Z., Wen, L., Lei, Z., Vasconcelos, N., Li, S.Z.: Robust deformable and occluded object tracking with dynamic graph. TIP **23**(12) (2014) 5497–5509
10. Chen, C.Y., Grauman, K.: Efficient activity detection with max-subgraph search. In: CVPR. (2012) 1274–1281
11. Leordeanu, M., Hebert, M.: A spectral technique for correspondence problems using pairwise constraints. In: ICCV. (2005) 1482–1489
12. Cour, T., Srinivasan, P., Shi, J.: Balanced graph matching. In: NIPS. (2007) 313–320
13. Cho, M., Lee, J., Lee, K.M.: Reweighted random walk for graph matching. In: ECCV. (2010) 492–505
14. Gold, S., Rangarajan, A.: A graduated assignment algorithm for graph matching. PAMI **18**(4) (1996) 377–388
15. Zaslavskiy, M., Bach, F., Vert, J.P.: A path following algorithm for the graph matching problem. PAMI **31**(12) (2009) 2227–2242
16. Zhou, F., De la Torre, F.: Factorized graph matching. In: CVPR. (2012) 127–134
17. Liu, Z., Qiao, H.: GNCCP - graduated nonconvexity and concavity procedure. PAMI **36**(6) (2014) 1258–1267
18. Liu, Z., Qiao, H., Yang, X., Hoi, S.C.H.: Graph matching by simplified convex-concave relaxation procedure. IJCV **109**(3) (2014) 169–186
19. Zass, R., Shashua, A.: Probabilistic graph and hypergraph matching. In: CVPR. (2008) 1–8
20. Egozi, A., Keller, Y., Guterman, H.: A probabilistic approach to spectral graph matching. PAMI **35**(1) (2013) 18–27
21. Allgower, E.L., Georg, K.: Numerical continuation methods. Springer (1990)
22. Keller, H.B.: Lectures on numerical methods in bifurcation theory. Tata Institute of Fundamental Research Lectures on Mathematics and Physics 79, Springer-Verlag, Berlin (1987)
23. Dickson, K.I., Kelley, C.T., Ipsen, I.C.F., Kevrekidis, I.G.: Condition estimates for pseudo-arclength continuation. SIAM J. NUMER. ANAL **45**(1) (2007) 263–276
24. Conte, D., Foggia, P., Sansone, C., Vento, M.: Thirty years of graph matching in pattern recognition. International Journal of Pattern Recognition and Artificial Intelligence **18**(3) (2004) 265–298

25. Foggia, P., Percannella, G., Vento, M.: Graph matching and learning in pattern recognition in the last 10 years. International Journal of Pattern Recognition and Artificial Intelligence **28**(1) (2014) 1–40
26. Yan, J., Zhang, C., Zha, H., Liu, W., Yang, X., Chu, S.M.: Discrete hyper-graph matching. In: CVPR. (2015) 1520–1528
27. Adamczewski, K., Suh, Y., Lee, K.M.: Discrete tabu search for graph matching. In: ICCV. (2015) 109–117
28. Leordeanu, M., Hebert, M.: An integer projected fixed point method for graph matching and map inference. In: NIPS. (2009)
29. Wang, T., Ling, H.: Path following with adaptive path estimation for graph matching. In: AAAI. (2016) 3625–3631
30. Frank, M., Wolfe, P.: An algorithm for quadratic programming. Naval Research Logistics Quarterly **3** (1956) 95–100
31. Kuhn, H.W., Tucker, A.W.: Nonlinear programming. In: Proceedings of 2nd Berkeley Symposium. (1951) 481–492
32. Shacham, M.: Numerical solution of constrained nonlinear algebraic equations. International Journal for Numerical Methods in Engineering **23** (1986) 1455–1481
33. Dankowicz, H., Schilder, F.: An extended continuation problem for bifurcation analysis in the presence of constraints. J. Comput. Nonlinear Dynam **6**(3) (2010) 1–14
34. Crisfield, M.A.: Non-linear finite element analysis solids and structure. Wiley (1996)
35. Kudryavtsev, L.: Implicit function. Encyclopedia of Mathematics, Springer (2001)
36. Deuflhard, P.: Newton methods for nonlinear problems - affine invariance and adaptive algorithms. Series Computational Mathematics 35, Springer (2006)
37. Branch, M.A., Coleman, T.F., Li, Y.: A subspace, interior and conjugate gradient method for large-scale bound-constrained minimization problems. SIAM J. SCI. COMPUT. **21**(1) (1999) 1–23
38. Caetano, T.S., Caelli, T., Schuurmans, D., Barone, D.A.: Graphical models and point pattern matching. PAMI **28**(10) (2006) 1646–1663
39. Lee, D.T., Schachter, B.J.: Two algorithms for constructing a delaunay triangulation. Int. J. Computer Information Sci **9** (1980) 219–242
40. Leordeanu, M., Sukthankar, R., Hebert, M.: Unsupervised learning for graph matching. IJCV **96**(1) (2012) 28–45
41. Cho, M., Alahari, K., Ponce, J.: Learning graphs to match. In: ICCV. (2013) 25–32
42. Mikolajczyk, K., Schmid, C.: An affine invariant interest point detector. In: ECCV. (2002) 128–142
43. Lowe, D.G.: Distinctive image features from scale-invariant keypoints. IJCV **60**(2) (2004) 91–110
44. Donoser, M., Bischof, H.: Efficient maximally stable extremal region (mser) tracking. In: CVPR. (2006) 553–560
45. Cho, M., Lee, J., Lee, K.M.: Feature correspondence and deformable object matching via agglomerative correspondence clustering. In: ICCV. (2009) 1280–1287