Path Following with Adaptive Path Estimation for Graph Matching

Tao Wang[1] **Haibin Ling**[2,3]
[1] School of Computer &Information Technology, Beijing Jiaotong University, Beijing 100044, China
[2] Computer & Information Sciences Department, Temple University, Philadelphia 19122, USA
[3] Meitu HiScene Lab, HiScene Information Technologies, Shanghai, China
twang@bjtu.edu.cn, hbling@temple.edu

Abstract

Graph matching plays an important role in many fields in computer vision. It is a well-known general NP-hard problem and has been investigated for decades. Among the large amount of algorithms for graph matching, the algorithms utilizing the *path following* strategy exhibited state-of-art performances. However, the main drawback of this category of algorithms lies in their high computational burden. In this paper, we propose a novel path following strategy for graph matching aiming to improve its computation efficiency. We first propose a path estimation method to reduce the computational cost at each iteration, and subsequently a method of adaptive step length to accelerate the convergence. The proposed approach is able to be integrated into all the algorithms that utilize the path following strategy. To validate our approach, we compare our approach with several recently proposed graph matching algorithms on three benchmark image datasets. Experimental results show that, our approach improves significantly the computation efficiency of the original algorithms, and offers similar or better matching results.

Introduction

Graph matching involves establishing correspondences between the vertices of two graphs. It is a fundamental problem in computer science and closely relates to a lot of research problems in computer vision, including for instance feature registration (Torresani, Kolmogorov, and Rother, 2008; Jiang, Yu, and Martin, 2011), image classification (Lazebnik, Schmid, and Ponce, 2006; Wu et al., 2013), object recognition (Duchenne, Joulin, and Ponce, 2011) and shape matching (Berg, Berg, and Malik, 2005). Graph matching is a well-known general NP-hard problem of which global optimum is hardly guaranteed for graphs of reasonable sizes. Consequently, approximate algorithms seeking acceptable suboptimal solutions are popular, and the research interest of this problem mainly focuses on investigating more accurate and faster algorithms.

Although the graph matching problem has been investigated for decades, it is still a challenging problem due to the objective function being non-convex and the constraints on the solution being combinatorial. There are a large amount of algorithms (Gold and Rangarajan, 1996; Leordeanu and Hebert, 2005; Cour, Srinivasan, and Shi, 2007; Cho, Lee, and Lee, 2010; Leordeanu and Hebert, 2009; Zaslavskiy, Bach, and Vert, 2009; Zhou and Torre, 2012; Liu and Qiao, 2014; Liu et al., 2014; Zass and Shashua, 2008; Egozi, Keller, and Guterman, 2013) utilizing constraints relaxation to harness the solution searching. Among these algorithms, the ones utilizing the path-following strategy exhibited state-of-art performances (Zaslavskiy, Bach, and Vert, 2009; Zhou and Torre, 2012; Liu and Qiao, 2014). These approaches reformulate graph matching as a *convex-concave relaxation procedure* (CCRP) problem, which is solved by interpolating between two approximate simpler formulations, and they use the *path following* strategy to recast iteratively the *bistochastic matrix* [1] solution in the discrete domain. However, all these algorithms suffer from high computational burden and thus are hard to be applied in large graphs. It therefore demands research attention on how to reduce the computational burden of this category of algorithms.

Addressing the above mentioned issues, we propose a novel path following strategy to improve the computational efficiency of the path following algorithms, which achieving similar or even better matching accuracy. In particular, our strategy extends the traditional path following strategy in two aspects: (1) We propose a path estimation method that uses the temporal solutions in previous iterations to estimate the solution in current iteration, and then adopt this estimation as the start point to search the local minima. Since such estimation is usually closer to the local minima than the previous solution, we spend less time on searching for local minima in each iteration. (2) We derive an adaptive step length on the path estimation to dynamically adjust the step length to further accelerate the convergence.

As a general strategy, our approach can be fused into all the algorithms that utilize the origin path following strategy. In this work, we integrate the proposed approach into two state-of-the-art graph matching algorithms, FGM (Zhou and Torre, 2012) and GNCCP (Liu and Qiao, 2014), denote the enhanced algorithms as FGM+EST and GNCCP+EST

Copyright © 2015, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved.

[1] A bistochastic matrix is a square matrix of nonnegative real number, of which each row (and column) sums to one.

respectively. For a thorough evaluation, we compare the two algorithms with not only their baselines but also some recently proposed algorithms (Cho, Lee, and Lee, 2010; Leordeanu and Hebert, 2009; Egozi, Keller, and Guterman, 2013) on three public benchmarks. The results show that the proposed adaptive path prediction strategy offers dramatic improvement in computational efficiency. Specifically, FGM+EST and GNCCP+EST reduces the computation time of FGM and GNCCP by about about 7 and 50 times, respectively. As for matching accuracy, the proposed algorithms produce better or similar performances as their baselines, and in general outperform other graph matching algorithms.

This paper have several contributions: (1) we propose a path estimation strategy to reduce the computational burden of path following algorithms for graph matching; (2) we propose using adaptive step lengths for further acceleration; and (3) we integrate the proposed strategies to improve two state-of-the-art graph matching algorithms, and demonstrated their effectiveness in a thorough evaluation.

Related works

Since graph matching is in nature a combinatorial problem and there is no known efficient algorithm for global optimum, a common way is to search for approximate solutions under relaxed conditions or constraints. There are a large number of literatures dedicated to this problem, among which two comprehensive surveys are reported in (Conte et al., 2004; Foggia, Percannella, and Vento, 2014). Thoroughly reviewing all graph matching papers is beyond the scope of this paper, in the following we sample some related ones that inspire our study.

As graph matching is inherently a discrete optimization problem, an important class of approximate algorithms reformulate it in the continuous domain by relaxing related constraints. The continuous solution is later discretized towards the final solution. Gold and Rangarajan (1996) propose the graduated assignment algorithm which uses Taylor expansions to iteratively solve a series of linear approximations of the cost function. Leordeanu and Hebert (2005) present an efficient and robust solution by spectral relaxation that ignores the assignment constraints and is solved via eigen-analysis. The assignment constraints are then enforced during the discretization step. The same author (Leordeanu and Hebert, 2009) later propose an integer projection algorithm which focuses on the discretization step taking a continuous solution as input. Cour, Srinivasan, and Shi (2007) extend the spectral relaxation (Leordeanu and Hebert, 2005) by encoding affine constraints into the spectral decomposition. Another contribution of this work is to apply bistochastic normalization to balance the affinity matrix. Cho, Lee, and Lee (2010) reformulate graph matching as a vertex selection problem and introduce an affinity-preserving random walk algorithm. The algorithm is proved to be equivalent to the spectral relaxation (Leordeanu and Hebert, 2005) for the *integer quadratic programming* (IQP) formulation.

The *path following* algorithm involving several recent work is attracting research attention due to its excellent matching performance. This group of algorithms is firstly proposed in (Zaslavskiy, Bach, and Vert, 2009), where graph matching is reformulated as a *convex-concave relaxation procedure* (CCRP) problem. The authors propose the path following algorithm to iteratively search the solution by tracking a path of local minima of a series of functions that linearly interpolate between the two relaxations. Zhou and Torre (2012) apply the similar strategy to the general framework defined on affinity matrices, and factorize an affinity matrix into a Kronecker product of smaller matrices. Liu and Qiao (2014) propose the *graduated nonconvexity and concavity procedure* (GNCCP) to equivalently realize CCRP through a much simpler way without involving the convex or concave relaxation explicitly.

Another interesting class of algorithms is the probabilistic approach inspired by Zass and Shashua (2008), where a probabilistic framework is employed for (hyper)graph matching. The authors regard the pairwise affinity matrix being an empirical estimate of the pairwise assignment probability, and assume the statistical independence between the assignments of different vertices in graphs. Egozi, Keller, and Guterman (2013) extend the work by dropping the second assumption to achieve improvement on matching performance, and present a probabilistic interpretation of the spectral relaxation scheme proposed by Leordeanu and Hebert (2005).

Focusing on graph matching, our work is inspired by or build on top of these previous studies. The algorithms using the path following strategy exhibited state-of-art performances but suffer from high computational burden. Our work aims to reduce the computational cost of these algorithms, and the excellent experimental results further validate its advantage.

Path following for graph matching

Problem statement and notations

A graph $\mathbb{G} = (\mathbb{V}, \mathbb{E})$ of size n is defined by a finite set of n vertices $\mathbb{V} = \{v_1, \dots, v_n\}$ and a set of edges $\mathbb{E} \subset \mathbb{V} \times \mathbb{V}$. Such a graph can be equivalently represented by a symmetric adjacency matrix A of size $n \times n$, where $A(i, j) = 1$ if there is an edge between vertices v_i and v_j, and 0 otherwise. An important generalization is a weighted graph which is defined by associating a nonnegative real-valued weight $A(i, j) = w(i, j)$ to all pair of vertices. This generalization is important because these weights are crucial in capturing structural relations among vertices in many applications. In the rest of this paper, all adjacency matrices mentioned are weighted with real values unless otherwise stated.

Given two graphs $\mathbb{G}_1 = (\mathbb{V}_1, \mathbb{E}_1)$ and $\mathbb{G}_2 = (\mathbb{V}_2, \mathbb{E}_2)$ of size n_1 and n_2 respectively, the graph matching problem consists of finding a correspondence $X \in \{0, 1\}^{n_1 \times n_2}$ between vertices of the two graphs that maximizes the following score of global consistency:

$$\begin{aligned}\mathcal{E}_1(X) = \quad & \sum_{i_1, i_2} c(i_1, i_2) X(i_1, i_2) + \\ & \sum_{i_1, i_2, j_1, j_2} d(i_1, j_1, i_2, j_2) X(i_1, i_2) X(j_1, j_2),\end{aligned} \tag{1}$$

where $c(\cdot)$ measures the vertex compatibility and $d(\cdot)$ the edge compatibility. The assignment matrix X denotes the vertex correspondence, i.e., $X(i_1, i_2) = 1$ if the i_1-th vertex of $\mathbb{G}_1$ corresponds to the i_2-th vertex of $\mathbb{G}_2$. In most cases, X is constrained to be a one-to-one matching, i.e., $X\mathbf{1}_{n_2} \leq \mathbf{1}_{n_1}$ and $X^T\mathbf{1}_{n_1} \leq \mathbf{1}_{n_2}$ ($\mathbf{1}_n$ denotes vectors of n ones).

A commonly specified formulation of Eq. (1) for graph matching is defined on adjacency matrices

$$\mathcal{E}_2(X) = \mathrm{tr}(C^T X) + \alpha\|A_1 - XA_2X^T\|_F^2, \qquad (2)$$

where C is the cost matrix for vertex assignment, $\alpha \geq 0$ is the confidence of comparison of edges, A_1 and A_2 are adjacency matrices of graphs $\mathbb{G}_1$ and $\mathbb{G}_2$ respectively, and $\|\cdot\|_F$ is the Frobenius norm.

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

$$\mathcal{E}_3(x) = x^T K x, \qquad (3)$$

where $x = \mathrm{vec}(X)^T \in \{0,1\}^{n_1 n_2}$ is an indicator vector and $K \in \mathbb{R}^{n_1 n_2 \times n_1 n_2}$ is the affinity matrix computed as follows:

$$K(i_1 i_2, j_1 j_2) = \begin{cases} c(i_1, i_2) & \text{if } i_1 = j_1 \text{ and } i_2 = j_2, \\ d(i_1, j_1, i_2, j_2) & \text{if } A_1(i_1, j_1) A_2(i_2, j_2) > 0, \\ 0 & \text{otherwise.} \end{cases} \qquad (4)$$

This formulation is more general than Eq. (2) due to its ability to encode not only the difference of edge weight but also many complex compatibility functions. Consequently, we mainly discuss and test graph matching algorithms for Eq. (3).

The path following algorithm

Zaslavskiy, Bach, and Vert (2009) first reformulates $\mathcal{E}_2(X)$ in Eq. (2) by interpolating between two simpler relaxed formulations. The first relaxation is obtained by expanding the convex quadratic function $\mathcal{E}_2(X)$ from the set of permutation matrices $\mathcal{P}$ on the set of bistochastic matrices $\mathcal{D}$:

$$\min_{X \in \mathcal{D}} \mathcal{E}_2(X), \qquad (5)$$

and the second relaxation is a concave function

$$\min_{X \in \mathcal{D}} \mathcal{E}_4(X) = -\mathrm{tr}(\Delta X) - 2\mathrm{vec}(X)^T (L_{A_1}^T \otimes L_{A_2}^T)\mathrm{vec}(X), \qquad (6)$$

where matrices $\Delta(i,j) = (D_{A_1}(i,i) - D_{A_2}(j,j))^2$, D_A and L_A represent the diagonal degree matrix and the Laplacian matrix of an adjacency matrix A respectively, and $\otimes$ denotes the Kronecker product of two matrices. The permutation matrix that minimizes $\mathcal{E}_4(X)$ over $\mathcal{P}$ is the solution of the graph matching problem. Note that, the minimum of $\mathcal{E}_4(X)$ on $\mathcal{D}$ is in fact in $\mathcal{P}$ because of its concavity. This property is very important to guarantee the final solution is discretized.

The authors proposed a path following strategy to search the solution by tracking a path of local minima over $\mathcal{D}$ of a series of functions

$$\mathcal{E}_\lambda = (1-\lambda)\mathcal{E}_2 + \lambda\mathcal{E}_4, \qquad (7)$$

for $0 \leq \lambda \leq 1$. This approach starts at $\lambda = 0$ finding the unique local minimum of $\mathcal{E}_2$, and then iteratively searches the local minimum of $\mathcal{E}_{\lambda+d_\lambda}$ given the local minimum of $\mathcal{E}_\lambda$ as the start point using the Frank-Wolfe algorithm (Franke and Wolf, 1956). It ends at $\lambda = 1$ and takes the local minimum of $\mathcal{E}_4$ as the final solution. More details about the path following algorithm please see the literature (Zaslavskiy, Bach, and Vert, 2009).

Zhou and Torre (2012) applied the similar path following strategy to optimize the more general formulation $\mathcal{E}_3$ in Eq. (3), 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 (2014) proposed the *graduated nonconvexity and concavity procedure* (GNCCP) to equivalently realize CCRP on partial permutation matrix. This approach provides a much simple way for CCRP without involving the convex or concave relaxation explicitly.

The proposed approach

Numerical continuation method

As discussed in (Zaslavskiy, Bach, and Vert, 2009), the path following algorithm may be considered as a special case of numerical continuation methods (Allgower and Georg, 2003). These methods allow to estimate curves given in the following implicit form:

$$T(u) = 0, \text{ where } T \text{ is a mapping: } \mathbb{R}^{m+1} \to \mathbb{R}^m. \qquad (8)$$

In fact, the path following strategy corresponds to a particular implementation of the so-called *generic predictor corrector* approach (Allgower and Georg, 2003) widely used in numerical continuation methods.

In the case of path following in graph matching, we need to solve a set of constrained optimization problems

$$x^* = \arg\min_x \mathcal{E}_\lambda(x), \quad s.t. \begin{cases} Bx = \mathbf{1}_{2n}, \\ x \geq \mathbf{0}_{n^2}. \end{cases} \qquad (9)$$

parameterized by λ, where $Bx = \mathbf{1}_{2n}$ encodes the one-to-one matching constraints ($B \in \mathbb{R}^{2n \times n^2}$). This results in the following system of Karush-Kuhn-Tucker (KKT) equations (Kuhn and Tucher, 1951):

$$\begin{cases} \nabla\mathcal{E}_\lambda(x) + B^T\alpha + \mu = \mathbf{0}_{n^2}, \\ Bx - \mathbf{1}_{2n} = \mathbf{0}_{2n}, \\ \mathrm{dot}(\mu, x) = \mathbf{0}_{n^2}. \end{cases} \qquad (10)$$

for each λ, where α_i are Lagrange multiplier encoding equality constraints and μ_i are KKT multipliers encoding inequality constraints. Denote $T(\lambda, x, \alpha, \mu)$ the left-hand part of the KKT equation system, we thus reformulate the graph matching problem to a parameterized nonlinear equation system $T(\lambda, x, \alpha, \mu) = \mathbf{0}$.

From the implicit function theorem (Kudryavtsev, 2001), we know that solutions of the parameterized nonlinear equation system $T(\lambda, x, \alpha, \mu) = \mathbf{0}$ forms a smooth one-dimensional curve which can be parameterized by λ.

Algorithm 1: Path Estimation(x, d, t, k)

% $x = \{x_1, x_2, ..., x_{t-1}\}$: solutions of previous iterations
% $d = \{d_1, d_2, ..., d_{t-1}\}$: step lengths of previous iterations
% s', x': estimated speed and solution point at current iteration
% t: index of current iteration
% k: parameter for estimation
if $t < k + 1$ **then**
    $x' \leftarrow x_{t-1}$
else
    $s' \leftarrow 0$
    for $i = 1$ to k **do**
        $s_{t-i} \leftarrow (x_{t-i} - x_{t-i-1})/d_{t-i-1}$
        $s' \leftarrow s' + (k - i)s_{t-i}$
    end for
    $s' \leftarrow s' / \sum_{i=1}^{k-1} i$
    $x' \leftarrow x_{t-1} + s' d_{t-1}$
end if
return x'

Algorithm 2: GNCCP+EST, updated GNCCP algorithm

1 : Initialization: $x_1 \in \mathcal{D}, d_1 \leftarrow d_{\min}, \lambda \leftarrow 1, t \leftarrow 1$
2 : **while** $\lambda > -1$ and x_t not discretized **do**
3 : $x'_t \leftarrow$ Estimation(x, d, t, k) % path estimation
4 : $p \leftarrow x'_t$ % start point to search local minimum
5 :   **while** p not converged **do**   % Frank-Wolfe algorithm
6 : $y = \arg\min_y \operatorname{tr}(\nabla\mathcal{E}_\lambda(p)^T y), \text{s.t.} y \in \mathcal{D}$
7 : $\alpha = \arg\min_\alpha \mathcal{E}_\lambda(p + \alpha(y - p)), \text{s.t.} 0 \leq \alpha \leq 1$
8 : $p \leftarrow p + \alpha(y - p)$
9 :   **end while**
10:     $t \leftarrow t + 1, x_t \leftarrow p$
11:     $d_t \leftarrow$ AdaptiveStep(x'_t, x_t, d, t, k) % Eq. (13)
12:     $\lambda \leftarrow \lambda - d_t$
11: **end while**
12: **return** x_t

Path estimation

As discussed in the previous section, the path of solution points for the set of functions $\mathcal{E}_\lambda$ is a smooth one-dimensional curve and can be parameterized by λ. We thus are able to estimate the t-th solution point using the previous $t - 1$ solution points.

At each iteration t, denote $x_{t-k}, ..., x_{t-1}$ the previous k points of temporal solutions, the moving speed of each point is computed as $s_i = x_i - x_{i-1}, t - k + 1 \leq i \leq t - 1$. We estimate the moving speed s'_t and point x'_t at iteration t as

$$s'_t = \frac{\sum_{i=1}^{k-1} (k-i) \cdot s_{t-i}}{\sum_{i=1}^{k-1} i}, \quad (11)$$

$$x'_t = x_{t-1} + s'_t. \quad (12)$$

The motivation behind this estimation is the smoothness of the path, which implies the similarity between moving speeds of consecutive iterations.

Once the estimation x'_t is ready, it is used as the start point to search for the local minima x_t. Compared with x_{t-1}, which is used in the original path following algorithms as the start point, x'_t is in general closer to x_t and thus the search procedure is more efficient. The excellent experimental results described later further validate this point. The path estimation algorithm is summarized in Algorithms 1.

Adaptive step length

It is usually hard to choose a proper step length d_λ of each iteration in practice. Too small steps lead to high computational burden while too large ones may hurt matching accuracy. To address the issue, we propose using adaptive step lengths. Denote d_t the step length for iteration t, and d_t should depend on the estimation x'_t. When x'_t is very close to the local minimum x_t, suggesting that high smoothness of the path around x_t, we can afford using a large d_t; by contrast, when x'_t is far from x_t, a small d_t is more reasonable.

Our adaptive step length uses a *growing rate* ρ to dynamically adjust the step length. Initially, we set the step length as $d_1 = d_{\min}$, where $d_{\min}$ represents the minimum step length allowed. Then, at each iteration t, d_t increases or decreases by ρ times depends on the similarity between x'_t and x_t. Specifically, we have

$$d_t \leftarrow \begin{cases} \rho d_{t-1}, & \text{if } \operatorname{tr}((x_t - x'_t)^\top (x_t - x'_t)) \leq \theta \\ \max(d_{\min}, \frac{d_{t-1}}{\rho}), & \text{otherwise} \end{cases} \quad (13)$$

where θ is the threshold of the error tolerance of the estimation. Once the update on d_t is done, we run the optimization (Frank-Wolfe) for the new value $\lambda + d_t$. The idea behind this adaptation schema is to choose d_t which keeps the estimation x'_t close enough to the real local minimum x_t.

Upgrading existing path following algorithms

This proposed methods of path estimation and adaptive step length are able to be combined with all algorithms that utilize the path following strategy, for instance PATH (Zaslavskiy, Bach, and Vert, 2009), FGM (Zhou and Torre, 2012) and GNCCP (Liu and Qiao, 2014). Due to space limitation, we only illustrate the upgrading of GNCCP.

The upgraded algorithm (GNCCP+EST) is shown in Algorithm 2, where $\mathcal{E}_\lambda$ in lines 6 and 7 is the linearly interpolation between the convex relaxation and the concave relaxation of the original objective function (more details in (Liu and Qiao, 2014)). The parameter k controls the number of previous solution points that are used to estimate the next solution point. We set $k = 10, d_{\min} = 0.002$ and $\theta = 0.001$ throughout our experiments.

Experiments

We compare the proposed FGM+EST and GNCCP+EST algorithms with the original FGM and GNCCP algorithms and three recent graph matching algorithms, IPFP (Leordeanu and Hebert, 2009), RRWM (Cho, Lee, and Lee, 2010) and PSM (Egozi, Keller, and Guterman, 2013), and report experimental results on three benchmark datasets.

Experiments on the CMU house dataset

The CMU house image sequences is commonly used to test the performance of graph matching algorithms (Cho, Lee, and Lee, 2010; Zhou and Torre, 2012; Duchenne, Joulin, and Ponce, 2011; Torresani, Kolmogorov, and Rother, 2008). This dataset includes 110 frames of image sequences, in which 30 landmarks were manually labeled across all

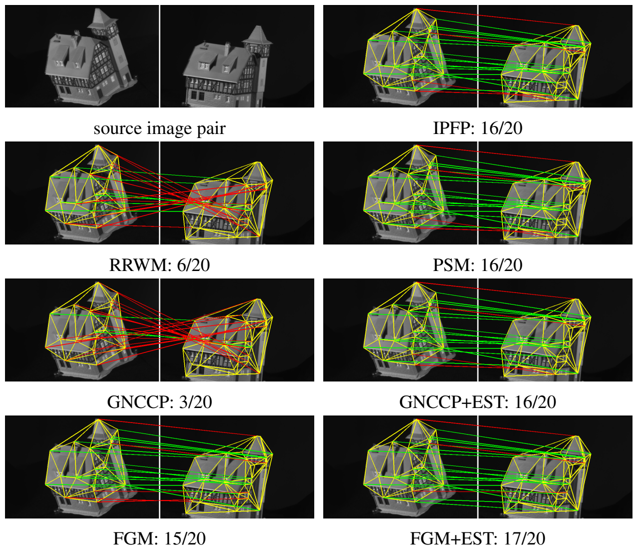

Figure 1: An example of graph matching on the CMU house dataset. Graph edges are represented by yellow lines, true matches by green lines and false matches by red lines. The algorithm, the number of true matches per ground truths for each subfigure are captioned (same for Fig. 3 and 5).

frames. We test all possible image pairs, total 550 pairs gaped by 10, 20, ..., 100, and use 3 different settings of nodes (n_1, n_2) =(30,30), (25, 30) and (20, 30). In the settings where $n_1 < 30$, we randomly choose n_1 points among 30 landmark points.

In the experiment, decreasing n_1 implies the increase of outlier, while increasing sampling gaps implies increase the degree of deformation. We use Delaunay triangulation (Lee and Schachter, 1980) to connect the landmarks, and edge weights are computed as the Euclidean distance between the connected nodes. The affinity matrix is conducted by $K(ia, jb) = \exp(-(A_1(i,j) - A_2(a,b))^2/2500)$, where $A_1(i,j)$ and $A_2(a,b)$ are the weights of the two edges.

Fig. 1 represents an example for graph matching with 10 outliers and significant deformation. Fig. 2 shows the performance curves for n_1 = 30, 25, and 20 with respect to variant sequence gaps. All algorithms except IPFP achieve perfect matching when no outliers existing ($n_1 = 30$) and have similar matching performances with five outliers ($n_1 = 25$). When we increase the number outliers to 10 ($n_1 = 20$), FGM, FGM+EST and PSM gain similar matching accuracy and outperform other algorithms. In the aspect of computational efficiency, our approach brings dramatically improvement into both GNCCP and FGM algorithms. The proposed GNCCP+EST and FGM+EST algorithms spend only about 4% and 18% of computational time of the original GNCCP and FGM algorithms respectively. The computational efficiency of them exceeds the PSM algorithm and is close to the RRWM and IPFP algorithms.

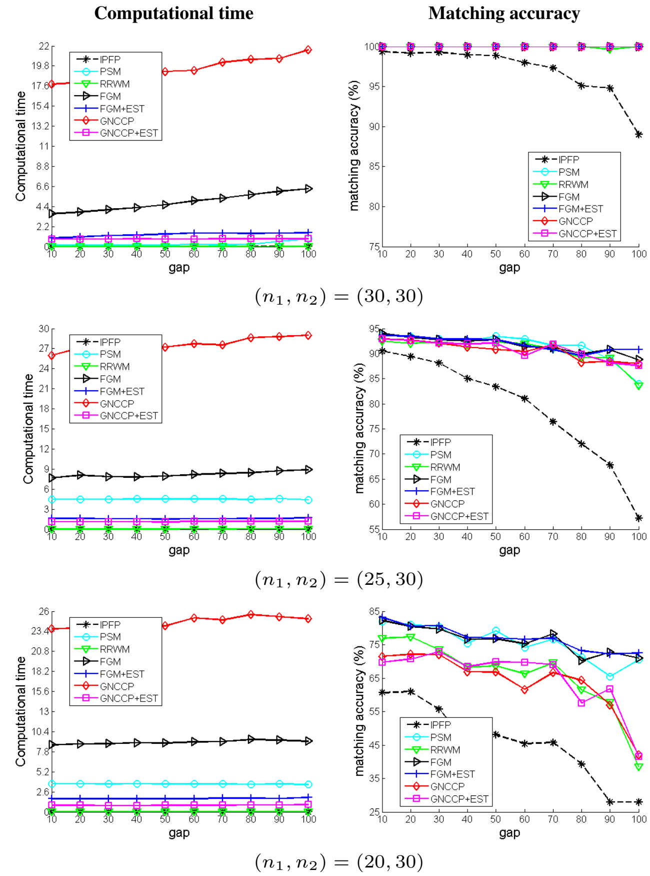

Figure 2: Comparison of graph matching with respect to sequence gap on the CMU house dataset.

Experiments on the Pascal dataset

This dataset from Leordeanu, Sukthankar, and Hebert (2012) consists of 30 pairs of car images and 20 pairs of motorbike images selected from Pascal 2007. Each pair contains $30 \sim 60$ ground-truth correspondences. To test the performance against noise, we randomly select $0 \sim 20$ outlier nodes from the background. We compute the feature, p_i, for each node i as its orientation of the normal vector at that point to the contour where the point was sampled. We utilize the Delaunay triangulation (Lee and Schachter, 1980) to connect the nodes and associate each edge c with a couple of values $[d_c, \theta_c]^T$, where d_c is the pairwise distance between the connected nodes and θ_c is the absolute angle between the edge and the horizontal line. Consequently, the node affinity between nodes i_1 and i_2 is computed as $\exp(-|p_{i_1} - p_{i_2}|)$, and the edge affinity between edges c_1 and c_2 is computed as $\exp(-(|d_{c_1} - d_{c_2}| + |\theta_{c_1} - \theta_{c_2}|)/2)$.

Fig. 3 represents an example for graph matching of motorbike images with 10 outliers on this dataset. The matching performance and computational time of each algorithm with respect to the outlier number was summarized in Fig. 4. The original GNCCP and FGM algorithms have very huge computational burden comparing to other algorithms. Fortunately, both of them are benefited significantly in com-

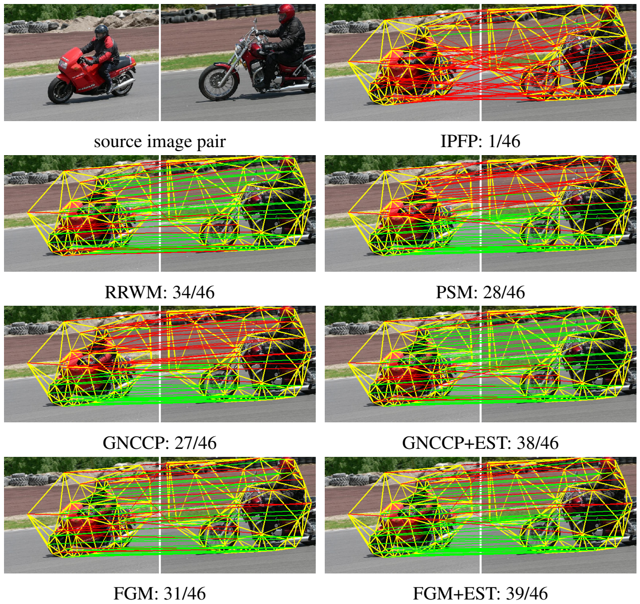

Figure 3: A matching example on the Pascal dataset.

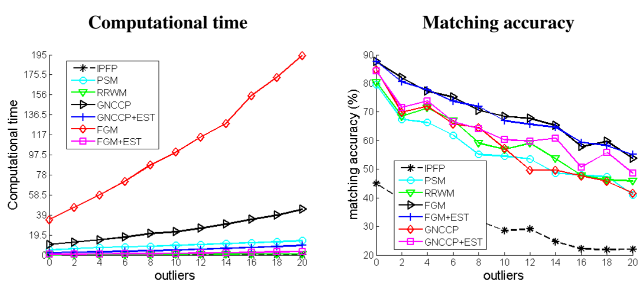

Figure 4: Comparison of graph matching with respect to the outlier number for the Pascal dataset.

putational efficiency from our new path following strategy. In fact, the computational costs of the GNCCP+EST and FGM+EST algorithms are reduced to as less as only about 2% and 15% of the original GNCCP and FGM algorithms respectively. The computational efficiency of them exceeds the PSM algorithm and is close to the RRWM and IPFP algorithms. The FGM+EST algorithms is in line with the FGM algorithm and outperforms other algorithms in terms of matching accuracy. It is interesting that the GNCCP+EST algorithm achieves remarkable improvement in matching accuracy comparing to the GNCCP algorithm with certain numbers of outliers.

Caltech image dataset

In this experiment, we test our approach on a real image dataset containing 30 image pairs provided in (Cho, Lee, and Lee, 2010) which are collected from Caltech-101 and MSRC datasets. For each image pair, the authors provide detected MSER keypoints (Donoser and Bischof, 2006), initial matches, affinity matrix, and manually labeled ground truth feature pairs. In (Cho, Lee, and Lee, 2010), candidate matches were filtered using the distance of 128-dim SIFT descriptor (Lowe, 2004) where all the low-quality candidate matches were filtered if the feature pair is more distant in SIFT feature space than a loose threshold $\delta = 0.6$. The FGM algorithm (and the corresponding FGM+EST algorithm) is absent in this experiment because the provided affinity matrices cannot be directly factorized by the FGM algorithm due to its omission of many candidate matches.

An example for matching cat images with dramatic non-grid deformation is shown in Fig. 5. The matching accuracy and computational time of each algorithm are summarized in table 1. The proposed GNCCP+EST algorithm spend only about 4% of computational time of the original GNCCP algorithm, and also exceeds the RRWM algorithm and the PSM algorithm in terms of computation efficiency. Our new algorithm also gains improvement in matching accuracy, and outperforms all other algorithms.

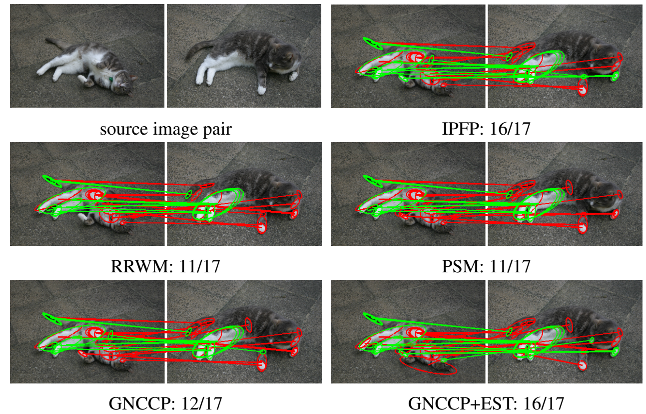

Figure 5: A matching example on the Caltech dataset.

Table 1: Graph matching results on the Caltech dataset.

Algorithm	IPFP	RRWM	PSM	GNCCP	GNCCP+EST
Accuracy	66.30	75.49	67.40	74.92	***75.60***
Time (sec)	***0.05***	0.61	15.53	6.79	0.27

Conclusion

We proposed a novel path following strategy for graph matching to improve the computational efficiency. We integrated our strategy into various path following algorithms. Experimental results reveal that our approach can consistently improve dramatically the computation efficiency of the original algorithms. Furthermore, in terms of matching accuracy, our efficient approach achieves better or similar results compared with the original ones.

Acknowledgments

This work is supported by the National Nature Science Foundation of China (nos. 61300071, 61272352, 61472028, 61301185 and 61370070), Beijing Natural Science Foundation (no.4142045), the Fundamental Research Funds for the Central Universities (no. 2015JBM037), and NSF grants IIS-1350521 and IIS-1218156.

References

Allgower, E. L., and Georg, K. 2003. *Introduction to Numerical Continuation Methods*. SIAM Classics in Applied Mathematics 45.

Berg, A.; Berg, T.; and Malik, J. 2005. Shape matching and object recognition using low distortion correspondences. In *CVPR*, 15–33.

Cho, M.; Lee, J.; and Lee, K. 2010. Reweighted random walk for graph matching. In *ECCV*, 492–505.

Conte, D.; Foggia, P.; Sansone, C.; and Vento, M. 2004. Thirty years of graph matching in pattern recognition. *International Journal of Pattern Recognition and Artificial Intelligence* 18(3):265–298.

Cour, T.; Srinivasan, P.; and Shi, J. 2007. Balanced graph matching. In *NIPS*, 313–320.

Donoser, M., and Bischof, H. 2006. Efficient maximally stable extremal region (mser) tracking. In *CVPR*, 553–560.

Duchenne, O.; Joulin, A.; and Ponce, J. 2011. A graph-matching kernel for object categorization. In *ICCV*, 1792–1799.

Egozi, A.; Keller, Y.; and Guterman, H. 2013. A probabilistic approach to spectral graph matching. *PAMI* 35(1):18–27.

Foggia, P.; Percannella, G.; and Vento, M. 2014. Graph matching and learning in pattern recognition in the last 10 years. *International Journal of Pattern Recognition and Artificial Intelligence* 28(1):1–40.

Franke, M., and Wolf, P. 1956. An algorithm for quadratic programming. *Naval Research Logistics Quarterly* 3:95–100.

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

Jiang, H.; Yu, S.; and Martin, D. 2011. Linear scale and rotation invaiant matching. *PAMI* 33(7):1339–1355.

Kudryavtsev, L. 2001. *Implicit function*. Encyclopedia of Mathematics, Springer.

Kuhn, H., and Tucher, A. 1951. Nonlinear programming. In *Proceedings of 2nd Berkeley Symposium*, 481–492.

Lazebnik, S.; Schmid, C.; and Ponce, J. 2006. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *CVPR*, 2169–2178.

Lee, D., and Schachter, B. 1980. Two algorithms for constructing a delaunay triangulation. *Int. J. Computer Information Sci* 9:219–242.

Leordeanu, M., and Hebert, M. 2005. A spectral technique for correspondence problems using pairwise constraints. In *ICCV*, 1482–1489.

Leordeanu, M., and Hebert, M. 2009. An integer projected fixed point method for graph matching and map inference. In *NIPS*.

Leordeanu, M.; Sukthankar, R.; and Hebert, M. 2012. Unsupervised learning for graph matching. *IJCV* 96(1):28–45.

Liu, Z., and Qiao, H. 2014. GNCCP - graduated nonconvexity and concavity procedure. *PAMI* 36(6):1258–1267.

Liu, Z.; Qiao, H.; Yang, X.; and Hoi, S. 2014. Graph matching by simplified convex-concave relaxation procedure. *IJCV* 109(3):169–186.

Lowe, D. 2004. Distinctive image features from scale-invariant keypoints. *IJCV* 60(2):91–110.

Torresani, L.; Kolmogorov, V.; and Rother, C. 2008. Feature correspondence via graph matching: Models and global optimization. In *ECCV*, 596–609.

Wu, J.; Shen, H.; Li, Y.; Xiao, Z.; Lu, M.; and Wang, C. 2013. Learning a hybrid similarity measure for image retrieval. *Pattern Recognition* 46(11):2927–2939.

Zaslavskiy, M.; Bach, F.; and Vert, J. 2009. A path following algorithm for the graph matching problem. *PAMI* 31(12):2227–2242.

Zass, R., and Shashua, A. 2008. Probabilistic graph and hypergraph matching. In *CVPR*, 1–8.

Zhou, F., and Torre, F. 2012. Factorized graph matching. In *CVPR*, 127–134.