



저작자표시-비영리-변경금지 2.0 대한민국

이용자는 아래의 조건을 따르는 경우에 한하여 자유롭게

- 이 저작물을 복제, 배포, 전송, 전시, 공연 및 방송할 수 있습니다.

다음과 같은 조건을 따라야 합니다:



저작자표시. 귀하는 원저작자를 표시하여야 합니다.



비영리. 귀하는 이 저작물을 영리 목적으로 이용할 수 없습니다.



변경금지. 귀하는 이 저작물을 개작, 변형 또는 가공할 수 없습니다.

- 귀하는, 이 저작물의 재이용이나 배포의 경우, 이 저작물에 적용된 이용허락조건을 명확하게 나타내어야 합니다.
- 저작권자로부터 별도의 허가를 받으면 이러한 조건들은 적용되지 않습니다.

저작권법에 따른 이용자의 권리는 위의 내용에 의하여 영향을 받지 않습니다.

이것은 [이용허락규약\(Legal Code\)](#)을 이해하기 쉽게 요약한 것입니다.

[Disclaimer](#)

M.S. THESIS

Graph Matching using Discrete Tabu Search
on the Penalized Association Graph

벌칙 포함 연합 그래프에서의 이산 타부 검색법에 기반한
그래프 정합

BY

KAMIL ADAMCZEWSKI

AUGUST 2015

DEPARTMENT OF ELECTRICAL ENGINEERING AND
COMPUTER SCIENCE
COLLEGE OF ENGINEERING
SEOUL NATIONAL UNIVERSITY

M.S. THESIS

Graph Matching using Discrete Tabu Search
on the Penalized Association Graph

벌칙 포함 연합 그래프에서의 이산 타부 검색법에 기반한
그래프 정합

BY

KAMIL ADAMCZEWSKI

AUGUST 2015

DEPARTMENT OF ELECTRICAL ENGINEERING AND
COMPUTER SCIENCE
COLLEGE OF ENGINEERING
SEOUL NATIONAL UNIVERSITY

Graph Matching using Discrete Tabu Search on the
Penalized Association Graph

별칭 포함 연합 그래프에서의 이산 타부 검색법에
기반한 그래프 정합

지도교수 이 경 무

이 논문을 공학석사 학위논문으로 제출함

2015 년 8 월

서울대학교 대학원

전기 컴퓨터 공학부

Kamil Adamczewski

Kamil Adamczewski의 공학석사 학위论문을 인준함

2015 년 8 월

위 원 장 _____
부위원장 _____
위 원 _____

Abstract

Graph matching is a fundamental problem in computer vision. In this paper, we propose a novel graph matching algorithm based on tabu search [12]. The proposed method solves graph matching problem by casting it into an equivalent weighted maximum clique problem of the corresponding penalized association graph, and then uses tabu search technique for the optimization. The distinct feature of tabu search optimization is that it utilizes the history of search to make more strategic decisions while looking for the optimal solution, thus effectively escaping local optima and in practice achieving superior results. The proposed method, unlike the existing algorithms, enables direct optimization in the original discrete space while encouraging rather than artificially enforcing hard one-to-one constraint, thus resulting in better solution. The experiments demonstrate the robustness of the algorithm in a variety of settings, presenting the state-of-the-art results.

Keywords: graph matching, feature correspondence, tabu search, object recognition

Student Number: 2013-23854

Contents

Abstract	i
Acknowledgements	ii
Chapter 1 Introduction	1
1.1 Motivation	1
1.2 Contribution	2
1.3 Thesis organization	3
Chapter 2 Related Works	4
2.1 Maximum weighted subgraph	4
2.2 Tabu search	5
2.3 Discrete space vs. continuous space optimization	5
Chapter 3 Graph Matching Problem	6
3.1 Formulation	7
Chapter 4 Equivalent weighted maximum subgraph problem on the penalized association graph	8
4.1 Association graph	8

4.2	Weighted maximum subgraph problem	9
4.3	The notion of subgraph as the integer constraint	9
4.4	Penalized association graph as one-to-one constraint	10
Chapter 5 Discrete Tabu Search for Graph Matching		13
5.1	Local search and search space	14
5.1.1	Decreasing the size of local search space	15
5.1.2	Example	16
5.2	Tabu Search	18
5.2.1	Short term memory	18
5.2.2	Implementation details	19
5.2.3	Example	20
5.2.4	Long term memory	22
5.2.5	Implementation details	22
Chapter 6 Experiments		25
6.1	Synthetic random graph matching	25
6.1.1	Experimental evaluation of the proposed one-to-one constraint	27
6.1.2	Evaluation of methods that work in discrete space on the synthetic dataset	29
6.1.3	Parameter settings	30
6.2	Real image feature correspondence	32
6.3	Repeated Structure Matching	38
Chapter 7 Summary and Conclusions		40
	초록	46

List of Figures

Figure 5.1	An example of using penalty to enforce one-to-one constraint through local search for (a) graph matching matching between \mathcal{G}_1 and \mathcal{G}_2 with initial matching ij',jk',kk' violating the constraint. (b) the corresponding association graph. The weights for non-conflicting matches are 1 and for conflicting matches -2 (for clarity, only the weights for the subgraph \mathcal{K} are shown). (c) affinity value of each node to \mathcal{K} given by Eq.(5.1) (d) The change Δ_{ab} to the IQP objective for interchanging top two matches from (c) according to Eq.(5.2). (e) Resulting subgraph with matches satisfying one-to-one constraint.	17
------------	--	----

Figure 5.2 An example of tabu search to find the maximum weighted clique. At each iteration we swap two non-tabu nodes to maximize the objective (3.1). After being swapped, the nodes (marked as shaded) remain tabu for 1 iteration. (0) initially, $\mathcal{K} = acf$ and $|acf| = 6$; no variables are tabu. (1) e substitutes a , and $|cef| = 8$. (2) tabu variables are e that entered and a that left, we choose between b and d , both choices lower the objective, choose b , $|bef| = 7$. (3) tabu variables are b and c , choose d to obtain the global optimum $|bed| = 9$ 21

Figure 6.1 Accuracy and score curves are plotted for four different conditions: varying the amount of deformation noise with fixed number of outliers (a) $n_{out} = 0$ and (b) $n_{out} = 10$, and varying the number of outliers with fixed deformation noise (c) $\sigma = 0$ (d) $\sigma = 0.15$) 26

Figure 6.2 Experimental ramifications of the proposed soft one-to-one constraint, where $p = -9$. Accuracy and score curves are plotted for four different conditions: varying the amount of deformation noise with fixed number of outliers (a) $n_{out} = 0$ and (b) $n_{out} = 10$, and varying the number of outliers with fixed deformation noise (c) $\sigma = 0$ (d) $\sigma = 0.25$ 28

Figure 6.3	Evaluation of DDMCMC [18] and SMCM [24], which solve one-to-one constrained IQP directly in the discrete space. Accuracy and score curves are plotted for four different conditions: varying the amount of deformation noise with fixed number of outliers (a) $n_{out} = 0$ and (b) $n_{out} = 10$, and varying the number of outliers with fixed deformation noise (c) $\sigma = 0$ (d) $\sigma = 0.25$	29
Figure 6.4	Example results of the real image matching on Willow Object dataset. Keypoints are distinguished by different colors in the model image and the matched locations are depicted for each algorithm	33
Figure 6.5	Example results of the real image matching on Willow Object dataset. Keypoints are distinguished by different colors in the model image and the matched locations are depicted for each algorithm	34
Figure 6.6	Example results of the real image matching on Willow Object dataset. Keypoints are distinguished by different colors in the model image and the matched locations are depicted for each algorithm	35
Figure 6.7	Example results of the real image matching on Willow Object dataset. Keypoints are distinguished by different colors in the model image and the matched locations are depicted for each algorithm	36
Figure 6.8	Example results of the real image matching on Willow Object dataset. Keypoints are distinguished by different colors in the model image and the matched locations are depicted for each algorithm	37

Figure 6.9 Accuracy and score curves are plotted for four different conditions: varying the amount of deformation noise with fixed number of outliers (a) $n_{out} = 0$ and (b) $n_{out} = 10$, and varying the number of outliers with fixed deformation noise (c) $\sigma = 0$ (d) $\sigma = 0.1$ 38

List of Tables

Table 5.1	Parameters used in Algorithm 1	24
Table 6.1	SNU Dataset Average Accuracy (%)	31
Table 6.2	Willow Object Dataset Average Accuracy (%)	31

Chapter 1

Introduction

1.1 Motivation

Graph matching is an important problem in both theoretical computer science, and practical applications of computer vision and machine learning. In computer vision, it has been proven to effectively deal with the problem of matching two sets of visual features making it the approach of choice in shape matching [3], object categorization [7] and tracking [17]. Typically, graph matching is formulated as an integer quadratic program (IQP) with an additional one-to-one constraint. The formulation is popular due to its generality to allow complex but more informative affinity measures between node correspondences such as transfer error [5]. Nonetheless, there are two main theoretical issues with optimizing this graph matching formulation; (1) IQP is generally an NP-hard non-convex problem with multiple local minima; (2) the one-to-one constraint is not straightforward to combine into the IQP optimization process. As a result, various methods have been developed to approximate the optimal solution

of the graph matching problem based on different approaches, including eigenvector analysis [25, 19], random walk [6], stochastic methods [24] and other optimization techniques [20, 30].

Conventionally, two stage procedure has been used to approximate the solution of IQP: first, solve the optimization problem in continuous domain by relaxing integer constraint, and then project the intermediate solution into the original feasible space of IQP [6, 19, 13]. Nonetheless, recent research devises methods that produce discrete solutions and suggests that post-processing discretization leads to substantial loss in accuracy, especially that the discretization step is independent of the optimization of IQP. The approach to one-to-one constraint is three-fold. The first way, which enforces one-to-one constraint in the post-processing stage, suffers from the above-mentioned post-processing loss [19]. The second approach, bistochastic normalization, in continuous space softly enforces the constraint during the optimization procedure [6, 13]. Finally, incorporating hard one-to-one during optimization results in overly restrictive search of the solution space. [1, 18, 24]

1.2 Contribution

In this work we propose a novel method for the graph matching based on tabu search, which solves the IQP problem effectively incorporating the one-to-one constraint. The contributions of this paper are manifold: (1) It proposes the penalized association graph framework which softly encourages the one-to-one constraint while directly searching in the discrete solution space. (2) Given the above framework, it adopts tabu search as an optimization technique. Tabu search, thanks to its strategic approach to searching the solution space has become a successful tool in optimizing hard combinatorial problems, while not

being popular yet in computer vision. (3) It extends the proposed framework to work on one-to-many graph matching, enhancing its usage in practice. (4) Presents experimental results which prove outstanding robustness of the algorithm in practical environment where noise and outliers exist.

1.3 Thesis organization

This thesis is organized as follows. First, previous research related to this work is briefly introduced in Chapter 2, underlying the differences between the proposed approach and the existing methods. Then in Chapter 3, the graph matching problem is defined as a constrained IQP and in Chapter 4 the equivalent problem, which incorporates graph matching constraints into the association graph, is proposed. In Chapter 5, the new graph matching algorithm based on the local search and tabu search optimization is presented. Finally in Chapter 6, the algorithm is evaluated through extensive experimental comparison with other state-of-the-art algorithms. At the end, conclusions follow.

Chapter 2

Related Works

2.1 Maximum weighted subgraph

The proposed method casts graph matching as a weighted maximum subgraph problem. In the literature, Horaud et al. [16] first proposed to formulate feature correspondence problem as finding the maximum clique between the nodes representing linear segments in two images. Similarly, Liu et al. [22] formulated graph matching as finding dense subgraphs within the association graph corresponding to local optima. The method was further generalized by allowing subgraph matching [21]. Zhao et al. [29] proposed heuristic label propagation as a post processing step following the initialization by [22]. In contrast, in this work, we redefine the notion of the association graph to incorporate soft one-to-one constraint and further propose tabu search method, which is less initialization dependent, to effectively explore given solution space searching beyond the local maxima, and thus resulting in more accurate solution.

2.2 Tabu search

Nevertheless, despite being a flexible and effective tool for solving combinatorial optimization problems [12], tabu search has not been actively applied in computer vision. Sorlin and Solnon [23] and Williams et al. [27] mentioned tabu search for graph matching focusing on simple incomplete synthetic graphs without noise nor outliers, omitting any tests on real world datasets. Moreover, [27] restricts the formulation to simple Bayesian consistency similarity measure and [23] uses cardinality functions.

2.3 Discrete space vs. continuous space optimization

This work is written in the spirit of recent research that advocates the importance of discretization in the course of the algorithm. Leordeanu et al. [20] iterates between maximizing linearly approximated objective in discrete domain and maximizing along the gradient direction in continuous domain. Moreover, Zhou et al. [30] induces discrete solution by gradually changing weights of concave and convex relaxations of the original objective. Although these methods mitigate the artifact of continuous relaxation, they still phase through the continuous solution space during the optimization.

On the other hand, few algorithms have been proposed to directly explore the feasible space of IQP without any relaxation. Yan et al. [28] propose discrete method for hyper graph matching. Lee et al. [18] built Markov chain Monte Carlo based algorithm and Suh et al. [24] used Sequential Monte Carlo sampling to deal with the exponentially large and discrete solution space. Though they circumvent the issue of relaxation mentioned above, instead, they suffer from inefficient exploration, because the one-to-one constraint significantly limits the possible moves in their approach.

Chapter 3

Graph Matching Problem

A graph \mathcal{G} is represented as a quadruple $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{W}^{\mathcal{V}}, \mathcal{W}^{\mathcal{E}})$. The elements $\mathcal{V}, \mathcal{E}, \mathcal{W}^{\mathcal{V}}, \mathcal{W}^{\mathcal{E}}$ respectively denote a set of nodes, a set of edges (which represent pairwise relations between the nodes), attributes of nodes and attributes of edges. In feature correspondence, a node attribute describes a local appearance of a feature in an image, and an edge geometrical relationship between the features.

The objective of graph matching is to find the correct correspondences between nodes of two graphs \mathcal{G}_1 and \mathcal{G}_2 with n_1 and n_2 nodes, respectively. In this work, “one-to-one” constraint is assumed, which requires one node in \mathcal{G}_1 to be matched to at most one node in \mathcal{G}_2 and vice versa¹ The case when a node is not matched can be interpreted as the node being an outlier.

The compatibility between the nodes of two graphs \mathcal{G}_1 and \mathcal{G}_2 is commonly

¹Note that the constraint is not one-to-one in the strict definition of a function since the correspondence is not a function as some of the nodes of \mathcal{G}_1 may not be matched with nodes in \mathcal{G}_2 .

encoded in the affinity matrix $\mathbf{A} \in \mathbb{R}^{n_1 n_2 \times n_1 n_2}$. A diagonal entry $\mathbf{A}_{ii';ii'}$ ² represents an affinity between attributes of $v_i \in \mathcal{V}_1$ and $v_{i'} \in \mathcal{V}_2$. A non-diagonal entry $\mathbf{A}_{ii';jj'}$ is a quantified information how well a match between $v_i \in \mathcal{V}_1$ and $v_{i'} \in \mathcal{V}_2$ corresponds to another match $v_j \in \mathcal{V}_1$ and $v_{j'} \in \mathcal{V}_2$. In subsequent sections for simplicity of notation, we will use superscript v_i^1 to denote the i -th node of graph \mathcal{G}_1 .

3.1 Formulation

The goal of graph matching problem is to find the set of matches between nodes of two graphs \mathcal{G}_1 and \mathcal{G}_2 that maximize the sum of affinity values of corresponding nodes and edges given by the affinity matrix \mathbf{A} . This formulation can be compactly represented by means of Integer Quadratic Programming (IQP):

$$\mathbf{x} = \arg \max_{\mathbf{x}} \mathbf{x}^T \mathbf{A} \mathbf{x} \quad (3.1)$$

$$s.t \quad \mathbf{x} \in \{0, 1\}^{n_1 n_2} = \mathbf{X}_{\text{int}}, \quad (3.2)$$

where the resulting matches are represented by a column vector \mathbf{x} , where $\mathbf{x}_{ii'} = 1$ if node v_i^1 is matched with $v_{i'}^2$ and 0 otherwise³. In accordance with the IQP graph matching conventional formulation, we assume the one-to-one constraint:

$$\begin{aligned} \mathbf{x} &\in \{\mathbf{x} | \forall i, \sum_{i'=1}^{n_2} \mathbf{x}_{ii'} \in \{0, 1\} \text{ and} \\ &\forall i', \sum_{i=1}^{n_1} \mathbf{x}_{ii'} \in \{0, 1\}\} \\ &= \mathbf{X}_{\text{one-to-one}}. \end{aligned} \quad (3.3)$$

²For ease of understanding, in this paper we use notation $\mathbf{A}_{ii';jj'}$ to refer actually to the affinity matrix entry $\mathbf{A}_{(i'-1)n_1+i, (j'-1)n_1+j}$.

³In a similar manner to the convention used for \mathbf{A} , we use $x_{ii'}$ to denote the entry $x_{(i-1)n_2+i'}$.

Chapter 4

Equivalent weighted maximum subgraph problem on the penalized association graph

In this section we build a graph matching framework based on the notion of the association graph, and incorporate graph matching constraints in an intuitive yet powerful way.

4.1 Association graph

Let us consider association graph $\mathcal{A} = (\mathcal{V}_{\mathcal{A}}, \mathcal{E}_{\mathcal{A}}, \mathcal{W}_{\mathcal{A}}^{\mathcal{V}}, \mathcal{W}_{\mathcal{A}}^{\mathcal{E}})$ constructed from two input graphs, \mathcal{G}_1 and \mathcal{G}_2 , which is a fully connected graph with real valued attributes for both nodes and edges. Each node represents a possible node correspondence between input graphs, i.e., $v \in \mathcal{V}_{\mathcal{A}}$ stands for $(v_i^1, v_{i'}^2) \in \mathcal{V}_1 \times \mathcal{V}_2$ and each edge $e \in \mathcal{E}_{\mathcal{A}}$ represents a relationship between the two node correspondences. The nodes and edges are respectively characterized by its weights $w_v \in \mathcal{W}_{\mathcal{A}}^{\mathcal{V}}$ and $w_e \in \mathcal{W}_{\mathcal{A}}^{\mathcal{E}}$.

4.2 Weighted maximum subgraph problem

Now, we can construct a graph \mathcal{A} , whose maximum subgraph problem is equivalent to the graph matching problem. The weighted maximum subgraph problem is to find a subgraph \mathcal{K} , also referred as a clique, in the graph \mathcal{A} which maximizes its potential, where the potential is the sum of all weights of included nodes and edges, i.e.,

$$\sum_{v \in \mathcal{V}_{\mathcal{K}}} w_v + \sum_{e \in \mathcal{E}_{\mathcal{K}}} w_e. \quad (4.1)$$

In the general formulation, we assign the attributes as follows: $w_v = \mathbf{A}_{ii'}$, where $v = (v_i^1, v_i^2)$, and $w_e = \mathbf{A}_{ii',jj'}$, where $e = (v, u)$, $v = (v_i^1, v_i^2)$, $u = (v_j^1, v_j^2)$. In Section 4.4 we extend this formulation to incorporate graph matching one-to-one constraint.

4.3 The notion of subgraph as the integer constraint

Then, denote \mathcal{K} as the corresponding subgraph of \mathcal{A} such that the subset of nodes $\mathcal{V}_{\mathcal{K}} \subset \mathcal{V}_{\mathcal{A}}$ and $\mathcal{E}_{\mathcal{K}} = \mathcal{E}_{\mathcal{A}} \cap \binom{\mathcal{V}_{\mathcal{K}}}{2}$.

Thus, we represent the selection of nodes in \mathcal{A} with a binary indicator vector $\mathbf{y} \in \{0, 1\}^{|\mathcal{V}_{\mathcal{A}}|}$, where $\mathbf{x}_a = 1$ if a -th node is selected, that is $v_i \in \mathcal{K}$ and $\mathbf{x}_a = 0$ if $v_i \in \mathcal{L}$. Let us note, that vector \mathbf{y} in this form satisfies the graph matching integer constraint and belongs to the feasible solution set of the constrained IQP (3.1). This is summarized by the following definition:

Definition 1. *Let $\mathbf{y} \in \mathcal{Y}$ be a vector representing the subgraph \mathcal{K} such that $\mathbf{y}_i = 1$ if $\mathbf{v}_i \in \mathcal{K}$ and 0 otherwise. Then the sum of the unary node and edge potentials (referred here as the graph potential) in \mathcal{K} is given by*

$$\mathbf{y}^T \mathbf{A} \mathbf{y}. \quad (4.2)$$

Furthermore, we show that the set of subgraphs $\mathcal{K} \in \mathcal{A}$ and the set of integer solutions to the graph matching IQP formulation (3.1) (which not necessarily satisfy one-to-one constraint (3.3)) are the same.

Lemma 1. *There exists bijection¹ between the set of subgraphs $\mathcal{K} \in \mathcal{A}$ and the set of correspondences between two graphs \mathcal{G}_1 and \mathcal{G}_2 represented by the vector $\mathbf{x} \in \mathbf{X}$.*

Proof. The proof follows from the construction. Let $\mathbf{x} \in \mathbf{X}$ be a vector in the set of all the integer solutions of the QP formulation (3.1) which represent the matches between nodes of graphs \mathcal{G}_1 and \mathcal{G}_2 , and $\mathbf{y} \in \mathbf{Y}$ be a vector representing a subgraph \mathcal{K} of \mathcal{A} . We show that function f that maps QP solution vectors to subgraphs is both injective (no two elements in the domain of the function are mapped to the same image) and surjective (any element in the codomain of the function has at least one preimage element in the domain). First, assume $\mathbf{y}_1 = \mathbf{y}_2$ that is $f(\mathbf{x}_1) = f(\mathbf{x}_2)$. Since the nodes in the subgraph are chosen in such a way that $\mathbf{y}_{ij;i'j'} = 1$ if there is a match between nodes v_i and v'_i , and v_j and v'_j then $\mathbf{y}_1 = \mathbf{x}_1$ and $\mathbf{y}_2 = \mathbf{x}_2$, and so $\mathbf{x}_1 = \mathbf{x}_2$. The surjection is based on the same principle. Given vector \mathbf{y} , build a vector \mathbf{x} such that $\mathbf{x}_{ij;i'j'} = 1$ whenever $\mathbf{y}_{ij;i'j'}$, and 0 otherwise. Therefore for each vector \mathbf{y} there exists a vector \mathbf{x} which is mapped to \mathbf{y} . Since the function f is both injective and surjective, it is therefore the bijection. \square

4.4 Penalized association graph as one-to-one constraint

In this section, we extend the notion of the association graph defined in Section 4.1 to account for the fact that graph matching problem requires “one-to-one” correspondence between the graph nodes. As a result, we introduce the penalty term which penalizes conflicting matches and softly yet effectively enforces one-to-one constraint.

In the penalized association graph formulation, we assign the attributes as follows: $w_v = \mathbf{A}_{ii'}$, where $v = (v_i^1, v_i^2)$, and as for edge weight w_e , where $e = (v, u)$, $v = (v_i^1, v_i^2)$, $u = (v_j^1, v_j^2)$, $w_e = p$, $p < 0$ when $i = j$ or $i' = j'$, which

¹Bijection is a function between the elements of two sets, where every element of one set is paired with exactly one element of the other set, and every element of the other set is paired with exactly one element of the first set.

is a conflicting match, and $w_e = \mathbf{A}_{ii',jj'}$ otherwise. If $p = -\infty$, then it becomes equivalent to maximizing the IQP with one-to-one constraint.

Subsequently more formally, we state the equivalence between the proposed formulation of the penalized association graph, where penalty is large enough, and the constrained IQP.

Theorem 1. *Let $p = -\infty$. A vector \mathbf{y} representing a subgraph \mathcal{K} in the association graph \mathcal{A} has finite potential if and only if it belongs to the feasible set of the one-to-one constrained IQP.*

Proof. We let the penalty $p = -\infty$. First, we assume a solution vector \mathbf{y} with finite potential that represents a subgraph \mathcal{K} in the association graph \mathcal{A} as given in the Definition 1. Then by Lemma 1 there is a corresponding solution $\mathbf{x} \in \mathbf{X}_{\text{int}}$. Furthermore, we argue that the solution $\mathbf{x} \in \mathbf{X}_{\text{one-to-one}}$. Assume, by contradiction, that the corresponding optimal solution \mathbf{x} is not one-to-one. Then one of the edges within the weighted subgraph has potential $-\infty$ and therefore, the sum of all the potentials within the subgraph is also $-\infty$. Yet we assumed that the sum of the affinity values is finite, which is a contradiction. Thus, $\mathbf{y} \in \mathbf{X}_{\text{one-to-one}} \cap \mathbf{X}_{\text{int}}$, and \mathbf{y} is in the feasible set to one-to-one constrained IQP.

The proof for a constrained IQP solution \mathbf{x} to be a solution vector \mathbf{y} of the subgraph \mathcal{K} follows in the similar fashion. \square

Corollary 1. *When $p = -\infty$, there is bijection between the set of subgraphs \mathcal{K} in the association graph \mathcal{A} with finite potential and the feasible set of the one-to-one constrained IQP.*

Corollary 2. *The problem of maximizing one-to-one constrained IQP is equivalent to the problem of maximizing weighted clique potential in the penalized association graph, where $p = -\infty$.*

Proof. The proof follows from the Theorem 1. First let us note that the objective function of the two problems is identical, that is Eq. (3.1) and Eq. (4.2) are the same. Furthermore, the feasible sets of the two problems are the same. As a result, the solution that maximizes constrained IQP also maximizes weighted clique potential in the penalized association graph and vice versa. \square

In this section, we proposed to incorporate the penalty term p into the association graph framework which enforces the one-to-one constraint in soft way. As our objective is to maximize the sum of affinities in the clique \mathcal{K} , choosing a match that violates one-to-one constraint (which is equivalent to selecting an edge with a negative value) lowers the objective function. The subsequent section describes the local search optimization which utilizes above framework to enforce the one-to-one constraint.

Chapter 5

Discrete Tabu Search for Graph Matching

In this section, we propose a tabu search-based graph matching algorithm that optimizes the framework built in the previous section. Tabu search solves the original optimization problem Eq.(3.1-3.3) by solving the equivalent weighted maximum subgraph problem (Section 4) in the penalized association graph \mathcal{A} . Tabu search, also called adaptive memory programming, has been introduced to deal with hard optimization problems through economical search in the solution space exploiting the search history [10, 11]. Due to its flexibility and excellent performance, it has been widely applied in various fields including engineering, economics and finance, spanning the applications ranging from pattern classification, VLSI design, investment planning, energy policy, biomedical analysis and environmental conservation [12]. Nonetheless, it has not been widely applied yet in computer vision. In this work, we use tabu search to explore the space of all possible cliques of the association graph.

Tabu search is a meta-algorithm and constitutes of two components. The first one is the base algorithm which describes the general mechanism how we iterate between solutions (Section 5.1). The second part consists of the criteria which solutions are allowed and which ones are forbidden at any given iteration (Section 5.2). The overall algorithm is described as Algorithm 1. The subsequent section describes the local search, where we propose how to utilize discrete solution space to iterate between solutions.

5.1 Local search and search space

Local search is an optimization technique which searches for the optimal solution among the set of candidate solutions. In our case, the set of candidate solutions is the neighborhood $N(\mathbf{x})$. The basic idea behind local search is to advance from one candidate solution \mathbf{x} to another solution \mathbf{x}' by an operation called a *move*. The movement occurs in the neighborhood $N(\mathbf{x})$ which we define as the set of solutions that differ with the candidate solution \mathbf{x} by one node:

$$N(\mathbf{x}) = \{\mathbf{x}' | \mathbf{x}' \cdot \mathbf{x} = k - 2, \|\mathbf{x}\| = \|\mathbf{x}'\| = k\}.$$

Since the goal of the graph matching is to maximize the affinity sum between the matches, local search moves from one solution to another in such a way to encounter a solution that maximizes the objective (3.1). To this aim, we need to make appropriate changes to the composition of \mathcal{K} , that is to remove the nodes that have little affinity to the rest of the subgraph and substitute them with the nodes which have high affinity with the rest of the subgraph.

Let α_a be the sum of affinity values between a node v_a and all the nodes in \mathcal{K} :

$$\alpha_a = \sum_{b|v_b \in \mathcal{K}} \mathbf{A}_{ab} \tag{5.1}$$

Then change in the objective (3.1) when interchanging $v_a \in \mathcal{K}$ and $v_b \in \mathcal{L}$ is equal to:

$$\Delta_{ab} = \alpha_b - \alpha_a - \mathbf{A}_{ab}. \quad (5.2)$$

For simplicity, we denote $\alpha_a^{\mathcal{K}}$ if $v_a \in \mathcal{K}$ and respectively $\alpha_a^{\mathcal{L}}$ if $v_a \in \mathcal{L}$. Computing affinity values and the affinity change are illustrated in Fig. 5.1(c,d) and in Algorithm 1, lines 6-7.

5.1.1 Decreasing the size of local search space

Note that for a single solution \mathbf{x} , the search space contains $|\mathcal{K}| \cdot |\mathcal{L}|$ possible moves and as the search space of local search is large, we attempt to decrease the scope of the search to the movements which are most likely to cause the large positive change Δ in the objective. To this aim similarly to the method in [26], we select the pairs of nodes from \mathcal{K} and \mathcal{L} which are likely to result in the largest positive change in the objective. To do so, we sort values $\alpha_a^{\mathcal{K}}$ in the increasing order and $\alpha_a^{\mathcal{L}}$ in the decreasing order and identify the top t nodes in each of the list for which we compute the change in the objective (Alg. 1, line 8). Thus, we narrow the search space from $|\mathcal{K}| \cdot |\mathcal{L}|$ to t^2 where $t^2 \ll |\mathcal{K}| \cdot |\mathcal{L}|$. Define the narrowed space as $\tilde{N}(\mathbf{x})$. Let us note that swapping top nodes from each of the list may not be optimal if the affinity between two nodes we swap is large. Subsequently, among t^2 candidate pairs we choose the pair of nodes that results in the highest gain and swap the two nodes.

Once the move is done and a node v_a in \mathcal{K} is replaced by a node v_b , then and therefore the node affinity values for each node in both \mathcal{K} and \mathcal{L} must be updated accordingly:

$$\alpha_c := \alpha_c - \mathbf{A}_{ca} + \mathbf{A}_{cb}. \quad (5.3)$$

Note two special cases when $c = a$ and $c = b$. The affinity \mathbf{A}_{cc} is 0 for all c , and therefore when v_a is replaced by v_b in the subgraph, the sum of affinities of v_a to the new subgraph increases by its affinity to the new node in the subgraph, v_b . The case for the new node v_b is analogical.

$$\begin{aligned}\alpha_c &= \alpha_b + \mathbf{A}_{ab}, \text{ if } c = a \\ \alpha_c &= \alpha_b - \mathbf{A}_{ab}, \text{ if } c = b.\end{aligned}\tag{5.4}$$

5.1.2 Example

In this subsection, we consider an example that combines and illustrates the idea of optimizing the penalized association graph in the previous chapter in Section 4.4 by means of local search presented in this chapter. Consider Fig. 5.1. We use penalty to enforce one-to-one constraint through local search for graph matching between \mathcal{G}_1 and \mathcal{G}_2 with initial matching ij', jk', kk' violating the constraint. Using two graphs, the corresponding association graph is constructed. The weights between two matches correspond to the entries in the affinity matrix \mathbf{A} , e.g. the weight between matches nodes v_i and $v_{i'}$, and v_j and $v_{j'}$ is given by the entry $\mathbf{A}_{ii':jj'}$. Here the weights are 1 for all non-conflicting matches, and for conflicting matches the penalty weight $p = -2$ is used (for clarity, only the weights for the subgraph \mathcal{K} are shown). Given the association graph, we compute the affinity values of each node to \mathcal{K} according to Eq.(5.1). Then according to Eq.(5.2), we compute the change Δ_{ab} to the IQP objective for interchanging a node in \mathcal{K} with a node in \mathcal{L} . The exhaustive search which results in the optimal move would require $3 \cdot 6 = 18$ computations, but by means of parameter $t = 2$, in this case, we narrow down the search space to 4 computations of change Δ , and still find the optimal solution. Among the four potential moves we choose

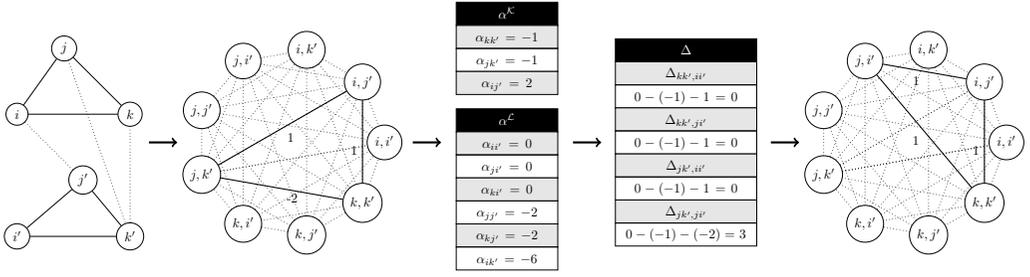


Figure 5.1 An example of using penalty to enforce one-to-one constraint through local search for (a) graph matching matching between \mathcal{G}_1 and \mathcal{G}_2 with initial matching ij', jk', kk' violating the constraint. (b) the corresponding association graph. The weights for non-conflicting matches are 1 and for conflicting matches -2 (for clarity, only the weights for the subgraph \mathcal{K} are shown). (c) affinity value of each node to \mathcal{K} given by Eq.(5.1) (d) The change Δ_{ab} to the IQP objective for interchanging top two matches from (c) according to Eq.(5.2). (e) Resulting subgraph with matches satisfying one-to-one constraint.

the one which results in the highest positive change of the objective, that is we exchange the nodes indicating match jk' with ji' . Thus we obtain the resulting subgraph with matches satisfying one-to-one constraint.

5.2 Tabu Search

The local search algorithm is a vanilla algorithm which forms the base for the tabu search. Local search defines the general direction of the search by aiming to choose the move which maximizes the change in the objective. On the other hand, tabu search defines the criteria whether a move proposed by local search can be executed or not.

In other words, the purpose of tabu search is to modify neighborhood $N(\mathbf{x})$ (usually by removing solutions) in such a way as to encourage or discourage certain moves in order to find the global solution. The advantages of the tabu search follow. Tabu search overcomes the shortcomings of simple descent method, which looks within the static neighborhood for the solutions that improve the objective, but terminates at a local optimum when no improving solution can be found. In contrast, the tabu search by redefining the neighborhood and forbidding certain moves, allows the objective to worsen in order to leave the current neighborhood in the search for more optimal solutions. As a result, optimization via tabu search is less dependent on the initialization as the tabu search has capability of moving away from the undesired states. Finally, the distinct feature of tabu search is its use of the history of search in making strategic changes to $N(\mathbf{x})$. The set of rules behind modification of search space are called *adaptive memory* and are described in detail in the following sections. The proposed method incorporates two types of adaptive memory, short-term memory and long-term memory.

5.2.1 Short term memory

The main component of tabu search is *short-term memory* which uses recent search history to modify the neighborhood of the current solution. Short-term memory is a selective memory that keeps track of solutions visited in the past to

forbid revisiting them in the future. Fig. 5.2 describes a simple example of short-term memory mechanism on a graph, which, in this work, is the association graph \mathcal{A} .

In the proposed algorithm, the search history of the optimization process of the objective (3.1) relies on a sequence of swaps between two nodes, $v_a \in \mathcal{K}$ and $v_b \in \mathcal{L}$. Thus, we consider a solution \mathbf{x} to be characterized by two nodes v_a and v_b that participate in the swap. The short-term memory is implemented to keep track of nodes that have been swapped and forbid them to participate in the swap within certain span of time in the future. Thus, the result of the short-term mechanism is a modified neighborhood $\tilde{N}(\mathbf{x}_{cur})$ which is the subset of the original neighborhood $N(\mathbf{x}_{cur})$.

Short-term memory is implemented via a list, called *tabu list* which keeps track of the nodes that are forbidden to participate in the swap at the current iteration. After every swap a node v_i that was swapped (there are two of them) is assigned a positive integer, s_i , randomly selected from the interval $[q, r]$ which prevents the variable to participate in swap for s_i subsequent iterations. Originally, all the values s_a are initialized to 0. These nodes who cannot participate in the swap are called *tabu variables*. Clearly, at subsequent iterations only *non-tabu* variables can be swapped.

5.2.2 Implementation details

In the proposed algorithm, at any given time we maintain pointers to three main solutions, \mathbf{x}_{cur} , the solution which indicates the point in solution space where the algorithm currently stands, \mathbf{x}_{cur}^* , the maximum solution in the current run of the tabu search and \mathbf{x}^* , the overall maximum solution of the objective (3.1). At every iteration, a subset of solutions given by parameter t is used for candidate moves from \mathbf{x}_{cur} .

Non-tabu solutions are computed from $\tilde{N}(\mathbf{x}_{cur})$ according to Eq. (5.2). Then the non-tabu solution with highest change Δ in the objective becomes the incumbent solution \mathbf{x}_{cur} (Algorithm 1, line 14). Notice that the algorithm is designed in such a way that \mathbf{x}_{cur} changes after every iteration and we are never stuck in one place in the solution space. Thus, the tabu search algorithm explores the solution space in a way that the objective function value (3.1) may fluctuate. Notice that if all the potential moves have negative value, then the objective function will decrease. The solution \mathbf{x}^* is updated when the new solution is larger. The proposed algorithm also incorporates the exception, called *aspiration criterium*, which allows tabu move to take place. The tabu solution becomes current solution if it is strictly bigger than the maximum solution \mathbf{x}^* (Algorithm 1, line 10).

5.2.3 Example

Let us subsequently consider a simple example of implementing short-term memory to find the maximum weighted subgraph. Consider Fig. 5.2. At each iteration we swap two non-tabu nodes to maximize the objective (3.1). After being swapped, the nodes (marked as shaded) remain tabu for 1 iteration. Initially, $\mathcal{K} = acf$ and $|acf| = 6$. As no swaps have been made previously, no variables are tabu and we can swap any two nodes. In the first iteration given the weights, the move where e substitutes a results in the highest change in the objective, and $|cef| = 8$. In the next iteration, the variables that participated in the move in the first iteration are tabu variables, that is e that entered (and cannot leave) and a that left (and cannot reenter). We choose between b and d despite the fact that both choices lower the objective. We then choose b to substitute c which results in the smallest loss. The subgraph weights becomes, $|bef| = 7$. Subsequently, tabu variables are b and c , which cannot participate

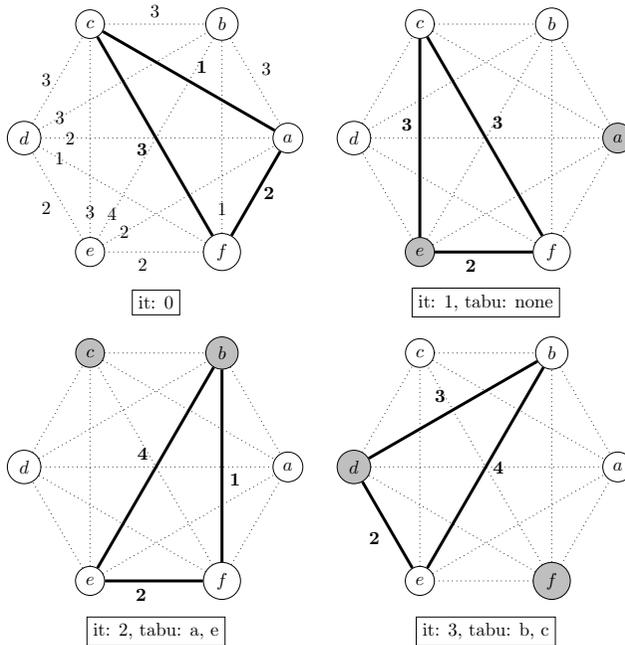


Figure 5.2 An example of tabu search to find the maximum weighted clique. At each iteration we swap two non-tabu nodes to maximize the objective (3.1). After being swapped, the nodes (marked as shaded) remain tabu for 1 iteration. (0) initially, $\mathcal{K} = acf$ and $|acf| = 6$; no variables are tabu. (1) e substitutes a , and $|cef| = 8$. (2) tabu variables are e that entered and a that left, we choose between b and d , both choices lower the objective, choose b , $|bef| = 7$. (3) tabu variables are b and c , choose d to obtain the global optimum $|bed| = 9$.

in the swap in the following round. choose d to obtain the global optimum $|bed| = 9$. This example illustrates well how tabu search allows us to escape the local maximum to arrive afterward at the global maximum.

5.2.4 Long term memory

Long-term memory, in contrast to short-term memory, takes into account the extended search history and is activated when the short-term criterion fails to improve the objective function. Long-term memory maintains a list of solutions (various criteria can be adopted to make this list) which can be used as new solutions from which local search may proceed. The search is restarted when the solution \mathbf{x}_{\max} is not improved for the given number of iterations (for clarity the loop not included in the Algorithm 1). The rationale behind the criterion is that the current region is not promising for finding a more optimal solution and we choose to relocate to a more promising region.

5.2.5 Implementation details

We implement long term memory as a list of “second” solutions S_{sec} which consists of solutions that at every iteration landed just behind the current solution \mathbf{x}_{cur} , called the “second” solution list [2]. Therefore, in the algorithm we also maintain pointers to a secondary solution \mathbf{x}_{sec} which, if better, replaces the worst solution in the list of secondary solutions. When the long-term termination criterion is activated, a search is restarted from the solution $\mathbf{x} \in S_{sec}$ with the highest objective. In the subsequent section, we present the experimental results of both short-term and long-term mechanism compared to the existing graph matching methods.

Algorithm 1: Discrete tabu search for Graph Matching

Input: Affinity matrix $\mathbf{A} \in \mathbb{R}^{n_1 n_2 \times n_1 n_2}$
Output: Solution vector $\mathbf{x}^* \in \{0, 1\}^{n_1 n_2 \times 1}$
penalize conflicting matches in \mathbf{A} with penalty p ;
 $\mathbf{S} \leftarrow \{\}$ // stores multiple solutions
for $i=1$ to sol **do**
 $it \leftarrow 0$; $nonImpIt \leftarrow 0$; $bestTabu \leftarrow -\infty$, $bestnTabu \leftarrow -\infty$; $tabuIt[] \leftarrow 0[]$; $S_{sec}[] = -\infty[]$;
 while $NonImpIter > endIt$ **do**
 initialize subgraph \mathcal{K} with random nodes from affinity graph \mathcal{A} ;
 $\alpha_a \leftarrow$ compute subgraph affinity for $v_a \in \mathcal{V}_{\mathcal{A}}$;
 $\Delta_{ab} \leftarrow$ compute the change (5.2) in the objective for exchanging top t nodes from sorted $\alpha_a^{\mathcal{K}}$ (increasingly) and $\alpha_a^{\mathcal{L}}$ (decreasingly) ;
 $\mathbf{x}_{tabu}, \mathbf{x}_{nontabu}, \mathbf{x}_{nontabu2} \leftarrow$ choose tabu ($tabuIt > it$), best non-tabu and second best tabu solution with highest Δ_{ab} ;
 if $f(\mathbf{x}_{tabu}) > f(\mathbf{x}_{nontabu})$ and $f(\mathbf{x}_{tabu}) > f(\mathbf{x}^*)$ **then**
 $\mathbf{x}_{cur} \leftarrow \mathbf{x}_{tabu}$;
 $\mathbf{x}_{sec} \leftarrow \mathbf{x}_{nontabu}$
 else
 $\mathbf{x}_{cur} \leftarrow \mathbf{x}_{nontabu}$;
 // can lower the objective.
 $\mathbf{x}_{sec} \leftarrow \mathbf{x}_{nontabu2}$
 end
 $tabuIt(a), tabuIt(b) \leftarrow rand([q, r]) + it$
 $\alpha_a \leftarrow$ recompute subgraph affinity for $v_a \in \mathcal{V}_{\mathcal{A}}$ for new solution \mathbf{x}_{cur} using (5.3) and (5.4);
 if $f(\mathbf{x}_{cur}) > f(\mathbf{x}_{cur}^*)$ **then**
 $\mathbf{x}_{cur}^* \leftarrow \mathbf{x}_{cur}$
 else
 $nonImpIt = nonImpIt + 1$;
 end
 if $f(\mathbf{x}_{cur}) > f(\mathbf{x}_{cur}^*)$ **then**
 $\mathbf{x}_{cur}^* \leftarrow \mathbf{x}_{cur}$
 end
 if $f(\mathbf{x}_{sec}) > \min(S_{sec}[])$ **then**
 $S_{sec}[] \leftarrow S_{sec}[] \cup \mathbf{x}_{sec} - \min(S_{sec}[])$
 end
 end
 if $f(\mathbf{x}_{cur}^*) > f(\mathbf{x}^*)$ **then**
 $\mathbf{x}^* \leftarrow \mathbf{x}_{cur}^*$
 end
end

Table 5.1 Parameters used in Algorithm 1

p	penalty for conflicting matches. $(-4 \cdot \max \mathcal{A})$
$q-r$	lower and upper bound of the possible tabu tenure. $(2-4)$
t	number of nodes in \mathcal{K} and \mathcal{L} which are considered for move in a given iteration. (5)
$endIt$	number of iterations that do not improve the objective to terminate. (1500)
sol	number of runs of the algorithm. (20)

Chapter 6

Experiments

This section presents extensive evaluation of the proposed method through three sets of experiments in both synthetic and real-world setting.

6.1 Synthetic random graph matching

The experiments perform an evaluation on the randomly synthesized graphs which simulate the practical environment, where deformation noise and outliers obfuscate the underlying matching. In this experiment, we construct two graphs, \mathcal{G}_1 and \mathcal{G}_2 , each with n nodes such that $n = n_{in} + n_{out}$ where n_{in} and n_{out} denote the number of inlier and outlier nodes, respectively. First, we construct \mathcal{G}_1 such that its edge attributes \mathbf{w}_{ij} are generated from the uniform distribution $[0, 1]$. Subsequently, \mathcal{G}_2 is created by perturbing the attributes of \mathcal{G}_1 with Gaussian noise $\epsilon \sim N(0, \sigma_s^2)$. The affinity matrix values between two potential matches v_i^1 with $v_{i'}^2$ and v_j^1 with $v_{j'}^2$ are computed using the negative exponential of $\mathcal{A}_{ii';jj'} = \exp(-\|\mathbf{w}_{ij} - \mathbf{w}_{i'j'}\|/\sigma^2)$, where $\sigma^2 = 0.1$.

In the synthetic setting, we evaluate the proposed discrete tabu search algo-

and objective score. Fig. 6.1 presents the results. In the first and second setting (Fig. 6.1a and 6.1b), the deformation noise σ varies between 0 and 0.4 by 0.05 increments while we fix the number of outliers to 0 and 10. In the third and fourth setting (Fig. 6.1c and 6.1d), the number of outliers varies between 0 and 20 by increments of 4 while the noise levels are fixed to be $\sigma = 0$ and $\sigma = 0.1$.

Under varying condition the proposed methods perform outstandingly well, noting the better performance of tabu search which includes only short-term memory. As the only evaluated method, it achieves 100% accuracy given high levels of noise ($\sigma = 0.25$ with no outliers and $\sigma = 0.1$ and $\sigma = 0.15$ with outliers). While varying outliers, the method slightly falls short to SEA, whose performance, however, considerably deteriorates when noise appears. The proposed method proves to be particularly robust in more practical settings where both outliers and noise are present. Given choice of parameters from Table 5.2.1 and varying *endIt* and *sol*, it obtains superior solutions in the presence of noise and outliers in 1.0-4.4 sec.; FGM - 23.5, RRWM - 0.2, IPFP - 0.2, GAGM - 0.4 and SEA - 1.1 (averaged results for results in Fig. 6.1(b,d)).

6.1.1 Experimental evaluation of the proposed one-to-one constraint

We further present experimental ramifications of the Theorem 1. In this work, we utilize softer constrain, letting $p = -9$, noting that there is no noticeable improvement when the penalty is higher. Fig. 6.2 shows the performance of our method with and without one-to-one constraint enforced artificially. We only include the results of the algorithm with the short term memory component. The solid lines show the performance of the algorithm in the setting where the penalty exists and when $p = 0$. The dotted lines indicate the performance of the aforementioned solutions which underwent the post-processing step enforcing strictly one-to-one constraint (in case the solution violated the constraint).

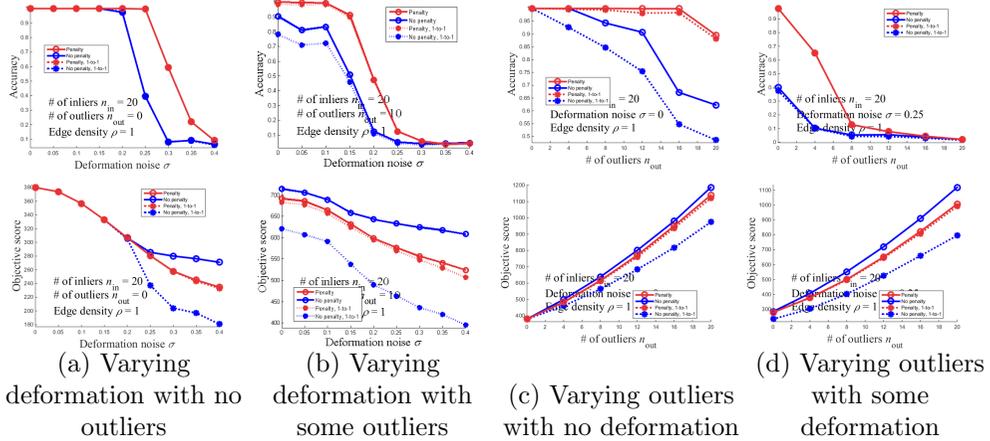


Figure 6.2 Experimental ramifications of the proposed soft one-to-one constraint, where $p = -9$. Accuracy and score curves are plotted for four different conditions: varying the amount of deformation noise with fixed number of outliers (a) $n_{out} = 0$ and (b) $n_{out} = 10$, and varying the number of outliers with fixed deformation noise (c) $\sigma = 0$ (d) $\sigma = 0.25$.

The results show that the penalty has considerable impact on the performance and in essence it produces solutions that are one-to-one. It is an important statement when comparing the score results between all the methods which we evaluate along with the proposed method. This is because the reported score for the existing methods in Fig. 6.1 is the score of the one-to-one solution. And the fact that a non-one-to-one solution can result in much higher score can be seen in Fig. 6.2, where non-constrained solutions have become susceptible to noise achieving higher score yet lower accuracy. Nonetheless as Fig. 6.2 shows, the score of the solution \mathbf{y} obtained from the optimization of the penalized association graph and the score of the one-to-one solution $\mathbf{y} \in \mathbf{X}_{\text{one-to-one}}$ obtained via post-processing is almost identical. Finally, we note the technicality that although we optimize the score on the penalized association graph, the reported score is the objective score given by the objective function (3.1) using

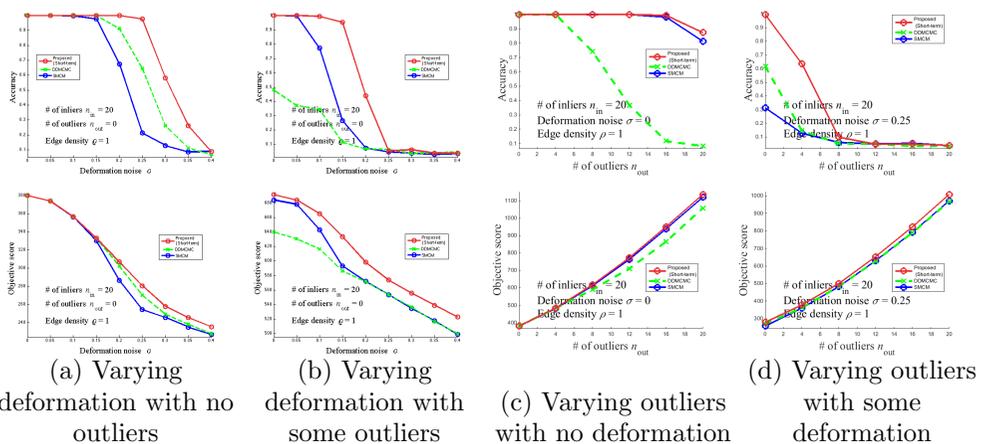


Figure 6.3 Evaluation of DDMCMC [18] and SMCM [24], which solve one-to-one constrained IQP directly in the discrete space. Accuracy and score curves are plotted for four different conditions: varying the amount of deformation noise with fixed number of outliers (a) $n_{out} = 0$ and (b) $n_{out} = 10$, and varying the number of outliers with fixed deformation noise (c) $\sigma = 0$ (d) $\sigma = 0.25$.

the original affinity matrix \mathbf{A} .

6.1.2 Evaluation of methods that work in discrete space on the synthetic dataset

Unlike in the previous section, which evaluated the methods which work (at least partially) in continuous space, in this section, we include experiments on the synthetic dataset of the methods that, similarly to ours, work in discrete space and present the results in Fig. 6.3. We evaluate two additional methods, DDMCMC [18] and SMCM [24]. SMCM deals well with outliers but underperforms in the noisy environment. In contrast, DDMCMC copes better with noise. The proposed method outperforms the two methods in all four experimental settings.

6.1.3 Parameter settings

In both above experiments and the experiments described in Section 6.1 of the main paper, we use the parameter settings as indicated in Table 5.2.1: $q = 2$, $r = 4$ as the lower and upper bound of how long a variable can stay on the tabu list. Moreover, $t = 5$ is used as the number of swap candidate nodes in \mathcal{K} and \mathcal{L} , and $endIt = 1500$ in the synthetic experiments and $endIt = 250$ in the feature correspondence experiments (described in the next section) as the number of iterations with no change in the objective before the tabu search terminates. In order to understand better the parameter settings, one can consider the following descriptions:

- p : empirically, the best results are obtained for $p = -3 \cdot \max(\mathcal{A})$ to $-5 \cdot \max(\mathcal{A})$. Lower p does not produce better results though may produce solution closer to be strictly one-to-one which may be undesirable in case of outliers present, i.e. we do not want to match outliers and prefer 1-to-2 matching if two points are close to each other and are similar. Higher p leads soft one-to-one constraint to be too lenient.
- $q - r$: larger q and r lead to more diversification, smaller q and r mean more intensification but a number of worsening iterations may not be enough to escape from local minima.
- t : lower number often misses optimal move, higher number increases complexity substantially.
- $endIt$: values 500-1500 bring similar superior results, with smaller $endIt$ improving running time.
- sol : empirically $endIt = 10 - 20$ produces comparable results with larger sol , if time is not a big concern increasing sol may have some little effect. $sol = 5 - 10$ produces results better than the methods we tested against

Table 6.1 SNU Dataset Average Accuracy (%)

Alg	SEA	GAGM	RRWM	IPFP	Proposed
Accuracy(%)	56.3	70.3	73.6	74.1	79.0

Table 6.2 Willow Object Dataset Average Accuracy (%)

Class	Car	Duck	Face	Motorbike	Winebottle	average
SEA	30.3	26.5	72.9	32.4	45.4	41.4
GAGM	38.6	35.4	79.4	38.3	53.9	49.1
RRWM	38.0	35.7	78.3	40.4	53.7	49.2
IPFP	37.6	36.9	80.4	40.4	56.0	50.3
Proposed	39.5	36.1	80.2	44.0	57.9	51.5

but the improvement margin is smaller.

Given the default parameters in the synthetic experiments, an average run of tabu search takes from 0.02 sec. on average for a matrix of size 400×400 (corresponding to matching two graphs of 20 nodes each) and no noise to 0.45 sec. on average for a matrix 1600×1600 and high levels of noise. We run the tabu search for several (six and nine, respectively) times for different number of outliers and levels of noise. Then to produce multiple solutions, we run the process 20 times. To obtain average results we rerun the entire experiment 50 times. Given this setting, the overall time of obtaining matching results for two graphs varies between about two minutes and one hour. In the feature correspondence experiments, the number of matches (nodes in the association graph) generally varies between 2000 and 4000. We use processor 2.3 GHz Intel Core i7 and 16 GB 1600 MHz DDR3 memory.

6.2 Real image feature correspondence

In this section, we evaluate the proposed method for feature correspondence task on the two datasets: SNU dataset [6] and Willow Object Class dataset [4]. RRWM, IPFP, GAGM, and SEA are compared with the proposed method ¹.

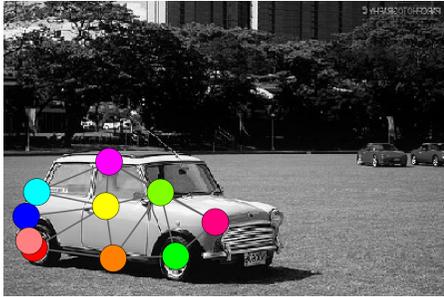
SNU dataset consists of 30 pairs of images most of which are collected from Caltech-101 [9] and MSRC ² dataset. Each pair of images have similar view-points with large intra-class variation. In addition to the original experimental setting [6], which did not use unary score, WHO [15] and their inner product are used as feature descriptor and unary score to enhance the overall performance. The average accuracy over 10 experiments is shown in Table 6.1. The proposed algorithm outperforms all the other methods.

Willow dataset consists of 5 classes, whose images are collected from Caltech-256 [14] and PASCAL VOC2007 dataset [8] with 10 manual keypoint annotations for each image. Each class contains 40 to 108 images from various view-points with clutter, therefore, feature correspondence is extremely challenging.

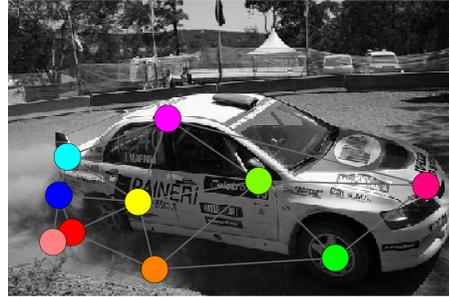
Therefore, we restricted the graph constructed from one side image to have only keypoints without outliers. WHO [15] and HARG [4] are used as node and edge descriptors, respectively. The average accuracy over 100 random pairs is shown in Table 6.2. Classes with low intra-class variation (face and winebottle) show high accuracy in overall, while the performance of the classes with large intra-class variation (car, duck, and motorbike) is relatively low. The proposed algorithm outperforms the other algorithms in car, motorbike and winebottle classes, while is comparable in face and motorbike classes. Some qualitative example results are shown in Fig. 6.4-6.8. The average running times are fol-

¹FGM is not compared because it cannot be applied in the experimental setting of [6], which uses initial candidate matches.

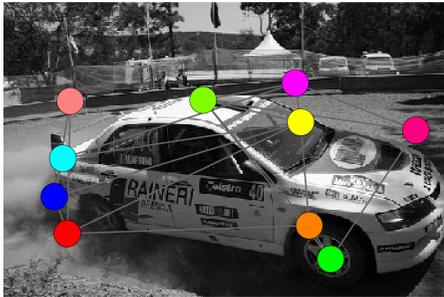
²<http://research.microsoft.com/en-us/projects/objectclassrecognition/>



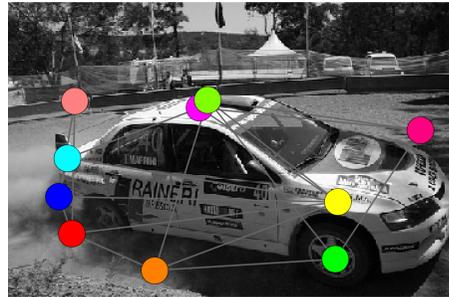
Model graph



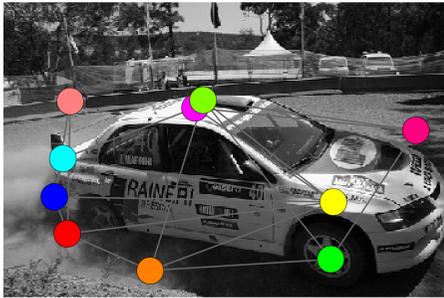
Proposed method



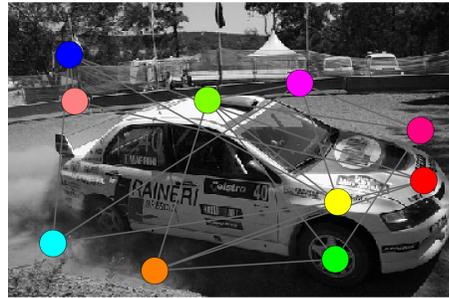
IPFP



RRWM

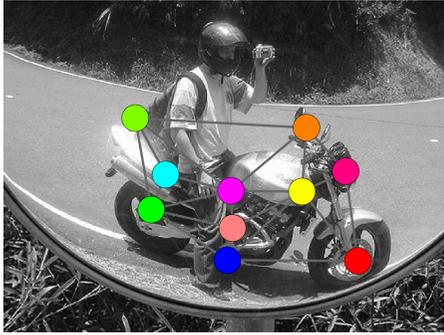


GAGM

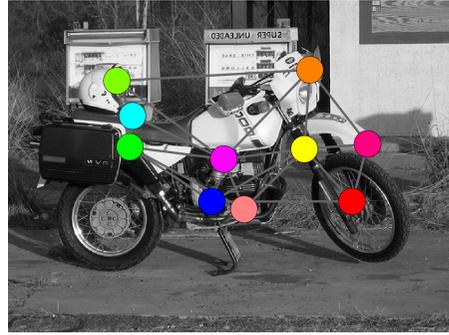


SEA

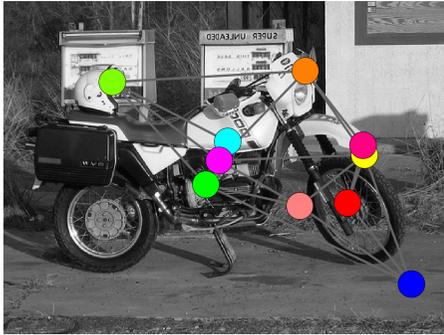
Figure 6.4 Example results of the real image matching on Willow Object dataset. Keypoints are distinguished by different colors in the model image and the matched locations are depicted for each algorithm



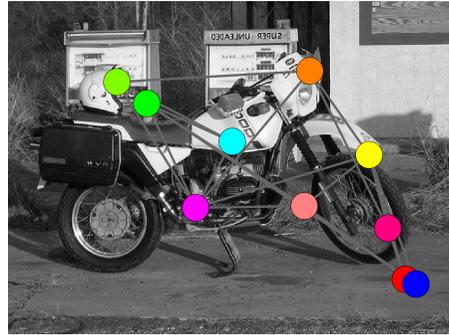
Model graph



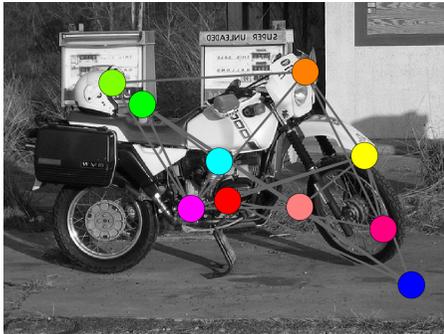
Proposed method



IPFP



RRWM

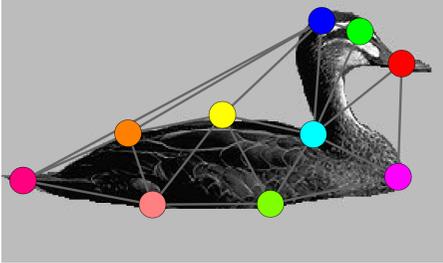


GAGM

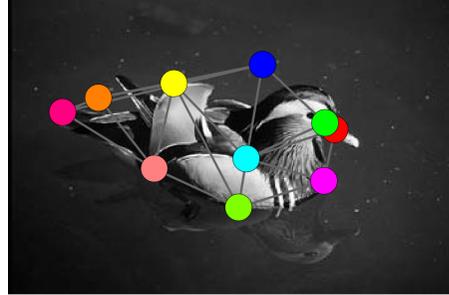


SEA

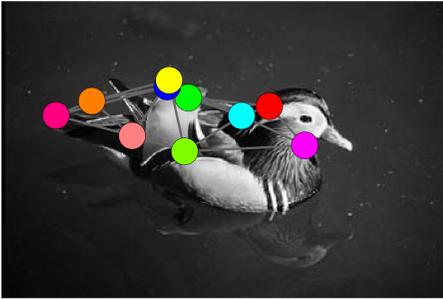
Figure 6.5 Example results of the real image matching on Willow Object dataset. Keypoints are distinguished by different colors in the model image and the matched locations are depicted for each algorithm



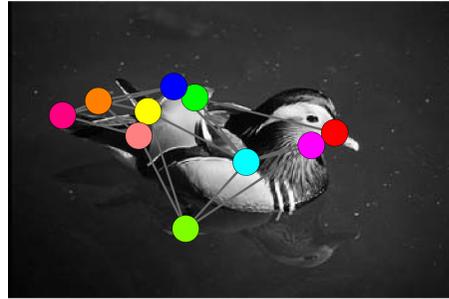
Model graph



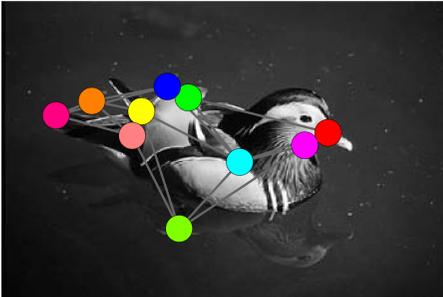
Proposed method



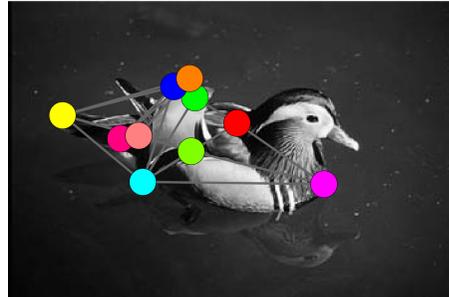
IPFP



RRWM



GAGM

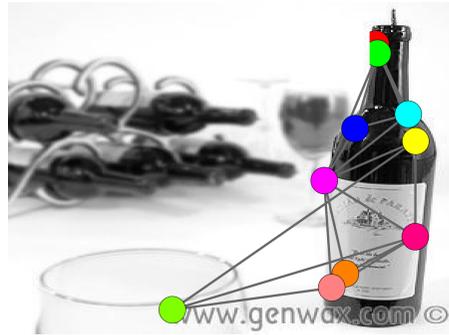


SEA

Figure 6.6 Example results of the real image matching on Willow Object dataset. Keypoints are distinguished by different colors in the model image and the matched locations are depicted for each algorithm



Model graph



Proposed method



IPFP



RRWM

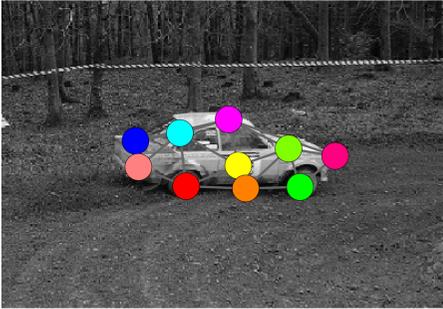


GAGM

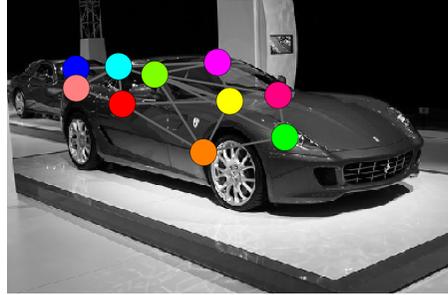


SEA

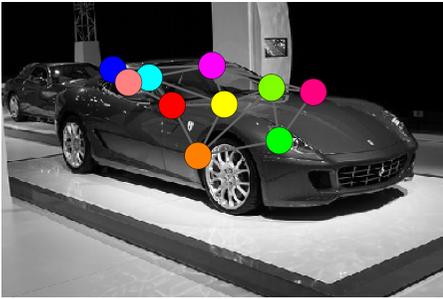
Figure 6.7 Example results of the real image matching on Willow Object dataset. Keypoints are distinguished by different colors in the model image and the matched locations are depicted for each algorithm



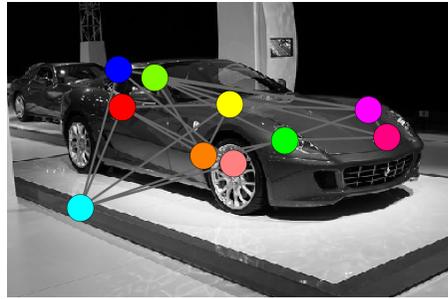
Model graph



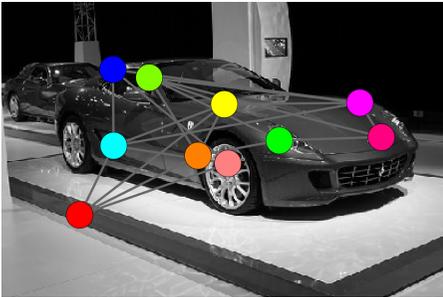
Proposed method



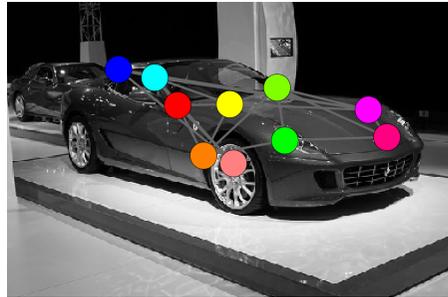
IPFP



RRWM



GAGM



SEA

Figure 6.8 Example results of the real image matching on Willow Object dataset. Keypoints are distinguished by different colors in the model image and the matched locations are depicted for each algorithm

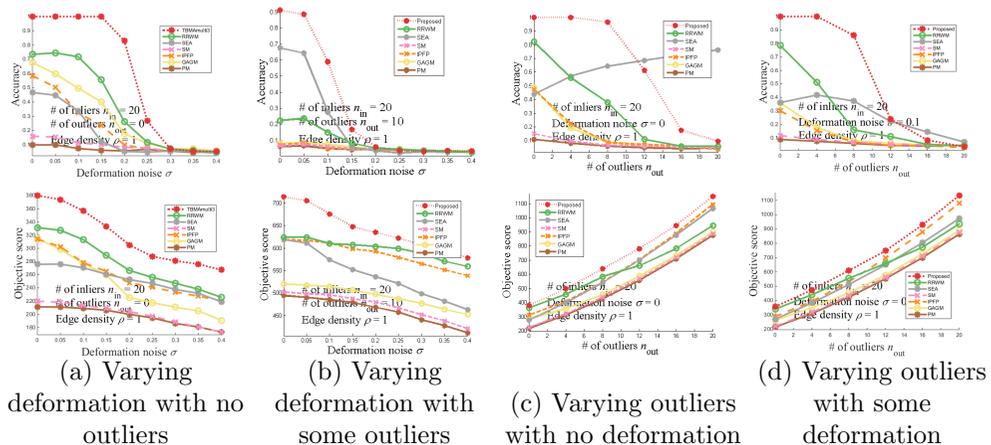


Figure 6.9 Accuracy and score curves are plotted for four different conditions: varying the amount of deformation noise with fixed number of outliers (a) $n_{out} = 0$ and (b) $n_{out} = 10$, and varying the number of outliers with fixed deformation noise (c) $\sigma = 0$ (d) $\sigma = 0.1$.

lowing: proposed - 0.93s, RRWM - 0.04s, IPFP - 0.03s, GAGM - 0.08s, SEA - 0.49s.

6.3 Repeated Structure Matching

In this set of experiments, we present particular suitability of tabu search method to multiple graph matching, attempting to discover repeated structures or multiple instances of an object. The advantage is due to the fact that the method produces multiple solutions (Alg. 1, line 2 & 32), each of which is a set of matches between two regions in two graphs. The multiple solutions may or may not correspond to the same regions in the original graphs. This setting makes graph matching more useful in practical applications, e.g., in feature correspondence, it overcomes the conventional restriction to allow one object per image.

In this set of experiments, we simulate the multiple object environment. We

construct \mathcal{G}_1^{mult} and \mathcal{G}_2^{mult} according to the procedure in Section (6.1) where $|\mathcal{G}_1^{mult}| = |\mathcal{G}_1^{mult}| = n$ and $|\mathcal{G}_2^{mult}| = 2|\mathcal{G}_1^{mult}| = 2n$ in such a way that \mathcal{G}_2^{mult} consists of two independently perturbed copies of \mathcal{G}_1^{mult} . The affinity matrix \mathbf{A}_{mult} has size $2n^2 \times 2n^2$. We subsequently perform tabu search optimization on the association graph \mathcal{A}_{mult} obtained from \mathbf{A}_{mult} .

We compare our method to SEA which also produces multiple solutions and the remaining methods evaluated in Section 6.1 (with the exception of FGM whose running time is too large for this set of experiments) adapted to multiple matching setting by means of the following sequential matching procedure. We perform matching m times (where $m = 2$ in our case) after each stage removing matched nodes and performing the subsequent round on the remaining nodes. The methods are evaluated in the same settings as in Section 6.1 where we vary deformation noise and the number of outliers. The results are presented in Fig. 6.9. The proposed method outperforms the existing methods by large margin when noise is varied. The improving accuracy of SEA with the increasing number of outliers is noteworthy. The proposed method again performs very robustly in a mixed environment with both noise and outliers present.

Chapter 7

Summary and Conclusions

In this work, we propose graph matching algorithm which utilizes tabu search to search discrete solution space directly and effectively. First, we build the penalized association graph which becomes the framework to move between its subgraphs by forbidding certain solutions and encouraging others. As a result unlike any of the existing works, this method works both in discrete space (by itself discrete graph matching optimization is rare in graph matching) and incorporates one-to-one constraint softly into the optimization process with no post-processing step. The proposed method is suitable for multiple graph matching as it produces different solutions that potentially correspond to multiple instances of an object. The algorithm proves to be outstandingly robust achieving superior results to the existing algorithms (in some cases over 40 percentage points, which given a long history of graph matching research, is notable) in the extensive combination of experiments where noise and outliers are present.

Bibliography

- [1] Andrea Albarelli, Samuel Rota Bulo, Andrea Torsello, and Marcello Pelillo. Matching as a non-cooperative game. In *International Conference on Computer Vision (ICCV)*, pages 1319–1326. IEEE, 2009.
- [2] Roberto Aringhieri, Roberto Cordone, and Yari Melzani. Tabu search versus grasp for the maximum diversity problem. *4OR*, 6(1):45–60, 2008.
- [3] Alexander C Berg, Tamara L Berg, and Jitendra Malik. Shape matching and object recognition using low distortion correspondences. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 1, pages 26–33. IEEE, 2005.
- [4] Minsu Cho, Karteek Alahari, and Jean Ponce. Learning graphs to match. In *International Conference on Computer Vision (ICCV)*, 2013.
- [5] Minsu Cho and Jungmin Lee. Feature correspondence and deformable object matching via agglomerative correspondence clustering. In *International Conference on Computer Vision (ICCV)*, pages 1280–1287. IEEE, 2009.

- [6] Minsu Cho, Jungmin Lee, and Kyoung Mu Lee. Reweighted random walks for graph matching. In *European Conference on Computer Vision (ECCV)*, 2010.
- [7] Olivier Duchenne, Armand Joulin, and Jean Ponce. A graph-matching kernel for object categorization. In *International Conference on Computer Vision (ICCV)*, pages 1792–1799. IEEE, 2011.
- [8] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes Challenge 2007 (VOC2007) Results. <http://www.pascal-network.org/challenges/VOC/voc2007/workshop/index.html>.
- [9] Li Fei-Fei, Rob Fergus, and Pietro Perona. Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories. *Computer Vision and Image Understanding*, 106(1):59–70, 2007.
- [10] Fred Glover. Future paths for integer programming and links to artificial intelligence. *Comput. Oper. Res.*, 13(5):533–549, May 1986.
- [11] Fred Glover. Tabu search-part i. *ORSA Journal on computing*, 1(3):190–206, 1989.
- [12] Fred Glover and Manuel Laguna. *Tabu search*. Springer, 1999.
- [13] Steven Gold and Anand Rangarajan. A graduated assignment algorithm for graph matching. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, (68):411–436, 1996.
- [14] Gregory Griffin, Alex Holub, and Pietro Perona. Caltech-256 object category dataset. 2007.

- [15] Bharath Hariharan, Jitendra Malik, and Deva Ramanan. Discriminative decorrelation for clustering and classification. In *European Conference on Computer Vision (ECCV)*, pages 459–472. Springer, 2012.
- [16] Radu Horaud and Thomas Skordas. Stereo correspondence through feature grouping and maximal cliques. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 11(11):1168–1180, 1989.
- [17] Suna Kim, Suha Kwak, Jan Feyereisl, and Bohyung Han. Online multi-target tracking by large margin structured learning. In *Asian Conference on Computer Vision (ACCV)*, pages 98–111. Springer, 2013.
- [18] Jungmin Lee, Minsu Cho, and Kyoung Mu Lee. A graph matching algorithm using data-driven markov chain monte carlo sampling. In *International Conference on Pattern Recognition (ICPR)*, pages 2816–2819. IEEE, 2010.
- [19] Marius Leordeanu and Martial Hebert. A spectral technique for correspondence problems using pairwise constraints. In *International Conference on Computer Vision (ICCV)*, 2005.
- [20] Marius Leordeanu and Martial Hebert. An integer projected fixed point method for graph matching and map inference. In *Conference on Neural Information Processing Systems (NIPS)*, 2009.
- [21] Hairong Liu, Longin Jan Latecki, and Shuicheng Yan. Fast detection of dense subgraphs with iterative shrinking and expansion. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 35, 2013.

- [22] Hairong Liu and Shuicheng Yan. Common visual pattern discovery via spatially coherent correspondences. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1609–1616. IEEE, 2010.
- [23] Sébastien Sorlin and Christine Solnon. Reactive tabu search for measuring graph similarity. In *Graph-Based Representations in Pattern Recognition*, pages 172–182. Springer, 2005.
- [24] Yumin Suh, Minsu Cho, and Kyoung Mu Lee. Graph matching via sequential monte carlo. In *European Conference on Computer Vision (ECCV)*, pages 624–637. 2012.
- [25] Shinji Umeyama. An eigendecomposition approach to weighted graph matching problems. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 10(5):695–703, 1988.
- [26] Yang Wang, Jin-Kao Hao, Fred Glover, and Zhipeng Lü. A tabu search based memetic algorithm for the maximum diversity problem. *Engineering Applications of Artificial Intelligence*, 27:103–114, 2014.
- [27] Mark L Williams, Richard C Wilson, and Edwin R Hancock. Deterministic search for relational graph matching. *Pattern Recognition*, 32(7):1255–1271, 1999.
- [28] Junchi Yan, Chao Zhang, Hongyuan Zha, Wei Liu, Xiaokang Yang, and Stephen M. Chu. Discrete hypergraph matching. June 2015.
- [29] Zhuoyi Zhao, Yu Qiao, Jie Yang, and Li Bai. From dense subgraph to graph matching: A label propagation approach. In *International Conference on Audio, Language and Image Processing (ICALIP)*, pages 301–306. IEEE, 2014.

- [30] Feng Zhou and Fernando De la Torre. Factorized graph matching. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.

초록

그래프 정합은 컴퓨터 비전의 근본적인 문제이다. 본 논문에서는 타부 검색법에 기반한 새로운 그래프 정합 알고리즘을 제안한다. 제안하는 방법은, 그래프 정합과 대응되는 연관 그래프의 최대 가중 클릭 문제(weighted maximum clique problem)를 만들어 타부 검색법을 이용하여 풀고 난 후 다시 원래의 해공간으로 가져오는 전략을 취한다. 타부 검색법은 지역해를 탈출하면서 최적의 해를 찾기 위해 이전까지의 검색 기록을 전략적으로 이용한다는 특징이 있다. 제안하는 방법은 기존의 방법과는 다르게 일대일 대응조건을 약하게 적용하여 본래의 이산 해 공간을 효과적으로 탐색하기 때문에 더 좋은 해를 얻을 수 있다. 합성 그래프 실험과 이를 실제 영상정합 문제에 적용한 실험을 수행하였고, 실험 결과를 통해 다양한 환경에서 알고리즘이 견고하게 동작함을 확인할 수 있다.

주요어: 그래프 정합, 타부 검색법, 영상 정합, 물체 인식

학번: 2013-23854